Completeness of Decreasing Diagrams for the Least Uncountable Cardinality

Ievgen Ivanov

Taras Shevchenko National University of Kyiv

Abstract

In [8] it was formally proved that the decreasing diagrams method [7] is sound for proving confluence: if a binary relation r has LD property defined in [8], then it has CR property defined in [6].

In this formal theory it is proved that if the cardinality of r does not exceed the first uncountable cardinal, then r has CR property if and only if r has LD property. As a consequence, the decreasing diagrams method is complete for proving confluence of relations of the least uncountable cardinality.

A paper that describes details of this proof has been submitted to the FSCD 2025 conference. This formalization extends formalizations [1, 5, 4, 2] and the paper [3].

Contents

1 Preliminaries

1.1 Formal definition of finite levels of the DCR hierarchy

theory *Finite-DCR-Hierarchy*
imports *Main*
begin

1.1.1 Auxiliary definitions

definition *confl-rel*
where *confl-rel* $r \equiv (\forall\, a\ b\ c.\ (a,b) \in r\hat{}{*} \wedge (a,c) \in r\hat{}{*} \longrightarrow (\exists\ d.\ (b,d) \in r\hat{}{*} \wedge (c,d) \in r\hat{}{*})\ )$

definition *jn00* :: $'a\ rel \Rightarrow 'a \Rightarrow 'a \Rightarrow bool$
where
$jn00\ r0\ b\ c \equiv (\exists\ d.\ (b,d) \in r0\hat{}{=} \wedge (c,d) \in r0\hat{}{=})$

definition *jn01* :: $'a\ rel \Rightarrow 'a\ rel \Rightarrow 'a \Rightarrow 'a \Rightarrow bool$
where
$jn01\ r0\ r1\ b\ c \equiv (\exists\, b'\ d.\ (b,b') \in r1\hat{}{=} \wedge (b',d) \in r0\hat{}{*} \wedge (c,d) \in r0\hat{}{*})$

definition *jn10* :: $'a\ rel \Rightarrow 'a\ rel \Rightarrow 'a \Rightarrow 'a \Rightarrow bool$
where
$jn10\ r0\ r1\ b\ c \equiv (\exists\, c'\ d.\ (b,d) \in r0\hat{}{*} \wedge (c,c') \in r1\hat{}{=} \wedge (c',d) \in r0\hat{}{*})$

definition *jn11* :: $'a\ rel \Rightarrow 'a\ rel \Rightarrow 'a \Rightarrow 'a \Rightarrow bool$
where
$jn11\ r0\ r1\ b\ c \equiv (\exists\, b'\ b''\ c'\ c''\ d.\ \ (b,b') \in r0\hat{}{*} \wedge (b',b'') \in r1\hat{}{=} \wedge (b'',d) \in r0\hat{}{*}$
$\wedge (c,c') \in r0\hat{}{*} \wedge (c',c'') \in r1\hat{}{=} \wedge (c'',d) \in r0\hat{}{*})$

definition *jn02* :: $'a\ rel \Rightarrow 'a\ rel \Rightarrow 'a\ rel \Rightarrow 'a \Rightarrow 'a \Rightarrow bool$
where
$jn02\ r0\ r1\ r2\ b\ c \equiv (\exists\, b'\ d.\ \ (b,b') \in r2\hat{}{=} \wedge (b',d) \in (r0 \cup r1)\hat{}{*} \wedge (c,d) \in (r0 \cup r1)\hat{}{*}\ )$

definition *jn12* :: $'a\ rel \Rightarrow 'a\ rel \Rightarrow 'a\ rel \Rightarrow 'a \Rightarrow 'a \Rightarrow bool$
where
$jn12\ r0\ r1\ r2\ b\ c \equiv (\exists\, b'\ b''\ d.\ \ (b,b') \in (r0)\hat{}{*} \wedge (b',b'') \in r2\hat{}{=} \wedge (b'',d) \in (r0 \cup r1)\hat{}{*}$
$\wedge (c,d) \in (r0 \cup r1)\hat{}{*})$

definition *jn22* :: $'a\ rel \Rightarrow 'a\ rel \Rightarrow 'a\ rel \Rightarrow 'a \Rightarrow 'a \Rightarrow bool$
where
$jn22\ r0\ r1\ r2\ b\ c \equiv (\exists\, b'\ b''\ c'\ c''\ d.\ \ (b,b') \in (r0 \cup r1)\hat{}{*} \wedge (b',b'') \in r2\hat{}{=} \wedge (b'',d) \in (r0 \cup r1)\hat{}{*}$

$\wedge\ (c,c') \in (r0 \cup r1)\hat{}* \wedge (c',c'') \in r2\hat{}= \wedge (c'',d) \in (r0 \cup r1)\hat{}*)$

definition *LD2* :: *'a rel* ⇒ *'a rel* ⇒ *'a rel* ⇒ *bool*
where
$LD2\ r\ r0\ r1 \equiv (\ r = r0 \cup r1$
$\wedge (\forall\ a\ b\ c.\ (a,b) \in r0 \wedge (a,c) \in r0 \longrightarrow jn00\ r0\ b\ c)$
$\wedge (\forall\ a\ b\ c.\ (a,b) \in r0 \wedge (a,c) \in r1 \longrightarrow jn01\ r0\ r1\ b\ c)$
$\wedge (\forall\ a\ b\ c.\ (a,b) \in r1 \wedge (a,c) \in r1 \longrightarrow jn11\ r0\ r1\ b\ c)\ )$

definition *LD3* :: *'a rel* ⇒ *'a rel* ⇒ *'a rel* ⇒ *'a rel* ⇒ *bool*
where
$LD3\ r\ r0\ r1\ r2 \equiv (\ r = r0 \cup r1 \cup r2$
$\wedge (\forall\ a\ b\ c.\ (a,b) \in r0 \wedge (a,c) \in r0 \longrightarrow jn00\ r0\ b\ c)$
$\wedge (\forall\ a\ b\ c.\ (a,b) \in r0 \wedge (a,c) \in r1 \longrightarrow jn01\ r0\ r1\ b\ c)$
$\wedge (\forall\ a\ b\ c.\ (a,b) \in r1 \wedge (a,c) \in r1 \longrightarrow jn11\ r0\ r1\ b\ c)$
$\wedge (\forall\ a\ b\ c.\ (a,b) \in r0 \wedge (a,c) \in r2 \longrightarrow jn02\ r0\ r1\ r2\ b\ c)$
$\wedge (\forall\ a\ b\ c.\ (a,b) \in r1 \wedge (a,c) \in r2 \longrightarrow jn12\ r0\ r1\ r2\ b\ c)$
$\wedge (\forall\ a\ b\ c.\ (a,b) \in r2 \wedge (a,c) \in r2 \longrightarrow jn22\ r0\ r1\ r2\ b\ c))$

definition *DCR2* :: *'a rel* ⇒ *bool*
where
$DCR2\ r \equiv (\ \exists r0\ r1.\ LD2\ r\ r0\ r1\ )$

definition *DCR3* :: *'a rel* ⇒ *bool*
where
$DCR3\ r \equiv (\ \exists r0\ r1\ r2.\ LD3\ r\ r0\ r1\ r2\ )$

definition $\mathfrak{L}1$ :: (*nat* ⇒ *'U rel*) ⇒ *nat* ⇒ *'U rel*
where
$\mathfrak{L}1\ g\ \alpha \equiv \bigcup\ \{A.\ \exists\ \alpha'.\ (\alpha' < \alpha) \wedge A = g\ \alpha'\}$

definition $\mathfrak{L}v$ :: (*nat* ⇒ *'U rel*) ⇒ *nat* ⇒ *nat* ⇒ *'U rel*
where
$\mathfrak{L}v\ g\ \alpha\ \beta \equiv \bigcup\ \{A.\ \exists\ \alpha'.\ (\alpha' < \alpha \vee \alpha' < \beta) \wedge A = g\ \alpha'\}$

definition $\mathfrak{D}$ :: (*nat* ⇒ *'U rel*) ⇒ *nat* ⇒ *nat* ⇒ (*'U* × *'U* × *'U* × *'U*) *set*
where
$\mathfrak{D}\ g\ \alpha\ \beta = \{(b,b',b'',d).\ (b,b') \in (\mathfrak{L}1\ g\ \alpha)\hat{}* \wedge (b',b'') \in (g\ \beta)\hat{}= \wedge (b'',d) \in (\mathfrak{L}v\ g\ \alpha\ \beta)\hat{}*\}$

definition *DCR-generating* :: (*nat* ⇒ *'U rel*) ⇒ *bool*
where
$DCR\text{-}generating\ g \equiv (\forall\ \alpha\ \beta\ a\ b\ c.\ (a,b) \in (g\ \alpha) \wedge (a,c) \in (g\ \beta)$
$\longrightarrow\ (\exists\ b'\ b''\ c'\ c''\ d.\ (b,b',b'',d) \in (\mathfrak{D}\ g\ \alpha\ \beta) \wedge (c,c',c'',d) \in (\mathfrak{D}\ g\ \beta\ \alpha)\ ))$

1.1.2 Result

The next definition formalizes the condition "an ARS with a reduction relation r belongs to the class DCR_n", where n is a natural number.

definition *DCR* :: *nat* ⇒ *'U rel* ⇒ *bool*
where
DCR n r ≡ (∃ *g*::(*nat* ⇒ *'U rel*). *DCR-generating g* ∧ *r* = ⋃ { *r'*. ∃ *α'*. *α'* < *n* ∧ *r'* = *g α'* })

end

1.2 Completeness of the DCR3 method for proving confluence of relations of the least uncountable cardinality

theory *DCR3-Method*
imports
HOL−Cardinals.Cardinals
Abstract−Rewriting.Abstract-Rewriting
Finite-DCR-Hierarchy
begin

1.2.1 Auxiliary definitions

abbreviation *ω-ord* **where** *ω-ord* ≡ *natLeq*

definition *sc-ord*::*'U rel* ⇒ *'U rel* ⇒ *bool*
where *sc-ord α α'* ≡ (*α* <*o α'* ∧ (∀ *β*::*'U rel*. *α* <*o β* ⟶ *α'* ≤*o β*))

definition *lm-ord*::*'U rel* ⇒ *bool*
where *lm-ord α* ≡ *Well-order α* ∧ ¬ (*α* = {} ∨ *isSuccOrd α*)

definition *nord* :: *'U rel* ⇒ *'U rel* **where** *nord α* = (*SOME α'*::*'U rel*. *α'* =*o α*)

definition 𝒪::*'U rel set* **where** 𝒪 ≡ *nord* ‘ {*α*. *Well-order α*}

definition *oord*::*'U rel rel* **where** *oord* ≡ (*Restr ordLeq* 𝒪)

definition *CCR* :: *'U rel* ⇒ *bool*
where
CCR r = (∀ *a* ∈ *Field r*. ∀ *b* ∈ *Field r*. ∃ *c* ∈ *Field r*. (*a*,*c*) ∈ *r*^* ∧ (*b*,*c*) ∈ *r*^*)

definition *Conelike* :: *'U rel* ⇒ *bool*
where
Conelike r = (*r* = {} ∨ (∃ *m* ∈ *Field r*. ∀ *a* ∈ *Field r*. (*a*,*m*) ∈ *r*^*))

definition *dncl* :: *'U rel* ⇒ *'U set* ⇒ *'U set*
where
dncl r A = ((*r*^*)^−*1*)‘‘*A*

definition *Inv* :: $'U$ *rel* ⇒ $'U$ *set set*
where
Inv r = { *A* :: $'U$ *set* . *r* ‘‘ *A* ⊆ *A* }

definition *SF* :: $'U$ *rel* ⇒ $'U$ *set set*
where
SF r = { *A* :: $'U$ *set*. *Field* (*Restr r A*) = *A* }

definition *SCF*::$'U$ *rel* ⇒ ($'U$ *set*) *set* **where**
SCF r ≡ { *B*::($'U$ *set*) . *B* ⊆ *Field r* ∧ (∀ *a* ∈ *Field r*. ∃ *b* ∈ *B*. (*a*,*b*) ∈ *r*^*) }

definition *cfseq* :: $'U$ *rel* ⇒ (*nat* ⇒ $'U$) ⇒ *bool*
where
cfseq r xi ≡ ((∀ *a* ∈ *Field r*. ∃ *i*. (*a*, *xi i*) ∈ *r*^*) ∧ (∀ *i*. (*xi i*, *xi* (*Suc i*)) ∈ *r*))

definition *rpth* :: $'U$ *rel* ⇒ $'U$ ⇒ $'U$ ⇒ *nat* ⇒ (*nat* ⇒ $'U$) *set*
where
rpth r a b n ≡ { *f*::(*nat* ⇒ $'U$). *f 0* = *a* ∧ *f n* = *b* ∧ (∀ *i*<*n*. (*f i*, *f*(*Suc i*)) ∈ *r*)
}

definition $\mathcal{F}$ :: $'U$ *rel* ⇒ $'U$ ⇒ $'U$ ⇒ $'U$ *set set*
where
$\mathcal{F}$ *r a b* ≡ { *F*::$'U$ *set*. ∃ *n*::*nat*. ∃ *f* ∈ *rpth r a b n*. *F* = *f*‘{*i*. *i*≤*n*} }

definition 𝔣 :: $'U$ *rel* ⇒ $'U$ ⇒ $'U$ ⇒ $'U$ *set*
where
𝔣 *r a b* ≡ (*if* ($\mathcal{F}$ *r a b* ≠ {}) *then* (*SOME F*. *F* ∈ $\mathcal{F}$ *r a b*) *else* {})

definition *dnEsc* :: $'U$ *rel* ⇒ $'U$ *set* ⇒ $'U$ ⇒ $'U$ *set set*
where
dnEsc r A a ≡ { *F*. ∃ *b*. ((*b* ∉ *dncl r A*) ∧ (*F* ∈ $\mathcal{F}$ *r a b*) ∧ (*F* ∩ *A* = {})) }

definition *dnesc* :: $'U$ *rel* ⇒ $'U$ *set* ⇒ $'U$ ⇒ $'U$ *set*
where
dnesc r A a = (*if* (*dnEsc r A a* ≠ {}) *then* (*SOME F*. *F* ∈ *dnEsc r A a*) *else* {
a })

definition *escl* :: $'U$ *rel* ⇒ $'U$ *set* ⇒ $'U$ *set* ⇒ $'U$ *set*
where
escl r A B = ⋃ ((*dnesc r A*) ‘ *B*)

definition *clterm* **where** *clterm s' r* ≡ (*Conelike s'* ⟶ *Conelike r*)

definition *spthlen*::$'U$ *rel* ⇒ $'U$ ⇒ $'U$ ⇒ *nat*
where
spthlen r a b ≡ (*LEAST n*::*nat*. (*a*,*b*) ∈ *r*^^*n*)

definition *spth* :: $'U$ *rel* ⇒ $'U$ ⇒ $'U$ ⇒ (*nat* ⇒ $'U$) *set*
where

$spth\ r\ a\ b = rpth\ r\ a\ b\ (spthlen\ r\ a\ b)$

definition $\mathfrak{U}$::$'U\ rel \Rightarrow ('U\ rel)\ set$ **where**
$\mathfrak{U}\ r \equiv \{\ s::('U\ rel)\ .\ CCR\ s \wedge s \subseteq r \wedge (\forall\ a \in Field\ r.\ \exists\ b \in Field\ s.\ (a,b) \in r\hat{\ }*)\ \}$

definition $RCC\text{-}rel$ :: $'U\ rel \Rightarrow 'U\ rel \Rightarrow bool$ **where**
$RCC\text{-}rel\ r\ \alpha \equiv (\mathfrak{U}\ r = \{\} \wedge \alpha = \{\}) \vee (\exists\ s \in \mathfrak{U}\ r.\ |s| =o\ \alpha \wedge (\ \forall\ s' \in \mathfrak{U}\ r.\ |s| \leq o\ |s'|\ ))$

definition RCC :: $'U\ rel \Rightarrow 'U\ rel\ (\|\text{-}\|)$
where $\|r\| \equiv (SOME\ \alpha.\ RCC\text{-}rel\ r\ \alpha)$

definition Den::$'U\ rel \Rightarrow ('U\ set)\ set$ **where**
$Den\ r \equiv \{\ B::('U\ set)\ .\ B \subseteq Field\ r \wedge (\forall\ a \in Field\ r.\ \exists\ b \in B.\ (a,b) \in r\hat{\ }=)\ \}$

definition $Span$::$'U\ rel \Rightarrow ('U\ rel)\ set$ **where**
$Span\ r \equiv \{\ s.\ s \subseteq r \wedge Field\ s = Field\ r\ \}$

definition $scf\text{-}rel$ :: $'U\ rel \Rightarrow 'U\ rel \Rightarrow bool$ **where**
$scf\text{-}rel\ r\ \alpha \equiv (\exists\ B \in SCF\ r.\ |B| =o\ \alpha \wedge (\ \forall\ B' \in SCF\ r.\ |B| \leq o\ |B'|\ ))$

definition scf :: $'U\ rel \Rightarrow 'U\ rel$
where $scf\ r \equiv (SOME\ \alpha.\ scf\text{-}rel\ r\ \alpha)$

definition $w\text{-}dncl$ :: $'U\ rel \Rightarrow 'U\ set \Rightarrow 'U\ set$
where
$w\text{-}dncl\ r\ A = \{\ a \in dncl\ r\ A.\ \forall\ b.\ \forall\ F \in \mathcal{F}\ r\ a\ b.\ (\ b \notin dncl\ r\ A \longrightarrow F \cap A \neq \{\}\ )\ \}$

definition $\mathfrak{L}$ :: $('U\ rel \Rightarrow 'U\ set) \Rightarrow 'U\ rel \Rightarrow 'U\ set$
where
$\mathfrak{L}\ f\ \alpha \equiv \bigcup\ \{A.\ \exists\ \alpha'.\ \alpha' <o\ \alpha \wedge A = f\ \alpha'\}$

definition Dbk :: $('U\ rel \Rightarrow 'U\ set) \Rightarrow 'U\ rel \Rightarrow 'U\ set\ (\ \nabla\ \text{-}\ \text{-}\ )$
where
$\nabla\ f\ \alpha \equiv f\ \alpha - (\mathfrak{L}\ f\ \alpha)$

definition $\mathcal{Q}$ :: $'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set) \Rightarrow 'U\ rel \Rightarrow 'U\ set$
where
$\mathcal{Q}\ r\ f\ \alpha \equiv (f\ \alpha - (dncl\ r\ (\mathfrak{L}\ f\ \alpha)))$

definition $\mathcal{W}$ :: $'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set) \Rightarrow 'U\ rel \Rightarrow 'U\ set$
where
$\mathcal{W}\ r\ f\ \alpha \equiv (f\ \alpha - (w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha)))$

definition $\mathcal{N}1$ :: $'U\ rel \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}1\ r\ \alpha 0 \equiv \{\ f\ .\ \forall \alpha\ \alpha'.\ (\ \alpha \leq o\ \alpha 0 \wedge \alpha' \leq o\ \alpha\ ) \longrightarrow (f\ \alpha') \subseteq (f\ \alpha)\ \}$

definition $\mathcal{N}2$:: $'U\ rel \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}2\ r\ \alpha 0 \equiv \{\ f\ .\ \forall \alpha.\ (\ \alpha \leq o\ \alpha 0 \wedge \neg\ (\alpha = \{\} \vee isSuccOrd\ \alpha)\ ) \longrightarrow (\nabla\ f\ \alpha) = \{\}\ \}$

definition $\mathcal{N}3$:: $'U\ rel \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}3\ r\ \alpha 0 \equiv \{\ f\ .\ \forall \alpha.\ (\ \alpha \leq o\ \alpha 0 \wedge (\alpha = \{\} \vee isSuccOrd\ \alpha)\ ) \longrightarrow$
$(\ \omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha| \longrightarrow ((escl\ r\ (\mathfrak{L}\ f\ \alpha)\ (f\ \alpha) \subseteq (f\ \alpha)) \wedge (clterm\ (Restr\ r\ (f\ \alpha))\ r))\ )\ \}$

definition $\mathcal{N}4$:: $'U\ rel \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}4\ r\ \alpha 0 \equiv \{\ f\ .\ \forall \alpha.\ (\ \alpha \leq o\ \alpha 0 \wedge (\alpha = \{\} \vee isSuccOrd\ \alpha)\ ) \longrightarrow$
$(\ \forall\ a \in (\mathfrak{L}\ f\ \alpha).\ (\ r``\{a\} \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha)\ ) \vee (\ r``\{a\} \cap (\mathcal{W}\ r\ f\ \alpha) \neq \{\}\ )\ )\ \}$

definition $\mathcal{N}5$ :: $'U\ rel \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}5\ r\ \alpha 0 \equiv \{\ f\ .\ \forall \alpha.\ \alpha \leq o\ \alpha 0 \longrightarrow (f\ \alpha) \in SF\ r\ \}$

definition $\mathcal{N}6$ :: $'U\ rel \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}6\ r\ \alpha 0 \equiv \{\ f.\ \forall \alpha.\ \alpha \leq o\ \alpha 0 \longrightarrow CCR\ (Restr\ r\ (f\ \alpha))\ \}$

definition $\mathcal{N}7$ :: $'U\ rel \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}7\ r\ \alpha 0 \equiv \{\ f.\ \forall \alpha.\ \alpha \leq o\ \alpha 0 \longrightarrow (\ \alpha <o\ \omega\text{-}ord \longrightarrow |f\ \alpha| <o\ \omega\text{-}ord\ ) \wedge (\omega\text{-}ord \leq o\ \alpha \longrightarrow |f\ \alpha| \leq o\ \alpha)\ \}$

definition $\mathcal{N}8$ :: $'U\ rel \Rightarrow 'U\ set\ set \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}8\ r\ Ps\ \alpha 0 \equiv \{\ f.\ \forall \alpha.\ \alpha \leq o\ \alpha 0 \wedge (\alpha = \{\} \vee isSuccOrd\ \alpha) \wedge (\ (\exists\ P.\ Ps = \{P\}) \vee (\neg\ finite\ Ps \wedge |Ps| \leq o\ |f\ \alpha|\ )) \longrightarrow$
$(\forall\ P \in Ps.\ ((f\ \alpha) \cap P) \in SCF\ (Restr\ r\ (f\ \alpha)))\ \}$

definition $\mathcal{N}9$ :: $'U\ rel \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}9\ r\ \alpha 0 \equiv \{\ f\ .\ \omega\text{-}ord \leq o\ \alpha 0 \longrightarrow Field\ r \subseteq (f\ \alpha 0)\ \}$

definition $\mathcal{N}10$ :: $'U\ rel \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}10\ r\ \alpha 0 \equiv \{\ f\ .\ \forall \alpha.\ \alpha \leq o\ \alpha 0 \longrightarrow ((\exists\ y::'U.\ \mathcal{Q}\ r\ f\ \alpha = \{y\}) \longrightarrow (Field\ r \subseteq dncl\ r\ (f\ \alpha)))\ \}$

definition $\mathcal{N}11$:: $'U\ rel \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}11\ r\ \alpha 0 \equiv \{\ f\ .\ \forall \alpha.\ (\ \alpha \leq o\ \alpha 0 \wedge isSuccOrd\ \alpha) \longrightarrow \mathcal{Q}\ r\ f\ \alpha = \{\} \longrightarrow (Field$

$r \subseteq dncl\ r\ (f\ \alpha))\ \}$

definition $\mathcal{N}12$:: $'U\ rel \Rightarrow 'U\ rel \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}12\ r\ \alpha 0 \equiv \{\ f\ .\ \forall \alpha.\ \alpha \leq o\ \alpha 0 \longrightarrow \omega\text{-}ord \leq o\ \alpha \longrightarrow \omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha|\ \}$

definition $\mathcal{N}$ :: $'U\ rel \Rightarrow 'U\ set\ set \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{N}\ r\ Ps \equiv \{\ f \in (\mathcal{N}1\ r\ |Field\ r|\ ) \cap (\mathcal{N}2\ r\ |Field\ r|\ ) \cap (\mathcal{N}3\ r\ |Field\ r|\ ) \cap (\mathcal{N}4$
$r\ |Field\ r|\ )$
$\cap\ (\mathcal{N}5\ r\ |Field\ r|\ ) \cap (\mathcal{N}6\ r\ |Field\ r|\ ) \cap (\mathcal{N}7\ r\ |Field\ r|\ ) \cap (\mathcal{N}8\ r\ Ps$
$|Field\ r|\ )$
$\cap\ (\mathcal{N}9\ r\ |Field\ r| \cap \mathcal{N}10\ r\ |Field\ r| \cap \mathcal{N}11\ r\ |Field\ r| \cap \mathcal{N}12\ r\ |Field\ r|\ ).$
$(\forall\ \ \alpha\ \beta.\ \alpha = o\ \beta \longrightarrow f\ \alpha = f\ \beta)\ \}$

definition $\mathcal{T}$ :: $('U\ rel \Rightarrow 'U\ set \Rightarrow 'U\ set) \Rightarrow ('U\ rel \Rightarrow 'U\ set)\ set$
where
$\mathcal{T}\ F \equiv \{\ f::'U\ rel \Rightarrow 'U\ set\ .$
$f\ \{\} = \{\}$
$\wedge\ (\forall\ \ \alpha 0\ \alpha::'U\ rel.\ (sc\text{-}ord\ \alpha 0\ \alpha \longrightarrow f\ \alpha = F\ \alpha 0\ (f\ \alpha 0)))$
$\wedge\ (\forall\ \ \alpha.\ (lm\text{-}ord\ \alpha \longrightarrow f\ \alpha = \bigcup\ \{\ D.\ \exists\ \ \beta.\ \beta < o\ \alpha \wedge D = f\ \beta\ \}))$
$\wedge\ (\forall \alpha\ \beta.\ \alpha = o\ \beta \longrightarrow f\ \alpha = f\ \beta)\ \}$

definition $\mathcal{E}p$ **where** $\mathcal{E}p\ r\ Ps\ A\ A' \equiv$
$(((\exists\ \ P.\ Ps = \{P\}) \vee ((\neg\ finite\ Ps) \wedge |Ps| \leq o\ |A|\ ))$
$\longrightarrow (\forall\ \ P \in Ps.\ (A' \cap P) \in SCF\ (Restr\ r\ A')\ ))$

definition $\mathcal{E}$ :: $'U\ rel \Rightarrow 'U \Rightarrow 'U\ set \Rightarrow 'U\ set\ set \Rightarrow 'U\ set\ set$
where
$\mathcal{E}\ r\ a\ A\ Ps \equiv \{\ A'.$
$(a \in Field\ r \longrightarrow a \in A') \wedge A \subseteq A'$
$\wedge\ (\ |A| < o\ \omega\text{-}ord \longrightarrow |A'| < o\ \omega\text{-}ord\ ) \wedge (\ \omega\text{-}ord \leq o\ |A| \longrightarrow |A'| \leq o\ |A|\ )$
$\wedge\ (A \in SF\ r \longrightarrow ($
$A' \in SF\ r$
$\wedge\ CCR\ (Restr\ r\ A')$
$\wedge\ (\ \forall\ \ a{\in}A.\ (r``\{a\} \subseteq w\text{-}dncl\ r\ A) \vee (r``\{a\} \cap (A'{-}w\text{-}dncl\ r\ A) \neq \{\})$
$)$
$\wedge\ ((\exists\ \ y.\ A' - dncl\ r\ A \subseteq \{y\}) \longrightarrow (Field\ r \subseteq (dncl\ r\ A')))$
$\wedge\ \mathcal{E}p\ r\ Ps\ A\ A'$
$\wedge\ (\ \omega\text{-}ord \leq o\ |A| \longrightarrow escl\ r\ A\ A' \subseteq A' \wedge clterm\ (Restr\ r\ A')\ r))\ )\ \}$

definition $wbase$::$'U\ rel \Rightarrow 'U\ set \Rightarrow ('U\ set)\ set$ **where**
$wbase\ r\ A \equiv \{\ B::'U\ set.\ A \subseteq w\text{-}dncl\ r\ B\ \}$

definition $wrank\text{-}rel$ :: $'U\ rel \Rightarrow 'U\ set \Rightarrow 'U\ rel \Rightarrow bool$ **where**
$wrank\text{-}rel\ r\ A\ \alpha \equiv (\exists\ \ B \in wbase\ r\ A.\ |B| = o\ \alpha \wedge (\ \forall\ \ B' \in wbase\ r\ A.\ |B| \leq o$
$|B'|\ ))$

definition $wrank$ :: $'U\ rel \Rightarrow 'U\ set \Rightarrow 'U\ rel$

where $\mathit{wrank}\ r\ A \equiv (\mathit{SOME}\ \alpha.\ \mathit{wrank\text{-}rel}\ r\ A\ \alpha)$

definition $\mathit{Mwn} :: {}'U\ \mathit{rel} \Rightarrow {}'U\ \mathit{rel} \Rightarrow {}'U\ \mathit{set}$
where
$\mathit{Mwn}\ r\ \alpha = \{\ a \in \mathit{Field}\ r.\ \alpha <o\ \mathit{wrank}\ r\ (r\ ``\{a\})\ \}$

definition $\mathit{Mwnm} :: {}'U\ \mathit{rel} \Rightarrow {}'U\ \mathit{set}$
where
$\mathit{Mwnm}\ r = \{\ a \in \mathit{Field}\ r.\ \|r\| \le o\ \mathit{wrank}\ r\ (r\ ``\{a\})\ \}$

definition $\mathit{wesc\text{-}rel} :: {}'U\ \mathit{rel} \Rightarrow ({}'U\ \mathit{rel} \Rightarrow {}'U\ \mathit{set}) \Rightarrow {}'U\ \mathit{rel} \Rightarrow {}'U \Rightarrow {}'U \Rightarrow \mathit{bool}$
where
$\mathit{wesc\text{-}rel}\ r\ f\ \alpha\ a\ b \equiv (\ b \in \mathcal{W}\ r\ f\ \alpha \wedge (a,b) \in (\mathit{Restr}\ r\ (\mathcal{W}\ r\ f\ \alpha))\hat{\ }*$
$\wedge\ (\forall \beta.\ \alpha <o\ \beta \wedge \beta <o\ |\mathit{Field}\ r| \wedge (\beta = \{\} \vee \mathit{isSuccOrd}\ \beta) \longrightarrow (r``\{b\} \cap (\mathcal{W}$
$r\ f\ \beta) \neq \{\}))\ )$

definition $\mathit{wesc} :: {}'U\ \mathit{rel} \Rightarrow ({}'U\ \mathit{rel} \Rightarrow {}'U\ \mathit{set}) \Rightarrow {}'U\ \mathit{rel} \Rightarrow {}'U \Rightarrow {}'U$
where
$\mathit{wesc}\ r\ f\ \alpha\ a \equiv (\mathit{SOME}\ b.\ \mathit{wesc\text{-}rel}\ r\ f\ \alpha\ a\ b)$

definition $\mathit{cardLeN1}::{}'a\ \mathit{set} \Rightarrow \mathit{bool}$
where
$\mathit{cardLeN1}\ A \equiv (\forall\ B \subseteq A.$
$(\ \forall\ C \subseteq B\ .\ ((\exists\ D\ f.\ D \subset C \wedge C \subseteq f`D\ ) \longrightarrow (\ \exists\ f.\ B \subseteq f`C\ ))\ )$
$\vee\ (\ \exists\ g\ .\ A \subseteq g`B\ )\ )$

1.2.2 Auxiliary lemmas

lemma *lem-Ldo-ldogen-ord*:
assumes $\forall \alpha\ \beta\ a\ b\ c.\ \alpha \le \beta \longrightarrow (a,\ b) \in g\ \alpha \wedge (a,\ c) \in g\ \beta \longrightarrow$
$(\exists b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ \beta\ \alpha)$
shows *DCR-generating g*
using *assms* **unfolding** *DCR-generating-def* **by** (*meson linear*)

lemma *lem-rtr-field*: $(x,y) \in r\hat{\ }* \Longrightarrow (x = y) \vee (x \in \mathit{Field}\ r \wedge y \in \mathit{Field}\ r)$
by (*metis Field-def Not-Domain-rtrancl Range.RangeI UnCI rtranclE*)

lemma *lem-fin-fl-rel*: *finite* (*Field r*) = *finite r*
using *finite-Field finite-subset trancl-subset-Field2* **by** *fastforce*

lemma *lem-Relprop-fld-sat*:
fixes $r\ s::{}'U\ \mathit{rel}$
assumes *a1*: $s \subseteq r$ **and** *a2*: $s' = \mathit{Restr}\ r\ (\mathit{Field}\ s)$
shows $s \subseteq s' \wedge \mathit{Field}\ s' = \mathit{Field}\ s$
proof −
have $s \subseteq (\mathit{Field}\ s) \times (\mathit{Field}\ s)$ **unfolding** *Field-def* **by** *force*
then have $s \subseteq s'$ **using** *a1 a2* **by** *blast*
moreover then have $\mathit{Field}\ s \subseteq \mathit{Field}\ s'$ **unfolding** *Field-def* **by** *blast*
moreover have $\mathit{Field}\ s' \subseteq \mathit{Field}\ s$ **using** *a2* **unfolding** *Field-def* **by** *blast*

ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Relprop-sat-un*:
fixes $r{::}'U\ rel$ **and** $S{::}'U\ set\ set$ **and** $A'{::}'U\ set$
assumes *a1*: $\forall A{\in}S.\ Field\ (Restr\ r\ A) = A$ **and** *a2*: $A' = \bigcup\ S$
shows $Field\ (Restr\ r\ A') = A'$
proof
show $Field\ (Restr\ r\ A') \subseteq A'$ **unfolding** *Field-def* **by** *blast*
next
show $A' \subseteq Field\ (Restr\ r\ A')$
proof
fix x
assume $x \in A'$
then obtain A **where** $A \in S \wedge x \in A$ **using** *a2* **by** *blast*
then have $x \in Field\ (Restr\ r\ A) \wedge A \subseteq A'$ **using** *a1 a2* **by** *blast*
moreover then have $Field\ (Restr\ r\ A) \subseteq Field\ (Restr\ r\ A')$ **unfolding** *Field-def* **by** *blast*
ultimately show $x \in Field\ (Restr\ r\ A')$ **by** *blast*
qed
qed

lemma *lem-nord-r*: $Well\text{-}order\ \alpha \Longrightarrow nord\ \alpha =o\ \alpha$ **unfolding** *nord-def* **by** (*meson ordIso-reflexive someI-ex*)

lemma *lem-nord-l*: $Well\text{-}order\ \alpha \Longrightarrow \alpha =o\ nord\ \alpha$ **unfolding** *nord-def* **by** (*meson ordIso-reflexive ordIso-symmetric someI-ex*)

lemma *lem-nord-eq*: $\alpha =o\ \beta \Longrightarrow nord\ \alpha = nord\ \beta$ **unfolding** *nord-def* **using** *ordIso-symmetric ordIso-transitive* **by** *metis*

lemma *lem-nord-req*: $Well\text{-}order\ \alpha \Longrightarrow Well\text{-}order\ \beta \Longrightarrow nord\ \alpha = nord\ \beta \Longrightarrow \alpha =o\ \beta$
using *lem-nord-l lem-nord-r ordIso-transitive* **by** *metis*

lemma *lem-Onord*: $\alpha \in \mathcal{O} \Longrightarrow \alpha = nord\ \alpha$ **unfolding** *$\mathcal{O}$-def* **using** *lem-nord-r lem-nord-eq* **by** *blast*

lemma *lem-Oeq*: $\alpha \in \mathcal{O} \Longrightarrow \beta \in \mathcal{O} \Longrightarrow \alpha =o\ \beta \Longrightarrow \alpha = \beta$ **using** *lem-Onord lem-nord-eq* **by** *metis*

lemma *lem-Owo*: $\alpha \in \mathcal{O} \Longrightarrow Well\text{-}order\ \alpha$ **unfolding** *$\mathcal{O}$-def* **using** *lem-nord-r ordIso-Well-order-simp* **by** *blast*

lemma *lem-fld-oord*: $Field\ oord = \mathcal{O}$ **using** *lem-Owo ordLeq-reflexive* **unfolding** *oord-def Field-def* **by** *blast*

lemma *lem-nord-less*: $\alpha <o\ \beta \Longrightarrow nord\ \beta \neq nord\ \alpha \wedge (nord\ \alpha,\ nord\ \beta) \in oord$
proof –

assume *b1*: $\alpha <o\ \beta$
then have *nord* $\alpha \in \mathcal{O} \wedge$ *nord* $\beta \in \mathcal{O} \wedge$ *nord* $\alpha =o\ \alpha \wedge$ *nord* $\beta =o\ \beta$
using *lem-nord-r ordLess-Well-order-simp* **unfolding** *$\mathcal{O}$-def* **by** *blast*
moreover have $\forall\ r\ A\ a\ b.\ (a,b) \in Restr\ r\ A = (a \in A \wedge b \in A \wedge (a,b) \in r)$
unfolding *Field-def* **by** *force*
ultimately show *nord* $\beta \neq$ *nord* $\alpha \wedge$(*nord* α, *nord* β) $\in$ *oord* **using** *b1* **unfolding** *oord-def*
by (*metis not-ordLess-ordIso ordIso-iff-ordLeq ordLeq-iff-ordLess-or-ordIso ordLeq-transitive*)
qed

lemma *lem-nord-ls*: $\alpha <o\ \beta \Longrightarrow$ *nord* $\alpha <o$ *nord* β
proof –
assume *a1*: $\alpha <o\ \beta$
then have *Well-order* $\alpha \wedge$ *Well-order* β **unfolding** *ordLess-def* **by** *blast*
then have *nord* $\alpha =o\ \alpha$ **and** *nord* $\beta =o\ \beta$ **using** *lem-nord-r* **by** *blast+*
then show *nord* $\alpha <o$ *nord* β **using** *a1*
using *ordIso-iff-ordLeq ordIso-ordLess-trans ordLess-ordLeq-trans* **by** *blast*
qed

lemma *lem-nord-le*: $\alpha \leq o\ \beta \Longrightarrow$ *nord* $\alpha \leq o$ *nord* β
proof –
assume *a1*: $\alpha \leq o\ \beta$
then have *Well-order* $\alpha \wedge$ *Well-order* β **unfolding** *ordLeq-def* **by** *blast*
then have *nord* $\alpha =o\ \alpha$ **and** *nord* $\beta =o\ \beta$ **using** *lem-nord-r* **by** *blast+*
then show *nord* $\alpha \leq o$ *nord* β **using** *a1* **by** (*meson ordIso-iff-ordLeq ordLeq-transitive*)
qed

lemma *lem-nordO-ls-l*: $\alpha <o\ \beta \Longrightarrow$ *nord* $\alpha \in \mathcal{O}$ **using** *$\mathcal{O}$-def ordLess-Well-order-simp* **by** *blast*

lemma *lem-nordO-ls-r*: $\alpha <o\ \beta \Longrightarrow$ *nord* $\beta \in \mathcal{O}$ **using** *$\mathcal{O}$-def ordLess-Well-order-simp* **by** *blast*

lemma *lem-nordO-le-l*: $\alpha \leq o\ \beta \Longrightarrow$ *nord* $\alpha \in \mathcal{O}$ **using** *$\mathcal{O}$-def ordLeq-Well-order-simp* **by** *blast*

lemma *lem-nordO-le-r*: $\alpha \leq o\ \beta \Longrightarrow$ *nord* $\beta \in \mathcal{O}$ **using** *$\mathcal{O}$-def ordLeq-Well-order-simp* **by** *blast*

lemma *lem-nord-ls-r*: $\alpha <o\ \beta \Longrightarrow \alpha <o$ *nord* β
using *lem-nord-ls*[*of* α β] *lem-nord-r*[*of* β] *lem-nord-l* **by** (*metis ordLess-ordIso-trans ordLess-Well-order-simp*)

lemma *lem-nord-ls-l*: $\alpha <o\ \beta \Longrightarrow$ *nord* $\alpha <o\ \beta$
using *lem-nord-ls*[*of* α β] *lem-nord-r*[*of* β] **by** (*metis ordLess-ordIso-trans ordLess-Well-order-simp*)

lemma *lem-nord-le-r*: $\alpha \leq o\ \beta \Longrightarrow \alpha \leq o$ *nord* β

using *lem-nord-le*[*of* α β] *lem-nord-r*[*of* β] *lem-nord-l* **by** (*metis ordLeq-ordIso-trans ordLeq-Well-order-simp*)

lemma *lem-nord-le-l*: $\alpha \le o\ \beta \Longrightarrow nord\ \alpha \le o\ \beta$
using *lem-nord-le*[*of* α β] *lem-nord-r*[*of* β] **by** (*metis ordLeq-ordIso-trans ordLeq-Well-order-simp*)

lemma *lem-oord-wo*: *Well-order oord*
proof –
let *?oleqO* = *Restr ordLeq* $\mathcal{O}$
have *Well-order ?oleqO*
proof –
have *c1*: *Field ordLeq* = {α::′*U rel*. *Well-order* α}
using *ordLeq-Well-order-simp ordLeq-reflexive* **unfolding** *Field-def* **by** *blast*
then have *Refl ordLeq* **using** *ordLeq-refl-on* **by** *metis*
then have *Preorder ordLeq* **using** *ordLeq-trans* **unfolding** *preorder-on-def*
using *subset-antisym c1* **by** *auto*
then have *Preorder ?oleqO* **using** *Preorder-Restr* **by** *blast*
moreover have ∀ α β::′*U rel*. (α, β) ∈ *?oleqO* ⟶ (β, α) ∈ *?oleqO* ⟶ α = β
proof (*intro allI impI*)
fix α β::′*U rel*
assume *d1*: (α, β) ∈ *?oleqO* **and** *d2*: (β, α) ∈ *?oleqO*
then have $\alpha \le o\ \beta \wedge \beta \le o\ \alpha$ **by** *blast*
then have $\alpha =o\ \beta$ **using** *ordIso-iff-ordLeq* **by** *blast*
moreover have $\alpha \in \mathcal{O} \wedge \beta \in \mathcal{O}$ **using** *d1* **by** *blast*
ultimately show α = β **using** *lem-Oeq* **by** *blast*
qed
moreover have ∀ α ∈ *Field* (*?oleqO*::′*U rel rel*). ∀ β ∈ *Field ?oleqO*. α ≠ β ⟶
(α, β) ∈ *?oleqO* ∨ (β, α) ∈ *?oleqO*
proof (*intro ballI impI*)
fix α β::′*U rel*
assume *d1*: α ∈ *Field ?oleqO* **and** *d2*: β ∈ *Field ?oleqO* **and** α ≠ β
then have *Well-order* α ∧ *Well-order* β **using** *c1* **unfolding** *Field-def*
by (*metis* (*no-types*, *lifting*) *Field-Un Field-def Un-def mem-Collect-eq sup-inf-absorb*)
then have $\alpha \le o\ \beta \vee \beta \le o\ \alpha$ **using** *ordLess-imp-ordLeq ordLess-or-ordLeq* **by** *blast*
moreover have $\alpha \in \mathcal{O} \wedge \beta \in \mathcal{O}$ **using** *d1 d2* **unfolding** *Field-def* **by** *blast*
ultimately show (α, β) ∈ *?oleqO* ∨ (β, α) ∈ *?oleqO* **by** *blast*
qed
ultimately have *Linear-order ?oleqO* **unfolding** *linear-order-on-def partial-order-on-def total-on-def antisym-def preorder-on-def* **by** *blast*
moreover have *wf* ((*?oleqO*::′*U rel rel*) − *Id*)
proof –
have *Restr* (*ordLess*::′*U rel rel*) $\mathcal{O}$ ⊆ *?oleqO* − *Id*
using *not-ordLeq-ordLess ordLeq-iff-ordLess-or-ordIso* **by** *blast*
moreover have (*?oleqO*::′*U rel rel*) − *Id* ⊆ *Restr ordLess* $\mathcal{O}$
using *lem-Oeq ordLeq-iff-ordLess-or-ordIso* **by** *blast*

ultimately have (*?oleqO*::*'U rel rel*) − *Id* = *Restr ordLess* $\mathcal{O}$ **by** *blast*
moreover have *wf* (*Restr ordLess* $\mathcal{O}$)
using *wf-ordLess Restr-subset wf-subset*[*of ordLess Restr ordLess* $\mathcal{O}$] **by** *blast*
ultimately show *?thesis* **by** *simp*
qed
ultimately show *?thesis* **unfolding** *well-order-on-def* **by** *blast*
qed
moreover have *Well-order* |(*UNIV* − $\mathcal{O}$)::*'U rel set*| **using** *card-of-Well-order* **by** *blast*
moreover have *Field* (*Restr ordLeq* $\mathcal{O}$) ∩ *Field* (|(*UNIV* − $\mathcal{O}$)::*'U rel set*|) = {}
proof −
have *Field* (*Restr ordLeq* $\mathcal{O}$) ⊆ $\mathcal{O}$ **unfolding** *Field-def* **by** *blast*
moreover have *Field* (|(*UNIV* − $\mathcal{O}$)::*'U rel set*|) ⊆ *UNIV* − $\mathcal{O}$ **by** *simp*
ultimately show *?thesis* **by** *blast*
qed
ultimately show *?thesis* **unfolding** *oord-def* **using** *Osum-Well-order* **by** *blast*
qed

lemma *lem-lmord-inf*:
fixes α::*'U rel*
assumes *lm-ord* α
shows ¬ *finite* (*Field* α)
proof −
have *finite* (*Field* α) ⟶ *False*
proof
assume *c1*: *finite* (*Field* α)
have *c2*: *Well-order* α **using** *assms* **unfolding** *lm-ord-def* **by** *blast*
have α ≠ {} **using** *assms lm-ord-def* **by** *blast*
then have *Field* α ≠ {} **unfolding** *Field-def* **by** *force*
then have *wo-rel.isMaxim* α (*Field* α) (*wo-rel.maxim* α (*Field* α))
using *c1 c2 wo-rel.maxim-isMaxim*[*of* α *Field* α] **unfolding** *wo-rel-def* **by** *blast*
then have ∃ *j*∈*Field* α. ∀ *i*∈*Field* α. (*i*, *j*) ∈ α
using *c2 wo-rel.isMaxim-def*[*of* α *Field* α] **unfolding** *wo-rel-def* **by** *blast*
then have *isSuccOrd* α **using** *c2 wo-rel.isSuccOrd-def* **unfolding** *wo-rel-def* **by** *blast*
then show *False* **using** *assms* **unfolding** *lm-ord-def* **by** *blast*
qed
then show *?thesis* **by** *blast*
qed

lemma *lem-sucord-ex*:
fixes α β::*'U rel*
assumes α <*o* β
shows ∃ α'::*'U rel. sc-ord* α α'
proof −
obtain *S*::*'U rel set* **where** *b1*: *S* = { γ::*'U rel.* α <*o* γ } **by** *blast*
then have *S* ≠ {} ∧ (∀ α ∈ *S*. *Well-order* α) **using** *assms ordLess-Well-order-simp*

by *blast*
then obtain α' **where** $\alpha' \in S \wedge (\forall \alpha \in S.\ \alpha' \leq o\ \alpha)$
using *BNF-Wellorder-Constructions.exists-minim-Well-order*[*of S*] **by** *blast*
then show *?thesis* **unfolding** *b1 sc-ord-def* **by** *blast*
qed

lemma *lem-osucc-eq*: *isSuccOrd* $\alpha \Longrightarrow \alpha =o\ \beta \Longrightarrow$ *isSuccOrd* β
proof −
assume *a1*: *isSuccOrd* α **and** *a2*: $\alpha =o\ \beta$
moreover then have *a3*: *wo-rel* α **and** *a4*: *wo-rel* β **unfolding** *ordIso-def wo-rel-def* **by** *blast+*
obtain *j* **where** *a5*: $j \in$ *Field* α **and** *a6*: $\forall i \in$*Field* α. $(i, j) \in \alpha$ **using** *a1 a3 wo-rel.isSuccOrd-def* **by** *blast*
obtain *f* **where** *a7*: *iso* $\alpha\ \beta\ f$ **using** *a2* **unfolding** *ordIso-def* **by** *blast*
have $(f\ j) \in$ *Field* β **using** *a5 a7* **unfolding** *iso-def bij-betw-def* **by** *blast*
moreover have $\forall\ i' \in$ *Field* β. $(i', f\ j) \in \beta$
proof
fix i'
assume *b1*: $i' \in$ *Field* β
then obtain *i* **where** *b2*: $i \in$ *Field* $\alpha \wedge i' = f\ i$ **using** *a7* **unfolding** *iso-def bij-betw-def* **by** *blast*
then have $(i, j) \in \alpha$ **using** *a6* **by** *blast*
then have $(f\ i,\ f\ j) \in \beta$ **using** *a2 a7* **by** (*meson iso-oproj oproj-in ordIso-Well-order-simp*)
then show $(i', f\ j) \in \beta$ **using** *b2* **by** *blast*
qed
ultimately have $\exists j \in$*Field* β. $\forall i \in$*Field* β. $(i, j) \in \beta$ **by** *blast*
then show *isSuccOrd* β **using** *a4 wo-rel.isSuccOrd-def* **by** *blast*
qed

lemma *lem-ord-subemp*: $(\alpha$::$'a$ *rel*$) \leq o$ $(\{\}$::$'b$ *rel*$) \Longrightarrow \alpha = \{\}$
proof −
assume $\alpha \leq o$ $(\{\}$::$'b$ *rel*$)$
then obtain *f* **where** *embed* α $(\{\}$::$'b$ *rel*$)$ *f* **unfolding** *ordLeq-def* **by** *blast*
then show $\alpha = \{\}$ **unfolding** *embed-def bij-betw-def Field-def under-def* **by** *force*
qed

lemma *lem-ordint-sucord*:
fixes $\alpha 0$::$'a$ *rel* **and** α::$'b$ *rel*
assumes $\alpha 0 <o\ \alpha \wedge (\forall\ \gamma$::$'b$ *rel*. $\alpha 0 <o\ \gamma \longrightarrow \alpha \leq o\ \gamma)$
shows *isSuccOrd* α
proof −
have *c1*: *Well-order* α **using** *assms* **unfolding** *ordLess-def* **by** *blast*
obtain *f* **where** *e3*: *Well-order* $\alpha 0 \wedge$ *Well-order* $\alpha \wedge$ *embedS* $\alpha 0\ \alpha\ f$ **using** *assms* **unfolding** *ordLess-def* **by** *blast*
moreover have *e4*: f ‘ *Field* $\alpha 0 \subseteq$ *Field* α **using** *e3 embed-in-Field*[*of* $\alpha 0\ \alpha\ f$] **unfolding** *embedS-def* **by** *blast*
have f ‘ *Field* $\alpha 0 \neq$ *Field* α **using** *e3 embed-inj-on* **unfolding** *bij-betw-def*

embedS-def **by** *blast*
then obtain *j0* **where** *e5*: $j0 \in Field\ \alpha \wedge j0 \notin f\ `\ Field\ \alpha 0$ **using** *e4* **by** *blast*
moreover have $\forall\ i \in Field\ \alpha.\ (i, j0) \in \alpha$
proof
fix *i*
assume $i \in Field\ \alpha$
moreover then have $(i, i) \in \alpha$ **using** *e3* **unfolding** *well-order-on-def linear-order-on-def partial-order-on-def preorder-on-def refl-on-def* **by** *blast*
moreover have $(j0, i) \in \alpha \longrightarrow (i, j0) \in \alpha$
proof
assume *g1*: $(j0, i) \in \alpha$
obtain γ **where** *g2*: $\gamma = Restr\ \alpha\ (under\ \alpha\ j0)$ **by** *blast*
then have *g3*: *Well-order* γ **using** *e3 Well-order-Restr* **by** *blast*
have $\alpha 0 <o\ \gamma$
proof −
have *h1*: $\forall\ a \in Field\ \alpha 0.\ f\ a \in under\ \alpha\ j0$
proof
fix *a*
assume *i1*: $a \in Field\ \alpha 0$
then have *i2*: *bij-betw* $f\ (under\ \alpha 0\ a)\ (under\ \alpha\ (f\ a))$ **using** *e3* **unfolding** *embedS-def embed-def* **by** *blast*
have $(j0, f\ a) \in \alpha \longrightarrow False$
proof
assume $(j0, f\ a) \in \alpha$
then obtain *b* **where** $j0 = f\ b \wedge b \in under\ \alpha 0\ a$ **using** *i2* **unfolding** *under-def bij-betw-def* **by** (*simp*, *blast*)
moreover then have $b \in Field\ \alpha 0$ **unfolding** *under-def Field-def* **by** *blast*
ultimately show *False* **using** *e5* **by** *blast*
qed
moreover have *i3*: $j0 \in Field\ \alpha$ **using** *g1* **unfolding** *Field-def* **by** *blast*
moreover have $f\ a \in Field\ \alpha$ **using** *i1 e3 embed-Field* **unfolding** *embedS-def* **by** *blast*
ultimately have *i4*: $(f\ a, j0) \in \alpha$
using *e3* **unfolding** *well-order-on-def linear-order-on-def total-on-def partial-order-on-def preorder-on-def refl-on-def* **by** *metis*
then show $f\ a \in under\ \alpha\ j0$ **unfolding** *under-def* **by** *blast*
qed
then have *compat* $\alpha 0\ \gamma\ f$
using *e3 g2 embed-compat* **unfolding** *Field-def embedS-def compat-def* **by** *blast*
moreover have *ofilter* $\gamma\ (f\ `\ Field\ \alpha 0)$
proof −
have *ofilter* $\alpha\ (under\ \alpha\ j0)$ **using** *e3 wo-rel.under-ofilter*[*of* α] **unfolding** *wo-rel-def* **by** *blast*
moreover have *ofilter* $\alpha\ (f\ `\ Field\ \alpha 0)$
using *e3 embed-iff-compat-inj-on-ofilter*[*of* $\alpha 0\ \alpha\ f$] **unfolding** *embedS-def* **by** *blast*
moreover have $f\ `\ Field\ \alpha 0 \subseteq under\ \alpha\ j0$ **using** *h1* **by** *blast*

ultimately show *ofilter* γ (*f* ' *Field* $\alpha 0$)
using *g2 e3 ofilter-Restr-subset*[*of* α *f* ' *Field* $\alpha 0$ *under* α *j0*] **by** *blast*
qed
moreover have *inj-on f* (*Field* $\alpha 0$)
using *e3 embed-iff-compat-inj-on-ofilter*[*of* $\alpha 0$ α *f*] **unfolding** *embedS-def* **by** *blast*
ultimately have *embed* $\alpha 0$ γ *f* **using** *g3 e3 embed-iff-compat-inj-on-ofilter*[*of* $\alpha 0$ γ *f*] **by** *blast*
moreover have *bij-betw f* (*Field* $\alpha 0$) (*Field* γ) $\longrightarrow$ *False*
proof
assume *i1*: *bij-betw f* (*Field* $\alpha 0$) (*Field* γ)
have (*j0*, *j0*) $\in \alpha$ **using** *e3 e5* **unfolding** *well-order-on-def linear-order-on-def partial-order-on-def preorder-on-def refl-on-def* **by** *blast*
then have *j0* $\in$ *Field* γ **using** *g2* **unfolding** *under-def Field-def* **by** *blast*
then show *False* **using** *i1 e5* **unfolding** *bij-betw-def* **by** *blast*
qed
ultimately have *embedS* $\alpha 0$ γ *f* **unfolding** *embedS-def* **by** *blast*
then show *?thesis* **using** *g3 e3* **unfolding** *ordLess-def* **by** *blast*
qed
then have α *=o* γ **using** *assms g2 e3 under-Restr-ordLeq*[*of* α *j0*] *ordIso-iff-ordLeq* **by** *blast*
then obtain *f1* **where** *iso* α γ *f1* **unfolding** *ordIso-def* **by** *blast*
then have *g4*: *embed* α γ *f1* $\wedge$ *bij-betw f1* (*Field* α) (*Field* γ) **unfolding** *iso-def* **by** *blast*
then have *f1* ' *under* α *i* = *under* γ (*f1 i*) **using** *g1* **unfolding** *bij-betw-def embed-def Field-def* **by** *blast*
then have (*f1 i*, *j0*) $\in \alpha$ **using** *g1* **unfolding** *g2 under-def* **by** *blast*
moreover have *f1 i* = *i*
proof −
have *Restr* α (*Field* α) = α **unfolding** *Field-def* **by** *force*
moreover have *ofilter* α (*under* α *j0*) **using** *e3 wo-rel.under-ofilter*[*of* α] **unfolding** *wo-rel-def* **by** *blast*
moreover have *ofilter* α (*Field* α) **unfolding** *ofilter-def under-def Field-def* **by** *blast*
moreover have *under* α *j0* $\subseteq$ *Field* α **unfolding** *under-def Field-def* **by** *blast*
ultimately have *embed* γ α *id* **using** *g2 e3 ofilter-subset-embed* **by** *metis*
then have *embed* α α (*id* $\circ$ *f1*) **using** *g4 e3 comp-embed* **by** *blast*
then have *embed* α α *f1* **by** *simp*
moreover have *embed* α α *id* **unfolding** *embed-def id-def bij-betw-def inj-on-def* **by** *blast*
ultimately have $\forall$ *k* $\in$ *Field* α. *f1 k* = *k* **using** *e3 embed-unique*[*of* α α *f1 id*] **unfolding** *id-def* **by** *blast*
moreover have *i* $\in$ *Field* α **using** *g1* **unfolding** *Field-def* **by** *blast*
ultimately show *?thesis* **by** *blast*
qed
ultimately show (*i*, *j0*) $\in \alpha$ **by** *metis*
qed

ultimately show $(i, j0) \in \alpha$
using *e3 e5* **unfolding** *well-order-on-def linear-order-on-def total-on-def* **by** *metis*
qed
ultimately show *isSuccOrd* α **using** *c1 wo-rel.isSuccOrd-def*[*of* α] **unfolding** *wo-rel-def* **by** *blast*
qed

lemma *lem-sucord-ordint*:
fixes α::'*U rel*
assumes *Well-order* $\alpha \wedge$ *isSuccOrd* α
shows $\exists\ \alpha 0$::'*U rel*. $\alpha 0 <o\ \alpha \wedge (\forall\ \gamma$::'*U rel*. $\alpha 0 <o\ \gamma \longrightarrow \alpha \leq o\ \gamma)$
proof −
obtain *j* **where** *b1*: $j \in$ *Field* $\alpha \wedge (\forall i \in$ *Field* α. $(i, j) \in \alpha)$
using *assms wo-rel.isSuccOrd-def* **unfolding** *wo-rel-def* **by** *blast*
moreover obtain $\alpha 0$ **where** *b2*: $\alpha 0$ = *Restr* α (*UNIV* − $\{j\}$) **by** *blast*
moreover have $\forall\ i.\ (j, i) \in \alpha \longrightarrow i = j$ **using** *assms b1* **unfolding** *Field-def well-order-on-def*
linear-order-on-def partial-order-on-def antisym-def **by** *blast*
ultimately have *b3*: *embedS* $\alpha 0\ \alpha$ *id*
unfolding *Field-def embedS-def embed-def id-def bij-betw-def under-def inj-on-def*

apply *simp*
by *blast*
moreover have *b4*: *Well-order* $\alpha 0$ **using** *assms b2 Well-order-Restr* **by** *blast*
ultimately have $\alpha 0 <o\ \alpha$ **using** *assms* **unfolding** *ordLess-def* **by** *blast*
moreover have $\forall\ \gamma$::'*U rel*. $\alpha 0 <o\ \gamma \longrightarrow \alpha \leq o\ \gamma$
proof (*intro allI impI*)
fix γ::'*U rel*
assume *c1*: $\alpha 0 <o\ \gamma$
then have *c2*: *Well-order* γ **unfolding** *ordLess-def* **by** *blast*
obtain *f* **where** *embedS* $\alpha 0\ \gamma$ *f* **using** *c1* **unfolding** *ordLess-def* **by** *blast*
then have *c3*: *embed* $\alpha 0\ \gamma$ *f* $\wedge \neg$ *bij-betw f* (*Field* $\alpha 0$) (*Field* γ) **unfolding** *embedS-def* **by** *blast*
have $\gamma <o\ \alpha \longrightarrow$ *False*
proof
assume *d1*: $\gamma <o\ \alpha$
obtain *g* **where** *embedS* $\gamma\ \alpha$ *g* **using** *d1* **unfolding** *ordLess-def* **by** *blast*
then have *d3*: *embed* $\gamma\ \alpha$ *g* $\wedge \neg$ *bij-betw g* (*Field* γ) (*Field* α) **unfolding** *embedS-def* **by** *blast*
have *d4*: $j \in g$ ‘ *Field* $\gamma \longrightarrow$ *False*
proof
assume $j \in g$ ‘ *Field* γ
then obtain *a* **where** $a \in$ *Field* $\gamma \wedge g\ a = j$ **by** *blast*
then have *bij-betw g* (*under* γ *a*) (*under* α *j*) **using** *d3* **unfolding** *embed-def* **by** *blast*
moreover have *under* α *j* = *Field* α **using** *b1* **unfolding** *under-def Field-def* **by** *blast*
ultimately have *bij-betw g* (*under* γ *a*) (*Field* α) **by** *simp*

then have *g ‘ Field* $\gamma \neq$ *Field* $\alpha \wedge$ *g ‘ Field* $\gamma \subseteq$ *Field* $\alpha \wedge$ *g ‘ under* γ *a* = *Field* α
using *c2 d3 embed-inj-on*[*of* γ α *g*] *embed-Field*[*of* γ α *g*] **unfolding** *bij-betw-def* **by** *blast*
moreover have *under* γ *a* $\subseteq$ *Field* γ **unfolding** *under-def Field-def* **by** *blast*
ultimately show *False* **by** *blast*
qed
have *Field* $\gamma \subseteq$ *f ‘ Field* $\alpha 0$
proof
fix *a*
assume *e1*: $a \in$ *Field* γ
then have *bij-betw g* (*under* γ *a*) (*under* α (*g a*)) **using** *d3* **unfolding** *embed-def* **by** *blast*
have *g a* $\in$ *Field* $\alpha - \{j\}$ **using** *e1 c2 d3 d4 embed-Field* **by** *blast*
moreover then have (*g a*, *g a*) $\in \alpha$ **using** *assms* **unfolding** *Field-def well-order-on-def*
linear-order-on-def partial-order-on-def preorder-on-def refl-on-def **by** *blast*
ultimately have *e2*: *g a* $\in$ *Field* $\alpha 0$ **using** *b2* **unfolding** *Field-def* **by** *blast*
have *embed* $\alpha 0$ α ($g \circ f$) **using** *b4 c3 d3 comp-embed*[*of* $\alpha 0$ γ *f* α *g*] **by** *blast*
then have $\forall$ $x \in$ *Field* $\alpha 0$. *g* (*f x*) = *x* **using** *assms b3 b4 embed-unique*[*of* $\alpha 0$ α $g \circ f$ *id*]
unfolding *embedS-def comp-def id-def* **by** *blast*
then have *g* (*f* (*g a*)) = *g a* **using** *e2* **by** *blast*
moreover have *inj-on g* (*Field* γ) **using** *c2 d3 embed-inj-on*[*of* γ α *g*] **by** *blast*
moreover have *f* (*g a*) $\in$ *Field* γ **using** *e2 b4 c3 embed-Field*[*of* $\alpha 0$ γ *f*] **by** *blast*
ultimately have *f* (*g a*) = *a* **using** *e1* **unfolding** *inj-on-def* **by** *blast*
then show $a \in$ *f ‘ Field* $\alpha 0$ **using** *e2* **by** *force*
qed
then have *bij-betw f* (*Field* $\alpha 0$) (*Field* γ)
using *b4 c3 embed-inj-on*[*of* $\alpha 0$ γ *f*] *embed-Field*[*of* $\alpha 0$ γ *f*] **unfolding** *bij-betw-def* **by** *blast*
then show *False* **using** *c3* **by** *blast*
qed
then show $\alpha \leq o$ γ **using** *assms c2* **by** *simp*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-sclm-ordind*:
fixes *P*::*'U rel* $\Rightarrow$ *bool*
assumes *a1*: *P* {}
and *a2*: $\forall$ $\alpha 0$ α::*'U rel*. (*sc-ord* $\alpha 0$ $\alpha \wedge P$ $\alpha 0 \longrightarrow P$ α)
and *a3*: $\forall$ α. ((*lm-ord* $\alpha \wedge$ ($\forall$ β. $\beta <o$ $\alpha \longrightarrow P$ β)) $\longrightarrow P$ α)
shows $\forall$ α. *Well-order* $\alpha \longrightarrow P$ α

```
proof −
  obtain Q where b1: Q = (λ α. Well-order α ⟶ P α) by blast
  have ∀ α. (∀ β. β <o α ⟶ Q β) ⟶ Q α
  proof (intro allI impI)
    fix α::'U rel
    assume c1: ∀ β. β <o α ⟶ Q β
    then have c2: ∀ β. β <o α ⟶ P β unfolding b1 ordLess-def by blast
    show Q α
    proof (cases ∃ α0. sc-ord α0 α)
      assume ∃ α0. sc-ord α0 α
      then obtain α0 where sc-ord α0 α by blast
      then show Q α using c2 b1 a2 unfolding sc-ord-def by blast
    next
      assume ¬ (∃ α0. sc-ord α0 α)
      then have (¬ Well-order α) ∨ α = {} ∨ lm-ord α
        using lem-sucord-ordint unfolding sc-ord-def lm-ord-def by blast
      moreover have lm-ord α ⟶ P α using c2 a3 by blast
      ultimately show Q α using a1 b1 by blast
    qed
  qed
  then show ?thesis using b1 wf-induct[of ordLess Q] wf-ordLess by blast
qed

lemma lem-ordseq-rec-sets:
fixes E::'U set and F::'U rel ⇒ 'U set ⇒ 'U set
assumes ∀ α β. α =o β ⟶ F α = F β
shows ∃ f::('U rel ⇒ 'U set).
          f {} = E
        ∧ (∀ α0 α::'U rel. (sc-ord α0 α ⟶ f α = F α0 (f α0)))
        ∧ (∀ α. lm-ord α ⟶ f α = ⋃ { D. ∃ β. β <o α ∧ D = f β })
        ∧ (∀ α β. α =o β ⟶ f α = f β)
proof −
  obtain cmp::'U rel rel where b1: cmp = oord by blast
   then interpret cmp: wo-rel cmp unfolding wo-rel-def using lem-oord-wo by
blast
  obtain L where b2: L = (λ g::'U rel ⇒ 'U set. λ α::'U rel. ⋃ (g ` (underS cmp
α))) by blast
  then have b3: adm-woL cmp L unfolding cmp.adm-woL-def by blast
  obtain fo where b4: fo = (worecZSL cmp E F L) by blast
  obtain f where b5: f = (λ α::'U rel. fo (nord α)) by blast
  have b6: fo (zero cmp) = E using b3 b4 cmp.worecZSL-zero by simp
  have b7: ∀ α. aboveS cmp α ≠ {} ⟶ fo (succ cmp α) = F α (fo α)
    using b3 b4 cmp.worecZSL-succ by metis
  have b8: ∀ α. isLim cmp α ∧ α ≠ zero cmp ⟶ fo α = ⋃ (fo ` (underS cmp
α))
    using b2 b3 b4 cmp.worecZSL-isLim by metis
  have b9: zero cmp = {} ∧ nord ({}::'U rel) = {}
  proof −
    obtain isz where c1: isz = (λ α. α ∈ Field cmp ∧ (∀β∈Field cmp. (α, β) ∈
```

cmp)) **by** *blast*
have *c2*: {} ∈ (𝒪::′*U rel set*)
proof −
have *Well-order* ({}::′*U rel*) **by** *simp*
moreover then have *nord* ({}::′*U rel*) = {} **using** *lem-nord-r lem-ord-subemp ordIso-iff-ordLeq* **by** *blast*
ultimately show *?thesis* **unfolding** *𝒪-def* **by** *blast*
qed
moreover have ∀ *β* ∈ 𝒪::(′*U rel set*). ({}, *β*) ∈ *oord*
proof
fix *β*::′*U rel*
assume *d1*: *β* ∈ 𝒪
then have *Well-order β* **using** *lem-Owo* **by** *blast*
then have {} ≤*o β* **using** *ozero-ordLeq* **unfolding** *ozero-def* **by** *blast*
then show ({}, *β*) ∈ *oord* **using** *d1 c2* **unfolding** *oord-def* **by** *blast*
qed
ultimately have *isz* {} **using** *c1 b1 lem-fld-oord* **by** *blast*
moreover have ∀ *α*. *isz α* ⟶ *α* = {}
proof (*intro allI impI*)
fix *α*
assume *d1*: *isz α*
then have *d2*: *α* ∈ 𝒪 ∧ (∀ *β* ∈ 𝒪. (*α*, *β*) ∈ *oord*) **using** *c1 b1 lem-fld-oord* **by** *blast*
have *Well-order* ({}::′*U rel*) **by** *simp*
then have *α* ≤*o nord* ({}::′*U rel*) ∧ *nord* ({}::′*U rel*) =*o* ({}::′*U rel*)
using *d2 lem-nord-r* **unfolding** *oord-def 𝒪-def* **by** *blast*
then have *α* ≤*o* ({}::′*U rel*) **using** *ordLeq-ordIso-trans* **by** *blast*
then show *α* = {} **using** *lem-ord-subemp* **by** *blast*
qed
ultimately have (*THE α*. *isz α*) = {} **by** (*simp only*: *the-equality*)
then have *zero cmp* = {} **unfolding** *c1 cmp.zero-def cmp.minim-def cmp.isMinim-def* **by** *blast*
moreover have *nord* ({}::′*U rel*) = {} **using** *c2 lem-Onord* **by** *blast*
ultimately show *?thesis* **by** *blast*
qed
have *b10*: ∀ *α α*′::′*U rel*. *aboveS cmp α* ≠ {} ∧ *α*′ = *succ cmp α* ⟶ (*α* ∈ 𝒪 ∧ *α*′ ∈ 𝒪 ∧ *α* <*o α*′ ∧ (∀ *β*::′*U rel*. *α* <*o β* ⟶ *α*′ ≤*o β*))
proof (*intro allI impI*)
fix *α α*′
assume *aboveS cmp α* ≠ {} ∧ *α*′ = *succ cmp α*
moreover then have *AboveS cmp* {*α*} ⊆ *Field cmp* ∧ *AboveS cmp* {*α*} ≠ {}
unfolding *AboveS-def aboveS-def Field-def* **by** *blast*
ultimately have *c4*: *isMinim cmp* (*AboveS cmp* {*α*}) *α*′
using *cmp.minim-isMinim* **unfolding** *cmp.succ-def cmp.suc-def* **by** *blast*
have *c5*: (*α*, *α*′) ∈ *cmp* ∧ *α* ≠ *α*′ **using** *c4 lem-fld-oord* **unfolding** *cmp.isMinim-def AboveS-def* **by** *blast*
then have *α* ≤*o α*′ ∧ ¬ (*α* =*o α*′) **using** *b1 lem-Oeq* **unfolding** *oord-def* **by** *blast*
then have *α* <*o α*′ **using** *ordLeq-iff-ordLess-or-ordIso* **by** *blast*

moreover have $\forall\ \beta::'U\ rel.\ \alpha <o\ \beta \longrightarrow \alpha' \leq o\ \beta$
proof (*intro allI impI*)
fix $\beta::'U\ rel$
assume *d1*: $\alpha <o\ \beta$
have $nord\ \beta \neq nord\ \alpha \wedge (nord\ \alpha,\ nord\ \beta) \in cmp$ **using** *d1 b1 lem-nord-less* **by** *blast*
moreover then have $nord\ \beta \in Field\ cmp$ **unfolding** *Field-def* **by** *blast*
ultimately have $nord\ \beta \in AboveS\ cmp\ \{nord\ \alpha\}$ **unfolding** *AboveS-def* **by** *blast*
moreover have $\alpha = nord\ \alpha$ **using** *c5 b1 lem-Onord* **unfolding** *oord-def* **by** *blast*
ultimately have $(\alpha',\ nord\ \beta) \in cmp$ **using** *c4* **unfolding** *cmp.isMinim-def* **by** *metis*
then have $\alpha' \leq o\ nord\ \beta$ **unfolding** *b1 oord-def* **by** *blast*
moreover have $nord\ \beta =o\ \beta$ **using** *d1 lem-nord-r ordLess-Well-order-simp* **by** *blast*
ultimately show $\alpha' \leq o\ \beta$ **using** *ordLeq-ordIso-trans* **by** *blast*
qed
moreover have $\alpha \in \mathcal{O} \wedge \alpha' \in \mathcal{O}$ **using** *c5 b1* **unfolding** *oord-def* **by** *blast*
ultimately show $\alpha \in \mathcal{O} \wedge \alpha' \in \mathcal{O} \wedge \alpha <o\ \alpha' \wedge (\forall\ \beta::'U\ rel.\ \alpha <o\ \beta \longrightarrow \alpha' \leq o\ \beta)$ **by** *blast*
qed
then have *b11*: $\forall\ \alpha::'U\ rel.\ Well\text{-}order\ \alpha \wedge \neg\ (\alpha = \{\} \vee isSuccOrd\ \alpha) \longrightarrow isLim\ cmp\ \alpha$
using *lem-ordint-sucord* **unfolding** *cmp.isLim-def cmp.isSucc-def* **by** *metis*
have $f\ \{\} = E$ **using** *b5 b6 b9* **by** *simp*
moreover have $(\forall\ \alpha\ \alpha'::'U\ rel.\ (\alpha <o\ \alpha' \wedge (\forall\ \beta::'U\ rel.\ \alpha <o\ \beta \longrightarrow \alpha' \leq o\ \beta) \longrightarrow f\ \alpha' = F\ \alpha\ (f\ \alpha)))$
proof (*intro allI impI*)
fix $\alpha\ \alpha'::'U\ rel$
assume *c1*: $\alpha <o\ \alpha' \wedge (\forall\ \beta::'U\ rel.\ \alpha <o\ \beta \longrightarrow \alpha' \leq o\ \beta)$
then have *c2*: $(aboveS\ cmp\ (nord\ \alpha)) \neq \{\}$ **using** *lem-nord-less* **unfolding** *b1 aboveS-def* **by** *fast*
obtain γ **where** *c3*: $\gamma = succ\ cmp\ (nord\ \alpha)$ **by** *blast*
have *c4*: $\gamma \in \mathcal{O} \wedge (nord\ \alpha) <o\ \gamma \wedge (\forall \beta::'U\ rel.\ (nord\ \alpha) <o\ \beta \longrightarrow \gamma \leq o\ \beta)$ **using** *c2 c3 b10* **by** *blast*
moreover have $nord\ \alpha =o\ \alpha$ **using** *c1 lem-nord-r ordLess-Well-order-simp* **by** *blast*
ultimately have $\alpha <o\ \gamma \wedge (\forall \beta::'U\ rel.\ \alpha <o\ \beta \longrightarrow \gamma \leq o\ \beta)$ **using** *ordIso-iff-ordLeq ordLeq-ordLess-trans* **by** *blast*
then have $\alpha' =o\ \gamma$ **using** *c1 ordIso-iff-ordLeq* **by** *blast*
then have $f\ \alpha' = f\ \gamma$ **using** *b5 lem-nord-eq* **by** *metis*
moreover have $\gamma = nord\ \gamma$ **using** *c4 lem-Onord* **by** *blast*
moreover have $fo\ \gamma = F\ (nord\ \alpha)\ (f\ \alpha)$ **using** *c2 c3 b5 b7* **by** *blast*
moreover have $F\ (nord\ \alpha)\ (f\ \alpha) = F\ \alpha\ (f\ \alpha)$ **using** *assms c1 lem-nord-r ordLess-Well-order-simp* **by** *metis*
ultimately show $f\ \alpha' = F\ \alpha\ (f\ \alpha)$ **using** *b5* **by** *metis*
qed
moreover have $\forall\ \alpha.\ (Well\text{-}order\ \alpha \wedge \neg\ (\alpha = \{\} \vee isSuccOrd\ \alpha)) \longrightarrow f\ \alpha =$

$\bigcup$ { *D*. ∃ *β*. *β* <*o* *α* ∧ *D* = *f* *β* }
proof (*intro allI impI*)
fix *α*::′*U rel*
assume *c1*: *Well-order* *α* ∧ ¬ (*α* = {} ∨ *isSuccOrd* *α*)
then have *Well-order* (*nord* *α*) **using** *lem-nord-l* **unfolding** *ordIso-def* **by** *blast*
moreover have *nord* *α* ≠ {} ∧ ¬ *isSuccOrd* (*nord* *α*)
using *c1 lem-ord-subemp ordIso-iff-ordLeq lem-osucc-eq*[*of nord α α*] *lem-nord-r*[*of α*] **by** *metis*
ultimately have *c2*: *fo* (*nord* *α*) = $\bigcup$ (*fo* ‘ (*underS cmp* (*nord* *α*))) **using** *b8 b9 b11* **by** *metis*
obtain *A* **where** *c3*: *A* = $\bigcup$ { *D*. ∃ *β*::′*U rel*. *β* <*o* *α* ∧ *D* = *f* *β* } **by** *blast*
have ∀ *γ* ∈ *underS cmp* (*nord* *α*). ∃ *β*::′*U rel*. *β* <*o* *α* ∧ *fo* *γ* = *f* *β*
proof
fix *γ*::′*U rel*
assume *γ* ∈ *underS cmp* (*nord* *α*)
then have *γ* ≠ *nord* *α* ∧ (*γ*, *nord* *α*) ∈ *oord* **unfolding** *b1 underS-def* **by** *blast*
then have *γ* ≤*o* *nord* *α* ∧ *γ* ∈ $\mathcal{O}$ ∧ ¬ (*γ* =*o* *nord* *α*) **using** *lem-Oeq* **unfolding** *oord-def* **by** *blast*
then have *γ* <*o* *nord* *α* ∧ *γ* = *nord* *γ* **using** *lem-Onord ordLeq-iff-ordLess-or-ordIso* **by** *blast*
moreover have *nord* *α* =*o* *α* **using** *c1 lem-nord-r* **by** *blast*
ultimately have *γ* <*o* *α* ∧ *fo* *γ* = *f* *γ* **unfolding** *b5* **using** *ordIso-imp-ordLeq ordLess-ordLeq-trans* **by** *metis*
then show ∃ *β*::′*U rel*. *β* <*o* *α* ∧ *fo* *γ* = *f* *β* **by** *blast*
qed
then have *c4*: *f* *α* ⊆ *A* **unfolding** *c2 c3 b5* **by** *blast*
have ∀ *β*::′*U rel*. *β* <*o* *α* ⟶ (∃ *γ* ∈ *underS cmp* (*nord* *α*). *f* *β* = *fo* *γ*)
proof (*intro allI impI*)
fix *β*::′*U rel*
assume *β* <*o* *α*
then have (*nord* *β*, *nord* *α*) ∈ *cmp* ∧ *nord* *β* ≠ *nord* *α* **using** *b1 lem-nord-less* **by** *blast*
then have *nord* *β* ∈ *underS cmp* (*nord* *α*) **unfolding** *underS-def* **by** *blast*
then show ∃ *γ* ∈ *underS cmp* (*nord* *α*). *f* *β* = *fo* *γ* **unfolding** *b5* **by** *blast*
qed
then have *A* ⊆ *f* *α* **unfolding** *c2 c3 b5* **by** *force*
then show *f* *α* = $\bigcup$ { *D*. ∃ *β*::′*U rel*. *β* <*o* *α* ∧ *D* = *f* *β* } **using** *c3 c4* **by** *blast*
qed
moreover have ∀ *α* *β*. *α* =*o* *β* ⟶ *f* *α* = *f* *β* **using** *b5 lem-nord-eq* **by** *metis*
ultimately show *?thesis* **unfolding** *sc-ord-def lm-ord-def* **by** *blast*
qed

lemma *lem-lmord-prec*:
fixes *α*::′*a rel* **and** *α*′::′*b rel*
assumes *a1*: *α*′ <*o* *α* **and** *a2*: *isLimOrd* *α*
shows ∃ *β*::(′*a rel*). *α*′ <*o* *β* ∧ *β* <*o* *α*

proof −
have ¬ *isSuccOrd* α **using** *a1 a2 wo-rel.isLimOrd-def* **unfolding** *ordLess-def wo-rel-def* **by** *blast*
then obtain β::'*a rel* **where** $\alpha' <o\ \beta \wedge \neg (\alpha \leq o\ \beta)$ **using** *a1 lem-ordint-sucord*[*of* α' α] **by** *blast*
then have $\alpha' <o\ \beta \wedge \beta <o\ \alpha$ **using** *a1 ordIso-imp-ordLeq ordLess-Well-order-simp*

ordLess-imp-ordLeq ordLess-or-ordIso **by** *metis*
then show *?thesis* **by** *blast*
qed

lemma *lem-inford-ge-w*:
fixes α::'*U rel*
assumes *Well-order* α **and** ¬ *finite* (*Field* α)
shows ω-*ord* $\leq o\ \alpha$
using *assms card-of-least infinite-iff-natLeq-ordLeq ordLeq-transitive* **by** *blast*

lemma *lem-ge-w-inford*:
fixes α::'*U rel*
assumes ω-*ord* $\leq o\ \alpha$
shows ¬ *finite* (*Field* α)
using *assms cinfinite-def cinfinite-mono natLeq-cinfinite* **by** *blast*

lemma *lem-fin-card*: *finite* $|A|$ = *finite* A
proof
assume *finite* $|A|$
then show *finite* A **using** *finite-Field* **by** *fastforce*
next
assume *finite* A
then show *finite* $|A|$ **using** *lem-fin-fl-rel* **by** *fastforce*
qed

lemma *lem-cardord-emp*: *Card-order* ({}::'*U rel*)
by (*metis Well-order-empty card-order-on-def ozero-def ozero-ordLeq well-order-on-Well-order*)

lemma *lem-card-emprel*: |{}::'*U rel*| =*o* ({}::'*U rel*)
proof −
have ({}::'*U rel*) =*o* |{}::'*U set*| **using** *lem-cardord-emp BNF-Cardinal-Order-Relation.card-of-unique* **by** *simp*
then show *?thesis* **using** *card-of-empty-ordIso ordIso-symmetric ordIso-transitive* **by** *blast*
qed

lemma *lem-cord-lin*: *Card-order* $\alpha \Longrightarrow$ *Card-order* $\beta \Longrightarrow (\alpha \leq o\ \beta) = (\neg (\beta <o\ \alpha))$ **by** *simp*

lemma *lem-co-one-ne-min*:
fixes α::'*U rel* **and** *a*::'*a*
assumes *Well-order* α **and** $\alpha \neq \{\}$

shows |{*a*}| ≤*o* *α*
proof −
  have *Field α* ≠ {} **using** *assms* **unfolding** *Field-def* **by** *force*
  then have |{*a*}| ≤*o* |*Field α*| **using** *assms* **by** *simp*
  moreover have |*Field α*| ≤*o* *α* **using** *assms card-of-least* **by** *blast*
  ultimately show *?thesis* **using** *ordLeq-transitive* **by** *blast*
qed

lemma *lem-rel-inf-fld-card*:
fixes *r*::*'U rel*
assumes ¬ *finite r*
shows |*Field r*| =*o* |*r*|
proof −
  obtain *f1*::*'U* × *'U* ⇒ *'U* **where** *b1*: *f1* = (λ (*x*,*y*). *x*) **by** *blast*
  obtain *f2*::*'U* × *'U* ⇒ *'U* **where** *b2*: *f2* = (λ (*x*,*y*). *y*) **by** *blast*
  then have *f1* ' *r* = *Domain r* ∧ *f2* ' *r* = *Range r* **using** *b1 b2* **by** *force*
  then have *b3*: |*Domain r*| ≤*o* |*r*| ∧ |*Range r*| ≤*o* |*r*|
    using *card-of-image*[*of f1 r*] *card-of-image*[*of f2 r*] **by** *simp*
  have |*Domain r*| ≤*o* |*Range r*| ∨ |*Range r*| ≤*o* |*Domain r*| **by** (*simp add*: *ordLeq-total*)
  moreover have |*Domain r*| ≤*o* |*Range r*| ⟶ |*Field r*| ≤*o* |*r*|
  proof
    assume *c1*: |*Domain r*| ≤*o* |*Range r*|
    moreover have *finite* (*Domain r*) ∧ *finite* (*Range r*) ⟶ *finite* (*Field r*) **unfolding** *Field-def* **by** *blast*
    ultimately have ¬ *finite* (*Range r*)
      using *assms lem-fin-fl-rel card-of-ordLeq-finite* **by** *blast*
    then have |*Field r*| =*o* |*Range r*| **using** *c1 card-of-Un-infinite* **unfolding** *Field-def* **by** *blast*
    then show |*Field r*| ≤*o* |*r*| **using** *b3 ordIso-ordLeq-trans* **by** *blast*
  qed
  moreover have |*Range r*| ≤*o* |*Domain r*| ⟶ |*Field r*| ≤*o* |*r*|
  proof
    assume *c1*: |*Range r*| ≤*o* |*Domain r*|
    moreover have *finite* (*Domain r*) ∧ *finite* (*Range r*) ⟶ *finite* (*Field r*) **unfolding** *Field-def* **by** *blast*
    ultimately have ¬ *finite* (*Domain r*)
      using *assms lem-fin-fl-rel card-of-ordLeq-finite* **by** *blast*
    then have |*Field r*| =*o* |*Domain r*| **using** *c1 card-of-Un-infinite* **unfolding** *Field-def* **by** *blast*
    then show |*Field r*| ≤*o* |*r*| **using** *b3 ordIso-ordLeq-trans* **by** *blast*
  qed
  ultimately have |*Field r*| ≤*o* |*r*| **by** *blast*
  moreover have |*r*| ≤*o* |*Field r*|
  proof −
    have *r* ⊆ (*Field r*) × (*Field r*) **unfolding** *Field-def* **by** *force*
    then have *c1*: |*r*| ≤*o* |*Field r* × *Field r*| **by** *simp*
    have ¬ *finite* (*Field r*) **using** *assms lem-fin-fl-rel* **by** *blast*
    then have *c2*: |*Field r* × *Field r*| =*o* |*Field r*| **by** *simp*

show *?thesis* **using** *c1 c2* **using** *ordLeq-ordIso-trans* **by** *blast*
qed
ultimately show *?thesis* **using** *ordIso-iff-ordLeq* **by** *blast*
qed

lemma *lem-cardreleq-cardfldeq-inf*:
fixes *r1 r2*:: *'U rel*
assumes *a1*: $|r1| =o |r2|$ **and** *a2*: $\neg$ *finite r1* $\vee$ $\neg$ *finite r2*
shows $|Field\ r1| =o |Field\ r2|$
proof −
have $\neg$ *finite r1* $\wedge$ $\neg$ *finite r2* **using** *a1 a2* **by** *simp*
then have $|Field\ r1| =o |r1| \wedge |Field\ r2| =o |r2|$ **using** *lem-rel-inf-fld-card* **by** *blast*
then show $|Field\ r1| =o |Field\ r2|$ **using** *a1* **by** (*meson ordIso-symmetric ordIso-transitive*)
qed

lemma *lem-card-un-bnd*:
fixes *S*::*'a set set* **and** α::*'U rel*
assumes *a3*: $\forall A \in S.\ |A| \leq o\ \alpha$ **and** *a4*: $|S| \leq o\ \alpha$ **and** *a5*: ω-*ord* $\leq o\ \alpha$
shows $|\bigcup S| \leq o\ \alpha$
proof −
obtain α' **where** *b0*: $\alpha' = |Field\ \alpha|$ **by** *blast*
have *a3'*: $\forall A \in S.\ |A| \leq o\ \alpha'$
proof
fix *A*
assume $A \in S$
then have $|A| \leq o\ \alpha$ **using** *a3* **by** *blast*
moreover have *Card-order* $|A|$ **by** *simp*
ultimately show $|A| \leq o\ \alpha'$ **using** *b0 card-of-unique card-of-mono2 ordIso-ordLeq-trans* **by** *blast*
qed
have *Card-order* $|S|$ **by** *simp*
then have *a4'*: $|S| \leq o\ \alpha'$ **using** *b0 a4 card-of-unique card-of-mono2 ordIso-ordLeq-trans* **by** *blast*
have *a5'*: $\neg$ *finite* (*Field* α')
proof −
have *Card-order* α' **using** *b0* **by** *simp*
then have $|Field\ \alpha| =o |Field\ \alpha'|$ **using** *b0 card-of-unique* **by** *blast*
moreover have $\neg$ *finite* (*Field* α) **using** *a5 lem-ge-w-inford* **by** *blast*
ultimately show $\neg$ *finite* (*Field* α') **by** *simp*
qed
have *a0'*: $\alpha' \leq o\ \alpha$ **using** *b0 a4* **by** *simp*
obtain *r* **where** *b1*: $r = \bigcup S$ **by** *blast*
have $\forall\ A \in S.\ |A| \leq o\ \alpha'$ **using** *a3' ordIso-ordLeq-trans* **by** *blast*
moreover have $r = (\bigcup A \in S.\ A)$ **using** *b1* **by** *blast*
moreover have *Card-order* α' **using** *b0* **by** *simp*
ultimately have $|r| \leq o\ \alpha'$ **using** *a4' a5' card-of-UNION-ordLeq-infinite-Field*[*of* α' *S* λ *x. x*] **by** *blast*

then have | $\bigcup$ *S* | $\leq o$ α' **unfolding** *b1* **using** *ordLeq-transitive* **by** *blast*
then show | $\bigcup$ *S* | $\leq o$ α **using** *a0'* *ordLeq-transitive* **by** *blast*
qed

lemma *lem-ord-suc-ge-w*:
fixes $\alpha 0$ α::*'U rel*
assumes *a1*: ω*-ord* $\leq o$ α **and** *a2*: *sc-ord* $\alpha 0$ α
shows ω*-ord* $\leq o$ $\alpha 0$
proof −
obtain *N*::*'U set* **where** *b1*: $|N|$ $=o$ ω*-ord* **using** *a1*
by (*metis card-of-nat Field-natLeq card-of-mono2 internalize-card-of-ordLeq ordIso-symmetric ordIso-transitive*)
have $\alpha 0$ $<o$ $|N|$ $\longrightarrow$ *False*
proof
assume *c1*: $\alpha 0$ $<o$ $|N|$
have *Well-order* ω*-ord* $\wedge$ *isLimOrd* ω*-ord*
by (*metis natLeq-Well-order Field-natLeq card-of-nat card-order-infinite-isLimOrd infinite-iff-natLeq-ordLeq natLeq-Card-order ordIso-iff-ordLeq*)
then have $\neg$ *isSuccOrd* ω*-ord* **using** *wo-rel.isLimOrd-def* **unfolding** *wo-rel-def* **by** *blast*
then have $\neg$ *isSuccOrd* $|N|$ **using** *b1 lem-osucc-eq* **by** *blast*
then have $\neg$ ($\forall\gamma$::*'U rel.* $\alpha 0$ $<o$ γ $\longrightarrow$ $|N|$ $\leq o$ γ)
using *c1* **unfolding** *sc-ord-def* **using** *lem-ordint-sucord*[*of* $\alpha 0$ $|N|$] **by** *blast*
then obtain β::*'U rel* **where** $\alpha 0$ $<o$ β $\wedge$ β $<o$ $|N|$
using *card-of-Well-order not-ordLeq-iff-ordLess ordLess-Well-order-simp* **by** *blast*
moreover then have α $\leq o$ β **using** *a2* **unfolding** *sc-ord-def* **by** *blast*
ultimately have α $<o$ $|N|$ **using** *ordLeq-ordLess-trans* **by** *blast*
then show *False* **using** *a1 b1* **using** *not-ordLess-ordLeq ordIso-iff-ordLeq ordLeq-transitive* **by** *blast*
qed
moreover have *Well-order* $\alpha 0$ **using** *a2* **unfolding** *sc-ord-def ordLess-def* **by** *blast*
moreover have *Well-order* $|N|$ **by** *simp*
ultimately show *?thesis* **using** *b1 not-ordLess-iff-ordLeq ordIso-iff-ordLeq ordLeq-transitive* **by** *blast*
qed

lemma *lem-restr-ordbnd*:
fixes *r*::*'U rel* **and** *A*::*'U set* **and** α::*'U rel*
assumes *a1*: ω*-ord* $\leq o$ α **and** *a2*: $|A|$ $\leq o$ α
shows $|$*Restr r A*$|$ $\leq o$ α
proof (*cases finite A*)
assume *finite A*
then have *finite* (*Restr r A*) **by** *blast*
then have $|$*Restr r A*$|$ $<o$ ω*-ord* **using** *finite-iff-ordLess-natLeq* **by** *blast*
then show $|$*Restr r A*$|$ $\leq o$ α **using** *a1 ordLeq-transitive ordLess-imp-ordLeq* **by** *blast*
next

assume ¬ *finite A*
then have $|A \times A| =o |A|$ **by** *simp*
moreover have $|Restr\ r\ A| \leq o\ |A \times A|$ **by** *simp*
ultimately show $|Restr\ r\ A| \leq o\ \alpha$ **using** *a2 ordLeq-ordIso-trans ordLeq-transitive* **by** *blast*
qed

lemma *lem-card-inf-lim*:
fixes $r::'U\ rel$
assumes *a1*: *Card-order* α **and** *a2*: $\omega\text{-}ord \leq o\ \alpha$
shows $\neg(\alpha = \{\} \vee isSuccOrd\ \alpha)$
proof −
obtain *s* **where** $s = Field\ \alpha$ **by** *blast*
then have $|s| =o\ \alpha$ **using** *a1 card-of-Field-ordIso* **by** *blast*
moreover then have $\neg (|s| <o\ |UNIV :: nat\ set|)$ **using** *a2* **by** (*metis card-of-nat ordLess-ordIso-trans not-ordLess-ordIso ordLeq-iff-ordLess-or-ordIso ordLeq-ordLess-trans*)
ultimately have ¬ *finite* (*Field* α) **using** *lem-fin-card lem-fin-fl-rel* **by** (*metis finite-iff-cardOf-nat ordIso-finite-Field*)
moreover then have $\alpha \neq \{\}$ **by** *force*
moreover have *wo-rel* α **using** *a1* **unfolding** *wo-rel-def card-order-on-def* **by** *blast*
ultimately show *?thesis* **using** *a1 card-order-infinite-isLimOrd wo-rel.isLimOrd-def* **by** *blast*
qed

lemma *lem-card-nreg-inf-osetlm*:
fixes $\alpha::'U\ rel$
assumes *a1*: *Card-order* α **and** *a2*: ¬ *regularCard* α **and** *a3*: ¬ *finite* (*Field* α)
shows $\exists\ S::'U\ rel\ set.\ |S| <o\ \alpha \wedge (\forall\ \alpha'{\in}S.\ \alpha' <o\ \alpha) \wedge (\forall\ \alpha'::'U\ rel.\ \alpha' <o\ \alpha \longrightarrow (\exists\ \beta \in S.\ \alpha' \leq o\ \beta))$
proof −
obtain $K::'U\ set$ **where** *b1*: $K \subseteq Field\ \alpha \wedge cofinal\ K\ \alpha$ **and** *b2*: $\neg\ |K| =o\ \alpha$
using *a2* **unfolding** *regularCard-def* **by** *blast*
have *b3*: $|K| <o\ \alpha$
proof −
have $|K| \leq o\ |Field\ \alpha|$ **using** *b1* **by** *simp*
moreover have $|Field\ \alpha| =o\ \alpha$ **using** *a1 card-of-Field-ordIso* **by** *blast*
ultimately show $|K| <o\ \alpha$ **using** *a1 b2* **by** (*metis card-of-Well-order card-order-on-def not-ordLeq-ordLess ordIso-or-ordLess ordIso-ordLess-trans*)
qed
have *b4*: *isLimOrd* α **using** *a1 a3 card-order-infinite-isLimOrd* **by** *blast*
obtain $f::'U \Rightarrow 'U\ rel$ **where** *b5*: $f = (\lambda\ a.\ Restr\ \alpha\ (under\ \alpha\ a))$ **by** *blast*
obtain $S::'U\ rel\ set$ **where** *b6*: $S = f\ `\ K$ **by** *blast*
then have $|S| <o\ \alpha$ **using** *b3 card-of-image ordLeq-ordLess-trans* **by** *blast*
moreover have $\forall\ \alpha'{\in}S.\ \alpha' <o\ \alpha$
proof
fix $\alpha'::'U\ rel$

assume *c1*: $\alpha' \in S$
then obtain *a* **where** *c2*: $a \in K \wedge \alpha' = Restr\ \alpha\ (under\ \alpha\ a)$ **using** *b5 b6* **by** *blast*
then have *c3*: *Well-order* $\alpha' \wedge$ *Well-order* α **using** *a1 Well-order-Restr* **unfolding** *card-order-on-def* **by** *blast*
moreover have *embed* $\alpha'\ \alpha$ *id*
proof −
have *ofilter* α (*under* α *a*) **using** *c3 wo-rel.under-ofilter*[*of* α] **unfolding** *wo-rel-def* **by** *blast*
moreover then have *under* α *a* $\subseteq$ *Field* α **unfolding** *ofilter-def* **by** *blast*
ultimately show *?thesis* **using** *c2 c3 ofilter-embed*[*of* α *under* α *a*] **by** *blast*
qed
moreover have *bij-betw id* (*Field* α') (*Field* α) $\longrightarrow$ *False*
proof
assume *bij-betw id* (*Field* α') (*Field* α)
then have *d1*: *Field* α' = *Field* α **unfolding** *bij-betw-def* **by** *simp*
have $a \in$ *Field* α **using** *c2 b1* **by** *blast*
then obtain *b* **where** *d2*: $b \in$ *aboveS* α *a*
using *b4 c3 wo-rel.isLimOrd-aboveS*[*of* α *a*] **unfolding** *wo-rel-def* **by** *blast*
then have $b \in$ *Field* α' **using** *d1* **unfolding** *aboveS-def Field-def* **by** *blast*
then have $b \in$ *under* α *a* **using** *c2* **unfolding** *Field-def* **by** *blast*
then show *False* **using** *a1 d2* **unfolding** *under-def aboveS-def card-order-on-def well-order-on-def linear-order-on-def partial-order-on-def antisym-def* **by** *blast*
qed
ultimately show $\alpha' <o\ \alpha$ **using** *embedS-def* **unfolding** *ordLess-def* **by** *blast*
qed
moreover have $\forall\ \alpha'::'U\ rel.\ \alpha' <o\ \alpha \longrightarrow (\exists\ \beta \in S.\ \alpha' \leq o\ \beta)$
proof (*intro allI impI*)
fix $\alpha'::'U\ rel$
assume *c1*: $\alpha' <o\ \alpha$
then obtain *g* **where** *c2*: *embed* $\alpha'\ \alpha\ g \wedge \neg$ *bij-betw g* (*Field* α') (*Field* α)
using *embedS-def* **unfolding** *ordLess-def* **by** *blast*
then have *g* ‘ *Field* $\alpha' \neq$ *Field* α
using *c1 embed-inj-on* **unfolding** *ordLess-def bij-betw-def* **by** *blast*
moreover have *g* ‘ *Field* $\alpha' \subseteq$ *Field* α
using *c1 c2 embed-in-Field*[*of* $\alpha'\ \alpha$ *g*] **unfolding** *ordLess-def* **by** *fast*
ultimately obtain *a* **where** *c3*: $a \in$ *Field* α − (*g* ‘ *Field* α') **by** *blast*
then obtain *b* β **where** *c4*: $b \in K \wedge (a,\ b) \in \alpha \wedge \beta = f\ b$ **using** *b1* **unfolding** *cofinal-def* **by** *blast*
then have $\beta \in S$ **using** *b6* **by** *blast*
moreover have $\alpha' \leq o\ \beta$
proof −
have *d1*: *Well-order* β **using** *c4 b5 a1 Well-order-Restr* **unfolding** *card-order-on-def* **by** *blast*
moreover have *embed* $\alpha'\ \beta$ *g*
proof −
have *e1*: $\forall\ x\ y.\ (x,\ y) \in \alpha' \longrightarrow (g\ x,\ g\ y) \in \beta$
proof (*intro allI impI*)

fix $x\ y$
assume *f1*: $(x,\ y) \in \alpha'$
then have *f2*: $(g\,x,\ g\ y) \in \alpha$ **using** *c2 embed-compat* **unfolding** *compat-def* **by** *blast*
moreover have $g\ y \in$ *under* $\alpha\ b$
proof −
have $(b,\ g\ y) \in \alpha \longrightarrow$ *False*
proof
assume $(b,\ g\ y) \in \alpha$
moreover have $(a,\ b) \in \alpha$ **using** *c4* **by** *blast*
ultimately have $(a,\ g\ y) \in \alpha$ **using** *a1* **unfolding** *under-def card-order-on-def well-order-on-def linear-order-on-def partial-order-on-def preorder-on-def trans-def* **by** *blast*
then have $a \in$ *under* $\alpha\ (g\ y)$ **unfolding** *under-def* **by** *blast*
moreover have *bij-betw* g (*under* $\alpha'\ y$) (*under* $\alpha\ (g\ y)$)
using *f1 c2* **unfolding** *embed-def Field-def* **by** *blast*
ultimately obtain y' **where** $y' \in$ *under* $\alpha'\ y \wedge a = g\ y'$ **unfolding** *bij-betw-def* **by** *blast*
moreover then have $y' \in$ *Field* α' **unfolding** *under-def Field-def* **by** *blast*
ultimately have $a \in g$ ‘ *Field* α' **by** *blast*
then show *False* **using** *c3* **by** *blast*
qed
moreover have $g\ y \in$ *Field* $\alpha \wedge b \in$ *Field* α **using** *f2 c4* **unfolding** *Field-def* **by** *blast*
ultimately have $(g\ y,\ b) \in \alpha$ **using** *a1* **unfolding** *card-order-on-def well-order-on-def linear-order-on-def partial-order-on-def preorder-on-def refl-on-def total-on-def* **by** *metis*
then show *?thesis* **unfolding** *under-def* **by** *blast*
qed
moreover then have $g\ x \in$ *under* $\alpha\ b$ **using** *a1 f2* **unfolding** *under-def card-order-on-def well-order-on-def linear-order-on-def partial-order-on-def preorder-on-def trans-def* **by** *blast*
ultimately have $(g\ x,\ g\ y) \in$ *Restr* α (*under* $\alpha\ b$) **by** *blast*
then show $(g\ x,\ g\ y) \in \beta$ **using** *c4 b5* **by** *blast*
qed
have *e2*: $\forall x \in g$ ‘ *Field* α'. *under* $\beta\ x \subseteq g$ ‘ *Field* α'
proof
fix x
assume $x \in g$ ‘ *Field* α'
then obtain c **where** *f1*: $c \in$ *Field* $\alpha' \wedge x = g\ c$ **by** *blast*
have $\forall\ x'.\ (x',\ x) \in \beta \longrightarrow x' \in g$ ‘ *Field* α'
proof (*intro allI impI*)
fix x'
assume $(x',\ x) \in \beta$
then have $(x',\ g\ c) \in$ *Restr* α (*under* $\alpha\ b$) **using** *b5 f1 c4* **by** *blast*

then have $x' \in$ *under* α $(g\ c)$ **unfolding** *under-def* **by** *blast*
moreover have *bij-betw* g (*under* α' c) (*under* α $(g\ c)$) **using** *f1 c2* **unfolding** *embed-def* **by** *blast*
ultimately obtain c' **where** $x' = g\ c' \wedge c' \in$ *under* α' c **unfolding** *bij-betw-def* **by** *blast*
moreover then have $c' \in$ *Field* α' **unfolding** *under-def Field-def* **by** *blast*
ultimately show $x' \in g$ ' *Field* α' **by** *blast*
qed
then show *under* β $x \subseteq g$ ' *Field* α' **unfolding** *under-def* **by** *blast*
qed
have *compat* α' β g **using** *e1* **unfolding** *compat-def* **by** *blast*
moreover then have *ofilter* β (g ' *Field* α') **using** *e2* **unfolding** *ofilter-def compat-def Field-def* **by** *blast*
moreover have *inj-on* g (*Field* α') **using** *c1 c2 embed-inj-on* **unfolding** *ordLess-def* **by** *blast*
ultimately show *?thesis* **using** *d1 c1 embed-iff-compat-inj-on-ofilter*[*of* α' β g]
unfolding *ordLess-def* **by** *blast*
qed
ultimately show *?thesis* **using** *c1* **unfolding** *ordLess-def ordLeq-def* **by** *blast*
qed
ultimately show $\exists\ \beta \in S.\ \alpha' \leq o\ \beta$ **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-card-un-bnd-stab*:
fixes S::'*a set set* **and** α::'*U rel*
assumes *stable* α **and** $\forall A \in S.\ |A| <o\ \alpha$ **and** $|S| <o\ \alpha$
shows $|\bigcup S| <o\ \alpha$
using *assms stable-UNION*[*of* α S λ x. x] **by** *simp*

lemma *lem-finwo-cardord*: *finite* $\alpha \Longrightarrow$ *Well-order* $\alpha \Longrightarrow$ *Card-order* α
proof –
assume *a1*: *finite* α **and** *a2*: *Well-order* α
have $\forall r$. *well-order-on* (*Field* α) $r \longrightarrow \alpha \leq o\ r$
proof (*intro allI impI*)
fix r
assume *well-order-on* (*Field* α) r
moreover have *well-order-on* (*Field* α) α **using** *a2* **by** *blast*
moreover have *finite* (*Field* α) **using** *a1 finite-Field* **by** *fastforce*
ultimately have $\alpha =o\ r$ **using** *finite-well-order-on-ordIso* **by** *blast*
then show $\alpha \leq o\ r$ **using** *ordIso-iff-ordLeq* **by** *blast*
qed
then show *?thesis* **using** *a2* **unfolding** *card-order-on-def* **by** *blast*
qed

lemma *lem-finwo-le-w*: *finite* $\alpha \Longrightarrow$ *Well-order* $\alpha \Longrightarrow \alpha <o$ *natLeq*
proof –
assume *a1*: *finite* α **and** *a2*: *Well-order* α
then have $|$*Field* $\alpha| =o\ \alpha$ **using** *lem-finwo-cardord* **by** (*metis card-of-Field-ordIso*)
moreover have *finite* (*Field* α) **using** *a1 finite-Field* **by** *fastforce*
moreover then have $|$*Field* $\alpha| <o$ *natLeq* **using** *finite-iff-ordLess-natLeq* **by** *blast*
ultimately show $\alpha <o$ *natLeq* **using** *ordIso-iff-ordLeq ordLeq-ordLess-trans* **by** *blast*
qed

lemma *lem-wolew-fin*: $\alpha <o$ *natLeq* $\Longrightarrow$ *finite* α
proof –
assume *a1*: $\alpha <o$ *natLeq*
then have *Well-order* α **using** *a1* **unfolding** *ordLess-def* **by** *blast*
then have $|$*Field* $\alpha| \leq o\ \alpha$ **using** *card-of-least*[*of Field* α α] **by** *blast*
then have $\neg$ (*natLeq* $\leq o$ $|$*Field* $\alpha|$) **using** *a1* **by** (*metis BNF-Cardinal-Order-Relation.ordLess-Field not-ordLeq-ordLess*)
then have *finite* (*Field* α) **using** *infinite-iff-natLeq-ordLeq* **by** *blast*
then show *finite* α **using** *finite-subset trancl-subset-Field2* **by** *fastforce*
qed

lemma *lem-wolew-nat*:
assumes *a1*: $\alpha <o$ *natLeq* **and** *a2*: n = *card* (*Field* α)
shows $\alpha =o$ (*natLeq-on* n)
proof –
have *b1*: *Well-order* α **using** *a1* **unfolding** *ordLess-def* **by** *blast*
have *b2*: *finite* α **using** *a1 lem-wolew-fin* **by** *blast*
then have *finite* (*Field* α) **using** *a1 finite-Field* **by** *fastforce*
then have $|$*Field* $\alpha| =o$ *natLeq-on* n **using** *a2 finite-imp-card-of-natLeq-on*[*of Field* α] **by** *blast*
moreover have $|$*Field* $\alpha| =o\ \alpha$ **using** *b1 b2 lem-finwo-cardord* **by** (*metis card-of-Field-ordIso*)
ultimately show *?thesis* **using** *ordIso-symmetric ordIso-transitive* **by** *blast*
qed

lemma *lem-cntset-enum*: $|A| =o$ *natLeq* $\Longrightarrow$ ($\exists$ f. $A = f$ ‘ (*UNIV*::*nat set*))
proof –
assume $|A| =o$ *natLeq*
moreover have $|$*UNIV*::*nat set*$| =o$ *natLeq* **using** *card-of-nat* **by** *blast*
ultimately have $|$*UNIV*::*nat set*$| =o$ $|A|$ **by** (*meson ordIso-iff-ordLeq ordIso-ordLeq-trans*)
then obtain f **where** *bij-betw* f (*UNIV*::*nat set*) A **using** *card-of-ordIso* **by** *blast*
then have $A = f$ ‘ (*UNIV*::*nat set*) **unfolding** *bij-betw-def* **by** *blast*
then show *?thesis* **by** *blast*
qed

lemma *lem-oord-int-card-le-inf*:
fixes α::*'U rel*
assumes ω*-ord* $\leq o\ \alpha$

shows $|\{ \gamma \in \mathcal{O}::'U\ rel\ set.\ \gamma <o\ \alpha \}| \le o\ \alpha$
proof −
  obtain $f::'U \Rightarrow 'U\ rel$ **where** $b1$: $f = (\lambda\ a.\ nord\ (Restr\ \alpha\ (underS\ \alpha\ a)))$ **by** *blast*
  have $\forall\ \gamma \in \mathcal{O}::'U\ rel\ set.\ \gamma <o\ \alpha \longrightarrow \gamma \in f\ `\ (Field\ \alpha)$
  proof (*intro ballI impI*)
    fix $\gamma::'U\ rel$
    assume $c1$: $\gamma \in \mathcal{O}$ **and** $c2$: $\gamma <o\ \alpha$
    have $\exists\ a \in Field\ \alpha.\ \gamma =o\ Restr\ \alpha\ (underS\ \alpha\ a)$
      using *c2 ordLess-iff-ordIso-Restr*[*of* $\alpha\ \gamma$] **unfolding** *ordLess-def* **by** *blast*
    then obtain a **where** $a \in Field\ \alpha \wedge \gamma =o\ Restr\ \alpha\ (underS\ \alpha\ a)$ **by** *blast*
    moreover then have $\gamma = f\ a$ **using** *c1 b1 lem-nord-eq lem-Onord* **by** *blast*
    ultimately show $\gamma \in f\ `\ (Field\ \alpha)$ **by** *blast*
  qed
  then have $\{ \gamma \in \mathcal{O}::'U\ rel\ set.\ \gamma<o\ \alpha \} \subseteq f\ `\ (Field\ \alpha)$ **by** *blast*
  then have $|\{ \gamma \in \mathcal{O}::'U\ rel\ set.\ \gamma <o\ \alpha \}| \le o\ |f\ `\ (Field\ \alpha)|$ **by** *simp*
  moreover have $|f\ `\ (Field\ \alpha)| \le o\ |Field\ \alpha|$ **by** *simp*
  ultimately have $|\{ \gamma \in \mathcal{O}::'U\ rel\ set.\ \gamma <o\ \alpha \}| \le o\ |Field\ \alpha|$ **using** *ordLeq-transitive* **by** *blast*
  moreover have $|Field\ \alpha| \le o\ \alpha$ **using** *assms* **by** *simp*
  ultimately show *?thesis* **using** *ordLeq-transitive* **by** *blast*
qed

lemma *lem-oord-card-le-int-inf*:
fixes $\alpha::'U\ rel$
assumes $a1$: *Card-order* α **and** $a2$: $\omega\text{-}ord \le o\ \alpha$
shows $\alpha \le o\ |\{ \gamma \in \mathcal{O}::'U\ rel\ set.\ \gamma <o\ \alpha \}|$
proof −
  obtain α' **where** $b0$: $\alpha' = |Field\ \alpha|$ **by** *blast*
  then have $b0'$: *Card-order* $\alpha' \wedge \alpha =o\ \alpha'$ **using** *a1 card-of-unique* **by** *simp*
  then have $b0''$: $\omega\text{-}ord \le o\ \alpha'$ **using** *a2 ordLeq-ordIso-trans* **by** *blast*
  obtain $f::'U \Rightarrow 'U\ rel$ **where** $b1$: $f = (\lambda\ a.\ Restr\ \alpha'\ (under\ \alpha'\ a))$ **by** *blast*
  have $b2$: *Well-order* α' **using** *b0* **by** *simp*
  have $b3$: $\forall\ a \in Field\ \alpha'.\ \forall\ b \in Field\ \alpha'.\ f\ a =o\ f\ b \longrightarrow a = b$
  proof (*intro ballI impI*)
    fix $a\ b$
    assume $d1$: $a \in Field\ \alpha'$ **and** $d2$: $b \in Field\ \alpha'$ **and** $f\ a =o\ f\ b$
    then have $d3$: $f\ a \le o\ f\ b \wedge f\ b \le o\ f\ a$ **using** *ordIso-iff-ordLeq* **by** *blast*
    obtain $A\ B$ **where** $d4$: $A = under\ \alpha'\ a \wedge B = under\ \alpha'\ b$ **by** *blast*
    have $d5$: *Well-order* α' **using** *b0* **by** *simp*
    moreover then have *wo-rel.ofilter* $\alpha'\ A \wedge$ *wo-rel.ofilter* $\alpha'\ B$
      using *d4 wo-rel-def wo-rel.under-ofilter*[*of* α'] **by** *blast*
    moreover have $Restr\ \alpha'\ A \le o\ Restr\ \alpha'\ B$ **and** $Restr\ \alpha'\ B \le o\ Restr\ \alpha'\ A$
      using *d3 d4 b1* **by** *blast+*
    ultimately have $A = B$ **using** *ofilter-subset-ordLeq*[*of* α'] **by** *blast*
    then have $under\ \alpha'\ a = under\ \alpha'\ b$ **using** *d4* **by** *blast*
    moreover have $(a,a) \in \alpha' \wedge (b,b) \in \alpha'$ **using** *d1 d2 d5*
      by (*metis preorder-on-def partial-order-on-def linear-order-on-def*
        well-order-on-def refl-on-def)

ultimately have $(a,b) \in \alpha' \wedge (b,a) \in \alpha'$ **unfolding** *under-def* **by** *blast*
then show $a = b$ **using** *d5*
by (*metis partial-order-on-def linear-order-on-def well-order-on-def antisym-def*)
qed
have *b4*: $\forall\ a \in Field\ \alpha'.\ f\ a <o\ \alpha'$
proof
fix *a*
assume *c1*: $a \in Field\ \alpha'$
have $under\ \alpha'\ a \subset Field\ \alpha'$
proof −
have $\neg$ *finite* α' **using** *b0''* *Field-natLeq finite-Field infinite-UNIV-nat ordLeq-finite-Field* **by** *metis*
then have $\neg$ *finite* (*Field* α') **using** *lem-fin-fl-rel* **by** *blast*
then obtain a' **where** $a' \in Field\ \alpha' \wedge a \neq a' \wedge (a,\ a') \in \alpha'$
using *c1 b0'* *infinite-Card-order-limit*[*of* α' *a*] **by** *blast*
moreover then have $(a',\ a) \notin \alpha'$ **using** *b2* **unfolding** *well-order-on-def linear-order-on-def partial-order-on-def antisym-def* **by** *blast*
ultimately show *?thesis* **unfolding** *under-def Field-def* **by** *blast*
qed
moreover have *ofilter* α' (*under* α' *a*)
using *b2 wo-rel.under-ofilter*[*of* α'] **unfolding** *wo-rel-def* **by** *blast*
ultimately show $f\ a <o\ \alpha'$ **unfolding** *b1* **using** *b2 ofilter-ordLess* **by** *blast*
qed
obtain *g* **where** *b5*: $g = nord \circ f$ **by** *blast*
have $\forall x \in Field\ \alpha'.\ \forall y \in Field\ \alpha'.\ g\ x = g\ y \longrightarrow x = y$
proof (*intro ballI impI*)
fix *x y*
assume *c1*: $x \in Field\ \alpha'$ **and** *c2*: $y \in Field\ \alpha'$ **and** $g\ x = g\ y$
then have *Well-order* (*f x*) $\wedge$ *Well-order* (*f y*) $\wedge$ *nord* (*f x*) = *nord* (*f y*)
using *b4 b5* **unfolding** *ordLess-def* **by** *simp*
then have $f\ x =o\ f\ y$ **using** *lem-nord-req* **by** *blast*
then show $x = y$ **using** *c1 c2 b3* **by** *blast*
qed
then have *inj-on g* (*Field* α') **unfolding** *inj-on-def* **by** *blast*
moreover have $\forall\ a \in Field\ \alpha'.\ g\ a \in \mathcal{O} \wedge g\ a <o\ \alpha'$
proof
fix *a*
assume $a \in Field\ \alpha'$
then have $f\ a <o\ \alpha'$ **using** *b4* **by** *blast*
then have *nord* (*f a*) $<o\ \alpha' \wedge$ *nord* (*f a*) $\in \mathcal{O}$ **using** *lem-nord-ls-l lem-nordO-ls-l* **by** *blast*
then show $g\ a \in \mathcal{O} \wedge g\ a <o\ \alpha'$ **using** *b5* **by** *simp*
qed
ultimately have $|Field\ \alpha'| \leq o\ |\{\gamma \in \mathcal{O}::'U\ rel\ set.\ \gamma <o\ \alpha'\}|$
using *card-of-ordLeq*[*of Field* α' $\{\gamma \in \mathcal{O}::'U\ rel\ set.\ \gamma <o\ \alpha'\}$] **by** *blast*
moreover have $\alpha =o\ |Field\ \alpha'|$ **using** *b0 a1* **by** *simp*
moreover have $\{\gamma \in \mathcal{O}::'U\ rel\ set.\ \gamma <o\ \alpha'\} = \{\gamma \in \mathcal{O}::'U\ rel\ set.\ \gamma <o\ \alpha\}$
using *b0'* **using** *ordIso-iff-ordLeq ordLess-ordLeq-trans* **by** *blast*

ultimately show *?thesis* **using** *ordIso-ordLeq-trans* **by** *simp*
qed

lemma *lem-ord-int-card-le-inf*:
fixes α::*'U rel* **and** *f* :: *'U rel* ⇒ *'a*
assumes ∀ α β. α =*o* β ⟶ *f* α = *f* β **and** ω*-ord* ≤*o* α
shows |*f* ' { γ::*'U rel*. γ <*o* α }| ≤*o* α
proof −
obtain *I* **where** *b1*: *I* = { γ ∈ $\mathcal{O}$::*'U rel set*. γ <*o* α } **by** *blast*
have *f*'{ γ::*'U rel*. γ <*o* α } ⊆ *f*'*I*
proof
fix *a*
assume *a* ∈ *f*'{ γ::*'U rel*. γ <*o* α }
then obtain γ **where** *a* = *f* γ ∧ γ <*o* α **by** *blast*
moreover then have *nord* γ =*o* γ ∧ *nord* γ ∈ *I*
using *b1 lem-nord-r lem-nord-ls-l lem-nordO-ls-l ordLess-def* **by** *blast*
ultimately have *a* = *f* (*nord* γ) ∧ *nord* γ ∈ *I* **using** *assms* **by** *metis*
then show *a* ∈ *f*'*I* **by** *blast*
qed
then have |*f*'{ γ::*'U rel*. γ <*o* α }| ≤*o* |*f*'*I*| **by** *simp*
moreover have |*f*'*I*| ≤*o* |*I*| **by** *simp*
moreover have |*I*| ≤*o* α **using** *b1 assms lem-oord-int-card-le-inf* **by** *blast*
ultimately show *?thesis* **using** *ordLeq-transitive* **by** *metis*
qed

lemma *lem-card-setcv-inf-stab*:
fixes α::*'U rel* **and** *A*::*'U set*
assumes *a1*: *Card-order* α **and** *a2*: ω*-ord* ≤*o* α **and** *a3*: |*A*| ≤*o* α
shows ∃ *f*::(*'U rel* ⇒ *'U*). *A* ⊆ *f* '{ γ::*'U rel*. γ <*o* α } ∧ (∀ γ*1* γ*2*. γ*1* =*o* γ*2* ⟶ *f* γ*1* = *f* γ*2*)
proof −
obtain *B* **where** *b1*: *B* = { γ ∈ $\mathcal{O}$::*'U rel set*. γ <*o* α } **by** *blast*
then have |*A*| ≤*o* |*B*|
using *a1 a2 a3 lem-oord-card-le-int-inf*[*of* α] *ordLeq-transitive* **by** *blast*
then obtain *g* **where** *b2*: *A* ⊆ *g* '*B* **by** (*metis card-of-ordLeq2 empty-subsetI order-refl*)
obtain *f* **where** *b3*: *f* = *g* ∘ *nord* **by** *blast*
have *A* ⊆ *f* '{ γ::*'U rel*. γ <*o* α }
proof
fix *a*
assume *a* ∈ *A*
then obtain γ::*'U rel* **where** γ ∈ $\mathcal{O}$ ∧ γ <*o* α ∧ *a* = *g* γ **using** *b1 b2* **by** *blast*
moreover then have *f* γ = *g* γ **using** *b3 lem-Onord* **by** *force*
ultimately show *a* ∈ *f* '{ γ::*'U rel*. γ <*o* α } **by** *force*
qed
moreover have ∀ γ*1* γ*2*. γ*1* =*o* γ*2* ⟶ *f* γ*1* = *f* γ*2* **using** *b3 lem-nord-eq* **by** *force*
ultimately show *?thesis* **by** *blast*
qed

```
lemma lem-jnfix-gen:
fixes I::'i set and leI::'i rel and L::'l set
  and t::'i×'l ⇒ 'i ⇒ 'n and jnN::'n ⇒ 'n ⇒ 'n
assumes a1:¬ finite L
    and a2: |L| <o |I|
    and a3: ∀α∈I. (α,α) ∈ leI
    and a4: ∀α∈I. ∀β∈I. ∀γ∈I. (α,β)∈leI ∧ (β,γ)∈leI ⟶ (α,γ)∈leI
    and a5: ∀α∈I. ∀β∈I. (α,β) ∈ leI ∨ (β,α) ∈ leI
    and a6: ∀β∈I. |{α∈I. (α,β) ∈ leI}| ≤o |L|
    and a7: ∀α∈I. ∃α'∈I. (α,α') ∈ leI ∧ (α',α) ∉ leI
shows ∃ h. ∀α∈I. ∀β∈I. ∀i∈L. ∀j∈L. ∃ γ∈I. (α,γ)∈leI ∧ (β,γ)∈leI ∧ (γ,α)∉leI
∧ (γ,β)∉leI
          ∧ h γ = jnN (t (α,i) γ) (t (β,j) γ)
proof −
  obtain inc where p1: inc = (λ α. SOME α'. α' ∈ I ∧ (α,α') ∈ leI ∧ (α',α) ∉
leI) by blast
  have p2: ⋀ α. α ∈ I ⟹ (inc α) ∈ I ∧ (α, inc α) ∈ leI ∧ (inc α, α) ∉ leI
  proof −
    fix α
    assume α ∈ I
    moreover obtain P where c1: P = (λ α'. α' ∈ I ∧ (α,α') ∈ leI ∧ (α',α) ∉
leI) by blast
    ultimately have ∃ α'. P α' using a7 by blast
    then have P (SOME x. P x) using someI-ex by metis
    moreover have inc α = (SOME x. P x) using c1 p1 by blast
    ultimately show (inc α) ∈ I ∧ (α,inc α) ∈ leI ∧ (inc α, α) ∉ leI using c1
by simp
  qed
  obtain mxI where m0: mxI = (λ α β. (if ((α,β) ∈ leI) then β else α)) by blast
  then have m1: ∀α∈I. ∀β∈I. mxI α β ∈ I by simp
  obtain maxI where b0: maxI = (λ α β. inc (mxI α β)) by blast
  have q1: ∀α∈I. ∀β∈I. maxI α β ∈ I using p2 b0 m0 by simp
  have q2: ∀α∈I. ∀β∈I. (α, maxI α β) ∈ leI ∧ (β, maxI α β) ∈ leI
  proof (intro ballI)
    fix α β
    assume c1: α ∈ I and c2: β ∈ I
    moreover then have c3: (α, mxI α β) ∈ leI ∧ (β, mxI α β) ∈ leI ∧ mxI α
β ∈ I
      using m0 m1 a5 by force+
    ultimately have (mxI α β, maxI α β) ∈ leI ∧ maxI α β ∈ I using b0 p2 by
blast
    then show (α, maxI α β) ∈ leI ∧ (β, maxI α β) ∈ leI using c1 c2 c3 a4 by
blast
  qed
  have q3: ∀ α∈I. ∀β∈I. ∀γ∈I. (maxI α β, γ) ∈ leI ⟶ (α,γ)∈leI ∧ (β,γ)∈leI
∧ (γ,α)∉leI ∧ (γ,β)∉leI
  proof (intro ballI impI)
    fix α β γ
```

assume *c1*: $\alpha{\in}I$ **and** *c2*: $\beta{\in}I$ **and** *c3*: $\gamma{\in}I$ **and** *c4*: $(maxI\ \alpha\ \beta,\ \gamma) \in leI$
moreover then have *c5*: $(mxI\ \alpha\ \beta,\ maxI\ \alpha\ \beta) \in leI \wedge maxI\ \alpha\ \beta \in I$
$\wedge\ (maxI\ \alpha\ \beta,\ mxI\ \alpha\ \beta) \notin leI \wedge mxI\ \alpha\ \beta \in I$ **using** *b0 p2 m1* **by** *blast*
ultimately have *c6*: $(mxI\ \alpha\ \beta,\ \gamma) \in leI$ **using** *a4* **by** *blast*
have $(\alpha,\gamma){\in}leI \wedge (\beta,\gamma){\in}leI$
proof (*cases* $(\alpha,\beta) \in leI$)
assume $(\alpha,\beta) \in leI$
moreover then have $(\beta,\gamma) \in leI$ **using** *m0 c6* **by** *simp*
ultimately show $(\alpha,\gamma){\in}leI \wedge (\beta,\gamma){\in}leI$ **using** *c1 c2 c3 a4* **by** *blast*
next
assume $(\alpha,\beta) \notin leI$
then have $(\beta,\alpha) \in leI \wedge (\alpha,\gamma) \in leI$ **using** *m0 c1 c2 c6 a5* **by** *force*
then show $(\alpha,\gamma){\in}leI \wedge (\beta,\gamma){\in}leI$ **using** *c1 c2 c3 a4* **by** *blast*
qed
moreover have $(\gamma,\alpha) \in leI \longrightarrow False$
proof
assume $(\gamma,\alpha) \in leI$
moreover have $(\alpha,\ mxI\ \alpha\ \beta) \in leI \wedge mxI\ \alpha\ \beta \in I$ **using** *c1 c2 m0 a5* **by**
force
ultimately have $(\gamma,\ mxI\ \alpha\ \beta) \in leI$ **using** *c1 c3 a4* **by** *blast*
then show *False* **using** *c3 c4 c5 a4* **by** *blast*
qed
moreover have $(\gamma,\beta) \in leI \longrightarrow False$
proof
assume $(\gamma,\beta) \in leI$
moreover have $(\beta,\ mxI\ \alpha\ \beta) \in leI \wedge mxI\ \alpha\ \beta \in I$ **using** *c1 c2 m0 a5* **by**
force
ultimately have $(\gamma,\ mxI\ \alpha\ \beta) \in leI$ **using** *c2 c3 a4* **by** *blast*
then show *False* **using** *c3 c4 c5 a4* **by** *blast*
qed
ultimately show $(\alpha,\gamma){\in}leI \wedge (\beta,\gamma){\in}leI \wedge (\gamma,\alpha){\notin}leI \wedge (\gamma,\beta){\notin}leI$ **by** *blast*
qed
have $\exists\ d.\ d`I = I{\times}L{\times}I$
proof −
have *c1*: $\neg$ *finite I* **using** *a1 a2* **by** (*metis card-of-ordLeq-infinite ordLess-imp-ordLeq*)
then have $I \neq \{\} \wedge L \neq \{\}$ **using** *a1* **by** *blast*
moreover then have $|I| \leq o\ |L{\times}I| \wedge |L{\times}I| =o\ |I| \wedge L \neq \{\}$
using *c1 a1 a2* **by** (*metis card-of-Times-infinite[of I L] ordLess-imp-ordLeq*
ordIso-iff-ordLeq)
moreover then have $\neg$ *finite* $(L{\times}I)$ **using** *c1 a1* **by** (*metis finite-cartesian-productD2*)
ultimately have $|I{\times}(L{\times}I)| \leq o\ |I|$
by (*metis card-of-Times-infinite[of L×I I] ordIso-transitive ordIso-iff-ordLeq*)
moreover have $I{\times}L{\times}I \neq \{\}$ **using** *c1 a1* **by** *force*
ultimately show *?thesis* **using** *card-of-ordLeq2[of I×(L×I) I]* **by** *blast*
qed
then obtain *d* **where** *b1*: $d`I = I{\times}(L{\times}I)$ **by** *blast*
obtain μ **where** *b2*: $\mu = (\lambda\ \gamma.\ SOME\ m.\ m`L = (\{\alpha{\in}I.\ (\alpha,\gamma){\in}leI\}{\times}L){\times}(\{\alpha{\in}I.$
$(\alpha,\gamma){\in}leI\}{\times}L)\ )$ **by** *blast*
have *b3*: $\bigwedge\gamma.\ \gamma \in I \Longrightarrow (\mu\ \gamma)`L = (\{\alpha{\in}I.\ (\alpha,\gamma){\in}leI\}{\times}L){\times}(\{\alpha{\in}I.\ (\alpha,\gamma){\in}leI\}{\times}L)$

proof $-$
fix γ
assume *c1*: $\gamma \in I$
obtain A **where** *c2*: $A = \{\alpha{\in}I.\ (\alpha,\gamma){\in}leI\}$ **by** *blast*
have *c3*: $A \neq \{\}$ **using** *c1 c2 a3* **unfolding** *refl-on-def* **by** *blast*
moreover have $L \neq \{\}$ **using** *a1* **by** *blast*
ultimately have $(A{\times}L){\times}(A{\times}L) \neq \{\}$ **using** *a1* **by** *simp*
moreover have $|(A{\times}L){\times}(A{\times}L)| \leq o\ |L|$
proof $-$
have $|A| \leq o\ |L|$ **using** *c1 c2 a6* **by** *blast*
then have $|A{\times}L| \leq o\ |L|$ **using** *c3 a1* **by** (*metis card-of-Times-infinite*[*of L A*] *ordIso-iff-ordLeq*)
moreover have $\neg$ *finite* $(A{\times}L)$ **using** *c3 a1* **by** (*metis finite-cartesian-productD2*)
ultimately show *?thesis*
by (*metis card-of-Times-same-infinite*[*of* $A{\times}L$] *ordIso-iff-ordLeq ordLeq-transitive*)
qed
ultimately have $\exists\, m.\ m\text{‘}L = (\{\alpha{\in}I.\ (\alpha,\gamma){\in}leI\}{\times}L){\times}(\{\alpha{\in}I.\ (\alpha,\gamma){\in}leI\}{\times}L)$
using *c2 card-of-ordLeq2*[*of* $(A{\times}L){\times}(A{\times}L)$ *L*] **by** *blast*
then show $(\mu\ \gamma)\text{‘}L = (\{\alpha{\in}I.\ (\alpha,\gamma){\in}leI\}{\times}L){\times}(\{\alpha{\in}I.\ (\alpha,\gamma){\in}leI\}{\times}L)$
using *b2 someI-ex*[*of* $\lambda\ m.\ m\text{‘}L = (\{\alpha{\in}I.\ (\alpha,\gamma){\in}leI\}{\times}L){\times}(\{\alpha{\in}I.\ (\alpha,\gamma){\in}leI\}{\times}L)$] **by** *blast*
qed
obtain φ **where** *b4*: $\varphi = (\lambda\ x.\ \mu\ (fst\ (d\ x))\ (fst\ (snd\ (d\ x))))$ **by** *blast*
obtain h **where** *b5*: $h = (\lambda\ x.\ jnN\ (t\ (fst\ (\varphi\ x))\ x)\ (t\ (snd\ (\varphi\ x))\ x))$ **by** *blast*
have $\forall\,\alpha{\in}I.\ \forall\,\beta{\in}I.\ \forall\, i{\in}L.\ \forall\, j{\in}L.\ \exists\ \gamma{\in}I.$
$(maxI\ \alpha\ \beta,\ \gamma) \in leI \wedge h\ \gamma = jnN\ (t\ (\alpha,i)\ \gamma)\ (t\ (\beta,j)\ \gamma)$
proof (*intro ballI*)
fix $\alpha\ \beta\ i\ j$
assume *c1*: $\alpha \in I$ **and** *c2*: $\beta \in I$ **and** *c3*: $i \in L$ **and** *c4*: $j \in L$
obtain D **where** *c5*: $D = (\{\alpha' \in I.\ (\alpha',\ maxI\ \alpha\ \beta) \in leI\} \times L) \times \{\alpha' \in I.\ (\alpha',\ maxI\ \alpha\ \beta) \in leI\} \times L$ **by** *blast*
have *c6*: $maxI\ \alpha\ \beta \in I$ **using** *c1 c2 q1* **by** *blast*
have $\alpha \in \{\alpha' \in I.\ (\alpha',\ maxI\ \alpha\ \beta) \in leI\}$ **using** *c1 c2 q2* **by** *blast*
moreover have $\beta \in \{\alpha' \in I.\ (\alpha',\ maxI\ \alpha\ \beta) \in leI\}$ **using** *c1 c2 q2* **by** *blast*
ultimately have $((\alpha,i),(\beta,j)) \in D$ **using** *c3 c4 c5* **by** *blast*
moreover have $\mu\ (maxI\ \alpha\ \beta)\ \text{‘}\ L = D$ **using** *c5 c6 b3*[*of maxI* α β] **by** *blast*
ultimately obtain v **where** *c7*: $v \in L \wedge (\mu\ (maxI\ \alpha\ \beta))\ v = ((\alpha,i),(\beta,j))$ **by** *force*
obtain A **where** *c8*: $A = \{maxI\ \alpha\ \beta\} \times (\{v\} \times I)$ **by** *blast*
then have $A \subseteq I \times L \times I$ **using** *c6 c7* **by** *blast*
then have $\forall\, a{\in}A.\ \exists\ x{\in}I.\ d\ x = a$ **using** *b1* **by** (*metis imageE set-rev-mp*)
moreover obtain X **where** *c9*: $X = \{\ x{\in}I.\ d\ x \in A\ \}$ **by** *blast*
ultimately have $A = d\ \text{‘}\ X$ **by** *force*
then have $|A| \leq o\ |X|$ **by** *simp*
moreover have $|I| =o\ |A|$
proof $-$
obtain f **where** $f = (\lambda\ x{::}'i.\ (maxI\ \alpha\ \beta,\ v,\ x))$ **by** *blast*
then have *bij-betw f I A* **using** *c8* **unfolding** *bij-betw-def inj-on-def* **by** *force*
then show $|I| =o\ |A|$ **using** *card-of-ordIsoI*[*of f I A*] **by** *blast*

qed
ultimately have *c10*: $|L| <o\ |X|$ **using** *a2* **by** (*metis ordLess-ordIso-trans ordLess-ordLeq-trans*)
have $\forall\, y{\in}I.\ X \subseteq \{x{\in}I.\ (x,y) \in leI\} \longrightarrow False$
proof (*intro ballI impI*)
fix *y*
assume $y \in I$ **and** $X \subseteq \{x{\in}I.\ (x,y) \in leI\}$
then have $y \in I \wedge X \subseteq \{x{\in}I.\ (x,y) \in leI\}$ **by** *blast*
moreover then have $|\{x{\in}I.\ (x,y) \in leI\}| \leq o\ |L|$ **using** *a6* **by** *blast*
ultimately have $|X| \leq o\ |L|$ **using** *card-of-mono1 ordLeq-transitive* **by** *blast*
then show *False* **using** *c10* **by** (*metis not-ordLeq-ordLess*)
qed
then obtain γ **where** *c11*: $\gamma \in X \wedge (\gamma,\ maxI\ \alpha\ \beta) \notin leI$ **using** *c6 c9* **by** *blast*
then obtain *w* **where** *c12*: $\gamma \in I \wedge\ d\ \gamma = (maxI\ \alpha\ \beta,\ v,\ w)$ **using** *c8 c9* **by** *blast*
moreover have $(maxI\ \alpha\ \beta,\ \gamma) \in leI$ **using** *c11 c12 c6 a5* **by** *blast*
moreover have $h\ \gamma = jnN\ (t\ (\alpha,i)\ \gamma)\ (t\ (\beta,j)\ \gamma)$
proof −
have $\varphi\ \gamma = \mu\ (fst\ (d\ \gamma))\ (fst\ (snd\ (d\ \gamma)))$ **using** *b4* **by** *blast*
then have $\varphi\ \gamma = \mu\ (maxI\ \alpha\ \beta)\ v$ **using** *c12* **by** *simp*
then have $\varphi\ \gamma = ((\alpha,i),(\beta,j))$ **using** *c7* **by** *simp*
moreover have $h\ \gamma = jnN\ (t\ (fst\ (\varphi\ \gamma))\ \gamma)\ (t\ (snd\ (\varphi\ \gamma))\ \gamma)$ **using** *b5* **by** *blast*
ultimately show $h\ \gamma = jnN\ (t\ (\alpha,i)\ \gamma)\ (t\ (\beta,j)\ \gamma)$ **by** *simp*
qed
ultimately show $\exists\ \gamma{\in}I.\ (maxI\ \alpha\ \beta,\ \gamma) \in leI \wedge h\ \gamma = jnN\ (t\ (\alpha,i)\ \gamma)\ (t\ (\beta,j)\ \gamma)$ **by** *blast*
qed
then show *?thesis* **using** *q3* **by** *blast*
qed

lemma *lem-jnfix-card*:
fixes $\kappa{::}'U\ rel$ **and** $L{::}'l\ set$ **and** $t{::}('U\ rel){\times}'l \Rightarrow\ 'U\ rel \Rightarrow\ 'n$ **and** $jnN{::}'n \Rightarrow\ 'n \Rightarrow\ 'n$
and $S{::}'U\ rel\ set$
assumes *a1*: *Card-order* κ **and** *a2*: $\neg$ *finite* L **and** *a3*: $|L| <o\ \kappa$
and *a4*: $\forall\ \alpha \in S.\ |Field\ \alpha| \leq o\ |L|$
and *a5*: $S \subseteq \mathcal{O}$ **and** *a6*: $|\{\alpha \in \mathcal{O}{::}'U\ rel\ set.\ \alpha <o\ \kappa\}| \leq o\ |S|$
and *a7*: $\forall\ \alpha \in S.\ \exists\ \beta \in S.\ \alpha <o\ \beta$
shows $\exists\ h.\ \forall\ \alpha \in S.\ \forall\ \beta \in S.\ \forall\, i{\in}L.\ \forall\, j{\in}L.$
$(\exists\ \gamma \in S.\ \alpha <o\ \gamma \wedge \beta <o\ \gamma \wedge h\ \gamma = jnN\ (t\ (\alpha,i)\ \gamma)\ (t\ (\beta,j)\ \gamma)\ )$
proof −
obtain $I{::}('U\ rel)\ set$ **where** *c1*: $I = S$ **by** *blast*
obtain $leI{::}'U\ rel\ rel$ **where** *c2*: $leI = oord$ **by** *blast*
have $\neg$ *finite* L **using** *a2* **by** *blast*
moreover have $|L| <o\ |I|$
proof −
have $\omega\text{-}ord \leq o\ |L|$ **using** *a2* **by** (*metis infinite-iff-natLeq-ordLeq*)
then have $\omega\text{-}ord \leq o\ \kappa$ **using** *a3* **by** (*metis ordLeq-ordLess-trans ordLess-imp-ordLeq*)

then obtain f::$'U$ *rel* $\Rightarrow$ $'U$ **where**
d1: *Field* $\kappa \subseteq f$ ‘ $\{\gamma.\ \gamma <o\ \kappa\}$ **and** *d2*: $\forall \gamma 1\ \gamma 2.\ \gamma 1 =o\ \gamma 2 \longrightarrow f\ \gamma 1 = f\ \gamma 2$
using *a1 lem-card-setcv-inf-stab*[*of* κ *Field* κ] **by** (*metis card-of-Field-ordIso ordIso-imp-ordLeq*)
then have $|$*Field* $\kappa| \leq o\ |f$ ‘ $\{\gamma.\ \gamma <o\ \kappa\}|$ **by** *simp*
then have $\kappa \leq o\ |f$ ‘ $\{\gamma.\ \gamma <o\ \kappa\}|$ **using** *a1*
by (*metis card-of-Field-ordIso ordIso-imp-ordLeq ordLeq-transitive ordIso-symmetric*)
moreover have $|f$ ‘ $\{\gamma.\ \gamma <o\ \kappa\}| \leq o\ |\{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}|$
proof −
have $\kappa \neq \{\}$ **using** *a2 a3*
using *lem-cardord-emp* **by** (*metis Field-empty card-of-Field-ordIso card-of-empty not-ordLess-ordIso ordLeq-ordLess-trans*)
then have $(\{\}::'U\ rel) <o\ \kappa$ **using** *a1*
by (*metis ozero-def iso-ozero-empty card-order-on-well-order-on ordIso-symmetric ordLeq-iff-ordLess-or-ordIso ozero-ordLeq*)
then have *e1*: f ‘ $\{\gamma.\ \gamma <o\ \kappa\} \neq \{\}$ **by** *blast*
moreover have f ‘ $\{\gamma.\ \gamma <o\ \kappa\} \subseteq f$ ‘ $\{\alpha \in \mathcal{O}.\ \alpha <o\ \kappa\}$
proof
fix y
assume $y \in f$ ‘ $\{\gamma.\ \gamma <o\ \kappa\}$
then obtain $\gamma\ \alpha$ **where** *f1*: $\gamma <o\ \kappa \wedge y = f\ \gamma \wedge \alpha =$ *nord* γ **by** *blast*
moreover then have *f2*: $\alpha \in \mathcal{O} \wedge \alpha =o\ \gamma$ **using** *lem-nord-r* **unfolding** $\mathcal{O}$*-def ordLess-def* **by** *blast*
ultimately have $\alpha <o\ \kappa$ **using** *d2 ordIso-ordLess-trans* **by** *blast*
moreover have $y = f\ \alpha$ **using** *d2 f1 f2* **by** *fastforce*
ultimately show $y \in f$ ‘ $\{\alpha \in \mathcal{O}.\ \alpha <o\ \kappa\}$ **using** *f2* **by** *blast*
qed
ultimately have f ‘ $\{\alpha \in \mathcal{O}.\ \alpha <o\ \kappa\} = f$ ‘ $\{\gamma.\ \gamma <o\ \kappa\}$ **by** *blast*
then show *?thesis* **using** *e1 card-of-ordLeq2*[*of* f ‘ $\{\gamma.\ \gamma <o\ \kappa\}$ $\{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}$] **by** *blast*
qed
ultimately have $\kappa \leq o\ |\{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}|$ **using** *ordLeq-transitive* **by** *blast*
moreover have $I = S$ **using** *c1* **by** *blast*
moreover then have $|\{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}| \leq o\ |I|$ **using** *a6* **by** *blast*
ultimately have $\kappa \leq o\ |I|$ **using** *c1* **using** *ordLeq-transitive* **by** *blast*
then show *?thesis* **using** *a3* **by** (*metis ordLess-ordLeq-trans*)
qed
moreover have $\forall \alpha \in I.\ (\alpha,\alpha) \in leI$
using *c1 c2 a5 lem-fld-oord lem-oord-wo* **unfolding** *well-order-on-def linear-order-on-def*
partial-order-on-def preorder-on-def refl-on-def **by** *blast*
moreover have $\forall \alpha \in I.\ \forall \beta \in I.\ \forall \gamma \in I.\ (\alpha,\beta) \in leI \wedge (\beta,\gamma) \in leI \longrightarrow (\alpha,\gamma) \in leI$
using *c2 lem-oord-wo* **unfolding** *well-order-on-def linear-order-on-def*
partial-order-on-def preorder-on-def trans-def **by** *blast*
moreover have $\forall \alpha \in \mathcal{O}.\ \forall \beta \in \mathcal{O}.\ (\alpha,\beta) \in leI \vee (\beta,\alpha) \in leI$
using *c1 c2 lem-fld-oord lem-oord-wo* **unfolding** *well-order-on-def linear-order-on-def total-on-def*
partial-order-on-def preorder-on-def refl-on-def **by** *metis*

moreover then have ∀α∈I. ∀β∈I. (α,β) ∈ leI ∨ (β,α) ∈ leI **using** c1 a5 **by** blast
moreover have ∀β∈I. |{α∈I. (α,β) ∈ leI}| ≤o |L|
proof
fix β
assume d1: β ∈ I
show |{α∈I. (α,β) ∈ leI}| ≤o |L|
proof (cases ω-ord ≤o β)
assume e1: ω-ord ≤o β
obtain C **where** e2: C = nord ‘ {α::'U rel. α <o β} **by** blast
have {α∈I. (α,β) ∈ leI} ⊆ C ∪ {β}
proof
fix γ
assume γ ∈ {α∈I. (α,β) ∈ leI}
then have γ ∈ 𝒪 ∧ (γ <o β ∨ γ = β)
using c2 lem-Oeq **unfolding** oord-def **using** ordLeq-iff-ordLess-or-ordIso **by** blast
moreover then have γ = nord γ **using** lem-Onord **by** blast
ultimately show γ ∈ C ∪ {β} **using** e2 **by** blast
qed
moreover have |C ∪ {β}| ≤o β
proof (cases finite C)
assume finite C
then have finite (C ∪ {β}) **by** blast
then have |C ∪ {β}| <o ω-ord **using** finite-iff-ordLess-natLeq **by** blast
then show ?thesis **using** e1 ordLess-ordLeq-trans ordLess-imp-ordLeq **by** blast
next
assume ¬ finite C
then have |C ∪ {β}| =o |C| **by** (metis card-of-singl-ordLeq finite.simps card-of-Un-infinite)
then show ?thesis **using** e1 e2 lem-nord-eq lem-ord-int-card-le-inf[of nord β] ordIso-ordLeq-trans **by** blast
qed
ultimately have |{α∈I. (α,β) ∈ leI}| ≤o β **by** (meson card-of-mono1 ordLeq-transitive)
moreover have ⋀ A::'U rel set. |A| ≤o β ⟹ |A| ≤o |Field β|
by (metis Field-card-of card-of-mono1 internalize-card-of-ordLeq)
ultimately have |{α∈I. (α,β) ∈ leI}| ≤o |Field β| **by** blast
moreover have |Field β| ≤o |L| **using** d1 c1 a4 **by** blast
ultimately show |{α∈I. (α,β) ∈ leI}| ≤o |L| **using** ordLeq-transitive **by** blast
next
assume ¬ ω-ord ≤o β
then have e1: β <o ω-ord **using** d1 c1 a5 **using** lem-Owo Field-natLeq natLeq-well-order-on **by** force
then have e2: β =o natLeq-on (card (Field β)) **using** lem-wolew-nat **by** blast
obtain A **where** e3: A = { n. n ≤ card (Field β) } **by** blast
obtain f **where** e4: f = (λn::nat. SOME α. α ∈ I ∧ α <o ω-ord ∧ card

$(Field\ \alpha) = n)$ **by** *blast*

have $\{\alpha{\in}I.\ (\alpha,\beta) \in leI\} \subseteq f\ `\ A$

proof

fix γ

assume *f1*: $\gamma \in \{\alpha{\in}I.\ (\alpha,\beta) \in leI\}$

then have *f2*: $\gamma \leq o\ \beta$ **using** *c2 oord-def* **by** *blast*

then have *f3*: $\gamma <o\ \omega\text{-}ord$ **using** *e1 ordLeq-ordLess-trans* **by** *blast*

then have *f4*: $\gamma =o\ natLeq\text{-}on\ (card\ (Field\ \gamma))$ **using** *lem-wolew-nat* **by** *blast*

then have $natLeq\text{-}on\ (card\ (Field\ \gamma)) \leq o\ natLeq\text{-}on\ (card\ (Field\ \beta))$

using *f2 e2* **by** (*meson ordIso-iff-ordLeq ordLeq-transitive*)

then have *f5*: $\gamma \in I \wedge card\ (Field\ \gamma) \in A$ **using** *f1 e3 natLeq-on-ordLeq-less-eq* **by** *blast*

moreover obtain γ' **where** *f6*: $\gamma' = f\ (card\ (Field\ \gamma))$ **by** *blast*

ultimately have $\gamma' \in I \wedge \gamma' <o\ \omega\text{-}ord \wedge card\ (Field\ \gamma') = card\ (Field\ \gamma)$

using *f3 e4 someI-ex*[*of* $\lambda\ \alpha.\ \alpha \in I \wedge \alpha <o\ \omega\text{-}ord \wedge card\ (Field\ \alpha) = card\ (Field\ \gamma)$] **by** *blast*

moreover then have $\gamma' =o\ natLeq\text{-}on\ (card\ (Field\ \gamma))$ **using** *lem-wolew-nat* **by** *force*

ultimately have $\gamma \in \mathcal{O} \wedge \gamma' \in \mathcal{O} \wedge \gamma' =o\ \gamma$ **using** *f1 f4 c1 a5 ordIso-symmetric ordIso-transitive* **by** *blast*

then have $\gamma' = \gamma$ **using** *lem-Oeq* **by** *blast*

moreover have $\gamma' \in f\ `\ A$ **using** *f5 f6* **by** *blast*

ultimately show $\gamma \in f\ `\ A$ **by** *blast*

qed

then have *finite* $\{\alpha{\in}I.\ (\alpha,\beta) \in leI\}$ **using** *e3 finite-subset* **by** *blast*

then show $|\{\alpha{\in}I.\ (\alpha,\beta) \in leI\}| \leq o\ |L|$ **using** *a2 ordLess-imp-ordLeq* **by** *force*

qed

qed

moreover have $\forall\alpha{\in}I.\ \exists\alpha'{\in}I.\ (\alpha,\alpha') \in leI \wedge (\alpha',\alpha) \notin leI$

proof

fix α

assume $\alpha \in I$

then obtain α' **where** *d1*: $\alpha \in S \wedge \alpha' \in S \wedge \alpha <o\ \alpha'$ **using** *c1 a7* **by** *blast*

then have *d2*: $\alpha \leq o\ \alpha' \wedge \alpha \in \mathcal{O} \wedge \alpha' \in \mathcal{O}$ **using** *a5 ordLess-imp-ordLeq* **by** *blast*

then have $\alpha' \in I \wedge (\alpha,\alpha') \in leI$ **using** *d1 c1 c2* **unfolding** *oord-def* **by** *blast*

moreover have $(\alpha',\alpha) \in leI \longrightarrow False$

proof

assume *e1*: $(\alpha',\alpha) \in leI$

then have $\alpha' \leq o\ \alpha$ **using** *c2* **unfolding** *oord-def* **by** *blast*

then have $\alpha' = \alpha$ **using** *d2 lem-Oeq ordIso-iff-ordLeq* **by** *blast*

then show *False* **using** *d1 ordLess-irreflexive* **by** *blast*

qed

ultimately show $\exists\alpha'{\in}I.\ (\alpha,\alpha') \in leI \wedge (\alpha',\alpha) \notin leI$ **by** *blast*

qed

ultimately obtain h **where**

c3: $\forall\alpha{\in}I.\ \forall\beta{\in}I.\ \forall i{\in}L.\ \forall j{\in}L.\ \exists\ \gamma{\in}I.$

$(\alpha,\gamma){\in}leI \wedge (\beta,\gamma) \in leI \wedge (\gamma,\alpha){\notin}leI \wedge (\gamma,\beta){\notin}leI \wedge h\ \gamma = jnN\ (t\ (\alpha,i)\ \gamma)$

$(t\ (\beta,j)\ \gamma)$
  using *lem-jnfix-gen*[*of L I leI jnN t*] **by** *blast*
 have $\forall\ \alpha \in S.\ \forall\ \beta \in S.\ \forall i{\in}L.\ \forall j{\in}L.$
   $(\exists\ \gamma \in S.\ \alpha <o\ \gamma \wedge \beta <o\ \gamma \wedge h\ \gamma = jnN\ (t\ (\alpha,i)\ \gamma)\ (t\ (\beta,j)\ \gamma))$
 proof (*intro allI ballI impI*)
  fix α::$'U$ *rel* **and** i::$'l$ **and** β::$'U$ *rel* **and** j::$'l$
  assume *d2*: $i \in L$ **and** *d3*: $j \in L$ **and** $\alpha \in S$ **and** $\beta \in S$
  then have *d4*: $\alpha \in I \wedge \beta \in I$ **using** *c1 a5* **by** *blast*
  then obtain γ **where** $\gamma \in I$ **and** $(\alpha,\gamma) \in leI \wedge (\beta,\gamma) \in leI$ **and** $(\gamma,\alpha) \notin leI \wedge (\gamma,\beta) \notin leI$
   and *d6*: $h\ \gamma = jnN\ (t\ (\alpha,i)\ \gamma)\ (t\ (\beta,j)\ \gamma)$ **using** *d2 d3 c3* **by** *blast*
  then have $\gamma \in \mathcal{O} \cap S \wedge \alpha <o\ \gamma \wedge \beta <o\ \gamma$
   using *d4 c1 c2 a5 lem-Oeq* **unfolding** *oord-def*
    by (*smt ordLeq-iff-ordLess-or-ordIso subsetCE Int-iff*)
  moreover have $h\ \gamma = jnN\ (t\ (\alpha,i)\ \gamma)\ (t\ (\beta,j)\ \gamma)$ **using** *d2 d3 d6* **by** *blast*
  ultimately show $\exists\ \gamma \in S.\ \alpha <o\ \gamma \wedge \beta <o\ \gamma \wedge h\ \gamma = jnN\ (t\ (\alpha,i)\ \gamma)\ (t\ (\beta,j)\ \gamma)$ **by** *blast*
 qed
 then show *?thesis* **by** *blast*
qed

lemma *lem-cardsuc-ls-fldcard*:
fixes κ::$'a$ *rel* **and** α::$'b$ *rel*
assumes *a1*: *Card-order* κ **and** *a2*: $\alpha <o$ *cardSuc* κ
shows $|Field\ \alpha| \leq o\ \kappa$
proof −
 have $\kappa <o\ |Field\ \alpha| \longrightarrow False$
 proof
  assume $\kappa <o\ |Field\ \alpha|$
  moreover have *Card-order* $|Field\ \alpha|$ **by** *simp*
  ultimately have *cardSuc* $\kappa \leq o\ |Field\ \alpha|$ **using** *a1 cardSuc-least* **by** *blast*
  moreover have $|Field\ \alpha| \leq o\ \alpha$ **using** *a2* **by** *simp*
  ultimately have *cardSuc* $\kappa \leq o\ \alpha$ **using** *ordLeq-transitive* **by** *blast*
  then show *False* **using** *a2 not-ordLeq-ordLess* **by** *blast*
 qed
 then show $|Field\ \alpha| \leq o\ \kappa$ **using** *a1* **by** *simp*
qed

lemma *lem-jnfix-cardsuc*:
fixes L::$'l$ *set* **and** κ::$'U$ *rel* **and** t::$('U\ rel) \times 'l \Rightarrow 'U\ rel \Rightarrow 'n$ **and** jnN::$'n \Rightarrow 'n \Rightarrow 'n$
 and S::$'U$ *rel set*
assumes *a1*: $\neg$ *finite* L **and** *a2*: $\kappa =o$ *cardSuc* $|L|$
  and *a3*: $S \subseteq \{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}$ **and** *a4*: $|\{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}| \leq o\ |S|$
  and *a5*: $\forall\ \alpha \in S.\ \exists\ \beta \in S.\ \alpha <o\ \beta$
shows $\exists\ h.\ \forall\ \alpha \in S.\ \forall\ \beta \in S.\ \forall i{\in}L.\ \forall j{\in}L.$
   $(\exists\ \gamma \in S.\ \alpha <o\ \gamma \wedge \beta <o\ \gamma \wedge h\ \gamma = jnN\ (t\ (\alpha,i)\ \gamma)\ (t\ (\beta,j)\ \gamma)\ )$
proof −

have *Card-order* κ **using** *a2* **by** (*metis Card-order-ordIso cardSuc-Card-order card-of-Card-order*)
moreover have $|L|$ $<o$ κ **using** *a2 cardSuc-greater*[*of* $|L|$]
by (*metis Field-card-of card-of-card-order-on ordIso-iff-ordLeq ordLess-ordLeq-trans*)
moreover have $\forall\, \alpha$::*'U rel*. α $<o$ κ $\longrightarrow$ $|$*Field* $\alpha|$ $\leq o$ $|L|$
using *a2* **using** *lem-cardsuc-ls-fldcard ordLess-ordIso-trans* **by** *force*
ultimately show *?thesis* **using** *a1 a3 a4 a5 lem-jnfix-card*[*of* κ *L S jnN t*] **by** *blast*
qed

lemma *lem-Relprop-cl-ccr*:
fixes *r*::*'U rel*
shows *Conelike r* $\Longrightarrow$ *CCR r*
unfolding *CCR-def Conelike-def* **by** *fastforce*

lemma *lem-Relprop-ccr-confl*:
fixes *r*::*'U rel*
shows *CCR r* $\Longrightarrow$ *confl-rel r*
using *lem-rtr-field*[*of - - r*] **unfolding** *CCR-def confl-rel-def* **by** *blast*

lemma *lem-Relprop-fin-ccr*:
fixes *r*::*'U rel*
shows *finite r* $\Longrightarrow$ *CCR r* = *Conelike r*
proof −
assume *a1*: *finite r*
have $r \neq \{\}$ $\wedge$ *CCR r* $\longrightarrow$ *Conelike r*
proof
assume *b1*: $r \neq \{\}$ $\wedge$ *CCR r*
have *b2*: *finite* (*Field r*) **using** *a1 finite-Field* **by** *fastforce*
have $\exists$ *xm* $\in$ *Field r*. $\forall$ *x* $\in$ *Field r*. (*x*, *xm*) $\in$ $r\hat{}*$
proof −
have $\{\} \subseteq$ *Field r* $\longrightarrow$ ($\exists$ *xm* $\in$ *Field r*. $\forall$ *x* $\in$ $\{\}$. (*x*, *xm*) $\in$ $r\hat{}*$) **using** *b1 Field-def* **by** *fastforce*
moreover have $\bigwedge$ *x F*. *finite F* $\Longrightarrow$ *x* $\notin$ *F* $\Longrightarrow$
F $\subseteq$ *Field r* $\longrightarrow$ ($\exists$ *xm* $\in$ *Field r*. $\forall$ *x* $\in$ *F*. (*x*, *xm*) $\in$ $r\hat{}*$) $\Longrightarrow$
insert x F $\subseteq$ *Field r* $\longrightarrow$ ($\exists$ *xm* $\in$ *Field r*. $\forall$ *x* $\in$ *insert x F*. (*x*, *xm*) $\in$ $r\hat{}*$)
proof
fix *x F*
assume *c1*: *finite F* **and** *c2*: *x* $\notin$ *F* **and** *c3*: *F* $\subseteq$ *Field r* $\longrightarrow$ ($\exists$ *xm*$\in$*Field r*. $\forall$ *x*$\in$*F*. (*x*, *xm*) $\in$ $r\hat{}*$)
and *c4*: *insert x F* $\subseteq$ *Field r*
then obtain *xm* **where** *c5*: *xm* $\in$ *Field r* $\wedge$ ($\forall$ *y*$\in$*F*. (*y*, *xm*) $\in$ $r\hat{}*$) **by** *blast*
then obtain *xm'* **where** *xm'* $\in$ *Field r* $\wedge$ (*x*, *xm'*) $\in$ $r\hat{}*$ $\wedge$ (*xm*, *xm'*) $\in$ $r\hat{}*$
using *b1 c4* **unfolding** *CCR-def* **by** *blast*
moreover then have $\forall$ *y*$\in$*insert x F*. (*y*, *xm'*) $\in$ $r\hat{}*$ **using** *c5* **by** *force*
ultimately show $\exists$ *xm*$\in$*Field r*. $\forall$ *x*$\in$*insert x F*. (*x*, *xm*) $\in$ $r\hat{}*$ **by** *blast*
qed

ultimately have (∃ *xm* ∈ *Field r*. ∀ *x* ∈ *Field r*. (*x*, *xm*) ∈ *r*^*)
using *b2 finite-induct*[*of Field r* λ *A′*. *A′* ⊆ *Field r* ⟶ (∃ *xm* ∈ *Field r*. ∀ *x* ∈ *A′*. (*x*, *xm*) ∈ *r*^*)] **by** *simp*
then show ∃ *xm* ∈ *Field r*. ∀ *x* ∈ *Field r*. (*x*, *xm*) ∈ *r*^* **by** *blast*
qed
then show *Conelike r* **using** *a1 b1* **unfolding** *Conelike-def* **by** *blast*
qed
then show *CCR r* = *Conelike r* **using** *lem-Relprop-cl-ccr* **unfolding** *Conelike-def* **by** *blast*
qed

lemma *lem-Relprop-ccr-ch-un*:
fixes *S*::*′U rel set*
assumes *a1*: ∀ *s*∈*S*. *CCR s* **and** *a2*: ∀ *s1*∈*S*. ∀ *s2*∈*S*. *s1* ⊆ *s2* ∨ *s2* ⊆ *s1*
shows *CCR* (⋃ *S*)
proof −
have ∀ *a*∈*Field* (⋃ *S*). ∀ *b*∈*Field* (⋃ *S*). ∃ *c*∈*Field* (⋃ *S*). (*a*, *c*) ∈ (⋃ *S*)^* ∧ (*b*, *c*) ∈ (⋃ *S*)^*
proof (*intro ballI*)
fix *a b*
assume *c1*: *a* ∈ *Field* (⋃ *S*) **and** *c2*: *b* ∈ *Field* (⋃ *S*)
then obtain *s1 s2* **where** *c3*: *s1* ∈ *S* ∧ *a* ∈ *Field s1* **and** *c4*: *s2* ∈ *S* ∧ *b* ∈ *Field s2*
unfolding *Field-def* **by** *blast*
show ∃ *c*∈*Field* (⋃ *S*). (*a*,*c*) ∈ (⋃ *S*)^* ∧ (*b*,*c*) ∈ (⋃ *S*)^*
proof (*cases s1* ⊆ *s2*)
assume *s1* ⊆ *s2*
then have *a* ∈ *Field s2* **using** *c3* **unfolding** *Field-def* **by** *blast*
then obtain *c* **where** *c* ∈ *Field s2* ∧ (*a*,*c*) ∈ *s2*^* ∧ (*b*,*c*) ∈ *s2*^*
using *a1 c4* **unfolding** *CCR-def* **by** *force*
moreover then have *c* ∈ *Field* (⋃ *S*) **using** *c4* **unfolding** *Field-def* **by** *blast*
moreover have *s2*^* ⊆ (⋃ *S*)^* **using** *c4 Transitive-Closure.rtrancl-mono*[*of s2* ⋃ *S*] **by** *blast*
ultimately show ∃ *c*∈*Field* (⋃ *S*). (*a*,*c*) ∈ (⋃ *S*)^* ∧ (*b*,*c*) ∈ (⋃ *S*)^* **by** *blast*
next
assume ¬ *s1* ⊆ *s2*
then have *s2* ⊆ *s1* **using** *a2 c3 c4* **by** *blast*
then have *b* ∈ *Field s1* **using** *c4* **unfolding** *Field-def* **by** *blast*
then obtain *c* **where** *c* ∈ *Field s1* ∧ (*a*,*c*) ∈ *s1*^* ∧ (*b*,*c*) ∈ *s1*^*
using *a1 c3* **unfolding** *CCR-def* **by** *force*
moreover then have *c* ∈ *Field* (⋃ *S*) **using** *c3* **unfolding** *Field-def* **by** *blast*
moreover have *s1*^* ⊆ (⋃ *S*)^* **using** *c3 Transitive-Closure.rtrancl-mono*[*of s1* ⋃ *S*] **by** *blast*
ultimately show ∃ *c*∈*Field* (⋃ *S*). (*a*,*c*) ∈ (⋃ *S*)^* ∧ (*b*,*c*) ∈ (⋃ *S*)^* **by** *blast*
qed

qed
then show *?thesis* **unfolding** *CCR-def* **by** *blast*
qed

lemma *lem-Relprop-restr-ch-un*:
fixes C::'U *set set* **and** r::'U *rel*
assumes $\forall A1 \in C.\ \forall A2 \in C.\ A1 \subseteq A2 \vee A2 \subseteq A1$
shows $Restr\ r\ (\bigcup\ C) = \bigcup\ \{\ s.\ \exists\ A \in C.\ s = Restr\ r\ A\ \}$
proof
show $Restr\ r\ (\bigcup\ C) \subseteq \bigcup\ \{\ s.\ \exists\ A \in C.\ s = Restr\ r\ A\ \}$
proof
fix p
assume $p \in Restr\ r\ (\bigcup\ C)$
then obtain $a\ b\ A1\ A2$ **where** $p = (a,b) \wedge a \in A1 \wedge b \in A2 \wedge p \in r \wedge A1 \in C \wedge A2 \in C$ **by** *blast*
moreover then have $A1 \subseteq A2 \vee A2 \subseteq A1$ **using** *assms* **by** *blast*
ultimately show $p \in \bigcup\ \{\ s.\ \exists\ A \in C.\ s = Restr\ r\ A\ \}$ **by** *blast*
qed
next
show $\bigcup\ \{\ s.\ \exists\ A \in C.\ s = Restr\ r\ A\ \} \subseteq Restr\ r\ (\bigcup\ C)$ **by** *blast*
qed

lemma *lem-Inv-restr-rtr*:
fixes r::'U *rel* **and** A::'U *set*
assumes $A \in Inv\ r$
shows $r\hat{}* \cap (A \times (UNIV::'U\ set)) \subseteq (Restr\ r\ A)\hat{}*$
proof −
have $\forall\ n.\ \forall\ a\ b.\ (a,b) \in r\hat{}\hat{}n \wedge a \in A \longrightarrow (a,b) \in (Restr\ r\ A)\hat{}*$
proof
fix n
show $\forall\ a\ b.\ (a,b) \in r\hat{}\hat{}n \wedge a \in A \longrightarrow (a,b) \in (Restr\ r\ A)\hat{}*$
proof (*induct n*)
show $\forall a\ b.\ (a,b) \in r\ \hat{}\hat{}\ 0 \wedge a \in A \longrightarrow (a,b) \in (Restr\ r\ A)\hat{}*$ **by** *simp*
next
fix n
assume *d1*: $\forall a\ b.\ (a,b) \in r\ \hat{}\hat{}\ n \wedge a \in A \longrightarrow (a,b) \in (Restr\ r\ A)\hat{}*$
show $\forall a\ b.\ (a,b) \in r\ \hat{}\hat{}\ (Suc\ n) \wedge a \in A \longrightarrow (a,b) \in (Restr\ r\ A)\hat{}*$
proof (*intro allI impI*)
fix $a\ b$
assume *e1*: $(a,b) \in r\ \hat{}\hat{}\ (Suc\ n) \wedge a \in A$
moreover then obtain c **where** *e2*: $(a,c) \in r\hat{}\hat{}n \wedge (c,b) \in r$ **by** *force*
ultimately have *e3*: $(a,c) \in (Restr\ r\ A)\hat{}*$ **using** *d1* **by** *blast*
moreover then have $c \in A$ **using** *e1* **using** *rtranclE* **by** *force*
then have $(c,b) \in Restr\ r\ A$ **using** *assms e2* **unfolding** *Inv-def* **by** *blast*
then show $(a,b) \in (Restr\ r\ A)\hat{}*$ **using** *e3* **by** (*meson rtrancl.rtrancl-into-rtrancl*)
qed
qed
qed
then show *?thesis* **using** *rtrancl-power* **by** *blast*

qed

lemma *lem-Inv-restr-rtr2*:
fixes $r::'U\ rel$ **and** $A::'U\ set$
assumes $A \in Inv\ r$
shows $r\hat{}* \cap (A\times(UNIV::'U\ set)) \subseteq (Restr\ r\ A)\hat{}* \cap ((UNIV::'U\ set)\times A)$
proof −
  have $\forall\ n.\ \forall\ a\ b.\ (a,b) \in r\hat{}\hat{}n \wedge a \in A \longrightarrow (a,b) \in (Restr\ r\ A)\hat{}* \cap ((UNIV::'U\ set)\times A)$
  proof
    fix n
    show $\forall\ a\ b.\ (a,b) \in r\hat{}\hat{}n \wedge a \in A \longrightarrow (a,b) \in (Restr\ r\ A)\hat{}* \cap ((UNIV::'U\ set)\times A)$
    proof (*induct n*)
      show $\forall a\ b.\ (a,b) \in r\ \hat{}\hat{}\ 0 \wedge a \in A \longrightarrow (a,b) \in (Restr\ r\ A)\hat{}* \cap ((UNIV::'U\ set)\times A)$ **by** *simp*
    next
      fix n
      assume *d1*: $\forall a\ b.\ (a,b) \in r\ \hat{}\hat{}\ n \wedge a \in A \longrightarrow (a,b) \in (Restr\ r\ A)\hat{}* \cap ((UNIV::'U\ set)\times A)$
      show $\forall a\ b.\ (a,b) \in r\ \hat{}\hat{}\ (Suc\ n) \wedge a \in A \longrightarrow (a,b) \in (Restr\ r\ A)\hat{}* \cap ((UNIV::'U\ set)\times A)$
      proof (*intro allI impI*)
        fix $a\ b$
        assume *e1*: $(a,b) \in r\ \hat{}\hat{}(Suc\ n) \wedge a \in A$
        moreover then obtain c **where** *e2*: $(a,c) \in r\hat{}\hat{}n \wedge (c,b) \in r$ **by** *force*
        ultimately have *e3*: $(a,c) \in (Restr\ r\ A)\hat{}*$ **using** *d1* **by** *blast*
        moreover then have $c \in A$ **using** *e1* **using** *rtranclE* **by** *force*
        then have *e4*: $(c,b) \in Restr\ r\ A$ **using** *assms e2* **unfolding** *Inv-def* **by** *blast*
      ultimately have $(a,b) \in (Restr\ r\ A)\hat{}*$ **using** *e3* **by** (*meson rtrancl.rtrancl-into-rtrancl*)
        then show $(a,b) \in (Restr\ r\ A)\hat{}* \cap ((UNIV::'U\ set)\times A)$ **using** *e4* **by** *blast*
      qed
    qed
  qed
  then show *?thesis* **using** *rtrancl-power* **by** *blast*
qed

lemma *lem-inv-rtr-mem*:
fixes $r::'U\ rel$ **and** $A::'U\ set$ **and** $a\ b::'U$
assumes $A \in Inv\ r$ **and** $a \in A$ **and** $(a,b) \in r\hat{}*$
shows $b \in A$
  using *assms lem-Inv-restr-rtr*[*of A r*] *rtranclE*[*of a b*] **by** *blast*

lemma *lem-Inv-ccr-restr*:
fixes $r::'U\ rel$ **and** $A::'U\ set$
assumes $CCR\ r$ **and** $A \in Inv\ r$
shows $CCR\ (Restr\ r\ A)$
proof −

have $\forall a \in Field\ (Restr\ r\ A).\ \forall b \in Field\ (Restr\ r\ A).\ \exists c \in Field\ (Restr\ r\ A).$ $(a,c) \in (Restr\ r\ A)\hat{}* \wedge (b,c) \in (Restr\ r\ A)\hat{}*$
proof (*intro ballI*)
fix *a b*
assume *c1*: $a \in Field\ (Restr\ r\ A)$ **and** *c2*: $b \in Field\ (Restr\ r\ A)$
moreover then obtain *c* **where** $c \in Field\ r$ **and** $(a,c) \in r\hat{}* \wedge (b,c) \in r\hat{}*$ **using** *assms* **unfolding** *CCR-def Field-def* **by** *blast*
ultimately have $(a,c) \in r\hat{}* \cap (A\times(UNIV::'U\ set)) \wedge (b,c) \in r\hat{}* \cap (A\times(UNIV::'U\ set))$ **unfolding** *Field-def* **by** *blast*
then have $(a,c) \in (Restr\ r\ A)\hat{}* \wedge (b,c) \in (Restr\ r\ A)\hat{}*$ **using** *assms lem-Inv-restr-rtr* **by** *blast*
moreover then have $c \in Field\ (Restr\ r\ A)$ **using** *c1 lem-rtr-field*[*of a c*] **by** *blast*
ultimately show $\exists c \in Field\ (Restr\ r\ A).\ (a,c) \in (Restr\ r\ A)\hat{}* \wedge (b,c) \in (Restr\ r\ A)\hat{}*$ **by** *blast*
qed
then show *?thesis* **unfolding** *CCR-def* **by** *blast*
qed

lemma *lem-Inv-cl-restr*:
fixes *r*::*'U rel* **and** *A*::*'U set*
assumes *Conelike r* **and** $A \in Inv\ r$
shows *Conelike* (*Restr r A*)
proof(*cases* $r = \{\}$)
assume $r = \{\}$
then show *?thesis* **unfolding** *Conelike-def* **by** *blast*
next
assume $r \neq \{\}$
then obtain *m* **where** *b1*: $\forall\ a \in Field\ r.\ (a,m) \in r\hat{}*$ **using** *assms* **unfolding** *Conelike-def* **by** *blast*
show *Conelike* (*Restr r A*)
proof (*cases* $m \in Field\ (Restr\ r\ A)$)
assume $m \in Field\ (Restr\ r\ A)$
moreover have $\forall a \in Field\ (Restr\ r\ A).\ (a,m) \in (Restr\ r\ A)\hat{}*$
using *assms lem-Inv-restr-rtr b1* **unfolding** *Field-def* **by** *blast*
ultimately show *Conelike* (*Restr r A*) **unfolding** *Conelike-def* **by** *blast*
next
assume *c1*: $m \notin Field\ (Restr\ r\ A)$
have $(Field\ r) \cap A \subseteq \{m\}$
proof
fix *a0*
assume $a0 \in (Field\ r) \cap A$
then have $(a0,m) \in r\hat{}* \cap (A\times(UNIV::'U\ set))$ **using** *b1* **by** *blast*
then have $(a0,m) \in (Restr\ r\ A)\hat{}*$ **using** *assms lem-Inv-restr-rtr* **by** *blast*
then show $a0 \in \{m\}$ **using** *c1 lem-rtr-field* **by** (*metis* (*full-types*) *mem-Collect-eq singleton-conv*)
qed
then show *Conelike* (*Restr r A*) **unfolding** *Conelike-def Field-def* **by** *blast*
qed

```
qed

lemma lem-Inv-ccr-restr-invdiff:
fixes r::'U rel and A B::'U set
assumes a1: CCR (Restr r A) and a2: B ∈ Inv (r^-1)
shows CCR (Restr r (A − B))
proof −
  have (Restr r A) `` (A−B) ⊆ (A−B)
  proof
    fix b
    assume b ∈ (Restr r A) `` (A−B)
    then obtain a where c2: a ∈ A−B ∧ (a,b) ∈ (Restr r A) by blast
    moreover then have b ∉ B using a2 unfolding Inv-def by blast
    ultimately show b ∈ A − B by blast
  qed
  then have (A−B) ∈ Inv(Restr r A) unfolding Inv-def by blast
  then have CCR (Restr (Restr r A) (A − B)) using a1 lem-Inv-ccr-restr by
blast
  moreover have Restr (Restr r A) (A − B) = Restr r (A−B) by blast
  ultimately show ?thesis by metis
qed

lemma lem-Inv-dncl-invbk: dncl r A ∈ Inv (r^-1)
  unfolding dncl-def Inv-def apply clarify
  using converse-rtrancl-into-rtrancl by (metis ImageI rtrancl-converse rtrancl-converseI)

lemma lem-inv-sf-ext:
fixes r::'U rel and A::'U set
assumes A ⊆ Field r
shows ∃ A' ∈ SF r. A ⊆ A' ∧ (finite A ⟶ finite A') ∧ ((¬ finite A) ⟶ |A'| =o
|A| )
proof −
  obtain rs where b4: rs = r ∪ (r^-1) by blast
  obtain S where b1: S = (λ a. rs``{a} ) by blast
  obtain S' where b2: S' = (λ a. if (S a) ≠ {} then (S a) else {a}) by blast
  obtain f where f = (λ a. SOME b. b ∈ S' a) by blast
  moreover have ∀ a. ∃ b. b ∈ (S' a) unfolding b2 by force
  ultimately have ∀ a. (f a) ∈ (S' a) by (metis someI-ex)
  then have b3: ∀ a. (S a ≠ {} ⟶ f a ∈ S a) ∧ (S a = {} ⟶ f a = a)
    unfolding b2 by (clarsimp, metis singletonD)
  obtain A' where b5: A' = A ∪ (f ` A) by blast
  have A ∪ (f ` A) ⊆ Field (Restr r A')
  proof
    fix x
    assume x ∈ A ∪ (f ` A)
    then obtain a b where c1: a ∈ A ∧ b = f a ∧ x ∈ {a,b} by blast
    moreover then have rs `` {a} ≠ {} ⟶ (a, b) ∈ rs using assms b1 b3 by
blast
    moreover have rs `` {a} = {} ⟶ False using assms c1 b4 unfolding
```

Field-def **by** *blast*
moreover have $(a,b) \in rs \longrightarrow \{a,b\} \subseteq$ *Field* (*Restr r A'*) **using** *c1 b4 b5* **unfolding** *Field-def* **by** *blast*
ultimately show $x \in$ *Field* (*Restr r A'*) **by** *blast*
qed
then have $(A \subseteq A') \wedge (A' \in$ *SF r*) **using** *b5* **unfolding** *SF-def Field-def* **by** *blast*
moreover have *finite A* $\longrightarrow$ *finite A'* **using** *b5* **by** *blast*
moreover have ($\neg$ *finite A*) $\longrightarrow |A'| =o |A|$ **using** *b5* **by** *simp*
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-inv-sf-un*:
assumes $S \subseteq$ *SF r*
shows $(\bigcup S) \in$ *SF r*
using *assms* **unfolding** *SF-def Field-def* **by** *blast*

lemma *lem-Inv-ccr-sf-inv-diff*:
fixes *r*::*'U rel* **and** *A B*::*'U set*
assumes *a1*: $A \in$ *SF r* **and** *a2*: *CCR* (*Restr r A*) **and** *a3*: $B \in$ *Inv* (r^-1)
shows $(A-B) \in$ *SF r* $\vee$ ($\exists$ *y*::*'U*. $(A-B) = \{y\}$)
proof $-$
have $\forall\ a \in A - B.\ a \notin$ *Field* (*Restr r* $(A-B)$) $\longrightarrow A - B = \{a\}$
proof (*intro ballI impI*)
fix *a*
assume *b1*: $a \in A - B$ **and** *b2*: $a \notin$ *Field* (*Restr r* $(A-B)$)
then have $\neg$ ($\exists\ b \in A-B.\ (a,b) \in r \vee (b,a) \in r$) **unfolding** *Field-def* **by** *blast*
then have *b3*: $\forall\ b \in A.\ (a,b) \notin r$ **using** *a3 b1* **unfolding** *Inv-def* **by** *blast*
have *b4*: $\forall\ x \in$ *Field*(*Restr r A*). $(x,a) \in$ (*Restr r A*)^$*$
proof
fix *x*
assume $x \in$ *Field*(*Restr r A*)
moreover then have $a \in$ *Field* (*Restr r A*) **using** *b1 a1* **unfolding** *SF-def* **by** *blast*
ultimately obtain *y* **where** *c1*: $(a,y) \in$ (*Restr r A*)^$*$ $\wedge$ $(x,y) \in$ (*Restr r A*)^$*$
using *a2* **unfolding** *CCR-def* **by** *blast*
moreover have $(a,y) \in$ (*Restr r A*)^$+$ $\longrightarrow$ *False* **using** *b3 tranclD* **by** *force*
ultimately have $a = y$ **using** *rtrancl-eq-or-trancl* **by** *metis*
then show $(x,a) \in$ (*Restr r A*)^$*$ **using** *c1* **by** *blast*
qed
have $\forall\ b \in (A-B) - \{a\}.$ *False*
proof
fix *b*
assume *c1*: $b \in (A-B) - \{a\}$
then have $b \in$ *Field* (*Restr r A*) **using** *a1* **unfolding** *SF-def* **by** *blast*
then have $(b,a) \in$ (*Restr r A*)^$*$ **using** *b4* **by** *blast*
moreover have $(b,a) \in$ (*Restr r A*)^$+$ $\longrightarrow$ *False*
proof

assume $(b,a) \in (Restr\ r\ A)\hat{}+$
then obtain b' **where** *d1*: $(b,b') \in (Restr\ r\ A)\hat{}* \wedge (b',a) \in Restr\ r\ A$ **using** *tranclD2* **by** *metis*
have *d2*: $\forall\ r'\ a\ b.\ (a,b) \in Restr\ r'\ B = (a \in B \wedge b \in B \wedge (a,b) \in r')$
unfolding *Field-def* **by** *force*
have $(b,b') \in r\hat{}*$ **using** *d1 rtrancl-mono*[*of Restr r A*] **by** *blast*
then have $(b',b) \in (r\hat{}-1)\hat{}*$ **using** *rtrancl-converse* **by** *blast*
then have $b' \in B \longrightarrow (b',b) \in (Restr\ (r\hat{}-1)\ B)\hat{}*$ **using** *a3 lem-Inv-restr-rtr* **by** *blast*
then have $b' \in B \longrightarrow b \in B$ **using** *d2* **by** (*metis rtrancl-eq-or-trancl tranclD2*)
then have $b' \in A - B$ **using** *d1 c1* **by** *blast*
then have $(b',a) \in Restr\ r\ (A-B)$ **using** *b1 d1* **by** *blast*
then have $a \in Field\ (Restr\ r\ (A-B))$ **unfolding** *Field-def* **by** *blast*
then show *False* **using** *b2* **by** *blast*
qed
ultimately have $b = a$ **using** *rtrancl-eq-or-trancl*[*of b a*] **by** *blast*
then show *False* **using** *c1* **by** *blast*
qed
then show $A - B = \{a\}$ **using** *b1* **by** *blast*
qed
then show *?thesis* **unfolding** *SF-def Field-def* **by** *blast*
qed

lemma *lem-Inv-ccr-sf-dn-diff*:
fixes r::$'U\ rel$ **and** $A\ D\ A'$::$'U\ set$
assumes *a1*: $A \in SF\ r$ **and** *a2*: $CCR\ (Restr\ r\ A)$ **and** *a3*: $A' = (A - (dncl\ r\ D))$
shows $((A' \in SF\ r) \wedge CCR\ (Restr\ r\ A')) \vee (\exists\ y$::$'U.\ A' = \{y\})$
using *assms lem-Inv-ccr-restr-invdiff lem-Inv-ccr-sf-inv-diff lem-Inv-dncl-invbk* **by** *blast*

lemma *lem-rseq-tr*:
fixes r::$'U\ rel$ **and** xi::$nat \Rightarrow 'U$
assumes $\forall\ i.\ (xi\ i,\ xi\ (Suc\ i)) \in r$
shows $\forall\ i\ j.\ i < j \longrightarrow (xi\ i \in Field\ r \wedge (xi\ i,\ xi\ j) \in r\hat{}+)$
proof −
have $\bigwedge j.\ \forall\ i < j.\ xi\ i \in Field\ r \wedge (xi\ i,\ xi\ j) \in r\hat{}+$
proof −
fix *j0*
show $\forall\ i < j0.\ xi\ i \in Field\ r \wedge (xi\ i,\ xi\ j0) \in r\hat{}+$
proof (*induct j0*)
show $\forall i<0.\ xi\ i \in Field\ r \wedge (xi\ i,\ xi\ 0) \in r\hat{}+$ **by** *blast*
next
fix *j*
assume *d1*: $\forall i<j.\ xi\ i \in Field\ r \wedge (xi\ i,\ xi\ j) \in r\hat{}+$
show $\forall i<Suc\ j.\ xi\ i \in Field\ r \wedge (xi\ i,\ xi\ (Suc\ j)) \in r\hat{}+$
proof (*intro allI impI*)
fix *i*
assume *e1*: $i < Suc\ j$

```
        have e2: (xi j, xi (Suc j)) ∈ r using assms by simp
        show xi i ∈ Field r ∧ (xi i, xi (Suc j)) ∈ r^+
        proof (cases i < j)
          assume i < j
          then have xi i ∈ Field r ∧ (xi i, xi j) ∈ r^+ using d1 by blast
          then show ?thesis using e2 by force
        next
          assume ¬ i < j
          then have i = j using e1 by simp
          then show ?thesis using e2 unfolding Field-def by blast
        qed
      qed
    qed
  qed
  then show ?thesis by blast
qed

lemma lem-rseq-rtr:
fixes r::'U rel and xi::nat ⇒ 'U
assumes ∀ i. (xi i, xi (Suc i)) ∈ r
shows ∀ i j. i ≤ j ⟶ (xi i ∈ Field r ∧ (xi i, xi j) ∈ r^*)
proof (intro allI impI)
  fix i::nat and j::nat
  assume b1: i ≤ j
  then have xi i ∈ Field r using assms unfolding Field-def by blast
  moreover have (xi i, xi j) ∈ r^*
  proof (cases i = j)
    assume i = j
    then show ?thesis by blast
  next
    assume i ≠ j
    then have i < j using b1 by simp
    moreover have r^+ ⊆ r^* by force
    ultimately show ?thesis using assms lem-rseq-tr[of xi r] by blast
  qed
  ultimately show xi i ∈ Field r ∧ (xi i, xi j) ∈ r^* by blast
qed

lemma lem-rseq-svacyc-inv-tr:
fixes r::'U rel and xi::nat ⇒ 'U and a::'U
assumes a1: single-valued r and a2: ∀ i. (xi i, xi (Suc i)) ∈ r
shows ⋀ i. (xi i, a) ∈ r^+ ⟹ (∃ j. i<j ∧ a = xi j)
proof −
  fix i
  assume (xi i, a) ∈ r^+
  moreover have ⋀ n. ∀ i a. (xi i, a) ∈ r^^(Suc n) ⟶ (∃ j. i<j ∧ a = xi j)
  proof −
    fix n
    show ∀ i a. (xi i, a) ∈ r^^(Suc n) ⟶ (∃ j. i<j ∧ a = xi j)
```

proof (*induct n*)
show ∀ *i a*. (*xi i*, *a*) ∈ *r*^^(*Suc 0*) ⟶ (∃ *j*>*i*. *a* = *xi j*)
proof (*intro allI impI*)
fix *i a*
assume (*xi i*, *a*) ∈ *r*^^(*Suc 0*)
then have (*xi i*, *a*) ∈ *r* ∧ (*xi i*, *xi* (*Suc i*)) ∈ *r* **using** *a2* **by** *simp*
then have *a* = *xi* (*Suc i*) **using** *a1* **unfolding** *single-valued-def* **by** *blast*
then show ∃ *j*>*i*. *a* = *xi j* **by** *force*
qed
next
fix *n*
assume *d1*: ∀ *i a*. (*xi i*, *a*) ∈ *r*^^(*Suc n*) ⟶ (∃ *j*>*i*. *a* = *xi j*)
show ∀ *i a*. (*xi i*, *a*) ∈ *r* ^^ *Suc* (*Suc n*) ⟶ (∃ *j*>*i*. *a* = *xi j*)
proof (*intro allI impI*)
fix *i a*
assume (*xi i*, *a*) ∈ *r*^^(*Suc* (*Suc n*))
then obtain *b* **where** (*xi i*, *b*) ∈ *r*^^(*Suc n*) ∧ (*b*, *a*) ∈ *r* **by** *force*
moreover then obtain *j* **where** *e1*: *j* > *i* ∧ *b* = *xi j* **using** *d1* **by** *blast*
ultimately have (*xi j*, *a*) ∈ *r* ∧ (*xi j*, *xi* (*Suc j*)) ∈ *r* **using** *a2* **by** *blast*
then have *a* = *xi* (*Suc j*) **using** *a1* **unfolding** *single-valued-def* **by** *blast*
moreover have *Suc j* > *i* **using** *e1* **by** *force*
ultimately show ∃ *j*>*i*. *a* = *xi j* **by** *blast*
qed
qed
qed
ultimately show ∃ *j*. *i*<*j* ∧ *a* = *xi j* **using** *trancl-power*[*of* - *r*] **by** (*metis Suc-pred′*)
qed

lemma *lem-rseq-svacyc-inv-rtr*:
fixes *r*::′*U rel* **and** *xi*::*nat* ⇒ ′*U* **and** *a*::′*U*
assumes *a1*: *single-valued r* **and** *a2*: ∀ *i*. (*xi i*, *xi* (*Suc i*)) ∈ *r*
shows ⋀ *i*. (*xi i*, *a*) ∈ *r*^∗ ⟹ (∃ *j*. *i*≤*j* ∧ *a* = *xi j*)
proof −
fix *i*
assume *b1*: (*xi i*, *a*) ∈ *r*^∗
show ∃ *j*. *i*≤*j* ∧ *a* = *xi j*
proof (*cases xi i* = *a*)
assume *xi i* = *a*
then show *?thesis* **by** *force*
next
assume *xi i* ≠ *a*
then have (*xi i*, *a*) ∈ *r*^+ **using** *b1* **by** (*meson rtranclD*)
then obtain *j* **where** *i*<*j* ∧ *a* = *xi j* **using** *assms lem-rseq-svacyc-inv-tr*[*of r xi i a*] **by** *blast*
then have *i* ≤ *j* ∧ *a* = *xi j* **by** *force*
then show *?thesis* **by** *blast*
qed
qed

lemma *lem-ccrsv-cfseq*:
fixes *r*::*'U rel*
assumes *a1*: *r* ≠ {} **and** *a2*: *CCR r* **and** *a3*: *single-valued r* **and** *a4*: ∀ *x*∈*Field r*. *r*‘‘{*x*} ≠ {}
shows ∃ *xi*. *cfseq r xi*
proof −
  have *b1*: *Field r* ≠ {} ∧ (∀ *x* ∈ *Field r*. ∃ *y*. (*x*,*y*) ∈ *r*)
    using *a1 a4* **unfolding** *Field-def* **by** *force*
  moreover obtain *f* **where** *f* = (λ *x*. *SOME y*. (*x*,*y*) ∈ *r*) **by** *blast*
  ultimately have *b2*: ∀ *x* ∈ *Field r*. (*x*, *f x*) ∈ *r* **by** (*metis someI-ex*)
  obtain *x0* **where** *b3*: *x0* ∈ *Field r* **using** *b1* **unfolding** *Field-def* **by** *blast*
  obtain *xi*::*nat* ⇒ *'U* **where** *b4*: *xi* = (λ *n*::*nat*. (*f*^^*n*) *x0*) **by** *blast*
  obtain *A* **where** *b5*: *A* = *xi* ‘ *UNIV* **by** *blast*
  have *r* ‘‘ *A* ⊆ *A*
  proof
    fix *a*
    assume *a* ∈ *r*‘‘*A*
    then obtain *i* **where** (*xi i*, *a*) ∈ *r* **using** *b5* **by** *blast*
    moreover then have (*xi i*, *f* (*xi i*)) ∈ *r* **using** *b2* **unfolding** *Field-def* **by** *blast*
    moreover have *f* (*xi i*) = *xi* (*Suc i*) **using** *b4* **by** *simp*
    ultimately have *a* = *xi* (*Suc i*) **using** *a3* **unfolding** *single-valued-def* **by** *blast*
    then show *a* ∈ *A* **using** *b5* **by** *blast*
  qed
  then have *b6*: *A* ∈ *Inv r* **unfolding** *Inv-def* **by** *blast*
  have ∀ *a* ∈ *Field r*. ∃ *i*. (*a*, *xi i*) ∈ *r*^*
  proof
    fix *a*
    assume *a* ∈ *Field r*
    then obtain *b* **where** (*a*,*b*) ∈ *r*^* ∧ (*x0*,*b*) ∈ *r*^* **using** *b3 a2* **unfolding** *CCR-def* **by** *blast*
    moreover have *x0* = *xi 0* **using** *b4* **by** *simp*
    ultimately have (*a*,*b*) ∈ *r*^* ∧ *b* ∈ *A* **using** *b5 b6 lem-inv-rtr-mem*[*of A r x0 b*] **by** *blast*
    then show ∃ *i*. (*a*, *xi i*) ∈ *r*^* **using** *b5* **by** *blast*
  qed
  moreover have ⋀ *i*. (*xi i*, *xi* (*Suc i*)) ∈ *r*
  proof −
    fix *i0*
    show (*xi i0*, *xi* (*Suc i0*)) ∈ *r*
    proof (*induct i0*)
      show (*xi 0*, *xi* (*Suc 0*)) ∈ *r* **using** *b2 b3 b4* **by** *simp*
    next
      fix *i*
      assume (*xi i*, *xi* (*Suc i*)) ∈ *r*
      then have *xi* (*Suc i*) ∈ *Field r* **unfolding** *Field-def* **by** *blast*
      then show (*xi* (*Suc i*), *xi* (*Suc* (*Suc i*))) ∈ *r* **using** *b2 b3 b4* **by** *simp*

qed
qed
ultimately show *?thesis* **unfolding** *cfseq-def* **by** *blast*
qed

lemma *lem-cfseq-fld*: *cfseq r xi* $\Longrightarrow$ *xi ‘ UNIV* $\subseteq$ *Field r*
using *lem-rseq-rtr*[*of xi r*] **unfolding** *cfseq-def* **by** *blast*

lemma *lem-cfseq-inv*: *cfseq r xi* $\Longrightarrow$ *single-valued r* $\Longrightarrow$ *xi ‘ UNIV* $\in$ *Inv r*
unfolding *cfseq-def single-valued-def Inv-def* **by** *blast*

lemma *lem-scfinv-scf-int*: $A \in SCF\ r \cap Inv\ r \Longrightarrow B \in SCF\ r \Longrightarrow (A \cap B) \in SCF\ r$
proof −
assume *a1*: $A \in SCF\ r \cap Inv\ r$ **and** *a2*: $B \in SCF\ r$
moreover have $\forall\ a \in Field\ r.\ \exists b{\in}A \cap B.\ (a,\ b) \in r\hat{}*$
proof
fix *a*
assume $a \in Field\ r$
then obtain a' **where** *b1*: $a' \in A \wedge a' \in Field\ r \wedge (a,a') \in r\hat{}*$ **using** *a1* **unfolding** *SCF-def* **by** *blast*
moreover then obtain *b* **where** *b2*: $b \in B \wedge (a',b) \in r\hat{}*$ **using** *a2* **unfolding** *SCF-def* **by** *blast*
ultimately have $(a,\ b) \in r\hat{}*$ **by** *force*
moreover have $b \in A \cap B$ **using** *b1 b2 a1 lem-inv-rtr-mem*[*of A r a′ b*] **by** *blast*
ultimately show $\exists\ b \in A \cap B.\ (a,\ b) \in r\hat{}*$ **by** *blast*
qed
ultimately show $(A \cap B) \in SCF\ r$ **unfolding** *SCF-def Inv-def* **by** *blast*
qed

lemma *lem-scf-minr*: $a \in Field\ r \Longrightarrow B \in SCF\ r \Longrightarrow \exists\ b \in B.\ (a,b) \in (r \cap ((UNIV{-}B) \times UNIV))\hat{}*$
proof −
assume *a1*: $a \in Field\ r$ **and** *a2*: $B \in SCF\ r$
then obtain b' **where** *b1*: $b' \in B \wedge (a,b') \in r\hat{}*$ **unfolding** *SCF-def* **by** *blast*
then obtain *n* **where** $(a,b') \in r\hat{}\hat{}n$ **using** *rtrancl-power* **by** *blast*
then obtain *f* **where** *b2*: $f\ (0{::}nat) = a \wedge f\ n = b'$ **and** *b3*: $\forall i{<}n.\ (f\ i,\ f\ (Suc\ i)) \in r$
using *relpow-fun-conv*[*of a b′*] **by** *blast*
obtain *N* **where** *b4*: $N = \{\ i.\ f\ i \in B\ \}$ **by** *blast*
obtain *s* **where** *b5*: $s = r \cap ((UNIV{-}B) \times UNIV)$ **by** *blast*
obtain *m* **where** $m = (LEAST\ i.\ i \in N)$ **by** *blast*
moreover have $n \in N$ **using** *b1 b2 b4* **by** *blast*
ultimately have $m \in N \wedge m \le n \wedge (\forall\ i \in N.\ m \le i)$ **by** (*metis LeastI Least-le*)
then have $m \le n \wedge f\ m \in B \wedge (\forall\ i < m.\ f\ i \notin B)$ **using** *b4* **by** *force*
then have $f\ 0 = a \wedge f\ m \in B \wedge (\forall i{<}m.\ (f\ i,\ f\ (Suc\ i)) \in s)$ **using** *b2 b3 b5* **by** *force*
then have $f\ m \in B \wedge (a,\ f\ m) \in s\hat{}*$

    using *relpow-fun-conv*[*of a f m*] *rtrancl-power*[*of - s*] **by** *metis*
  then show ∃ $b \in B$. $(a,b) \in (r \cap ((UNIV-B) \times UNIV))$^* **using** *b5* **by** *blast*
qed

lemma *lem-cfseq-ncl*:
fixes *r*::*'U rel* **and** *xi*::*nat* ⇒ *'U*
assumes *a1*: *cfseq r xi* **and** *a2*: ¬ *Conelike r*
shows ∀ *n*. ∃ *k*. $n \leq k \wedge (xi\ (Suc\ k), xi\ k) \notin r$^*
proof
  fix *n*
  have (∀ *k*. $n \leq k \longrightarrow (xi\ (Suc\ k), xi\ k) \in r$^*) ⟶ *False*
  proof
    assume *c1*: ∀ *k*. $n \leq k \longrightarrow (xi\ (Suc\ k), xi\ k) \in r$^*
    have ⋀ *k*. $n \leq k \longrightarrow (xi\ k, xi\ n) \in r$^*
    proof −
      fix *k*
      show $n \leq k \longrightarrow (xi\ k, xi\ n) \in r$^*
      proof (*induct k*)
        show $n \leq 0 \longrightarrow (xi\ 0, xi\ n) \in r$^* **by** *blast*
      next
        fix *k*
        assume *e1*: $n \leq k \longrightarrow (xi\ k, xi\ n) \in r$^*
        show $n \leq Suc\ k \longrightarrow (xi\ (Suc\ k), xi\ n) \in r$^*
        proof
          assume *f1*: $n \leq Suc\ k$
          show $(xi\ (Suc\ k), xi\ n) \in r$^*
          proof (*cases n = Suc k*)
            assume $n = Suc\ k$
            then show *?thesis* **using** *c1* **by** *blast*
          next
            assume $n \neq Suc\ k$
            then have $(xi\ k, xi\ n) \in r$^* ∧ $(xi\ (Suc\ k), xi\ k) \in r$^* **using** *f1 e1 c1*
by *simp*
            then show *?thesis* **by** *force*
          qed
        qed
      qed
    qed
    moreover have ∀ $k \leq n$. $(xi\ k, xi\ n) \in r$^* **using** *a1 lem-rseq-rtr* **unfolding**
cfseq-def **by** *blast*
    moreover have ∀ *k*::*nat*. $k \leq n \vee n \leq k$ **by** *force*
    ultimately have *b1*: ∀ *k*. $(xi\ k, xi\ n) \in r$^* **by** *blast*
    have $xi\ n \in Field\ r$ **using** *a1* **unfolding** *cfseq-def Field-def* **by** *blast*
    moreover have *b2*: ∀ $a \in Field\ r$. $(a, xi\ n) \in r$^*
    proof
      fix *a*
      assume $a \in Field\ r$
      then obtain *i* **where** $(a, xi\ i) \in r$^* **using** *a1* **unfolding** *cfseq-def* **by** *blast*
      moreover have $(xi\ i, xi\ n) \in r$^* **using** *b1* **by** *blast*

ultimately show $(a,\ xi\ n) \in r\hat{}*$ **by** *force*
qed
ultimately have *Conelike r* **unfolding** *Conelike-def* **by** *blast*
then show *False* **using** *a2* **by** *blast*
qed
then show $\exists\ k.\ n \leq k \wedge (xi\ (Suc\ k),\ xi\ k) \notin r\hat{}*$ **by** *blast*
qed

lemma *lem-cfseq-inj*:
fixes $r::'U\ rel$ **and** $xi::nat \Rightarrow\ 'U$
assumes *a1*: *cfseq r xi* **and** *a2*: *acyclic r*
shows *inj xi*
proof −
have $\forall\ i\ j.\ xi\ i = xi\ j \longrightarrow i = j$
proof (*intro allI impI*)
fix *i j*
assume *c1*: $xi\ i = xi\ j$
have $i < j \longrightarrow False$
proof
assume $i < j$
then have $(xi\ i,\ xi\ j) \in r\hat{}+$ **using** *a1 lem-rseq-tr* **unfolding** *cfseq-def* **by** *blast*
then show *False* **using** *c1 a2* **unfolding** *acyclic-def* **by** *force*
qed
moreover have $j < i \longrightarrow False$
proof
assume $j < i$
then have $(xi\ j,\ xi\ i) \in r\hat{}+$ **using** *a1 lem-rseq-tr* **unfolding** *cfseq-def* **by** *blast*
then show *False* **using** *c1 a2* **unfolding** *acyclic-def* **by** *force*
qed
ultimately show $i = j$ **by** *simp*
qed
then show *?thesis* **unfolding** *inj-on-def* **by** *blast*
qed

lemma *lem-cfseq-rmon*:
fixes $r::'U\ rel$ **and** $xi::nat \Rightarrow\ 'U$
assumes *a1*: *cfseq r xi* **and** *a2*: *single-valued r* **and** *a3*: *acyclic r*
shows $\forall\ i\ j.\ (xi\ i,\ xi\ j) \in r\hat{}+ \longrightarrow i < j$
proof (*intro allI impI*)
fix *i j*
assume *c1*: $(xi\ i,\ xi\ j) \in r\hat{}+$
then obtain j' **where** *c2*: $i < j' \wedge xi\ j' = xi\ j$
using *a1 a2 lem-rseq-svacyc-inv-tr*[*of r xi i*] **unfolding** *cfseq-def* **by** *metis*
have $j \leq i \longrightarrow False$
proof
assume *d1*: $j \leq i$
then have $(xi\ j,\ xi\ i) \in r\hat{}*$ **using** *c2 a1 lem-rseq-rtr* **unfolding** *cfseq-def* **by**

blast
  then have $(xi\ i,\ xi\ i) \in r\hat{}+$ **using** *c1* **by** *force*
  then show *False* **using** *a3* **unfolding** *acyclic-def* **by** *blast*
 qed
 then show $i < j$ **by** *simp*
qed

lemma *lem-rseq-hd*:
assumes $\forall\, i{<}n.\ (f\ i,\ f\ (Suc\ i)) \in r$
shows $\forall\, i{\le}n.\ (f\ 0,\ f\ i) \in r\hat{}*$
proof (*intro allI impI*)
 fix *i*
 assume $i \le n$
 then have $\forall\, j{<}i.\ (f\ j,\ f\ (Suc\ j)) \in r$ **using** *assms* **by** *force*
 then have $(f\ 0,\ f\ i) \in r\hat{}\hat{}i$ **using** *relpow-fun-conv* **by** *metis*
 then show $(f\ 0,\ f\ i) \in r\hat{}*$ **using** *relpow-imp-rtrancl* **by** *blast*
qed

lemma *lem-rseq-tl*:
assumes $\forall\, i{<}n.\ (f\ i,\ f\ (Suc\ i)) \in r$
shows $\forall\, i{\le}n.\ (f\ i,\ f\ n) \in r\hat{}*$
proof (*intro allI impI*)
 fix *i*
 assume *b1*: $i \le n$
 obtain *g* **where** *b2*: $g = (\lambda\ j.\ f\ (i + j))$ **by** *blast*
 then have $\forall\, j{<}n{-}i.\ (g\ j,\ g\ (Suc\ j)) \in r$ **using** *assms* **by** *force*
 moreover have $g\ 0 = f\ i \wedge g\ (n{-}i) = f\ n$ **using** *b1 b2* **by** *simp*
 ultimately have $(f\ i,\ f\ n) \in r\hat{}\hat{}(n{-}i)$ **using** *relpow-fun-conv* **by** *metis*
 then show $(f\ i,\ f\ n) \in r\hat{}*$ **using** *relpow-imp-rtrancl* **by** *blast*
qed

lemma *lem-ccext-ntr-rpth*: $(a,b) \in r\hat{}\hat{}n = (rpth\ r\ a\ b\ n \neq \{\})$
proof
 assume $rpth\ r\ a\ b\ n \neq \{\}$
 then obtain *f* **where** $f \in rpth\ r\ a\ b\ n$ **by** *blast*
 then show $(a,b) \in r\hat{}\hat{}n$ **unfolding** *rpth-def* **using** *relpow-fun-conv*[*of a b*] **by** *blast*
next
 assume $(a,b) \in r\hat{}\hat{}n$
 then obtain *f* **where** $f \in rpth\ r\ a\ b\ n$ **unfolding** *rpth-def* **using** *relpow-fun-conv*[*of a b*] **by** *blast*
 then show $rpth\ r\ a\ b\ n \neq \{\}$ **by** *blast*
qed

lemma *lem-ccext-rtr-rpth*: $(a,b) \in r\hat{}* \Longrightarrow \exists\ n.\ rpth\ r\ a\ b\ n \neq \{\}$
 using *rtrancl-power lem-ccext-ntr-rpth* **by** *metis*

lemma *lem-ccext-rpth-rtr*: $rpth\ r\ a\ b\ n \neq \{\} \Longrightarrow (a,b) \in r\hat{}*$
 using *rtrancl-power lem-ccext-ntr-rpth* **by** *metis*

lemma *lem-ccext-rtr-Fne*:
fixes r::$'U$ *rel* **and** a b::$'U$
shows $(a,b) \in r\hat{}* = (\mathcal{F}\ r\ a\ b \neq \{\})$
proof
assume $(a,b) \in r\hat{}*$
then obtain n f **where** $f \in rpth\ r\ a\ b\ n$ **using** *lem-ccext-rtr-rpth*[*of* a b r] **by** *blast*
then have $f`\{i.\ i{\le}n\} \in \mathcal{F}\ r\ a\ b$ **unfolding** $\mathcal{F}$-*def* **by** *blast*
then show $\mathcal{F}\ r\ a\ b \neq \{\}$ **by** *blast*
next
assume $\mathcal{F}\ r\ a\ b \neq \{\}$
then obtain F **where** $F \in \mathcal{F}\ r\ a\ b$ **by** *blast*
then obtain n::*nat* **and** f::*nat* $\Rightarrow$ $'U$ **where** $F = f`\{i.\ i{\le}n\} \wedge f \in rpth\ r\ a\ b\ n$ **unfolding** $\mathcal{F}$-*def* **by** *blast*
then show $(a,b) \in r\hat{}*$ **using** *lem-ccext-rpth-rtr*[*of* r] **by** *blast*
qed

lemma *lem-ccext-fprop*: $\mathcal{F}\ r\ a\ b \neq \{\} \Longrightarrow \mathfrak{f}\ r\ a\ b \in \mathcal{F}\ r\ a\ b$ **unfolding** $\mathfrak{f}$-*def* **using** *some-in-eq* **by** *metis*

lemma *lem-ccext-ffin*: *finite* $(\mathfrak{f}\ r\ a\ b)$
proof (*cases* $\mathcal{F}\ r\ a\ b = \{\}$)
assume $\mathcal{F}\ r\ a\ b = \{\}$
then show *finite* $(\mathfrak{f}\ r\ a\ b)$ **unfolding** $\mathfrak{f}$-*def* **by** *simp*
next
assume $\mathcal{F}\ r\ a\ b \neq \{\}$
then have $\mathfrak{f}\ r\ a\ b \in \mathcal{F}\ r\ a\ b$ **using** *lem-ccext-fprop*[*of* r] **by** *blast*
then show *finite* $(\mathfrak{f}\ r\ a\ b)$ **unfolding** $\mathcal{F}$-*def* **by** *force*
qed

lemma *lem-ccr-fin-subr-ext*:
fixes r s::$'U$ *rel*
assumes *a1*: *CCR* r **and** *a2*: $s \subseteq r$ **and** *a3*: *finite* s
shows $\exists\ s'$::$('U\ rel)$. *finite* $s' \wedge$ *CCR* $s' \wedge s \subseteq s' \wedge s' \subseteq r$
proof −
have *CCR* $\{\}$ **unfolding** *CCR-def Field-def* **by** *blast*
then have $\{\} \subseteq r \longrightarrow (\exists\ r''.\ CCR\ r'' \wedge \{\} \subseteq r'' \wedge r'' \subseteq r \wedge finite\ r'')$ **by** *blast*
moreover have $\bigwedge\ p\ R.\ finite\ R \Longrightarrow p \notin R \Longrightarrow$
$R \subseteq r \longrightarrow (\exists\ r''.\ CCR\ r'' \wedge R \subseteq r'' \wedge r'' \subseteq r \wedge finite\ r'') \Longrightarrow$
$insert\ p\ R \subseteq r \longrightarrow (\exists\ r''.\ CCR\ r'' \wedge\ insert\ p\ R \subseteq r'' \wedge r'' \subseteq r \wedge finite\ r'')$
proof
fix p R
assume *c1*: *finite* R **and** *c2*: $p \notin R$
and *c3*: $R \subseteq r \longrightarrow (\exists\ r''.\ CCR\ r'' \wedge R \subseteq r'' \wedge r'' \subseteq r \wedge finite\ r'')$ **and** *c4*: $insert\ p\ R \subseteq r$
then obtain r'' **where** *c5*: $CCR\ r'' \wedge R \subseteq r'' \wedge r'' \subseteq r \wedge finite\ r''$ **by** *blast*
show $\exists\ r'''.\ CCR\ r''' \wedge insert\ p\ R \subseteq r''' \wedge r''' \subseteq r \wedge finite\ r'''$
proof (*cases* $r'' = \{\}$)

assume $r'' = \{\}$
then have *insert p R* $\subseteq \{p\}$ **using** *c5* **by** *blast*
moreover have *CCR* $\{p\}$ **unfolding** *CCR-def Field-def* **by** *fastforce*
ultimately show $\exists\ r'''.\ CCR\ r''' \wedge insert\ p\ R \subseteq r''' \wedge r''' \subseteq r \wedge finite\ r'''$ **using** *c4* **by** *blast*
next
assume *d1*: $r'' \neq \{\}$
then obtain *xm* **where** *d2*: $xm \in Field\ r'' \wedge (\forall\ x \in Field\ r''.\ (x, xm) \in r''\hat{}*)$
using *c5 lem-Relprop-fin-ccr*[*of* r''] **unfolding** *Conelike-def* **by** *blast*
then have *d3*: $xm \in Field\ r$ **using** *c5* **unfolding** *Field-def* **by** *blast*
obtain *xp yp* **where** *d4*: $p = (xp, yp)$ **by** *force*
then have *d5*: $yp \in Field\ r$ **using** *c4* **unfolding** *Field-def* **by** *blast*
then obtain *t* **where** *d6*: $t \in Field\ r \wedge (xm, t) \in r\hat{}* \wedge (yp, t) \in r\hat{}*$ **using** *a1 d3* **unfolding** *CCR-def* **by** *blast*
then obtain *n m* **where** *d7*: $(xm, t) \in r\hat{}\hat{}n \wedge (yp, t) \in r\hat{}\hat{}m$ **using** *rtrancl-power* **by** *blast*
obtain *fn* **where** *d8*: $fn\ (0::nat) = xm \wedge fn\ n = t \wedge (\forall\, i<n.\ (fn\ i, fn(Suc\ i)) \in r)$ **using** *d7 relpow-fun-conv*[*of xm t*] **by** *blast*
obtain *fm* **where** *d9*: $fm\ (0::nat) = yp \wedge fm\ m = t \wedge (\forall\, i<m.\ (fm\ i, fm(Suc\ i)) \in r)$ **using** *d7 relpow-fun-conv*[*of yp t*] **by** *blast*
obtain *A* **where** *d10*: $A = Field\ r'' \cup \{\ xp\ \} \cup \{\ x.\ \exists\ i\leq n.\ x = fn\ i\ \} \cup \{\ x.\ \exists\ i\leq m.\ x = fm\ i\ \}$ **by** *blast*
obtain r''' **where** *d11*: $r''' = r \cap (A \times A)$ **by** *blast*
have *d12*: $r'' \subseteq r'''$ **using** *d10 d11 c5* **unfolding** *Field-def* **by** *fastforce*
then have *d13*: $Field\ r'' \subseteq Field\ r'''$ **unfolding** *Field-def* **by** *blast*
have *d14*: $r''\hat{}* \subseteq r'''\hat{}*$ **using** *d12 rtrancl-mono* **by** *blast*
have *d15*: $\forall\ i.\ i<n \longrightarrow (fn\ i, fn(Suc\ i)) \in r'''$
proof
fix *i*
show $i<n \longrightarrow (fn\ i, fn(Suc\ i)) \in r'''$
proof (*induct i*)
show $0 < n \longrightarrow (fn\ 0, fn\ (Suc\ 0)) \in r'''$
proof
assume $0 < n$
moreover then have $(Suc\ 0) \leq n$ **by** *force*
ultimately have $fn\ 0 \in A \wedge fn(Suc\ 0) \in A \wedge (fn\ 0, fn(Suc\ 0)) \in r$ **using** *d8 d10* **by** *fastforce*
then show $(fn\ 0, fn\ (Suc\ 0)) \in r'''$ **using** *d11* **by** *blast*
qed
next
fix *i*
assume *g1*: $i < n \longrightarrow (fn\ i, fn\ (Suc\ i)) \in r'''$
show $Suc\ i < n \longrightarrow (fn\ (Suc\ i), fn\ (Suc\ (Suc\ i))) \in r'''$
proof
assume $Suc\ i < n$
moreover then have $Suc\ (Suc\ i) \leq n$ **by** *simp*
moreover then have $(fn\ i, fn\ (Suc\ i)) \in r'''$ **using** *g1* **by** *simp*
ultimately show $(fn\ (Suc\ i), fn\ (Suc\ (Suc\ i))) \in r'''$ **using** *d8 d10 d11*

by *blast*
qed
qed
qed
have *d16*: $\forall\ i.\ i<m \longrightarrow (fm\ i,\ fm(Suc\ i)) \in r'''$
proof
fix *i*
show $i<m \longrightarrow (fm\ i,\ fm(Suc\ i)) \in r'''$
proof (*induct i*)
show $0 < m \longrightarrow (fm\ 0,\ fm\ (Suc\ 0)) \in r'''$
proof
assume $0 < m$
moreover then have $(Suc\ 0) \leq m$ **by** *force*
ultimately have $fm\ 0 \in A \wedge fm(Suc\ 0) \in A \wedge (fm\ 0,\ fm(Suc\ 0)) \in r$ **using** *d9 d10* **by** *fastforce*
then show $(fm\ 0,\ fm\ (Suc\ 0)) \in r'''$ **using** *d11* **by** *blast*
qed
next
fix *i*
assume *g1*: $i < m \longrightarrow (fm\ i,\ fm\ (Suc\ i)) \in r'''$
show $Suc\ i < m \longrightarrow (fm\ (Suc\ i),\ fm\ (Suc\ (Suc\ i))) \in r'''$
proof
assume $Suc\ i < m$
moreover then have $Suc\ (Suc\ i) \leq m$ **by** *simp*
moreover then have $(fm\ i,\ fm\ (Suc\ i)) \in r'''$ **using** *g1* **by** *simp*
ultimately show $(fm\ (Suc\ i),\ fm\ (Suc\ (Suc\ i))) \in r'''$ **using** *d9 d10 d11* **by** *blast*
qed
qed
qed
have *d17*: $(xm,\ t) \in r'''\hat{}*$ **using** *d8 d15 relpow-fun-conv*[*of xm t n* r'''] *rtrancl-power* **by** *blast*
then have *d18*: $t \in Field\ r'''$ **using** *d2 d13* **by** (*metis FieldI2 rtrancl.cases subsetCE*)
have *d19*: $(yp,\ t) \in r'''\hat{}*$ **using** *d9 d16 relpow-fun-conv*[*of yp t m* r'''] *rtrancl-power* **by** *blast*
have *d20*: $\forall\ j\leq n.\ (fn\ j,\ t) \in r'''\hat{}*$
proof (*intro allI impI*)
fix *j*
assume $j \leq n$
moreover obtain f' **where** $f' = (\lambda k.\ fn\ (j + k))$ **by** *blast*
ultimately have $f'\ 0 = fn\ j \wedge f'\ (n - j) = t \wedge (\forall\, i < n - j.\ (f'\ i,\ f'\ (Suc\ i)) \in r''')$
using *d8 d15* **by** *simp*
then show $(fn\ j,\ t) \in r'''\hat{}*$
using *relpow-fun-conv*[*of fn j t* $n - j$ r'''] *rtrancl-power* **by** *blast*
qed
have *d21*: $\forall\ j\leq m.\ (fm\ j,\ t) \in r'''\hat{}*$
proof (*intro allI impI*)

fix *j*
assume $j \le m$
moreover obtain f' **where** $f' = (\lambda k.\ fm\ (j + k))$ **by** *blast*
ultimately have $f'\ 0 = fm\ j \wedge f'\ (m - j) = t \wedge (\forall\, i < m - j.\ (f'\ i,\ f'\ (Suc\ i)) \in r''')$
using *d9 d16* **by** *simp*
then show $(fm\ j,\ t) \in r'''\hat{}*$
using *relpow-fun-conv*[*of fm j t m* $-$ *j r'''*] *rtrancl-power* **by** *blast*
qed
have $r''' \subseteq r$ **using** *d11* **by** *blast*
moreover have *d22*: *insert p R* $\subseteq r'''$
proof $-$
have $p \in r'''$ **using** *c4 d4 d9 d10 d11* **by** *blast*
moreover have $R \subseteq r'''$
proof
fix p'
assume $p' \in R$
moreover then have $p' \in$ *Field R* $\times$ *Field R* **using** *Restr-Field* **by** *blast*
moreover have *Field R* $\subseteq$ *Field r''* **using** *c5* **unfolding** *Field-def* **by** *blast*
ultimately show $p' \in r'''$ **using** *c4 d10 d11* **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed
moreover have *finite r'''* **using** *c5 d10 d11 finite-Field* **by** *fastforce*
moreover have *CCR r'''*
proof $-$
let *?jn* = λ *a b*. $\exists\, c \in$ *Field r'''*. $(a,c) \in r'''\hat{}* \wedge (b,c) \in r'''\hat{}*$
have $\forall\, a \in$ *Field r'''*. $\forall\, b \in$ *Field r'''*. *?jn a b*
proof (*intro ballI*)
fix *a b*
assume *f1*: $a \in$ *Field r'''* **and** *f2*: $b \in$ *Field r'''*
then have *f3*: $a \in A \wedge b \in A$ **using** *d11* **unfolding** *Field-def* **by** *blast*
have *f4*: $(xp,\ t) \in r'''\hat{}*$ **using** *d4 d19 d22* **by** *force*
have $a \in$ *Field r''* $\longrightarrow$ *?jn a b*
proof
assume *g1*: $a \in$ *Field r''*
then have *g2*: $(a,\ t) \in r'''\hat{}*$ **using** *d2 d14 d17* **by** *fastforce*
have $b \in$ *Field r''* $\longrightarrow$ *?jn a b* **using** *c5 d13 d14 g1* **unfolding** *CCR-def* **by** *blast*
moreover have *?jn a xp* **using** *d4 d18 d19 d22 g2* **by** *force*
moreover have $\forall\ j{\le}n.$ *?jn a* (*fn j*) **using** *d18 d20 g2* **by** *blast*
moreover have $\forall\ j{\le}m.$ *?jn a* (*fm j*) **using** *d18 d21 g2* **by** *blast*
ultimately show *?jn a b* **using** *d10 f3* **by** *blast*
qed
moreover have *?jn xp b*
proof $-$
have $b \in$ *Field r''* $\longrightarrow$ *?jn xp b*
proof

assume $b \in Field\ r''$
then have $(b,\ xm) \in r'''\hat{}*$ **using** *d14 d2* **by** *blast*
then show *?jn xp b* **using** *d17 d18 f4* **by** *force*
qed
moreover have *?jn xp xp* **using** *d4 d22* **unfolding** *Field-def* **by** *blast*
moreover have $\forall\ j \le n.$ *?jn xp (fn j)* **using** *d18 d20 f4* **by** *blast*
moreover have $\forall\ j \le m.$ *?jn xp (fm j)* **using** *d18 d21 f4* **by** *blast*
ultimately show *?jn xp b* **using** *d10 f3* **by** *blast*
qed
moreover have $\forall\ i \le n.$ *?jn (fn i) b*
proof (*intro allI impI*)
fix *i*
assume *g1*: $i \le n$
have $b \in Field\ r'' \longrightarrow$ *?jn (fn i) b*
proof
assume $b \in Field\ r''$
then have $(b,\ t) \in r'''\hat{}*$ **using** *d2 d14 d17* **by** *fastforce*
then show *?jn (fn i) b* **using** *d18 d20 g1* **by** *blast*
qed
moreover have *?jn (fn i) xp* **using** *d18 d20 f4 g1* **by** *blast*
moreover have $\forall\ j \le n.$ *?jn (fn i) (fn j)* **using** *d18 d20 g1* **by** *blast*
moreover have $\forall\ j \le m.$ *?jn (fn i) (fm j)* **using** *d18 d20 d21 g1* **by** *blast*
ultimately show *?jn (fn i) b* **using** *d10 f3* **by** *blast*
qed
moreover have $\forall\ i \le m.$ *?jn (fm i) b*
proof (*intro allI impI*)
fix *i*
assume *g1*: $i \le m$
have $b \in Field\ r'' \longrightarrow$ *?jn (fm i) b*
proof
assume $b \in Field\ r''$
then have $(b,\ t) \in r'''\hat{}*$ **using** *d2 d14 d17* **by** *fastforce*
then show *?jn (fm i) b* **using** *d18 d21 g1* **by** *blast*
qed
moreover have *?jn (fm i) xp* **using** *d18 d21 f4 g1* **by** *blast*
moreover have $\forall\ j \le n.$ *?jn (fm i) (fn j)* **using** *d18 d20 d21 g1* **by** *blast*
moreover have $\forall\ j \le m.$ *?jn (fm i) (fm j)* **using** *d18 d21 g1* **by** *blast*
ultimately show *?jn (fm i) b* **using** *d10 f3* **by** *blast*
qed
ultimately show *?jn a b* **using** *d10 f3* **by** *blast*
qed
then show *?thesis* **unfolding** *CCR-def* **by** *blast*
qed
ultimately show $\exists\ r'''.\ CCR\ r''' \wedge insert\ p\ R \subseteq r''' \wedge r''' \subseteq r \wedge finite\ r'''$
by *blast*
qed
qed
ultimately have $\exists\ r''.\ CCR\ r'' \wedge s \subseteq r'' \wedge r'' \subseteq r \wedge finite\ r''$
using *a2 a3 finite-induct*[*of s* $\lambda\ h.\ h \subseteq r \longrightarrow (\exists\ r''.\ CCR\ r'' \wedge h \subseteq r'' \wedge r''$

$\subseteq r \wedge$ *finite* $r'')$] **by** *simp*
then show *?thesis* **by** *blast*
qed

lemma *lem-Ccext-fint*:
fixes r s::$'U$ *rel* **and** a b::$'U$
assumes *a1*: *Restr* r $(\mathfrak{f}\ r\ a\ b) \subseteq s$ **and** *a2*: $(a,b) \in r\hat{\ }*$
shows $\{a,\ b\} \subseteq \mathfrak{f}\ r\ a\ b \wedge (\forall\ c \in \mathfrak{f}\ r\ a\ b.\ (a,c) \in s\hat{\ }* \wedge (c,b) \in s\hat{\ }*)$
proof −
obtain A **where** *b1*: $A = \mathfrak{f}\ r\ a\ b$ **by** *blast*
then have $A \in \mathcal{F}\ r\ a\ b$ **using** *a2 lem-ccext-rtr-Fne*[*of a b r*] *lem-ccext-fprop*[*of r*] **by** *blast*
then obtain n f **where** *b2*: $A = f$ ‘ $\{i.\ i \leq n\}$ **and** *b3*: $f \in$ *rpth* r a b n **unfolding** $\mathcal{F}$*-def* **by** *blast*
then have $\forall\ i<n.\ (f\ i,\ f\ (Suc\ i)) \in$ *Restr* r A **unfolding** *rpth-def* **by** *simp*
then have *b4*: $\forall\ i<n.\ (f\ i,\ f\ (Suc\ i)) \in s$ **using** *a1 b1* **by** *blast*
have $\{a,\ b\} \subseteq \mathfrak{f}\ r\ a\ b$ **using** *b1 b2 b3* **unfolding** *rpth-def* **by** *blast*
moreover have $\forall\ c \in \mathfrak{f}\ r\ a\ b.\ (a,c) \in s\hat{\ }* \wedge (c,b) \in s\hat{\ }*$
proof
fix c
assume $c \in \mathfrak{f}\ r\ a\ b$
then obtain k **where** *c1*: $k \leq n \wedge c = f\ k$ **using** *b1 b2* **by** *blast*
have $f \in$ *rpth* s a c k **using** *c1 b3 b4* **unfolding** *rpth-def* **by** *simp*
moreover have $(\lambda\ i.\ f\ (i + k)) \in$ *rpth* s c b $(n - k)$ **using** *c1 b3 b4* **unfolding** *rpth-def* **by** *simp*
ultimately show $(a,c) \in s\hat{\ }* \wedge (c,b) \in s\hat{\ }*$ **using** *lem-ccext-rpth-rtr*[*of s*] **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Ccext-subccr-eqfld*:
fixes r r'::$'U$ *rel*
assumes *CCR* r **and** $r \subseteq r'$ **and** *Field* r' = *Field* r
shows *CCR* r'
proof −
have $\forall\ a\in$*Field* r'. $\forall\ b\in$*Field* r'. $\exists\ c\in$*Field* r'. $(a,\ c) \in r'\hat{\ }* \wedge (b,\ c) \in r'\hat{\ }*$
proof (*intro ballI*)
fix a b
assume $a\in$*Field* r' **and** $b\in$*Field* r'
then have $a \in$ *Field* $r \wedge b \in$ *Field* r **using** *assms* **by** *blast*
then obtain c **where** $c \in$ *Field* $r \wedge (a,\ c) \in r\hat{\ }* \wedge (b,\ c) \in r\hat{\ }*$ **using** *assms* **unfolding** *CCR-def* **by** *blast*
then have $c \in$ *Field* $r' \wedge (a,\ c) \in r'\hat{\ }* \wedge (b,\ c) \in r'\hat{\ }*$ **using** *assms rtrancl-mono* **by** *blast*
then show $\exists\ c\in$*Field* r'. $(a,\ c) \in r'\hat{\ }* \wedge (b,\ c) \in r'\hat{\ }*$ **by** *blast*
qed
then show *CCR* r' **unfolding** *CCR-def* **by** *blast*
qed

lemma *lem-Ccext-finsubccr-pext*:
fixes r s::$'U$ *rel* **and** x::$'U$
assumes *a1*: *CCR* r **and** *a2*: $s \subseteq r$ **and** *a3*: *finite* s **and** *a5*: $x \in$ *Field* r
shows $\exists\ s'$::($'U$ *rel*). *finite* $s' \wedge$ *CCR* $s' \wedge s \subseteq s' \wedge s' \subseteq r \wedge x \in$ *Field* s'
proof −
  obtain y **where** *b1*: $(x,y) \in r \vee (y,x) \in r$ **using** *a5* **unfolding** *Field-def* **by** *blast*
  then obtain x' y' **where** *b2*: $\{x',y'\} = \{x,y\} \wedge (x',y') \in r$ **by** *blast*
  obtain *s1* **where** *b3*: $s1 = s \cup \{(x',y')\}$ **by** *blast*
  then have *finite s1* **using** *a3* **by** *blast*
  moreover have $s1 \subseteq r$ **using** *b2 b3 a2* **by** *blast*
  ultimately obtain s' **where** *b4*: *finite* $s' \wedge$ *CCR* $s' \wedge s1 \subseteq s' \wedge s' \subseteq r$ **using** *a1 lem-ccr-fin-subr-ext*[*of r s1*] **by** *blast*
  moreover have $x \in$ *Field s1* **using** *b2 b3* **unfolding** *Field-def* **by** *blast*
  ultimately have $x \in$ *Field* s' **unfolding** *Field-def* **by** *blast*
  then show *?thesis* **using** *b3 b4* **by** *blast*
qed

lemma *lem-Ccext-finsubccr-dext*:
fixes r::$'U$ *rel* **and** A::$'U$ *set*
assumes *a1*: *CCR* r **and** *a2*: $A \subseteq$ *Field* r **and** *a3*: *finite* A
shows $\exists\ s$::($'U$ *rel*). *finite* $s \wedge$ *CCR* $s \wedge s \subseteq r \wedge A \subseteq$ *Field* s
proof −
  have *finite* $\{\} \wedge \{\} \subseteq$ *Field* $r \longrightarrow (\exists s.$ *finite* $s \wedge$ *CCR* $s \wedge s \subseteq r \wedge \{\} \subseteq$ *Field* $s)$ **unfolding** *CCR-def Field-def* **by** *blast*
  moreover have $\forall\ x\ F.$ *finite* $F \longrightarrow x \notin F \longrightarrow$
    finite $F \wedge F \subseteq$ *Field* $r \longrightarrow (\exists s.$ *finite* $s \wedge$ *CCR* $s \wedge s \subseteq r \wedge F \subseteq$ *Field* $s) \longrightarrow$
    finite (*insert* x F) $\wedge$ *insert* x $F \subseteq$ *Field* $r \longrightarrow$
   $(\exists s.$ *finite* $s \wedge$ *CCR* $s \wedge s \subseteq r \wedge$ *insert* x $F \subseteq$ *Field* $s)$
  proof(*intro allI impI*)
   fix x F
   assume *c1*: *finite* F **and** *c2*: $x \notin F$ **and** *c3*: *finite* $F \wedge F \subseteq$ *Field* r
     and *c4*: $\exists s.$ *finite* $s \wedge$ *CCR* $s \wedge s \subseteq r \wedge F \subseteq$ *Field* s
     and *c5*: *finite* (*insert* x F) $\wedge$ *insert* x $F \subseteq$ *Field* r
   then obtain s **where** *c6*: *finite* $s \wedge$ *CCR* $s \wedge s \subseteq r \wedge F \subseteq$ *Field* s **by** *blast*
   moreover have $x \in$ *Field* r **using** *c5* **by** *blast*
   ultimately obtain s' **where** *finite* $s' \wedge$ *CCR* $s' \wedge s \subseteq s' \wedge s' \subseteq r \wedge x \in$ *Field* s'
    using *a1 lem-Ccext-finsubccr-pext*[*of r s x*] **by** *blast*
   moreover then have *insert* x $F \subseteq$ *Field* s' **using** *c6* **unfolding** *Field-def* **by** *blast*
   ultimately show $\exists s'.$ *finite* $s' \wedge$ *CCR* $s' \wedge s' \subseteq r \wedge$ *insert* x $F \subseteq$ *Field* s' **by** *blast*
  qed
  ultimately have *finite* $A \wedge A \subseteq$ *Field* $r \longrightarrow (\exists s.$ *finite* $s \wedge$ *CCR* $s \wedge s \subseteq r \wedge A \subseteq$ *Field* $s)$
   using *finite-induct*[*of A* λ A. *finite* $A \wedge A \subseteq$ *Field* $r \longrightarrow (\exists\ s.$ *finite* $s \wedge$ *CCR* $s \wedge s \subseteq r \wedge A \subseteq$ *Field* $s)$]

by *simp*
then show *?thesis* **using** *a2 a3* **by** *blast*
qed

lemma *lem-Ccext-infsubccr-pext*:
fixes *r s*::*'U rel* **and** *x*::*'U*
assumes *a1*: *CCR r* **and** *a2*: $s \subseteq r$ **and** *a3*: ¬ *finite s* **and** *a5*: $x \in$ *Field r*
shows ∃ *s'*::(*'U rel*). *CCR s'* ∧ *s* ⊆ *s'* ∧ *s'* ⊆ *r* ∧ |*s'*| =*o* |*s*| ∧ *x* ∈ *Field s'*
proof −
obtain *G*::*'U set* ⇒ *'U rel set* **where** *b1*: *G* = (λ *A*. {*t*::*'U rel*. *finite t* ∧ *CCR t* ∧ *t* ⊆ *r* ∧ *A* ⊆ *Field t*}) **by** *blast*
obtain *g*::*'U set* ⇒ *'U rel* **where** *b2*: *g* = (λ *A*. *if A* ⊆ *Field r* ∧ *finite A then* (*SOME t*. *t* ∈ *G A*) *else* {}) **by** *blast*
have *b3*: ∀ *A*. *A* ⊆ *Field r* ∧ *finite A* ⟶ *finite* (*g A*) ∧ *CCR* (*g A*) ∧ (*g A*) ⊆ *r* ∧ *A* ⊆ *Field* (*g A*)
proof (*intro allI impI*)
fix *A*
assume *c1*: *A* ⊆ *Field r* ∧ *finite A*
then have *g A* = (*SOME t*. *t* ∈ *G A*) **using** *b2* **by** *simp*
moreover have *G A* ≠ {} **using** *b1 a1 c1 lem-Ccext-finsubccr-dext*[*of r A*] **by** *blast*
ultimately have *g A* ∈ *G A* **using** *some-in-eq* **by** *metis*
then show *finite* (*g A*) ∧ *CCR* (*g A*) ∧ (*g A*) ⊆ *r* ∧ *A* ⊆ *Field* (*g A*) **using** *b1* **by** *blast*
qed
have *b4*: ∀ *A*. ¬ (*A* ⊆ *Field r* ∧ *finite A*) ⟶ *g A* = {} **using** *b2* **by** *simp*
obtain *H*::*'U set* ⇒ *'U set*
where *b5*: *H* = (λ *X*. *X* ∪ ⋃ {*S* . ∃ *a*∈*X*. ∃ *b*∈*X*. *S* = *Field* (*g* {*a*,*b*})}) **by** *blast*
obtain *ax bx* **where** *b6*: (*ax*, *bx*) ∈ *r* ∧ *x* ∈ {*ax*, *bx*} **using** *a5* **unfolding** *Field-def* **by** *blast*
obtain *D0*::*'U set* **where** *b7*: *D0* = *Field s* ∪ {*ax*, *bx*} **by** *blast*
obtain *Di*::*nat* ⇒ *'U set* **where** *b8*: *Di* = (λ *n*. (*H*^^*n*) *D0*) **by** *blast*
obtain *D*::*'U set* **where** *b9*: *D* = ⋃ {*X*. ∃ *n*. *X* = *Di n*} **by** *blast*
obtain *s'* **where** *b10*: *s'* = *Restr r D* **by** *blast*
have *b11*: ∀ *n*. (¬ *finite* (*Di n*)) ∧ |*Di n*| ≤*o* |*s*|
proof
fix *n0*
show (¬ *finite* (*Di n0*)) ∧ |*Di n0*| ≤*o* |*s*|
proof (*induct n0*)
have *finite* {*ax*, *bx*} **by** *blast*
moreover have ¬ *finite* (*Field s*) **using** *a3 lem-fin-fl-rel* **by** *blast*
ultimately have ¬ *finite* (*Field s*) ∧ |{*ax*, *bx*}| ≤*o* |*Field s*|
using *card-of-Well-order card-of-ordLeq-infinite ordLeq-total* **by** *metis*
then have |*D0*| =*o* |*Field s*| **using** *b7 card-of-Un-infinite* **by** *blast*
moreover have |*Field s*| =*o* |*s*| **using** *a3 lem-rel-inf-fld-card* **by** *blast*
ultimately have |*D0*| ≤*o* |*s*| **using** *ordIso-imp-ordLeq ordIso-transitive* **by** *blast*
moreover have ¬ *finite D0* **using** *a3 b7 lem-fin-fl-rel* **by** *blast*

ultimately show ¬ *finite* (*Di 0*) ∧ |*Di 0*| ≤*o* |*s*| **using** *b8* **by** *simp*
next
fix *n*
assume *d1*: (¬ *finite* (*Di n*)) ∧ |*Di n*| ≤*o* |*s*|
moreover then have |(*Di n*) × (*Di n*)| =*o* |*Di n*| **by** *simp*
ultimately have *d2*: |(*Di n*) × (*Di n*)| ≤*o* |*s*| **using** *ordIso-imp-ordLeq ordLeq-transitive* **by** *blast*
have *d3*: ∀ *a* ∈ (*Di n*). ∀ *b* ∈ (*Di n*). |*Field* (*g* {*a*, *b*})| ≤*o* |*s*|
proof (*intro ballI*)
fix *a b*
assume *a* ∈ (*Di n*) **and** *b* ∈ (*Di n*)
have *finite* (*g* {*a*, *b*}) **using** *b3 b4* **by** (*metis finite.emptyI*)
then have *finite* (*Field* (*g* {*a*, *b*})) **using** *lem-fin-fl-rel* **by** *blast*
then have |*Field* (*g* {*a*, *b*})| <*o* |*s*| **using** *a3 finite-ordLess-infinite2* **by** *blast*
then show |*Field* (*g* {*a*, *b*})| ≤*o* |*s*| **using** *ordLess-imp-ordLeq* **by** *blast*
qed
have *d4*: *Di* (*Suc n*) = *H* (*Di n*) **using** *b8* **by** *simp*
then have *Di n* ⊆ *Di* (*Suc n*) **using** *b5* **by** *blast*
then have ¬ *finite* (*Di* (*Suc n*)) **using** *d1 finite-subset* **by** *blast*
moreover have |*Di* (*Suc n*)| ≤*o* |*s*|
proof −
obtain *I* **where** *e1*: *I* = (*Di n*) × (*Di n*) **by** *blast*
obtain *f* **where** *e2*: *f* = (λ (*a*,*b*). *Field* (*g* {*a*,*b*})) **by** *blast*
have |*I*| ≤*o* |*s*| **using** *e1 d2* **by** *blast*
moreover have ∀ *i*∈*I*. |*f i*| ≤*o* |*s*| **using** *e1 e2 d3* **by** *simp*
ultimately have |⋃ *i*∈*I*. *f i*| ≤*o* |*s*| **using** *a3 card-of-UNION-ordLeq-infinite*[*of s I f*] **by** *blast*
moreover have *Di* (*Suc n*) = (*Di n*) ∪ (⋃ *i*∈*I*. *f i*) **using** *e1 e2 d4 b5* **by** *blast*
ultimately show *?thesis* **using** *d1 a3* **by** *simp*
qed
ultimately show (¬ *finite* (*Di* (*Suc n*))) ∧ |*Di* (*Suc n*)| ≤*o* |*s*| **by** *blast*
qed
qed
have *b12*: ∀ *m*. ∀ *n*. *n* ≤ *m* ⟶ *Di n* ≤ *Di m*
proof
fix *m0*
show ∀ *n*. *n* ≤ *m0* ⟶ *Di n* ≤ *Di m0*
proof (*induct m0*)
show ∀ *n*≤*0*. *Di n* ⊆ *Di 0* **by** *blast*
next
fix *m*
assume *d1*: ∀ *n*≤*m*. *Di n* ⊆ *Di m*
show ∀ *n*≤*Suc m*. *Di n* ⊆ *Di* (*Suc m*)
proof (*intro allI impI*)
fix *n*
assume *e1*: *n* ≤ *Suc m*
have *Di* (*Suc m*) = *H* (*Di m*) **using** *b8* **by** *simp*

moreover have *Di m* $\subseteq$ *H* (*Di m*) **using** *b5* **by** *blast*
ultimately have $n \leq m \longrightarrow$ *Di n* $\subseteq$ *Di* (*Suc m*) **using** *d1* **by** *blast*
moreover have $n =$ (*Suc m*) $\vee$ $n \leq m$ **using** *e1* **by** *force*
ultimately show *Di n* $\subseteq$ *Di* (*Suc m*) **by** *blast*
qed
qed
qed
have *Di 0* $\subseteq$ *D* **using** *b9* **by** *blast*
then have *b13*: *Field s* $\subseteq$ *D* **using** *b7 b8* **by** *simp*
then have *b14*: $s \subseteq s' \wedge s' \subseteq r$ **using** *a2 b10* **unfolding** *Field-def* **by** *force*
moreover have *b15*: $|D| \leq o\ |s|$
proof $-$
have $|UNIV::nat\ set| \leq o\ |s|$ **using** *a3 infinite-iff-card-of-nat* **by** *blast*
then have $|\bigcup\ n.\ Di\ n| \leq o\ |s|$ **using** *b11 a3 card-of-UNION-ordLeq-infinite*[*of s UNIV Di*] **by** *blast*
moreover have $D = (\bigcup\ n.\ Di\ n)$ **using** *b9* **by** *force*
ultimately show *?thesis* **by** *blast*
qed
moreover have $|s'| =o\ |s|$
proof $-$
have $\neg$ *finite* (*Field s*) **using** *a3 lem-fin-fl-rel* **by** *blast*
then have $\neg$ *finite D* **using** *b13 finite-subset* **by** *blast*
then have $|D \times D| =o\ |D|$ **by** *simp*
moreover have $s' \subseteq D \times D$ **using** *b10* **by** *blast*
ultimately have $|s'| \leq o\ |s|$ **using** *b15 card-of-mono1 ordLeq-ordIso-trans ordLeq-transitive* **by** *metis*
moreover have $|s| \leq o\ |s'|$ **using** *b14* **by** *simp*
ultimately show *?thesis* **using** *ordIso-iff-ordLeq* **by** *blast*
qed
moreover have $x \in$ *Field s'*
proof $-$
have *Di 0* $\subseteq$ *D* **using** *b9* **by** *blast*
then have $\{ax,\ bx\} \subseteq D$ **using** *b7 b8* **by** *simp*
then have $(ax,\ bx) \in s'$ **using** *b6 b10* **by** *blast*
then show *?thesis* **using** *b6* **unfolding** *Field-def* **by** *blast*
qed
moreover have *CCR s'*
proof $-$
have $\forall\ a \in$ *Field s'*. $\forall\ b \in$ *Field s'*. $\exists\ c \in$ *Field s'*. $(a,c) \in (s')\hat{}* \wedge (b,c) \in (s')\hat{}*$
proof (*intro ballI*)
fix *a b*
assume *d1*: $a \in$ *Field s'* **and** *d2*: $b \in$ *Field s'*
then have *d3*: $a \in D \wedge b \in D$ **using** *b10* **unfolding** *Field-def* **by** *blast*
then obtain *ia ib* **where** *d4*: $a \in$ *Di ia* $\wedge$ $b \in$ *Di ib* **using** *b9* **by** *blast*
obtain *k* **where** *d5*: $k =$ (*max ia ib*) **by** *blast*
then have $ia \leq k \wedge ib \leq k$ **by** *simp*
then have *d6*: $a \in$ *Di k* $\wedge$ $b \in$ *Di k* **using** *d4 b12* **by** *blast*
obtain *p* **where** *d7*: $p = g\ \{a,b\}$ **by** *blast*

have *Field p* $\subseteq$ *H* (*Di k*) **using** *b5 d6 d7* **by** *blast*
moreover have *H* (*Di k*) = *Di* (*Suc k*) **using** *b8* **by** *simp*
moreover have *Di* (*Suc k*) $\subseteq$ *D* **using** *b9* **by** *blast*
ultimately have *d8*: *Field p* $\subseteq$ *D* **by** *blast*
have {*a*, *b*} $\subseteq$ *Field r* **using** *d1 d2 b10* **unfolding** *Field-def* **by** *blast*
moreover have *finite* {*a*, *b*} **by** *simp*
ultimately have *d9*: *CCR p* $\wedge$ *p* $\subseteq$ *r* $\wedge$ {*a*,*b*} $\subseteq$ *Field p* **using** *d7 b3* **by** *blast*
then obtain *c* **where** *d10*: *c* $\in$ *Field p* $\wedge$ (*a*,*c*) $\in$ *p*^* $\wedge$ (*b*,*c*) $\in$ *p*^* **unfolding** *CCR-def* **by** *blast*
have (*p* `` *D*) $\subseteq$ *D* **using** *d8* **unfolding** *Field-def* **by** *blast*
then have *D* $\in$ *Inv p* **unfolding** *Inv-def* **by** *blast*
then have *p*^* $\cap$ (*D*$\times$(*UNIV*::'*U set*)) $\subseteq$ (*Restr p D*)^* **using** *lem-Inv-restr-rtr*[*of D p*] **by** *blast*
moreover have *Restr p D* $\subseteq$ *s'* **using** *d9 b10* **by** *blast*
moreover have (*a*,*c*) $\in$ *p*^* $\cap$ (*D*$\times$(*UNIV*::'*U set*)) $\wedge$ (*b*,*c*) $\in$ *p*^* $\cap$ (*D*$\times$(*UNIV*::'*U set*)) **using** *d10 d3* **by** *blast*
ultimately have (*a*,*c*) $\in$ (*s'*)^* $\wedge$ (*b*,*c*) $\in$ (*s'*)^* **using** *rtrancl-mono* **by** *blast*
moreover then have *c* $\in$ *Field s'* **using** *d1 lem-rtr-field* **by** *metis*
ultimately show $\exists$ *c* $\in$ *Field s'*. (*a*,*c*) $\in$ (*s'*)^* $\wedge$ (*b*,*c*) $\in$ (*s'*)^* **by** *blast*
qed
then show *?thesis* **unfolding** *CCR-def* **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Ccext-finsubccr-set-ext*:
fixes *r s*::'*U rel* **and** *A*::'*U set*
assumes *a1*: *CCR r* **and** *a2*: *s* $\subseteq$ *r* **and** *a3*: *finite s* **and** *a4*: *A* $\subseteq$ *Field r* **and** *a5*: *finite A*
shows $\exists$ *s'*::('*U rel*). *CCR s'* $\wedge$ *s* $\subseteq$ *s'* $\wedge$ *s'* $\subseteq$ *r* $\wedge$ *finite s'* $\wedge$ *A* $\subseteq$ *Field s'*
proof −
obtain *Pt*::'*U* $\Rightarrow$ '*U rel* **where** *p1*: *Pt* = (λ *x*. {*p* $\in$ *r*. *x* = *fst p* $\vee$ *x* = *snd p*}) **by** *blast*
obtain *pt*::'*U* $\Rightarrow$ '*U*$\times$'*U* **where** *p2*: *pt* = (λ *x*. (*SOME p*. *p* $\in$ *Pt x*)) **by** *blast*
have $\forall$ *x*$\in$*A*. *Pt x* $\neq$ {} **using** *a4* **unfolding** *p1 Field-def* **by** *force*
then have *p3*: $\forall$ *x*$\in$*A*. *pt x* $\in$ *Pt x* **unfolding** *p2* **by** (*metis* (*full-types*) *Collect-empty-eq Collect-mem-eq someI-ex*)
have *b2*: *pt*`*A* $\subseteq$ *r* **using** *p1 p3* **by** *blast*
obtain *s1* **where** *b3*: *s1* = *s* $\cup$ (*pt*`*A*) **by** *blast*
then have *finite s1* **using** *a3 a5* **by** *blast*
moreover have *s1* $\subseteq$ *r* **using** *b2 b3 a2* **by** *blast*
ultimately obtain *s'* **where** *b4*: *finite s'* $\wedge$ *CCR s'* $\wedge$ *s1* $\subseteq$ *s'* $\wedge$ *s'* $\subseteq$ *r* **using** *a1 lem-ccr-fin-subr-ext*[*of r s1*] **by** *blast*
moreover have *A* $\subseteq$ *Field s1*
proof
fix *x*
assume *c1*: *x* $\in$ *A*
then have *pt x* $\in$ *s1* **using** *b3* **by** *blast*
moreover obtain *ax bx* **where** *c2*: *pt x* = (*ax*,*bx*) **by** *force*

ultimately have *ax* ∈ *Field s1* ∧ *bx* ∈ *Field s1* **unfolding** *Field-def* **by** *force*
then show *x* ∈ *Field s1* **using** *c1 c2 p1 p3* **by** *force*
qed
ultimately have *A* ⊆ *Field s′* **unfolding** *Field-def* **by** *blast*
then show *?thesis* **using** *b3 b4* **by** *blast*
qed

lemma *lem-Ccext-infsubccr-set-ext*:
fixes *r s*::′*U rel* **and** *A*::′*U set*
assumes *a1*: *CCR r* **and** *a2*: *s* ⊆ *r* **and** *a3*: ¬ *finite s* **and** *a4*: *A* ⊆ *Field r* **and** *a5*: |*A*| ≤*o* |*Field s*|
shows ∃ *s′*::(′*U rel*). *CCR s′* ∧ *s* ⊆ *s′* ∧ *s′* ⊆ *r* ∧ |*s′*| =*o* |*s*| ∧ *A* ⊆ *Field s′*
proof −
obtain *G*::′*U set* ⇒ ′*U rel set* **where** *b1*: *G* = (λ *A*. {*t*::′*U rel*. *finite t* ∧ *CCR t* ∧ *t* ⊆ *r* ∧ *A* ⊆ *Field t*}) **by** *blast*
obtain *g*::′*U set* ⇒ ′*U rel* **where** *b2*: *g* = (λ *A*. *if A* ⊆ *Field r* ∧ *finite A then* (*SOME t*. *t* ∈ *G A*) *else* {}) **by** *blast*
have *b3*: ∀ *A*. *A* ⊆ *Field r* ∧ *finite A* ⟶ *finite* (*g A*) ∧ *CCR* (*g A*) ∧ (*g A*) ⊆ *r* ∧ *A* ⊆ *Field* (*g A*)
proof (*intro allI impI*)
fix *A*
assume *c1*: *A* ⊆ *Field r* ∧ *finite A*
then have *g A* = (*SOME t*. *t* ∈ *G A*) **using** *b2* **by** *simp*
moreover have *G A* ≠ {} **using** *b1 a1 c1 lem-Ccext-finsubccr-dext*[*of r A*] **by** *blast*
ultimately have *g A* ∈ *G A* **using** *some-in-eq* **by** *metis*
then show *finite* (*g A*) ∧ *CCR* (*g A*) ∧ (*g A*) ⊆ *r* ∧ *A* ⊆ *Field* (*g A*) **using** *b1* **by** *blast*
qed
have *b4*: ∀ *A*. ¬ (*A* ⊆ *Field r* ∧ *finite A*) ⟶ *g A* = {} **using** *b2* **by** *simp*
obtain *H*::′*U set* ⇒ ′*U set*
where *b5*: *H* = (λ *X*. *X* ∪ ⋃ {*S* . ∃ *a*∈*X*. ∃ *b*∈*X*. *S* = *Field* (*g* {*a*,*b*})}) **by** *blast*
obtain *Pt*::′*U* ⇒ ′*U rel* **where** *p1*: *Pt* = (λ *x*. {*p* ∈ *r*. *x* = *fst p* ∨ *x* = *snd p*}) **by** *blast*
obtain *pt*::′*U* ⇒ ′*U*×′*U* **where** *p2*: *pt* = (λ *x*. (*SOME p*. *p* ∈ *Pt x*)) **by** *blast*
have ∀ *x*∈*A*. *Pt x* ≠ {} **using** *a4* **unfolding** *p1 Field-def* **by** *force*
then have *p3*: ∀ *x*∈*A*. *pt x* ∈ *Pt x* **unfolding** *p2* **by** (*metis* (*full-types*) *Collect-empty-eq Collect-mem-eq someI-ex*)
obtain *D0* **where** *b7*: *D0* = *Field s* ∪ *fst*‘(*pt*‘*A*) ∪ *snd*‘(*pt*‘*A*) **by** *blast*
obtain *Di*::*nat* ⇒ ′*U set* **where** *b8*: *Di* = (λ *n*. (*H*^^*n*) *D0*) **by** *blast*
obtain *D*::′*U set* **where** *b9*: *D* = ⋃ {*X*. ∃ *n*. *X* = *Di n*} **by** *blast*
obtain *s′* **where** *b10*: *s′* = *Restr r D* **by** *blast*
have *b11*: ∀ *n*. (¬ *finite* (*Di n*)) ∧ |*Di n*| ≤*o* |*s*|
proof
fix *n0*
show (¬ *finite* (*Di n0*)) ∧ |*Di n0*| ≤*o* |*s*|
proof (*induct n0*)
have |*D0*| =*o* |*Field s*|

proof −
have $|fst`(pt`A)| \leq o\ |(pt`A)| \wedge |(pt`A)| \leq o\ |A|$ **by** *simp*
then have *c1*: $|fst`(pt`A)| \leq o\ |A|$ **using** *ordLeq-transitive* **by** *blast*
have $|snd`(pt`A)| \leq o\ |(pt`A)| \wedge |(pt`A)| \leq o\ |A|$ **by** *simp*
then have *c2*: $|snd`(pt`A)| \leq o\ |A|$ **using** *ordLeq-transitive* **by** *blast*
have $|fst`(pt`A)| \leq o\ |Field\ s| \wedge |snd`(pt`A)| \leq o\ |Field\ s|$
using *c1 c2 a5 ordLeq-transitive* **by** *blast*
moreover have ¬ *finite* (*Field s*) **using** *a3 lem-fin-fl-rel* **by** *blast*
ultimately have *c3*: $|D0| \leq o\ |Field\ s|$ **unfolding** *b7* **by** *simp*
have *Field s* ⊆ *D0* **unfolding** *b7* **by** *blast*
then have $|Field\ s| \leq o\ |D0|$ **by** *simp*
then show *?thesis* **using** *c3 ordIso-iff-ordLeq* **by** *blast*
qed
moreover have $|Field\ s| =o\ |s|$ **using** *a3 lem-rel-inf-fld-card* **by** *blast*
ultimately have $|D0| \leq o\ |s|$ **using** *ordIso-imp-ordLeq ordIso-transitive* **by** *blast*
moreover have ¬ *finite D0* **using** *a3 b7 lem-fin-fl-rel* **by** *blast*
ultimately show ¬ *finite* (*Di 0*) ∧ $|Di\ 0| \leq o\ |s|$ **using** *b8* **by** *simp*
next
fix *n*
assume *d1*: (¬ *finite* (*Di n*)) ∧ $|Di\ n| \leq o\ |s|$
moreover then have $|(Di\ n) \times (Di\ n)| =o\ |Di\ n|$ **by** *simp*
ultimately have *d2*: $|(Di\ n) \times (Di\ n)| \leq o\ |s|$ **using** *ordIso-imp-ordLeq ordLeq-transitive* **by** *blast*
have *d3*: ∀ *a* ∈ (*Di n*). ∀ *b* ∈ (*Di n*). $|Field\ (g\ \{a,\ b\})| \leq o\ |s|$
proof (*intro ballI*)
fix *a b*
assume *a* ∈ (*Di n*) **and** *b* ∈ (*Di n*)
have *finite* (*g* {*a*, *b*}) **using** *b3 b4* **by** (*metis finite.emptyI*)
then have *finite* (*Field* (*g* {*a*, *b*})) **using** *lem-fin-fl-rel* **by** *blast*
then have $|Field\ (g\ \{a,\ b\})| <o\ |s|$ **using** *a3 finite-ordLess-infinite2* **by** *blast*
then show $|Field\ (g\ \{a,\ b\})| \leq o\ |s|$ **using** *ordLess-imp-ordLeq* **by** *blast*
qed
have *d4*: *Di* (*Suc n*) = *H* (*Di n*) **using** *b8* **by** *simp*
then have *Di n* ⊆ *Di* (*Suc n*) **using** *b5* **by** *blast*
then have ¬ *finite* (*Di* (*Suc n*)) **using** *d1 finite-subset* **by** *blast*
moreover have $|Di\ (Suc\ n)| \leq o\ |s|$
proof −
obtain *I* **where** *e1*: *I* = (*Di n*) × (*Di n*) **by** *blast*
obtain *f* **where** *e2*: *f* = (λ (*a*,*b*). *Field* (*g* {*a*,*b*})) **by** *blast*
have $|I| \leq o\ |s|$ **using** *e1 d2* **by** *blast*
moreover have ∀ *i*∈*I*. $|f\ i| \leq o\ |s|$ **using** *e1 e2 d3* **by** *simp*
ultimately have $|\bigcup\ i{\in}I.\ f\ i| \leq o\ |s|$ **using** *a3 card-of-UNION-ordLeq-infinite*[*of s I f*] **by** *blast*
moreover have *Di* (*Suc n*) = (*Di n*) ∪ (⋃ *i*∈*I*. *f i*) **using** *e1 e2 d4 b5* **by** *blast*
ultimately show *?thesis* **using** *d1 a3* **by** *simp*
qed

ultimately show $(\neg$ *finite* $(Di\ (Suc\ n))) \wedge |Di\ (Suc\ n)| \leq o\ |s|$ **by** *blast*
qed
qed
have *b12*: $\forall\ m.\ \forall\ n.\ n \leq m \longrightarrow Di\ n \leq Di\ m$
proof
fix *m0*
show $\forall\ n.\ n \leq m0 \longrightarrow Di\ n \leq Di\ m0$
proof (*induct m0*)
show $\forall n \leq 0.\ Di\ n \subseteq Di\ 0$ **by** *blast*
next
fix *m*
assume *d1*: $\forall n \leq m.\ Di\ n \subseteq Di\ m$
show $\forall n \leq Suc\ m.\ Di\ n \subseteq Di\ (Suc\ m)$
proof (*intro allI impI*)
fix *n*
assume *e1*: $n \leq Suc\ m$
have $Di\ (Suc\ m) = H\ (Di\ m)$ **using** *b8* **by** *simp*
moreover have $Di\ m \subseteq H\ (Di\ m)$ **using** *b5* **by** *blast*
ultimately have $n \leq m \longrightarrow Di\ n \subseteq Di\ (Suc\ m)$ **using** *d1* **by** *blast*
moreover have $n = (Suc\ m) \vee n \leq m$ **using** *e1* **by** *force*
ultimately show $Di\ n \subseteq Di\ (Suc\ m)$ **by** *blast*
qed
qed
qed
have $Di\ 0 \subseteq D$ **using** *b9* **by** *blast*
then have *b13*: $Field\ s \subseteq D$ **using** *b7 b8* **by** *simp*
then have *b14*: $s \subseteq s' \wedge s' \subseteq r$ **using** *a2 b10* **unfolding** *Field-def* **by** *force*
moreover have *b15*: $|D| \leq o\ |s|$
proof –
have $|UNIV::nat\ set| \leq o\ |s|$ **using** *a3 infinite-iff-card-of-nat* **by** *blast*
then have $|\bigcup\ n.\ Di\ n| \leq o\ |s|$ **using** *b11 a3 card-of-UNION-ordLeq-infinite*[*of s UNIV Di*] **by** *blast*
moreover have $D = (\bigcup\ n.\ Di\ n)$ **using** *b9* **by** *force*
ultimately show *?thesis* **by** *blast*
qed
moreover have $|s'| =o\ |s|$
proof –
have $\neg$ *finite* (*Field s*) **using** *a3 lem-fin-fl-rel* **by** *blast*
then have $\neg$ *finite D* **using** *b13 finite-subset* **by** *blast*
then have $|D \times D| =o\ |D|$ **by** *simp*
moreover have $s' \subseteq D \times D$ **using** *b10* **by** *blast*
ultimately have $|s'| \leq o\ |s|$ **using** *b15 card-of-mono1 ordLeq-ordIso-trans ordLeq-transitive* **by** *metis*
moreover have $|s| \leq o\ |s'|$ **using** *b14* **by** *simp*
ultimately show *?thesis* **using** *ordIso-iff-ordLeq* **by** *blast*
qed
moreover have $A \subseteq Field\ s'$
proof
fix *x*

```
    assume c1: x ∈ A
    obtain ax bx where c2: ax = fst (pt x) ∧ bx = snd (pt x) by blast
    have pt x ∈ Pt x using c1 p3 by blast
    then have c3: (ax, bx) ∈ r ∧ x ∈ {ax,bx} using c2 p1 by simp
    have {ax, bx} ⊆ D0 using b7 c1 c2 by blast
    moreover have Di 0 ⊆ D using b9 by blast
    moreover have Di 0 = D0 using b8 by simp
    ultimately have {ax, bx} ⊆ D by blast
    then have (ax, bx) ∈ s' using c3 b10 by blast
    then show x ∈ Field s' using c3 unfolding Field-def by blast
  qed
  moreover have CCR s'
  proof −
    have ∀ a ∈ Field s'. ∀ b ∈ Field s'. ∃ c ∈ Field s'. (a,c) ∈ (s')^* ∧ (b,c) ∈ (s')^*
    proof (intro ballI)
      fix a b
      assume d1: a ∈ Field s' and d2: b ∈ Field s'
      then have d3: a ∈ D ∧ b ∈ D using b10 unfolding Field-def by blast
      then obtain ia ib where d4: a ∈ Di ia ∧ b ∈ Di ib using b9 by blast
      obtain k where d5: k = (max ia ib) by blast
      then have ia ≤ k ∧ ib ≤ k by simp
      then have d6: a ∈ Di k ∧ b ∈ Di k using d4 b12 by blast
      obtain p where d7: p = g {a,b} by blast
      have Field p ⊆ H (Di k) using b5 d6 d7 by blast
      moreover have H (Di k) = Di (Suc k) using b8 by simp
      moreover have Di (Suc k) ⊆ D using b9 by blast
      ultimately have d8: Field p ⊆ D by blast
      have {a, b} ⊆ Field r using d1 d2 b10 unfolding Field-def by blast
      moreover have finite {a, b} by simp
      ultimately have d9: CCR p ∧ p ⊆ r ∧ {a,b} ⊆ Field p using d7 b3 by blast
      then obtain c where d10: c ∈ Field p ∧ (a,c) ∈ p^* ∧ (b,c) ∈ p^* unfolding CCR-def by blast
      have (p `` D) ⊆ D using d8 unfolding Field-def by blast
      then have D ∈ Inv p unfolding Inv-def by blast
      then have p^* ∩ (D×(UNIV::'U set)) ⊆ (Restr p D)^* using lem-Inv-restr-rtr[of D p] by blast
      moreover have Restr p D ⊆ s' using d9 b10 by blast
      moreover have (a,c) ∈ p^* ∩ (D×(UNIV::'U set)) ∧ (b,c) ∈ p^* ∩ (D×(UNIV::'U set)) using d10 d3 by blast
      ultimately have (a,c) ∈ (s')^* ∧ (b,c) ∈ (s')^* using rtrancl-mono by blast
      moreover then have c ∈ Field s' using d1 lem-rtr-field by metis
      ultimately show ∃ c ∈ Field s'. (a,c) ∈ (s')^* ∧ (b,c) ∈ (s')^* by blast
    qed
    then show ?thesis unfolding CCR-def by blast
  qed
  ultimately show ?thesis by blast
qed
```

lemma *lem-Ccext-finsubccr-pext5*:
fixes *r*::′*U rel* **and** *A B*::′*U set* **and** *x*::′*U*
assumes *a1*: *CCR r* **and** *a2*: *finite A* **and** *a3*: *A* ∈ *SF r*
shows ∃ *A*′::(′*U set*). (*x* ∈ *Field r* ⟶ *x* ∈ *A*′) ∧ *A* ⊆ *A*′ ∧ *CCR* (*Restr r A*′) ∧ *finite A*′
∧ (∀ *a*∈*A*. *r*‘‘{*a*}⊆*B* ∨ *r*‘‘{*a*}∩(*A*′−*B*) ≠ {}) ∧ *A*′ ∈ *SF r*
∧ ((∃ *y*::′*U*. *A*′−*B* = {*y*}) ⟶ *Field r* ⊆ (*A*′∪*B*))
proof −
have *q1*: *Field* (*Restr r A*) = *A* **using** *a3* **unfolding** *SF-def* **by** *blast*
obtain *s* **where** *s* = (*Restr r A*) **by** *blast*
then have *q2*: *s* ⊆ *r* **and** *q3*: *finite s* **and** *q4*: *A* = *Field s*
using *a2 q1 lem-fin-fl-rel* **by** (*blast*, *metis*, *blast*)
obtain *S* **where** *b1*: *S* = (λ *a*. *r*‘‘{*a*} − *B*) **by** *blast*
obtain *S*′ **where** *b2*: *S*′ = (λ *a*. *if* (*S a*) ≠ {} *then* (*S a*) *else* {*a*}) **by** *blast*
obtain *f* **where** *f* = (λ *a*. *SOME b*. *b* ∈ *S*′ *a*) **by** *blast*
moreover have ∀ *a*. ∃ *b*. *b* ∈ (*S*′ *a*) **unfolding** *b2* **by** *force*
ultimately have ∀ *a*. *f a* ∈ *S*′ *a* **by** (*metis someI-ex*)
then have *b3*: ∀ *a*. (*S a* ≠ {} ⟶ *f a* ∈ *S a*) ∧ (*S a* = {} ⟶ *f a* = *a*)
unfolding *b2* **by** (*clarsimp*, *metis singletonD*)
obtain *y1 y2*::′*U* **where** *n1*: *Field r* ≠ {} ⟶ {*y1*, *y2*} ⊆ *Field r*
and *n2*: (¬ (∃ *y*::′*U*. *Field r* − *B* ⊆ {*y*})) ⟶ *y1* ∉ *B* ∧ *y2* ∉ *B* ∧ *y1* ≠ *y2* **by** *blast*
obtain *A1* **where** *b4*: *A1* = ({*x*,*y1*,*y2*} ∩ *Field r*) ∪ *A* ∪ (*f* ‘ *A*) **by** *blast*
have *A1* ⊆ *Field r*
proof −
have *c1*: *A* ⊆ *Field r* **using** *q4 q2* **unfolding** *Field-def* **by** *blast*
moreover have *f* ‘ *A* ⊆ *Field r*
proof
fix *x*
assume *x* ∈ *f* ‘ *A*
then obtain *a* **where** *d2*: *a* ∈ *A* ∧ *x* = *f a* **by** *blast*
show *x* ∈ *Field r*
proof (*cases S a* = {})
assume *S a* = {}
then have *x* = *a* **using** *c1 d2 b3* **by** *blast*
then show *x* ∈ *Field r* **using** *d2 c1* **by** *blast*
next
assume *S a* ≠ {}
then have *x* ∈ *S a* **using** *d2 b3* **by** *blast*
then show *x* ∈ *Field r* **using** *b1* **unfolding** *Field-def* **by** *blast*
qed
qed
ultimately show *A1* ⊆ *Field r* **using** *b4* **by** *blast*
qed
moreover have *s0*: *finite A1* **using** *b4 q3 q4 lem-fin-fl-rel* **by** *blast*
ultimately obtain *s*′ **where** *s1*: *CCR s*′ ∧ *s* ⊆ *s*′ ∧ *s*′ ⊆ *r* ∧ *finite s*′ ∧ *A1* ⊆ *Field s*′
using *a1 q2 q3 lem-Ccext-finsubccr-set-ext*[*of r s A1*] **by** *blast*
obtain *A*′ **where** *s2*: *A*′ = *Field s*′ **by** *blast*

obtain s'' **where** $s3$: $s'' = Restr\ r\ A'$ **by** *blast*
then have $s4$: $s' \subseteq s'' \wedge Field\ s'' = A'$ **using** *s1 s2 lem-Relprop-fld-sat*[*of* s' r s''] **by** *blast*
have $s5$: *finite* (*Field* s') **using** *s1 lem-fin-fl-rel* **by** *blast*
have $A1 \cup (\{x\} \cap Field\ r) \subseteq A'$ **using** *b4 s1 s2* **by** *blast*
moreover have *CCR* (*Restr r* A')
proof −
have *CCR* s'' **using** *s1 s2 s4 lem-Ccext-subccr-eqfld*[*of* s' s''] **by** *blast*
then show *?thesis* **using** *s3* **by** *blast*
qed
ultimately have $b6$: $A1 \cup (\{x\} \cap Field\ r) \subseteq A' \wedge CCR\ (Restr\ r\ A')$ **by** *blast*
moreover then have $A \cup (\{x\} \cap Field\ r) \subseteq A'$ **using** *b4* **by** *blast*
moreover have *finite* A' **using** *s2 s5* **by** *blast*
moreover have $\forall\, a{\in}A.\ r``\{a\} \subseteq B \vee r``\{a\} \cap (A'{-}B) \neq \{\}$
proof
fix *a*
assume $c1$: $a \in A$
have $\neg\ (r``\{a\} \subseteq B) \longrightarrow r``\{a\} \cap (A'{-}B) \neq \{\}$
proof
assume $\neg\ (r``\{a\} \subseteq B)$
then have $S\ a \neq \{\}$ **unfolding** *b1* **by** *blast*
then have $f\ a \in r``\{a\} - B$ **using** *b1 b3* **by** *blast*
moreover have $f\ a \in A'$ **using** *c1 b4 b6* **by** *blast*
ultimately show $r``\{a\} \cap (A'{-}B) \neq \{\}$ **by** *blast*
qed
then show $r``\{a\} \subseteq B \vee r``\{a\} \cap (A'{-}B) \neq \{\}$ **by** *blast*
qed
moreover have $A' \in SF\ r$ **using** *s3 s4* **unfolding** *SF-def* **by** *blast*
moreover have $(\exists\ y{::}'U.\ A' - B = \{y\}) \longrightarrow Field\ r \subseteq (A' \cup B)$
proof
assume $c1$: $\exists\ y{::}'U.\ A' - B = \{y\}$
moreover have $c2$: $A' \subseteq Field\ r$ **using** *s1 s2* **unfolding** *Field-def* **by** *blast*
ultimately have $Field\ r \neq \{\}$ **by** *blast*
then have $\{y1,\ y2\} \subseteq Field\ r$ **using** *n1* **by** *blast*
then have $\{y1,\ y2\} \subseteq A'$ **using** *b4 s1 s2* **by** *fast*
then have $\neg\ (\exists\, y.\ Field\ r - B \subseteq \{y\}) \longrightarrow \{y1,\ y2\} \subseteq A' - B \wedge y1 \neq y2$ **using** *n2* **by** *blast*
moreover have $\neg\ (\{y1,\ y2\} \subseteq A' - B \wedge y1 \neq y2)$ **using** *c1* **by** *force*
ultimately have $\exists\ y{::}'U.\ Field\ r - B \subseteq \{y\}$ **by** *blast*
then show $Field\ r \subseteq A' \cup B$ **using** *c1 c2* **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Ccext-infsubccr-pext5*:
fixes $r{::}'U\ rel$ **and** $A\ B{::}'U\ set$ **and** $x{::}'U$
assumes $a1$: *CCR r* **and** $a2$: $\neg$ *finite A* **and** $a3$: $A \in SF\ r$
shows $\exists\ A'{::}('U\ set).\ (x \in Field\ r \longrightarrow x \in A') \wedge A \subseteq A' \wedge CCR\ (Restr\ r\ A') \wedge |A'| =o\ |A|$

```
                  ∧ (∀ a∈A. r``{a}⊆B ∨ r``{a}∩(A'−B) ≠ {}) ∧ A' ∈ SF r
                  ∧ ((∃ y::'U. A'−B = {y}) ⟶ Field r ⊆ (A'∪B))
proof −
  have q1: Field (Restr r A) = A using a3 unfolding SF-def by blast
  obtain s where s = (Restr r A) by blast
  then have q2: s ⊆ r and q3: ¬ finite s and q4: A = Field s
    using a2 q1 lem-fin-fl-rel by (blast, metis, blast)
  obtain S where b1: S = (λ a. r``{a} − B ) by blast
  obtain S' where b2: S' = (λ a. if (S a) ≠ {} then (S a) else {a}) by blast
  obtain f where f = (λ a. SOME b. b ∈ S' a) by blast
  moreover have ∀ a. ∃ b. b ∈ (S' a) unfolding b2 by force
  ultimately have ∀ a. f a ∈ S' a by (metis someI-ex)
  then have b3: ∀ a. (S a ≠ {} ⟶ f a ∈ S a) ∧ (S a = {} ⟶ f a = a)
    unfolding b2 by (clarsimp, metis singletonD)
  obtain y1 y2::'U where n1: Field r ≠ {} ⟶ {y1, y2} ⊆ Field r
                  and n2: (¬ (∃ y::'U. Field r − B ⊆ {y})) ⟶ y1 ∉ B ∧ y2 ∉ B
∧ y1 ≠ y2 by blast
  obtain A1 where b4: A1 = ({x, y1, y2} ∩ Field r) ∪ A ∪ (f ` A) by blast
  have A1 ⊆ Field r
  proof −
    have c1: A ⊆ Field r using q4 q2 unfolding Field-def by blast
    moreover have f ` A ⊆ Field r
    proof
      fix x
      assume x ∈ f ` A
      then obtain a where d2: a ∈ A ∧ x = f a by blast
      show x ∈ Field r
      proof (cases S a = {})
        assume S a = {}
        then have x = a using c1 d2 b3 by blast
        then show x ∈ Field r using d2 c1 by blast
      next
        assume S a ≠ {}
        then have x ∈ S a using d2 b3 by blast
        then show x ∈ Field r using b1 unfolding Field-def by blast
      qed
    qed
    ultimately show A1 ⊆ Field r using b4 by blast
  qed
  moreover have s0: |A1| ≤o |Field s|
  proof −
    obtain C1 where c1: C1 = {x,y1,y2} ∩ Field r by blast
    obtain C2 where c2: C2 = A ∪ f ` A by blast
    have ¬ finite A using q4 q3 lem-fin-fl-rel by blast
    then have |C2| =o |A| using c2 b4 q3 by simp
    then have |C2| ≤o |Field s| unfolding q4 using ordIso-iff-ordLeq by blast
    moreover have c3: ¬ finite (Field s) using q3 lem-fin-fl-rel by blast
    moreover have |C1| ≤o |Field s|
    proof −
```

have |{x,$y1$,$y2$}| ≤o |*Field s*| **using** *c3*
by (*meson card-of-Well-order card-of-ordLeq-finite finite.emptyI finite.insertI ordLeq-total*)
moreover have |*C1*| ≤o |{x,$y1$,$y2$}| **unfolding** *c1* **by** *simp*
ultimately show *?thesis* **using** *ordLeq-transitive* **by** *blast*
qed
ultimately have |*C1* ∪ *C2*| ≤o |*Field s*| **unfolding** *b4* **using** *card-of-Un-ordLeq-infinite* **by** *blast*
moreover have *A1* = *C1* ∪ *C2* **using** *c1 c2 b4* **by** *blast*
ultimately show *?thesis* **by** *blast*
qed
ultimately obtain *s′* **where** *s1*: *CCR s′* ∧ *s* ⊆ *s′* ∧ *s′* ⊆ *r* ∧ |*s′*| =o |*s*| ∧ *A1* ⊆ *Field s′*
using *a1 q2 q3 lem-Ccext-infsubccr-set-ext*[*of r s A1*] **by** *blast*
obtain *A′* **where** *s2*: *A′* = *Field s′* **by** *blast*
obtain *s″* **where** *s3*: *s″* = *Restr r A′* **by** *blast*
then have *s4*: *s′* ⊆ *s″* ∧ *Field s″* = *A′* **using** *s1 s2 lem-Relprop-fld-sat*[*of s′ r s″*] **by** *blast*
have *s5*: |*Field s′*| =o |*Field s*| **using** *s1 q3 lem-cardreleq-cardfldeq-inf*[*of s′ s*] **by** *blast*
have *A1* ∪ ({x} ∩ *Field r*) ⊆ *A′* **using** *b4 s1 s2* **by** *blast*
moreover have *CCR* (*Restr r A′*)
proof −
have *CCR s″* **using** *s1 s2 s4 lem-Ccext-subccr-eqfld*[*of s′ s″*] **by** *blast*
then show *?thesis* **using** *s3* **by** *blast*
qed
moreover have |*A′*| =o |*A1*|
proof −
have *Field s* ⊆ *A1* **using** *q4 b4* **by** *blast*
then have |*Field s*| ≤o |*A1*| **by** *simp*
then have |*A′*| ≤o |*A1*| **using** *s2 s5 ordIso-ordLeq-trans* **by** *blast*
moreover have |*A1*| ≤o |*A′*| **using** *s1 s2* **by** *simp*
ultimately show *?thesis* **using** *ordIso-iff-ordLeq* **by** *blast*
qed
ultimately have *b6*: *A1* ∪ ({x} ∩ *Field r*) ⊆ *A′* ∧ *CCR* (*Restr r A′*) ∧ |*A′*| =o |*A1*| **by** *blast*
moreover then have *A* ∪ ({x} ∩ *Field r*) ⊆ *A′* **using** *b4* **by** *blast*
moreover have |*A′*| =o |*A*| **using** *s5 s2 q4* **by** *blast*
moreover have ∀ *a*∈*A*. *r*‘‘{*a*} ⊆ *B* ∨ *r*‘‘{*a*} ∩ (*A′*−*B*) ≠ {}
proof
fix *a*
assume *c1*: *a* ∈ *A*
have ¬ (*r*‘‘{*a*} ⊆ *B*) ⟶ *r*‘‘{*a*} ∩ (*A′*−*B*) ≠ {}
proof
assume ¬ (*r*‘‘{*a*} ⊆ *B*)
then have *S a* ≠ {} **unfolding** *b1* **by** *blast*
then have *f a* ∈ *r*‘‘{*a*} − *B* **using** *b1 b3* **by** *blast*
moreover have *f a* ∈ *A′* **using** *c1 b4 b6* **by** *blast*
ultimately show *r*‘‘{*a*} ∩ (*A′*−*B*) ≠ {} **by** *blast*

    qed
    then show $r``\{a\} \subseteq B \vee r``\{a\} \cap (A'{-}B) \neq \{\}$ **by** *blast*
  qed
  moreover have $A' \in SF\ r$ **using** *s3 s4* **unfolding** *SF-def* **by** *blast*
  moreover have $(\exists\ y{::}'U.\ A' - B = \{y\}) \longrightarrow Field\ r \subseteq (A' \cup B)$
  proof
    assume *c1*: $\exists\ y{::}'U.\ A' - B = \{y\}$
    moreover have *c2*: $A' \subseteq Field\ r$ **using** *s1 s2* **unfolding** *Field-def* **by** *blast*
    ultimately have $Field\ r \neq \{\}$ **by** *blast*
    then have $\{y1,\ y2\} \subseteq Field\ r$ **using** *n1* **by** *blast*
    then have $\{y1,\ y2\} \subseteq A'$ **using** *b4 s1 s2* **by** *fast*
    then have $\neg\ (\exists\, y.\ Field\ r - B \subseteq \{y\}) \longrightarrow \{y1,\ y2\} \subseteq A' - B \wedge y1 \neq y2$
using *n2* **by** *blast*
    moreover have $\neg\ (\{y1,\ y2\} \subseteq A' - B \wedge y1 \neq y2)$ **using** *c1* **by** *force*
    ultimately have $\exists\ y{::}'U.\ Field\ r - B \subseteq \{y\}$ **by** *blast*
    then show $Field\ r \subseteq A' \cup B$ **using** *c1 c2* **by** *blast*
  qed
  ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Ccext-subccr-pext5*:
fixes $r{::}'U\ rel$ **and** $A\ B{::}'U\ set$ **and** $x{::}'U$
assumes $CCR\ r$ **and** $A \in SF\ r$
shows $\exists\ A'{::}('U\ set).\ (x \in Field\ r \longrightarrow x \in A')$
$\wedge\ A \subseteq A'$
$\wedge\ A' \in SF\ r$
$\wedge\ (\forall\, a{\in}A.\ ((r``\{a\}{\subseteq}B) \vee (r``\{a\}{\cap}(A'{-}B) \neq \{\})))$
$\wedge\ ((\exists\ y{::}'U.\ A'{-}B = \{y\}) \longrightarrow Field\ r \subseteq (A'{\cup}B))$
$\wedge\ CCR\ (Restr\ r\ A')$
$\wedge\ ((finite\ A \longrightarrow finite\ A') \wedge (\ (\neg\ finite\ A) \longrightarrow |A'| =o\ |A|\ ))$
proof (*cases finite A*)
  assume *finite A*
  then show *?thesis* **using** *assms lem-Ccext-finsubccr-pext5*[*of r A x B*] **by** *blast*
next
  assume $\neg$ *finite A*
  then show *?thesis* **using** *assms lem-Ccext-infsubccr-pext5*[*of r A x B*] **by** *blast*
qed

lemma *lem-Ccext-finsubccr-set-ext-scf*:
fixes $r\ s{::}'U\ rel$ **and** $A\ P{::}'U\ set$
assumes *a1*: $CCR\ r$ **and** *a2*: $s \subseteq r$ **and** *a3*: *finite s* **and** *a4*: $A \subseteq Field\ r$ **and**
a5: *finite A*
    and *a6*: $P \in SCF\ r$
shows $\exists\ s'{::}('U\ rel).\ CCR\ s' \wedge s \subseteq s' \wedge s' \subseteq r \wedge finite\ s' \wedge A \subseteq Field\ s'$
$\wedge\ ((Field\ s' \cap P) \in SCF\ s')$
proof (*cases* $s = \{\} \wedge A = \{\}$)
  assume $s = \{\} \wedge A = \{\}$
  moreover obtain $s'{::}'U\ rel$ **where** $s' = \{\}$ **by** *blast*
  ultimately have $CCR\ s' \wedge s \subseteq s' \wedge s' \subseteq r \wedge finite\ s' \wedge A \subseteq Field\ s'$

$\wedge$ ((*Field* s' $\cap$ *P*) $\in$ *SCF* s') **unfolding** *CCR-def SCF-def Field-def*
by *blast*
  then show *?thesis* **by** *blast*
next
  assume *b1*: $\neg$ (s = {} $\wedge$ A = {})
  obtain *Pt*::$'U \Rightarrow 'U$ *rel* **where** *p1*: *Pt* = (λ *x*. {*p* $\in$ *r*. *x* = *fst p* $\vee$ *x* = *snd p*})
by *blast*
  obtain *pt*::$'U \Rightarrow 'U \times 'U$ **where** *p2*: *pt* = (λ *x*. (*SOME p*. *p* $\in$ *Pt x*)) **by** *blast*
  have $\forall$ *x*$\in$*A*. *Pt x* $\neq$ {} **using** *a4* **unfolding** *p1 Field-def* **by** *force*
  then have *p3*: $\forall$ *x*$\in$*A*. *pt x* $\in$ *Pt x* **unfolding** *p2* **by** (*metis* (*full-types*) *Collect-empty-eq Collect-mem-eq someI-ex*)
  have *b2*: *pt*‘*A* $\subseteq$ *r* **using** *p1 p3* **by** *blast*
  obtain *s1* **where** *b3*: *s1* = *s* $\cup$ (*pt*‘*A*) **by** *blast*
  then have *finite s1* **using** *a3 a5* **by** *blast*
  moreover have *s1* $\subseteq$ *r* **using** *b2 b3 a2* **by** *blast*
  ultimately obtain *s2* **where** *b4*: *finite s2* $\wedge$ *CCR s2* $\wedge$ *s1* $\subseteq$ *s2* $\wedge$ *s2* $\subseteq$ *r* **using** *a1 lem-ccr-fin-subr-ext*[*of r s1*] **by** *blast*
  moreover have *A* $\subseteq$ *Field s1*
  proof
    fix *x*
    assume *c1*: *x* $\in$ *A*
    then have *pt x* $\in$ *s1* **using** *b3* **by** *blast*
    moreover obtain *ax bx* **where** *c2*: *pt x* = (*ax*,*bx*) **by** *force*
    ultimately have *ax* $\in$ *Field s1* $\wedge$ *bx* $\in$ *Field s1* **unfolding** *Field-def* **by** *force*
    then show *x* $\in$ *Field s1* **using** *c1 c2 p1 p3* **by** *force*
  qed
  ultimately have *b5*: *A* $\subseteq$ *Field s2* **unfolding** *Field-def* **by** *blast*
  have *Conelike s2* **using** *b4 lem-Relprop-fin-ccr* **by** *blast*
  moreover have *s2* $\neq$ {} **using** *b1 b3 b4* **unfolding** *Field-def* **by** *blast*
  ultimately obtain *m* **where** *b6*: *m* $\in$ *Field s2* $\wedge$ ($\forall$ *a*$\in$*Field s2*. (*a*,*m*) $\in$ *s2*^*)
unfolding *Conelike-def* **by** *blast*
  then have *m* $\in$ *Field r* **using** *b4* **unfolding** *Field-def* **by** *blast*
  then obtain m' **where** *b7*: m' $\in$ *P* $\wedge$ (*m*,m') $\in$ *r*^* **using** *a6* **unfolding** *SCF-def*
by *blast*
  obtain *D* **where** *b8*: *D* = *Field s2* $\cup$ ($\mathfrak{f}$ *r m* m') **by** *blast*
  obtain s' **where** *b9*: s' = *Restr r D* **by** *blast*
  have *b10*: *s2* $\subseteq$ s' **using** *b4 b8 b9* **unfolding** *Field-def* **by** *force*
  have *b11*: $\forall$ *a* $\in$ *Field* s'. (*a*,m') $\in$ s'^*
  proof
    fix *a*
    assume *c1*: *a* $\in$ *Field* s'
    have *c2*: *Restr r* ($\mathfrak{f}$ *r m* m') $\subseteq$ s' **using** *b8 b9* **by** *blast*
    then have *c3*: (*m*,m') $\in$ s'^* **using** *b7 lem-Ccext-fint*[*of r m* m' s'] **by** *blast*
    show (*a*,m') $\in$ s'^*
    proof (*cases a* $\in$ *Field s2*)
      assume *a* $\in$ *Field s2*
      then have (*a*,*m*) $\in$ *s2*^* **using** *b6* **by** *blast*
      then have (*a*,*m*) $\in$ s'^* **using** *b10 rtrancl-mono* **by** *blast*
      then show (*a*,m') $\in$ s'^* **using** *c3* **by** *simp*

```
    next
      assume a ∉ Field s2
      then have a ∈ (f r m m′) using c1 b8 b9 unfolding Field-def by blast
      then show (a,m′) ∈ s′^* using c2 b7 lem-Ccext-fint[of r m m′ s′] by blast
    qed
  qed
  have b12: m′ ∈ Field s′
  proof −
    have m ∈ Field s′ using b6 b10 unfolding Field-def by blast
    then have m ∈ Field s′ ∧ (m,m′) ∈ s′^* using b11 by blast
    then show m′ ∈ Field s′ using lem-rtr-field by force
  qed
  have Field s ⊆ D using b3 b4 b8 unfolding Field-def by blast
  then have s ⊆ s′ using a2 b9 unfolding Field-def by force
  moreover have s′ ⊆ r using b9 by blast
  moreover have finite s′
  proof −
    have finite (Field s2) using b4 lem-fin-fl-rel by blast
    then have finite D using b8 lem-ccext-ffin by simp
    then show ?thesis using b9 by blast
  qed
  moreover have A ⊆ Field s′ using b5 b10 unfolding Field-def by blast
  moreover have CCR s′
  proof −
    have Conelike s′ using b11 b12 unfolding Conelike-def by blast
    then show ?thesis using lem-Relprop-cl-ccr by blast
  qed
  moreover have (Field s′ ∩ P) ∈ SCF s′ using b7 b11 b12 unfolding SCF-def
by blast
  ultimately show ?thesis by blast
qed

lemma lem-ccext-scf-sat:
assumes s ⊆ r and Field s = Field r
shows SCF s ⊆ SCF r
  using assms rtrancl-mono unfolding SCF-def by blast

lemma lem-Ccext-infsubccr-set-ext-scf2:
fixes r s::′U rel and A::′U set and Ps::′U set set
assumes a1: CCR r and a2: s ⊆ r and a3: ¬ finite s and a4: A ⊆ Field r
    and a5: |A| ≤o |Field s| and a6: Ps ⊆ SCF r ∧ |Ps| ≤o |Field s|
shows ∃ s′::(′U rel). CCR s′ ∧ s ⊆ s′ ∧ s′ ⊆ r ∧ |s′| =o |s| ∧ A ⊆ Field s′
            ∧ (∀ P∈Ps. (Field s′ ∩ P) ∈ SCF s′)
proof −
  obtain q where q0: q = (λ P a. SOME p. p ∈ P ∧ (a, p) ∈ r^*) by blast
  have q1: ∀ P∈Ps. ∀ a∈Field r. (q P a) ∈ Field r ∧ (q P a) ∈ P ∧ (a, q P a)
∈ r^*
  proof (intro ballI)
    fix P a
```

assume *P* ∈ *Ps* **and** *a* ∈ *Field r*
then show (*q P a*) ∈ *Field r* ∧ (*q P a*) ∈ *P* ∧ (*a*, *q P a*) ∈ *r^**
using *q0 a6 someI-ex*[*of* λ *p*. *p* ∈ *P* ∧ (*a*,*p*) ∈ *r^**] **unfolding** *SCF-def* **by** *blast*
qed
obtain *G*::'*U set* ⇒ '*U rel set* **where** *b1*: *G* = (λ *A*. {*t*::'*U rel*. *finite t* ∧ *CCR t* ∧ *t* ⊆ *r* ∧ *A* ⊆ *Field t*}) **by** *blast*
obtain *g*::'*U set* ⇒ '*U rel* **where** *b2*: *g* = (λ *A*. *if A* ⊆ *Field r* ∧ *finite A then* (*SOME t*. *t* ∈ *G A*) *else* {}) **by** *blast*
have *b3*: ∀ *A*. *A* ⊆ *Field r* ∧ *finite A* ⟶ *finite* (*g A*) ∧ *CCR* (*g A*) ∧ (*g A*) ⊆ *r* ∧ *A* ⊆ *Field* (*g A*)
proof (*intro allI impI*)
fix *A*
assume *c1*: *A* ⊆ *Field r* ∧ *finite A*
then have *g A* = (*SOME t*. *t* ∈ *G A*) **using** *b2* **by** *simp*
moreover have *G A* ≠ {} **using** *b1 a1 c1 lem-Ccext-finsubccr-dext*[*of r A*] **by** *blast*
ultimately have *g A* ∈ *G A* **using** *some-in-eq* **by** *metis*
then show *finite* (*g A*) ∧ *CCR* (*g A*) ∧ (*g A*) ⊆ *r* ∧ *A* ⊆ *Field* (*g A*) **using** *b1* **by** *blast*
qed
have *b4*: ∀ *A*. ¬ (*A* ⊆ *Field r* ∧ *finite A*) ⟶ *g A* = {} **using** *b2* **by** *simp*
obtain *H*::'*U set* ⇒ '*U set*
where *b5*: *H* = (λ *X*. *X* ∪ ⋃ {*S* . ∃ *a*∈*X*. ∃ *b*∈*X*. *S* = *Field* (*g* {*a*,*b*})} ∪ ⋃ {*S*. ∃ *P*∈*Ps*. ∃ *a*∈*X*. *S* = 𝔣 *r a* (*q P a*) }) **by** *blast*
obtain *Pt*::'*U* ⇒ '*U rel* **where** *p1*: *Pt* = (λ *x*. {*p* ∈ *r*. *x* = *fst p* ∨ *x* = *snd p*}) **by** *blast*
obtain *pt*::'*U* ⇒ '*U*×'*U* **where** *p2*: *pt* = (λ *x*. (*SOME p*. *p* ∈ *Pt x*)) **by** *blast*
have ∀ *x*∈*A*. *Pt x* ≠ {} **using** *a4* **unfolding** *p1 Field-def* **by** *force*
then have *p3*: ∀ *x*∈*A*. *pt x* ∈ *Pt x* **unfolding** *p2* **by** (*metis* (*full-types*) *Collect-empty-eq Collect-mem-eq someI-ex*)
obtain *D0* **where** *b7*: *D0* = *Field s* ∪ *fst*‘(*pt*‘*A*) ∪ *snd*‘(*pt*‘*A*) **by** *blast*
obtain *Di*::*nat* ⇒ '*U set* **where** *b8*: *Di* = (λ *n*. (*H*^^*n*) *D0*) **by** *blast*
obtain *D*::'*U set* **where** *b9*: *D* = ⋃ {*X*. ∃ *n*. *X* = *Di n*} **by** *blast*
obtain *s'* **where** *b10*: *s'* = *Restr r D* **by** *blast*
have *b11*: ∀ *n*. (¬ *finite* (*Di n*)) ∧ |*Di n*| ≤*o* |*s*|
proof
fix *n0*
show (¬ *finite* (*Di n0*)) ∧ |*Di n0*| ≤*o* |*s*|
proof (*induct n0*)
have |*D0*| =*o* |*Field s*|
proof −
have |*fst*‘(*pt*‘*A*)| ≤*o* |(*pt*‘*A*)| ∧ |(*pt*‘*A*)| ≤*o* |*A*| **by** *simp*
then have *c1*: |*fst*‘(*pt*‘*A*)| ≤*o* |*A*| **using** *ordLeq-transitive* **by** *blast*
have |*snd*‘(*pt*‘*A*)| ≤*o* |(*pt*‘*A*)| ∧ |(*pt*‘*A*)| ≤*o* |*A*| **by** *simp*
then have *c2*: |*snd*‘(*pt*‘*A*)| ≤*o* |*A*| **using** *ordLeq-transitive* **by** *blast*
have |*fst*‘(*pt*‘*A*)| ≤*o* |*Field s*| ∧ |*snd*‘(*pt*‘*A*)| ≤*o* |*Field s*|
using *c1 c2 a5 ordLeq-transitive* **by** *blast*
moreover have ¬ *finite* (*Field s*) **using** *a3 lem-fin-fl-rel* **by** *blast*

ultimately have *c3*: $|D0| \leq o\ |Field\ s|$ **unfolding** *b7* **by** *simp*
have *Field s* $\subseteq$ *D0* **unfolding** *b7* **by** *blast*
then have $|Field\ s| \leq o\ |D0|$ **by** *simp*
then show *?thesis* **using** *c3 ordIso-iff-ordLeq* **by** *blast*
qed
moreover have $|Field\ s| =o\ |s|$ **using** *a3 lem-rel-inf-fld-card* **by** *blast*
ultimately have $|D0| \leq o\ |s|$ **using** *ordIso-imp-ordLeq ordIso-transitive* **by** *blast*
moreover have ¬ *finite D0* **using** *a3 b7 lem-fin-fl-rel* **by** *blast*
ultimately show ¬ *finite* (*Di 0*) ∧ $|Di\ 0| \leq o\ |s|$ **using** *b8* **by** *simp*
next
fix *n*
assume *d1*: (¬ *finite* (*Di n*)) ∧ $|Di\ n| \leq o\ |s|$
moreover then have $|(Di\ n) \times (Di\ n)| =o\ |Di\ n|$ **by** *simp*
ultimately have *d2*: $|(Di\ n) \times (Di\ n)| \leq o\ |s|$ **using** *ordIso-imp-ordLeq ordLeq-transitive* **by** *blast*
have *d3*: ∀ *a* ∈ (*Di n*). ∀ *b* ∈ (*Di n*). $|Field\ (g\ \{a,\ b\})| \leq o\ |s|$
proof (*intro ballI*)
fix *a b*
assume *a* ∈ (*Di n*) **and** *b* ∈ (*Di n*)
have *finite* (*g* {*a*, *b*}) **using** *b3 b4* **by** (*metis finite.emptyI*)
then have *finite* (*Field* (*g* {*a*, *b*})) **using** *lem-fin-fl-rel* **by** *blast*
then have $|Field\ (g\ \{a,\ b\})| <o\ |s|$ **using** *a3 finite-ordLess-infinite2* **by** *blast*
then show $|Field\ (g\ \{a,\ b\})| \leq o\ |s|$ **using** *ordLess-imp-ordLeq* **by** *blast*
qed
have *d4*: *Di* (*Suc n*) = *H* (*Di n*) **using** *b8* **by** *simp*
then have *Di n* ⊆ *Di* (*Suc n*) **using** *b5* **by** *blast*
then have ¬ *finite* (*Di* (*Suc n*)) **using** *d1 finite-subset* **by** *blast*
moreover have $|Di\ (Suc\ n)| \leq o\ |s|$
proof −
obtain *I* **where** *e1*: *I* = (*Di n*) × (*Di n*) **by** *blast*
obtain *f* **where** *e2*: *f* = (λ (*a*,*b*). *Field* (*g* {*a*,*b*})) **by** *blast*
have $|I| \leq o\ |s|$ **using** *e1 d2* **by** *blast*
moreover have ∀ *i*∈*I*. $|f\ i| \leq o\ |s|$ **using** *e1 e2 d3* **by** *simp*
ultimately have $|\bigcup\ i{\in}I.\ f\ i| \leq o\ |s|$ **using** *a3 card-of-UNION-ordLeq-infinite*[*of s I f*] **by** *blast*
moreover have *Di* (*Suc n*) = (*Di n*) ∪ (⋃ *i*∈*I*. *f i*) ∪ (⋃ *P*∈*Ps*. (⋃ *a*∈(*Di n*). 𝔣 *r a* (*q P a*)))
using *e1 e2 d4 b5* **by** *blast*
moreover have $|\bigcup\ P{\in}Ps.\ (\bigcup\ a{\in}(Di\ n).\ \mathfrak{f}\ r\ a\ (q\ P\ a))| \leq o\ |s|$
proof −
have ⋀ *P*. *P* ∈ *Ps* ⟹ ∀ *a*∈(*Di n*). $|\mathfrak{f}\ r\ a\ (q\ P\ a)| \leq o\ |s|$
using *a3 lem-ccext-ffin* **by** (*metis card-of-Well-order card-of-ordLeq-infinite ordLeq-total*)
then have ⋀ *P*. *P* ∈ *Ps* ⟹ $|\bigcup\ a{\in}(Di\ n).\ \mathfrak{f}\ r\ a\ (q\ P\ a)| \leq o\ |s|$
using *d1 a3 card-of-UNION-ordLeq-infinite*[*of s Di n* λ *a*. 𝔣 *r a* (*q* - *a*)] **by** *blast*
moreover have $|Ps| \leq o\ |s|$ **using** *a3 a6 lem-rel-inf-fld-card*[*of s*]

lem-fin-fl-rel[*of s*]
            by (*metis ordIso-iff-ordLeq ordLeq-transitive*)
          ultimately show *?thesis*
            using *a3 card-of-UNION-ordLeq-infinite*[*of s Ps* λ *P*. $\bigcup$ *a*$\in$(*Di n*). $\mathfrak{f}$ *r a* (*q P a*)] **by** *blast*
        qed
        ultimately show *?thesis* **using** *d1 a3* **by** *simp*
      qed
      ultimately show ($\neg$ *finite* (*Di* (*Suc n*))) $\wedge$ |*Di* (*Suc n*)| $\leq o$ |*s*| **by** *blast*
    qed
  qed
  have *b12*: $\forall$ *m*. $\forall$ *n*. $n \leq m \longrightarrow$ *Di n* $\leq$ *Di m*
  proof
    fix *m0*
    show $\forall$ *n*. $n \leq m0 \longrightarrow$ *Di n* $\leq$ *Di m0*
    proof (*induct m0*)
      show $\forall n \leq 0$. *Di n* $\subseteq$ *Di 0* **by** *blast*
    next
      fix *m*
      assume *d1*: $\forall n \leq m$. *Di n* $\subseteq$ *Di m*
      show $\forall n \leq$*Suc m*. *Di n* $\subseteq$ *Di* (*Suc m*)
      proof (*intro allI impI*)
        fix *n*
        assume *e1*: $n \leq$ *Suc m*
        have *Di* (*Suc m*) = *H* (*Di m*) **using** *b8* **by** *simp*
        moreover have *Di m* $\subseteq$ *H* (*Di m*) **using** *b5* **by** *blast*
        ultimately have $n \leq m \longrightarrow$ *Di n* $\subseteq$ *Di* (*Suc m*) **using** *d1* **by** *blast*
        moreover have *n* = (*Suc m*) $\vee$ $n \leq m$ **using** *e1* **by** *force*
        ultimately show *Di n* $\subseteq$ *Di* (*Suc m*) **by** *blast*
      qed
    qed
  qed
  have *Di 0* $\subseteq$ *D* **using** *b9* **by** *blast*
  then have *b13*: *Field s* $\subseteq$ *D* **using** *b7 b8* **by** *simp*
  then have *b14*: $s \subseteq s' \wedge s' \subseteq r$ **using** *a2 b10* **unfolding** *Field-def* **by** *force*
  moreover have *b15*: |*D*| $\leq o$ |*s*|
  proof –
    have |*UNIV*::*nat set*| $\leq o$ |*s*| **using** *a3 infinite-iff-card-of-nat* **by** *blast*
    then have |$\bigcup$ *n*. *Di n*| $\leq o$ |*s*| **using** *b11 a3 card-of-UNION-ordLeq-infinite*[*of s UNIV Di*] **by** *blast*
    moreover have *D* = ($\bigcup$ *n*. *Di n*) **using** *b9* **by** *force*
    ultimately show *?thesis* **by** *blast*
  qed
  moreover have |*s'*| $=o$ |*s*|
  proof –
    have $\neg$ *finite* (*Field s*) **using** *a3 lem-fin-fl-rel* **by** *blast*
    then have $\neg$ *finite D* **using** *b13 finite-subset* **by** *blast*
    then have |*D* $\times$ *D*| $=o$ |*D*| **by** *simp*
    moreover have $s' \subseteq D \times D$ **using** *b10* **by** *blast*

ultimately have $|s'| \leq o\ |s|$ **using** *b15 card-of-mono1 ordLeq-ordIso-trans ordLeq-transitive* **by** *metis*
moreover have $|s| \leq o\ |s'|$ **using** *b14* **by** *simp*
ultimately show *?thesis* **using** *ordIso-iff-ordLeq* **by** *blast*
qed
moreover have $A \subseteq Field\ s'$
proof
fix x
assume *c1*: $x \in A$
obtain *ax bx* **where** *c2*: $ax = fst\ (pt\ x) \wedge bx = snd\ (pt\ x)$ **by** *blast*
have $pt\ x \in Pt\ x$ **using** *c1 p3* **by** *blast*
then have *c3*: $(ax,\ bx) \in r \wedge x \in \{ax,bx\}$ **using** *c2 p1* **by** *simp*
have $\{ax,\ bx\} \subseteq D0$ **using** *b7 c1 c2* **by** *blast*
moreover have $Di\ 0 \subseteq D$ **using** *b9* **by** *blast*
moreover have $Di\ 0 = D0$ **using** *b8* **by** *simp*
ultimately have $\{ax,\ bx\} \subseteq D$ **by** *blast*
then have $(ax,\ bx) \in s'$ **using** *c3 b10* **by** *blast*
then show $x \in Field\ s'$ **using** *c3* **unfolding** *Field-def* **by** *blast*
qed
moreover have $CCR\ s'$
proof −
have $\forall\ a \in Field\ s'.\ \forall\ b \in Field\ s'.\ \exists\ c \in Field\ s'.\ (a,c) \in (s')\hat{}* \wedge (b,c) \in (s')\hat{}*$
proof (*intro ballI*)
fix *a b*
assume *d1*: $a \in Field\ s'$ **and** *d2*: $b \in Field\ s'$
then have *d3*: $a \in D \wedge b \in D$ **using** *b10* **unfolding** *Field-def* **by** *blast*
then obtain *ia ib* **where** *d4*: $a \in Di\ ia \wedge b \in Di\ ib$ **using** *b9* **by** *blast*
obtain k **where** *d5*: $k = (max\ ia\ ib)$ **by** *blast*
then have $ia \leq k \wedge ib \leq k$ **by** *simp*
then have *d6*: $a \in Di\ k \wedge b \in Di\ k$ **using** *d4 b12* **by** *blast*
obtain p **where** *d7*: $p = g\ \{a,b\}$ **by** *blast*
have $Field\ p \subseteq H\ (Di\ k)$ **using** *b5 d6 d7* **by** *blast*
moreover have $H\ (Di\ k) = Di\ (Suc\ k)$ **using** *b8* **by** *simp*
moreover have $Di\ (Suc\ k) \subseteq D$ **using** *b9* **by** *blast*
ultimately have *d8*: $Field\ p \subseteq D$ **by** *blast*
have $\{a,\ b\} \subseteq Field\ r$ **using** *d1 d2 b10* **unfolding** *Field-def* **by** *blast*
moreover have $finite\ \{a,\ b\}$ **by** *simp*
ultimately have *d9*: $CCR\ p \wedge p \subseteq r \wedge \{a,b\} \subseteq Field\ p$ **using** *d7 b3* **by** *blast*
then obtain c **where** *d10*: $c \in Field\ p \wedge (a,c) \in p\hat{}* \wedge (b,c) \in p\hat{}*$ **unfolding** *CCR-def* **by** *blast*
have $(p\ \text{``}\ D) \subseteq D$ **using** *d8* **unfolding** *Field-def* **by** *blast*
then have $D \in Inv\ p$ **unfolding** *Inv-def* **by** *blast*
then have $p\hat{}* \cap (D\times(UNIV::'U\ set)) \subseteq (Restr\ p\ D)\hat{}*$ **using** *lem-Inv-restr-rtr*[*of D p*] **by** *blast*
moreover have $Restr\ p\ D \subseteq s'$ **using** *d9 b10* **by** *blast*
moreover have $(a,c) \in p\hat{}* \cap (D\times(UNIV::'U\ set)) \wedge (b,c) \in p\hat{}* \cap (D\times(UNIV::'U\ set))$ **using** *d10 d3* **by** *blast*
ultimately have $(a,c) \in (s')\hat{}* \wedge (b,c) \in (s')\hat{}*$ **using** *rtrancl-mono* **by** *blast*

moreover then have $c \in$ *Field s′* **using** *d1 lem-rtr-field* **by** *metis*
ultimately show $\exists\ c \in$ *Field s′*. $(a,c) \in (s')\hat{}* \wedge (b,c) \in (s')\hat{}*$ **by** *blast*
qed
then show *?thesis* **unfolding** *CCR-def* **by** *blast*
qed
moreover have $\forall\ P{\in}Ps.$ (*Field s′* $\cap P) \in$ *SCF s′*
proof −
have $\forall\ P \in Ps.\ \forall a{\in}$*Field s′*. $\exists b{\in}$(*Field s′* $\cap P$). $(a,\ b) \in s'\hat{}*$
proof (*intro ballI*)
fix *P a*
assume *d0*: $P \in Ps$ **and** *d1*: $a \in$ *Field s′*
then have $a \in D$ **using** *b10* **unfolding** *Field-def* **by** *blast*
then obtain n **where** $a \in Di\ n$ **using** *b9* **by** *blast*
then have $\mathfrak{f}\ r\ a\ (q\ P\ a) \subseteq H\ (Di\ n)$ **using** *d0 b5* **by** *blast*
moreover have $H\ (Di\ n) = Di\ (Suc\ n)$ **using** *b8* **by** *simp*
ultimately have *d2*: $\mathfrak{f}\ r\ a\ (q\ P\ a) \subseteq D$ **using** *b9* **by** *blast*
have $a \in$ *Field r* **using** *d1 b10* **unfolding** *Field-def* **by** *blast*
then have $q\ P\ a \in P \wedge (a,\ q\ P\ a) \in r\hat{}*$ **using** *d0 q1* **by** *blast*
moreover have *Restr r* $(\mathfrak{f}\ r\ a\ (q\ P\ a)) \subseteq s'$ **using** *d0 d2 b10* **by** *blast*
ultimately have $q\ P\ a \in P \wedge (a,\ q\ P\ a) \in s'\hat{}*$ **using** *lem-Ccext-fint*[*of r a q P a s′*] **by** *blast*
moreover then have $q\ P\ a \in$ *Field s′* **using** *d1 lem-rtr-field* **by** *metis*
ultimately show $\exists b{\in}$(*Field s′* $\cap P$). $(a,\ b) \in s'\hat{}*$ **by** *blast*
qed
then show *?thesis* **unfolding** *SCF-def* **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Ccext-finsubccr-pext5-scf2*:
fixes *r*::*′U rel* **and** *A B B′*::*′U set* **and** *x*::*′U* **and** *Ps*::*′U set set*
assumes *a1*: *CCR r* **and** *a2*: *finite A* **and** *a3*: $A \in$ *SF r* **and** *a4*: $Ps \subseteq$ *SCF r*
shows $\exists\ A'$::(*′U set*). $(x \in$ *Field r* $\longrightarrow x \in A') \wedge A \subseteq A' \wedge$ *CCR* (*Restr r A′*) $\wedge$ *finite A′*
$\wedge\ (\forall a{\in}A.\ r``\{a\}{\subseteq}B \vee r``\{a\}{\cap}(A'{-}B) \neq \{\}) \wedge A' \in$ *SF r*
$\wedge\ ((\exists\ y{::}'U.\ A'{-}B' = \{y\}) \longrightarrow$ *Field r* $\subseteq (A'{\cup}B'))$
$\wedge\ ((\exists\ P.\ Ps = \{P\}) \longrightarrow (\forall\ P \in Ps.\ (A' \cap P) \in$ *SCF* (*Restr r A′*)))
proof −
obtain *P* **where** *p0*: $P =$ (*if* $(Ps \neq \{\})$ *then* (*SOME P*. $P \in Ps$) *else Field r*) **by** *blast*
moreover have *Field r* $\in$ *SCF r* **unfolding** *SCF-def* **by** *blast*
ultimately have *p1*: $P \in$ *SCF r* **using** *a4* **by** (*metis contra-subsetD some-in-eq*)
have *p2*: $(\exists\ P.\ Ps = \{P\}) \longrightarrow Ps = \{P\}$ **using** *p0* **by** *fastforce*
have *q1*: *Field* (*Restr r A*) $= A$ **using** *a3* **unfolding** *SF-def* **by** *blast*
obtain *s* **where** $s =$ (*Restr r A*) **by** *blast*
then have *q2*: $s \subseteq r$ **and** *q3*: *finite s* **and** *q4*: $A =$ *Field s*
using *a2 q1 lem-fin-fl-rel* **by** (*blast*, *metis*, *blast*)
obtain *S* **where** *b1*: $S = (\lambda\ a.\ r``\{a\} - B\ )$ **by** *blast*

obtain S' **where** *b2*: $S' = (\lambda\ a.\ if\ (S\ a) \neq \{\}\ then\ (S\ a)\ else\ \{a\})$ **by** *blast*
obtain f **where** $f = (\lambda\ a.\ SOME\ b.\ b \in S'\ a)$ **by** *blast*
moreover have $\forall\ a.\ \exists\ b.\ b \in (S'\ a)$ **unfolding** *b2* **by** *force*
ultimately have $\forall\ a.\ f\ a \in S'\ a$ **by** (*metis someI-ex*)
then have *b3*: $\forall\ a.\ (S\ a \neq \{\} \longrightarrow f\ a \in S\ a) \wedge (S\ a = \{\} \longrightarrow f\ a = a)$
unfolding *b2* **by** (*clarsimp*, *metis singletonD*)
obtain *y1 y2*::$'U$ **where** *n1*: $Field\ r \neq \{\} \longrightarrow \{y1,\ y2\} \subseteq Field\ r$
and *n2*: $(\neg\ (\exists\ y{::}'U.\ Field\ r - B' \subseteq \{y\})) \longrightarrow y1 \notin B' \wedge y2 \notin B' \wedge y1 \neq y2$ **by** *blast*
obtain *A1* **where** *b4*: $A1 = (\{x,y1,y2\} \cap Field\ r) \cup A \cup (f\ `\ A)$ **by** *blast*
have $A1 \subseteq Field\ r$
proof −
have *c1*: $A \subseteq Field\ r$ **using** *q4 q2* **unfolding** *Field-def* **by** *blast*
moreover have $f\ `\ A \subseteq Field\ r$
proof
fix x
assume $x \in f\ `\ A$
then obtain a **where** *d2*: $a \in A \wedge x = f\ a$ **by** *blast*
show $x \in Field\ r$
proof (*cases* $S\ a = \{\}$)
assume $S\ a = \{\}$
then have $x = a$ **using** *c1 d2 b3* **by** *blast*
then show $x \in Field\ r$ **using** *d2 c1* **by** *blast*
next
assume $S\ a \neq \{\}$
then have $x \in S\ a$ **using** *d2 b3* **by** *blast*
then show $x \in Field\ r$ **using** *b1* **unfolding** *Field-def* **by** *blast*
qed
qed
ultimately show $A1 \subseteq Field\ r$ **using** *b4* **by** *blast*
qed
moreover have *s0*: *finite A1* **using** *b4 q3 q4 lem-fin-fl-rel* **by** *blast*
ultimately obtain s' **where** *s1*: $CCR\ s' \wedge s \subseteq s' \wedge s' \subseteq r \wedge finite\ s' \wedge A1 \subseteq Field\ s'$
and *s1'*: $(\exists\ P.\ Ps = \{P\}) \longrightarrow (Field\ s' \cap P) \in SCF\ s'$
using *p1 a1 a4 q2 q3 lem-Ccext-finsubccr-set-ext-scf*[*of r s A1 P*] **by** *metis*
obtain A' **where** *s2*: $A' = Field\ s'$ **by** *blast*
obtain s'' **where** *s3*: $s'' = Restr\ r\ A'$ **by** *blast*
then have *s4*: $s' \subseteq s'' \wedge Field\ s'' = A'$ **using** *s1 s2 lem-Relprop-fld-sat*[*of* s' r s''] **by** *blast*
have *s5*: *finite* $(Field\ s')$ **using** *s1 lem-fin-fl-rel* **by** *blast*
have $A1 \cup (\{x\} \cap Field\ r) \subseteq A'$ **using** *b4 s1 s2* **by** *blast*
moreover have $CCR\ (Restr\ r\ A')$
proof −
have $CCR\ s''$ **using** *s1 s2 s4 lem-Ccext-subccr-eqfld*[*of* s' s''] **by** *blast*
then show *?thesis* **using** *s3* **by** *blast*
qed
ultimately have *b6*: $A1 \cup (\{x\} \cap Field\ r) \subseteq A' \wedge CCR\ (Restr\ r\ A')$ **by** *blast*
moreover then have $A \cup (\{x\} \cap Field\ r) \subseteq A'$ **using** *b4* **by** *blast*

ultimately have $(x \in \mathit{Field}\ r \longrightarrow x \in A') \wedge A \subseteq A' \wedge \mathit{CCR}\ (\mathit{Restr}\ r\ A')$ **by** *blast*
moreover have *finite* A' **using** *s2 s5* **by** *blast*
moreover have $\forall a{\in}A.\ r``\{a\} \subseteq B \vee r``\{a\} \cap (A'{-}B) \neq \{\}$
proof
fix a
assume *c1*: $a \in A$
have $\neg\ (r``\{a\} \subseteq B) \longrightarrow r``\{a\} \cap (A'{-}B) \neq \{\}$
proof
assume $\neg\ (r``\{a\} \subseteq B)$
then have $S\ a \neq \{\}$ **unfolding** *b1* **by** *blast*
then have $f\ a \in r``\{a\} - B$ **using** *b1 b3* **by** *blast*
moreover have $f\ a \in A'$ **using** *c1 b4 b6* **by** *blast*
ultimately show $r``\{a\} \cap (A'{-}B) \neq \{\}$ **by** *blast*
qed
then show $r``\{a\} \subseteq B \vee r``\{a\} \cap (A'{-}B) \neq \{\}$ **by** *blast*
qed
moreover have $A' \in \mathit{SF}\ r$ **using** *s3 s4* **unfolding** *SF-def* **by** *blast*
moreover have $(\exists\ y{::}'U.\ A' - B' = \{y\}) \longrightarrow \mathit{Field}\ r \subseteq (A' \cup B')$
proof
assume *c1*: $\exists\ y{::}'U.\ A' - B' = \{y\}$
moreover have *c2*: $A' \subseteq \mathit{Field}\ r$ **using** *s1 s2* **unfolding** *Field-def* **by** *blast*
ultimately have $\mathit{Field}\ r \neq \{\}$ **by** *blast*
then have $\{y1,\ y2\} \subseteq \mathit{Field}\ r$ **using** *n1* **by** *blast*
then have $\{y1,\ y2\} \subseteq A'$ **using** *b4 s1 s2* **by** *fast*
then have $\neg\ (\exists y.\ \mathit{Field}\ r - B' \subseteq \{y\}) \longrightarrow \{y1,\ y2\} \subseteq A' - B' \wedge y1 \neq y2$ **using** *n2* **by** *blast*
moreover have $\neg\ (\{y1,\ y2\} \subseteq A' - B' \wedge y1 \neq y2)$ **using** *c1* **by** *force*
ultimately have $\exists\ y{::}'U.\ \mathit{Field}\ r - B' \subseteq \{y\}$ **by** *blast*
then show $\mathit{Field}\ r \subseteq A' \cup B'$ **using** *c1 c2* **by** *blast*
qed
moreover have $(\exists\ P.\ Ps = \{P\}) \longrightarrow (\forall\ P \in Ps.\ (A' \cap P) \in \mathit{SCF}\ (\mathit{Restr}\ r\ A'))$
proof −
have *c1*: $s' \subseteq r$ **using** *s3 s4* **by** *blast*
then have $\mathit{Field}\ s' = \mathit{Field}\ (\mathit{Restr}\ r\ (\mathit{Field}\ s'))$ **using** *lem-Relprop-fld-sat* **by** *blast*
moreover have $s' \subseteq \mathit{Restr}\ r\ (\mathit{Field}\ s')$ **using** *c1* **unfolding** *Field-def* **by** *force*
ultimately have $\mathit{SCF}\ s' \subseteq \mathit{SCF}\ (\mathit{Restr}\ r\ (\mathit{Field}\ s'))$ **using** *lem-ccext-scf-sat*[*of* s' *Restr* r (*Field* s')] **by** *blast*
then show *?thesis* **using** *p2 s1' s2* **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Ccext-infsubccr-pext5-scf2*:
fixes $r{::}'U\ \mathit{rel}$ **and** $A\ B\ B'{::}'U\ \mathit{set}$ **and** $x{::}'U$ **and** $Ps{::}'U\ \mathit{set}\ \mathit{set}$
assumes *a1*: $\mathit{CCR}\ r$ **and** *a2*: $\neg\ \mathit{finite}\ A$ **and** *a3*: $A \in \mathit{SF}\ r$ **and** *a4*: $Ps \subseteq \mathit{SCF}\ r$
shows $\exists\ A'{::}('U\ \mathit{set}).\ (x \in \mathit{Field}\ r \longrightarrow x \in A') \wedge A \subseteq A' \wedge \mathit{CCR}\ (\mathit{Restr}\ r\ A') \wedge$

|A′| =o |A|
∧ (∀ a∈A. r‘‘{a}⊆B ∨ r‘‘{a}∩(A′−B) ≠ {}) ∧ A′ ∈ SF r
∧ ((∃ y::′U. A′−B′ = {y}) ⟶ Field r ⊆ (A′∪B′))
∧ (|Ps| ≤o |A| ⟶ (∀ P ∈ Ps. (A′ ∩ P) ∈ SCF (Restr r A′)))
proof −
obtain *Ps′* **where** *p0*: *Ps′* = (*if* (|*Ps*| ≤*o* |*A*|) *then Ps else* {}) **by** *blast*
then have *p1*: *Ps′* ⊆ *SCF r* ∧ |*Ps′*| ≤*o* |*A*| **using** *a4* **by** *simp*
have *q1*: *Field* (*Restr r A*) = *A* **using** *a3* **unfolding** *SF-def* **by** *blast*
obtain *s* **where** *s* = (*Restr r A*) **by** *blast*
then have *q2*: *s* ⊆ *r* **and** *q3*: ¬ *finite s* **and** *q4*: *A* = *Field s*
using *a2 q1 lem-fin-fl-rel* **by** (*blast*, *metis*, *blast*)
obtain *S* **where** *b1*: *S* = (λ *a*. *r*‘‘{*a*} − *B*) **by** *blast*
obtain *S′* **where** *b2*: *S′* = (λ *a*. *if* (*S a*) ≠ {} *then* (*S a*) *else* {*a*}) **by** *blast*
obtain *f* **where** *f* = (λ *a*. *SOME b*. *b* ∈ *S′ a*) **by** *blast*
moreover have ∀ *a*. ∃ *b*. *b* ∈ (*S′ a*) **unfolding** *b2* **by** *force*
ultimately have ∀ *a*. *f a* ∈ *S′ a* **by** (*metis someI-ex*)
then have *b3*: ∀ *a*. (*S a* ≠ {} ⟶ *f a* ∈ *S a*) ∧ (*S a* = {} ⟶ *f a* = *a*)
unfolding *b2* **by** (*clarsimp*, *metis singletonD*)
obtain *y1 y2*::′*U* **where** *n1*: *Field r* ≠ {} ⟶ {*y1*, *y2*} ⊆ *Field r*
and *n2*: (¬ (∃ *y*::′*U*. *Field r* − *B′* ⊆ {*y*})) ⟶ *y1* ∉ *B′* ∧ *y2* ∉ *B′* ∧ *y1* ≠ *y2* **by** *blast*
obtain *A1* **where** *b4*: *A1* = ({*x*, *y1*, *y2*} ∩ *Field r*) ∪ *A* ∪ (*f* ‘ *A*) **by** *blast*
have *A1* ⊆ *Field r*
proof −
have *c1*: *A* ⊆ *Field r* **using** *q4 q2* **unfolding** *Field-def* **by** *blast*
moreover have *f* ‘ *A* ⊆ *Field r*
proof
fix *x*
assume *x* ∈ *f* ‘ *A*
then obtain *a* **where** *d2*: *a* ∈ *A* ∧ *x* = *f a* **by** *blast*
show *x* ∈ *Field r*
proof (*cases S a* = {})
assume *S a* = {}
then have *x* = *a* **using** *c1 d2 b3* **by** *blast*
then show *x* ∈ *Field r* **using** *d2 c1* **by** *blast*
next
assume *S a* ≠ {}
then have *x* ∈ *S a* **using** *d2 b3* **by** *blast*
then show *x* ∈ *Field r* **using** *b1* **unfolding** *Field-def* **by** *blast*
qed
qed
ultimately show *A1* ⊆ *Field r* **using** *b4* **by** *blast*
qed
moreover have *s0*: |*A1*| ≤*o* |*Field s*|
proof −
obtain *C1* **where** *c1*: *C1* = {*x*,*y1*,*y2*} ∩ *Field r* **by** *blast*
obtain *C2* **where** *c2*: *C2* = *A* ∪ *f* ‘ *A* **by** *blast*
have ¬ *finite A* **using** *q4 q3 lem-fin-fl-rel* **by** *blast*
then have |*C2*| =*o* |*A*| **using** *c2 b4 q3* **by** *simp*

then have $|C2| \leq o\ |Field\ s|$ **unfolding** *q4* **using** *ordIso-iff-ordLeq* **by** *blast*
moreover have *c3*: $\neg$ *finite* (*Field s*) **using** *q3 lem-fin-fl-rel* **by** *blast*
moreover have $|C1| \leq o\ |Field\ s|$
proof −
have $|\{x,y1,y2\}| \leq o\ |Field\ s|$ **using** *c3*
by (*meson card-of-Well-order card-of-ordLeq-finite finite.emptyI finite.insertI ordLeq-total*)
moreover have $|C1| \leq o\ |\{x,y1,y2\}|$ **unfolding** *c1* **by** *simp*
ultimately show *?thesis* **using** *ordLeq-transitive* **by** *blast*
qed
ultimately have $|C1 \cup C2| \leq o\ |Field\ s|$ **unfolding** *b4* **using** *card-of-Un-ordLeq-infinite* **by** *blast*
moreover have $A1 = C1 \cup C2$ **using** *c1 c2 b4* **by** *blast*
ultimately show *?thesis* **by** *blast*
qed
ultimately obtain s' **where** *s1*: $CCR\ s' \wedge s \subseteq s' \wedge s' \subseteq r \wedge |s'| =o\ |s| \wedge A1 \subseteq Field\ s'$
and *s1'*: $(\forall\ P \in Ps'.\ (Field\ s' \cap P) \in SCF\ s')$
using *p1 a1 q2 q3 q4 lem-Ccext-infsubccr-set-ext-scf2*[*of r s A1 Ps'*] **by** *blast*
obtain A' **where** *s2*: $A' = Field\ s'$ **by** *blast*
obtain s'' **where** *s3*: $s'' = Restr\ r\ A'$ **by** *blast*
then have *s4*: $s' \subseteq s'' \wedge Field\ s'' = A'$ **using** *s1 s2 lem-Relprop-fld-sat*[*of s' r s''*] **by** *blast*
have *s5*: $|Field\ s'| =o\ |Field\ s|$ **using** *s1 q3 lem-cardreleq-cardfldeq-inf*[*of s' s*] **by** *blast*
have $A1 \cup (\{x\} \cap Field\ r) \subseteq A'$ **using** *b4 s1 s2* **by** *blast*
moreover have *CCR* (*Restr r A'*)
proof −
have $CCR\ s''$ **using** *s1 s2 s4 lem-Ccext-subccr-eqfld*[*of s' s''*] **by** *blast*
then show *?thesis* **using** *s3* **by** *blast*
qed
moreover have $|A'| =o\ |A1|$
proof −
have $Field\ s \subseteq A1$ **using** *q4 b4* **by** *blast*
then have $|Field\ s| \leq o\ |A1|$ **by** *simp*
then have $|A'| \leq o\ |A1|$ **using** *s2 s5 ordIso-ordLeq-trans* **by** *blast*
moreover have $|A1| \leq o\ |A'|$ **using** *s1 s2* **by** *simp*
ultimately show *?thesis* **using** *ordIso-iff-ordLeq* **by** *blast*
qed
ultimately have *b6*: $A1 \cup (\{x\} \cap Field\ r) \subseteq A' \wedge CCR\ (Restr\ r\ A') \wedge |A'| =o\ |A1|$ **by** *blast*
moreover then have $A \cup (\{x\} \cap Field\ r) \subseteq A'$ **using** *b4* **by** *blast*
moreover have $|A'| =o\ |A|$ **using** *s5 s2 q4* **by** *blast*
moreover have $\forall a \in A.\ r``\{a\} \subseteq B \vee r``\{a\} \cap (A'-B) \neq \{\}$
proof
fix *a*
assume *c1*: $a \in A$
have $\neg\ (r``\{a\} \subseteq B) \longrightarrow r``\{a\} \cap (A'-B) \neq \{\}$
proof

assume $\neg$ ($r``\{a\} \subseteq B$)
then have $S\ a \neq \{\}$ **unfolding** *b1* **by** *blast*
then have $f\ a \in r``\{a\} - B$ **using** *b1 b3* **by** *blast*
moreover have $f\ a \in A'$ **using** *c1 b4 b6* **by** *blast*
ultimately show $r``\{a\} \cap (A'-B) \neq \{\}$ **by** *blast*
qed
then show $r``\{a\} \subseteq B \vee r``\{a\} \cap (A'-B) \neq \{\}$ **by** *blast*
qed
moreover have $A' \in SF\ r$ **using** *s3 s4* **unfolding** *SF-def* **by** *blast*
moreover have $(\exists\ y::'U.\ A' - B' = \{y\}) \longrightarrow Field\ r \subseteq (A' \cup B')$
proof
assume *c1*: $\exists\ y::'U.\ A' - B' = \{y\}$
moreover have *c2*: $A' \subseteq Field\ r$ **using** *s1 s2* **unfolding** *Field-def* **by** *blast*
ultimately have $Field\ r \neq \{\}$ **by** *blast*
then have $\{y1,\ y2\} \subseteq Field\ r$ **using** *n1* **by** *blast*
then have $\{y1,\ y2\} \subseteq A'$ **using** *b4 s1 s2* **by** *fast*
then have $\neg\ (\exists y.\ Field\ r - B' \subseteq \{y\}) \longrightarrow \{y1,\ y2\} \subseteq A' - B' \wedge y1 \neq y2$ **using** *n2* **by** *blast*
moreover have $\neg\ (\{y1,\ y2\} \subseteq A' - B' \wedge y1 \neq y2)$ **using** *c1* **by** *force*
ultimately have $\exists\ y::'U.\ Field\ r - B' \subseteq \{y\}$ **by** *blast*
then show $Field\ r \subseteq A' \cup B'$ **using** *c1 c2* **by** *blast*
qed
moreover have ($|Ps| \leq o\ |A| \longrightarrow (\forall\ P \in Ps.\ (A' \cap P) \in SCF\ (Restr\ r\ A'))$)
proof −
have *c1*: $s' \subseteq r$ **using** *s3 s4* **by** *blast*
then have $Field\ s' = Field\ (Restr\ r\ (Field\ s'))$ **using** *lem-Relprop-fld-sat* **by** *blast*
moreover have $s' \subseteq Restr\ r\ (Field\ s')$ **using** *c1* **unfolding** *Field-def* **by** *force*
ultimately have $SCF\ s' \subseteq SCF\ (Restr\ r\ (Field\ s'))$ **using** *lem-ccext-scf-sat*[*of* s' *Restr r* $(Field\ s')$] **by** *blast*
moreover have $|Ps| \leq o\ |A| \longrightarrow Ps' = Ps$ **using** *p0* **by** *simp*
ultimately show *?thesis* **using** *s1' s2* **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Ccext-subccr-pext5-scf2*:
fixes $r::'U\ rel$ **and** $A\ B\ B'::'U\ set$ **and** $x::'U$ **and** $Ps::'U\ set\ set$
assumes $CCR\ r$ **and** $A \in SF\ r$ **and** $Ps \subseteq SCF\ r$
shows $\exists\ A'::('U\ set).\ (x \in Field\ r \longrightarrow x \in A')$
$\wedge\ A \subseteq A'$
$\wedge\ A' \in SF\ r$
$\wedge\ (\forall a{\in}A.\ ((r``\{a\}{\subseteq}B) \vee (r``\{a\}{\cap}(A'{-}B) \neq \{\})))$
$\wedge\ ((\exists\ y::'U.\ A'{-}B' = \{y\}) \longrightarrow Field\ r \subseteq (A'{\cup}B'))$
$\wedge\ CCR\ (Restr\ r\ A')$
$\wedge\ ((finite\ A \longrightarrow finite\ A') \wedge (\ (\neg\ finite\ A) \longrightarrow |A'| =o\ |A|\ ))$
$\wedge\ (\ ((\exists\ P.\ Ps = \{P\}) \vee ((\neg\ finite\ Ps) \wedge |Ps| \leq o\ |A|\ )) \longrightarrow$
$(\forall\ P \in Ps.\ (A' \cap P) \in SCF\ (Restr\ r\ A')))$
proof (*cases finite A*)

```
  assume b1: finite A
  then obtain A'::'U set where b2: (x ∈ Field r ⟶ x ∈ A') ∧ A ⊆ A' ∧ CCR
(Restr r A')
                    ∧ (∀ a∈A. r``{a}⊆B ∨ r``{a}∩(A'−B) ≠ {}) ∧ A' ∈ SF r
                    ∧ ((∃ y::'U. A'−B' = {y}) ⟶ Field r ⊆ (A'∪B'))
                    and b3: finite A' ∧ ((∃ P. Ps = {P}) ⟶ (∀ P ∈ Ps. (A' ∩ P)
∈ SCF (Restr r A')))
                    using assms lem-Ccext-finsubccr-pext5-scf2[of r A Ps x B B'] by
metis
  have b4: ((finite A ⟶ finite A') ∧ ( (¬ finite A) ⟶ |A'| =o |A| ))
   and b5: ( ((∃ P. Ps = {P}) ∨ ((¬ finite Ps) ∧ |Ps| ≤o |A| )) ⟶ (∀ P ∈ Ps.
(A' ∩ P) ∈ SCF (Restr r A')))
      using b1 b3 card-of-ordLeq-finite by blast+
  show ?thesis
    apply (rule exI)
    using b2 b4 b5 by force
next
  assume b1: ¬ finite A
  then obtain A' where b2: (x ∈ Field r ⟶ x ∈ A') ∧ A ⊆ A' ∧ CCR (Restr
r A')
                    ∧ (∀ a∈A. r``{a}⊆B ∨ r``{a}∩(A'−B) ≠ {}) ∧ A' ∈ SF r
                    ∧ ((∃ y::'U. A'−B' = {y}) ⟶ Field r ⊆ (A'∪B'))
                and b3: |A'| =o |A| ∧ ( |Ps| ≤o |A| ⟶ (∀ P ∈ Ps. (A' ∩ P) ∈ SCF
(Restr r A')) )
    using assms lem-Ccext-infsubccr-pext5-scf2[of r A Ps x B B'] by metis
  have b4: ((finite A ⟶ finite A') ∧ ( (¬ finite A) ⟶ |A'| =o |A| ))
    using b1 b3 by metis
  have b5: ( ((∃ P. Ps = {P}) ∨ ((¬ finite Ps) ∧ |Ps| ≤o |A| )) ⟶ (∀ P ∈ Ps.
(A' ∩ P) ∈ SCF (Restr r A')))
    using b1 b3 by (metis card-of-singl-ordLeq finite.simps)
  show ?thesis
    apply (rule exI)
    using b2 b4 b5 by force
qed

lemma lem-dnEsc-el: F ∈ dnEsc r A a ⟹ a ∈ F ∧ finite F unfolding dnEsc-def
𝓕-def rpth-def by blast

lemma lem-dnEsc-emp: dnEsc r A a = {} ⟹ dnesc r A a = { a } unfolding
dnesc-def by simp

lemma lem-dnEsc-ne: dnEsc r A a ≠ {} ⟹ dnesc r A a ∈ dnEsc r A a
  unfolding dnesc-def using someI-ex[of λ F. F ∈ dnEsc r A a] by force

lemma lem-dnesc-in: a ∈ dnesc r A a ∧ finite (dnesc r A a)
  using lem-dnEsc-emp[of r A a] lem-dnEsc-el[of - r A a] lem-dnEsc-ne[of r A a]
by force

lemma lem-escl-incr: B ⊆ escl r A B using lem-dnesc-in[of - r A] unfolding
```

escl-def **by** *blast*

lemma *lem-escl-card*: (*finite B* ⟶ *finite* (*escl r A B*)) ∧ (¬ *finite B* ⟶ |*escl r A B*| ≤*o* |*B*|)
proof (*intro conjI impI*)
  assume *finite B*
  then show *finite* (*escl r A B*) **using** *lem-dnesc-in*[*of - r A*] **unfolding** *escl-def* **by** *blast*
next
  assume *b1*: ¬ *finite B*
  moreover have *escl r A B* = (⋃*x*∈*B*. ((*dnesc r A*) *x*)) **unfolding** *escl-def* **by** *blast*
  moreover have ∀ *x*. |(*dnesc r A*) *x*| ≤*o* |*B*|
  proof
    fix *x*
    have *finite* (*dnesc r A x*) **using** *lem-dnesc-in*[*of - r A*] **by** *blast*
    then show |*dnesc r A x*| ≤*o* |*B*| **using** *b1* **by** (*meson card-of-Well-order card-of-ordLeq-infinite ordLeq-total*)
  qed
  ultimately show |*escl r A B*| ≤*o* |*B*| **by** (*simp add*: *card-of-UNION-ordLeq-infinite*)
qed

lemma *lem-Ccext-infsubccr-set-ext-scf3*:
fixes *r s*::′*U rel* **and** *A A0*::′*U set* **and** *Ps*::′*U set set*
assumes *a1*: *CCR r* **and** *a2*: *s* ⊆ *r* **and** *a3*: ¬ *finite s* **and** *a4*: *A* ⊆ *Field r*
  and *a5*: |*A*| ≤*o* |*Field s*| **and** *a6*: *Ps* ⊆ *SCF r* ∧ |*Ps*| ≤*o* |*Field s*|
shows ∃ *s*′::(′*U rel*). *CCR s*′ ∧ *s* ⊆ *s*′ ∧ *s*′ ⊆ *r* ∧ |*s*′| =*o* |*s*| ∧ *A* ⊆ *Field s*′
  ∧ (∀ *P*∈*Ps*. (*Field s*′ ∩ *P*) ∈ *SCF s*′) ∧ (*escl r A0* (*Field s*′) ⊆ *Field s*′)
  ∧ (∃ *D*. *s*′ = *Restr r D*) ∧ (*Conelike s*′ ⟶ *Conelike r*)
proof −
  obtain *w* **where** *w0*: *w* = (λ *x*. *SOME y*. *y* ∈ *Field r* − *dncl r* {*x*}) **by** *blast*
  have *w1*: ⋀ *x*. *Field r* − *dncl r* {*x*} ≠ {} ⟹ *w x* ∈ *Field r* − *dncl r* {*x*}
  proof −
    fix *x*
    assume *Field r* − *dncl r* {*x*} ≠ {}
    then show *w x* ∈ *Field r* − *dncl r* {*x*}
      using *w0 someI-ex*[*of* λ *y*. *y* ∈ *Field r* − *dncl r* {*x*}] **by** *force*
  qed
  obtain *q* **where** *q0*: *q* = (λ *P a*. *SOME p*. *p* ∈ *P* ∧ (*a*, *p*) ∈ *r*^∗) **by** *blast*
  have *q1*: ∀ *P*∈*Ps*. ∀ *a*∈*Field r*. (*q P a*) ∈ *Field r* ∧ (*q P a*) ∈ *P* ∧ (*a*, *q P a*) ∈ *r*^∗
  proof (*intro ballI*)
    fix *P a*
    assume *P* ∈ *Ps* **and** *a* ∈ *Field r*
    then show (*q P a*) ∈ *Field r* ∧ (*q P a*) ∈ *P* ∧ (*a*, *q P a*) ∈ *r*^∗
      using *q0 a6 someI-ex*[*of* λ *p*. *p* ∈ *P* ∧ (*a*,*p*) ∈ *r*^∗] **unfolding** *SCF-def* **by** *blast*
  qed
  obtain *G*::′*U set* ⇒ ′*U rel set* **where** *b1*: *G* = (λ *A*. {*t*::′*U rel*. *finite t* ∧ *CCR*

$t \wedge t \subseteq r \wedge A \subseteq Field\ t\}$) **by** *blast*
obtain $g::'U\ set \Rightarrow 'U\ rel$ **where** *b2*: $g = (\lambda\ A.\ if\ A \subseteq Field\ r \wedge finite\ A\ then\ (SOME\ t.\ t \in G\ A)\ else\ \{\})$ **by** *blast*
have *b3*: $\forall\ A.\ A \subseteq Field\ r \wedge finite\ A \longrightarrow finite\ (g\ A) \wedge CCR\ (g\ A) \wedge (g\ A) \subseteq r \wedge A \subseteq Field\ (g\ A)$
proof (*intro allI impI*)
fix *A*
assume *c1*: $A \subseteq Field\ r \wedge finite\ A$
then have $g\ A = (SOME\ t.\ t \in G\ A)$ **using** *b2* **by** *simp*
moreover have $G\ A \neq \{\}$ **using** *b1 a1 c1 lem-Ccext-finsubccr-dext*[*of r A*] **by** *blast*
ultimately have $g\ A \in G\ A$ **using** *some-in-eq* **by** *metis*
then show $finite\ (g\ A) \wedge CCR\ (g\ A) \wedge (g\ A) \subseteq r \wedge A \subseteq Field\ (g\ A)$ **using** *b1* **by** *blast*
qed
have *b4*: $\forall\ A.\ \neg\ (A \subseteq Field\ r \wedge finite\ A) \longrightarrow g\ A = \{\}$ **using** *b2* **by** *simp*
obtain $H::'U\ set \Rightarrow 'U\ set$
where *b5*: $H = (\lambda\ X.\ X \cup \bigcup\ \{S.\ \exists\ a{\in}X.\ \exists\, b{\in}X.\ S = Field\ (g\ \{a,b\})\}$
$\cup \bigcup\ \{S.\ \exists\ P{\in}Ps.\ \exists\ a{\in}X.\ S = \mathfrak{f}\ r\ a\ (q\ P\ a)\ \}$
$\cup\ escl\ r\ A0\ X \cup (w`X)\ )$ **by** *blast*

obtain $Pt::'U \Rightarrow 'U\ rel$ **where** *p1*: $Pt = (\lambda\ x.\ \{p \in r.\ x = fst\ p \vee x = snd\ p\})$ **by** *blast*
obtain $pt::'U \Rightarrow 'U \times 'U$ **where** *p2*: $pt = (\lambda\ x.\ (SOME\ p.\ p \in Pt\ x))$ **by** *blast*
have $\forall\ x{\in}A.\ Pt\ x \neq \{\}$ **using** *a4* **unfolding** *p1 Field-def* **by** *force*
then have *p3*: $\forall\ x{\in}A.\ pt\ x \in Pt\ x$ **unfolding** *p2* **by** (*metis* (*full-types*) *Collect-empty-eq Collect-mem-eq someI-ex*)
obtain *D0* **where** *b7*: $D0 = Field\ s \cup fst`(pt`A) \cup snd`(pt`A)$ **by** *blast*
obtain $Di::nat \Rightarrow 'U\ set$ **where** *b8*: $Di = (\lambda\ n.\ (H\widehat{\ }\widehat{\ }n)\ D0)$ **by** *blast*
obtain $D::'U\ set$ **where** *b9*: $D = \bigcup\ \{X.\ \exists\ n.\ X = Di\ n\}$ **by** *blast*
obtain s' **where** *b10*: $s' = Restr\ r\ D$ **by** *blast*
have *b11*: $\forall\ n.\ (\neg\ finite\ (Di\ n)) \wedge |Di\ n| \leq o\ |s|$
proof
fix *n0*
show $(\neg\ finite\ (Di\ n0)) \wedge |Di\ n0| \leq o\ |s|$
proof (*induct n0*)
have $|D0| =o\ |Field\ s|$
proof −
have $|fst`(pt`A)| \leq o\ |(pt`A)| \wedge |(pt`A)| \leq o\ |A|$ **by** *simp*
then have *c1*: $|fst`(pt`A)| \leq o\ |A|$ **using** *ordLeq-transitive* **by** *blast*
have $|snd`(pt`A)| \leq o\ |(pt`A)| \wedge |(pt`A)| \leq o\ |A|$ **by** *simp*
then have *c2*: $|snd`(pt`A)| \leq o\ |A|$ **using** *ordLeq-transitive* **by** *blast*
have $|fst`(pt`A)| \leq o\ |Field\ s| \wedge |snd`(pt`A)| \leq o\ |Field\ s|$
using *c1 c2 a5 ordLeq-transitive* **by** *blast*
moreover have $\neg\ finite\ (Field\ s)$ **using** *a3 lem-fin-fl-rel* **by** *blast*
ultimately have *c3*: $|D0| \leq o\ |Field\ s|$ **unfolding** *b7* **by** *simp*
have $Field\ s \subseteq D0$ **unfolding** *b7* **by** *blast*
then have $|Field\ s| \leq o\ |D0|$ **by** *simp*
then show *?thesis* **using** *c3 ordIso-iff-ordLeq* **by** *blast*

qed
moreover have |*Field s*| =*o* |*s*| **using** *a3 lem-rel-inf-fld-card* **by** *blast*
ultimately have |*D0*| ≤*o* |*s*| **using** *ordIso-imp-ordLeq ordIso-transitive* **by** *blast*
moreover have ¬ *finite D0* **using** *a3 b7 lem-fin-fl-rel* **by** *blast*
ultimately show ¬ *finite* (*Di 0*) ∧ |*Di 0*| ≤*o* |*s*| **using** *b8* **by** *simp*
next
fix *n*
assume *d1*: (¬ *finite* (*Di n*)) ∧ |*Di n*| ≤*o* |*s*|
moreover then have |(*Di n*) × (*Di n*)| =*o* |*Di n*| **by** *simp*
ultimately have *d2*: |(*Di n*) × (*Di n*)| ≤*o* |*s*| **using** *ordIso-imp-ordLeq ordLeq-transitive* **by** *blast*
have *d3*: ∀ *a* ∈ (*Di n*). ∀ *b* ∈ (*Di n*). |*Field* (*g* {*a*, *b*})| ≤*o* |*s*|
proof (*intro ballI*)
fix *a b*
assume *a* ∈ (*Di n*) **and** *b* ∈ (*Di n*)
have *finite* (*g* {*a*, *b*}) **using** *b3 b4* **by** (*metis finite.emptyI*)
then have *finite* (*Field* (*g* {*a*, *b*})) **using** *lem-fin-fl-rel* **by** *blast*
then have |*Field* (*g* {*a*, *b*})| <*o* |*s*| **using** *a3 finite-ordLess-infinite2* **by** *blast*
then show |*Field* (*g* {*a*, *b*})| ≤*o* |*s*| **using** *ordLess-imp-ordLeq* **by** *blast*
qed
have *d4*: *Di* (*Suc n*) = *H* (*Di n*) **using** *b8* **by** *simp*
then have *Di n* ⊆ *Di* (*Suc n*) **using** *b5* **by** *blast*
then have ¬ *finite* (*Di* (*Suc n*)) **using** *d1 finite-subset* **by** *blast*
moreover have |*Di* (*Suc n*)| ≤*o* |*s*|
proof −
obtain *I* **where** *e1*: *I* = (*Di n*) × (*Di n*) **by** *blast*
obtain *f* **where** *e2*: *f* = (λ (*a*,*b*). *Field* (*g* {*a*,*b*})) **by** *blast*
have |*I*| ≤*o* |*s*| **using** *e1 d2* **by** *blast*
moreover have ∀ *i*∈*I*. |*f i*| ≤*o* |*s*| **using** *e1 e2 d3* **by** *simp*
ultimately have |⋃ *i*∈*I*. *f i*| ≤*o* |*s*| **using** *a3 card-of-UNION-ordLeq-infinite*[*of s I f*] **by** *blast*
moreover have *Di* (*Suc n*) = (*Di n*) ∪ (⋃ *i*∈*I*. *f i*)
∪ (⋃ *P*∈*Ps*. (⋃ *a*∈(*Di n*). 𝔣 *r a* (*q P a*))) ∪ *escl r A0* (*Di n*) ∪ (*w*‘(*Di n*))
using *e1 e2 d4 b5* **by** *blast*
moreover have |⋃ *P*∈*Ps*. (⋃ *a*∈(*Di n*). 𝔣 *r a* (*q P a*))| ≤*o* |*s*|
proof −
have ⋀ *P*. *P* ∈ *Ps* ⟹ ∀ *a*∈(*Di n*). |𝔣 *r a* (*q P a*)| ≤*o* |*s*|
using *a3 lem-ccext-ffin* **by** (*metis card-of-Well-order card-of-ordLeq-infinite ordLeq-total*)
then have ⋀ *P*. *P* ∈ *Ps* ⟹ |⋃ *a*∈(*Di n*). 𝔣 *r a* (*q P a*)| ≤*o* |*s*|
using *d1 a3 card-of-UNION-ordLeq-infinite*[*of s Di n* λ *a*. 𝔣 *r a* (*q* - *a*)] **by** *blast*
moreover have |*Ps*| ≤*o* |*s*| **using** *a3 a6 lem-rel-inf-fld-card*[*of s*] *lem-fin-fl-rel*[*of s*]
by (*metis ordIso-iff-ordLeq ordLeq-transitive*)
ultimately show *?thesis*

using *a3 card-of-UNION-ordLeq-infinite*[*of s Ps* λ *P*. ⋃ *a*∈(*Di n*). 𝔣 *r a* (*q P a*)] **by** *blast*
qed
moreover have |*escl r A0* (*Di n*)| ≤*o* |*s*|
using *d1 lem-escl-card*[*of Di n r A0*] **by** (*metis ordLeq-transitive*)
moreover have |*w*'(*Di n*)| ≤*o* |*s*| **using** *d1* **using** *card-of-image ordLeq-transitive* **by** *blast*
ultimately show *?thesis* **using** *d1 a3* **by** *simp*
qed
ultimately show (¬ *finite* (*Di* (*Suc n*))) ∧ |*Di* (*Suc n*)| ≤*o* |*s*| **by** *blast*
qed
qed
have *b12*: ∀ *m*. ∀ *n*. *n* ≤ *m* ⟶ *Di n* ≤ *Di m*
proof
fix *m0*
show ∀ *n*. *n* ≤ *m0* ⟶ *Di n* ≤ *Di m0*
proof (*induct m0*)
show ∀*n*≤*0*. *Di n* ⊆ *Di 0* **by** *blast*
next
fix *m*
assume *d1*: ∀*n*≤*m*. *Di n* ⊆ *Di m*
show ∀*n*≤*Suc m*. *Di n* ⊆ *Di* (*Suc m*)
proof (*intro allI impI*)
fix *n*
assume *e1*: *n* ≤ *Suc m*
have *Di* (*Suc m*) = *H* (*Di m*) **using** *b8* **by** *simp*
moreover have *Di m* ⊆ *H* (*Di m*) **using** *b5* **by** *blast*
ultimately have *n* ≤ *m* ⟶ *Di n* ⊆ *Di* (*Suc m*) **using** *d1* **by** *blast*
moreover have *n* = (*Suc m*) ∨ *n* ≤ *m* **using** *e1* **by** *force*
ultimately show *Di n* ⊆ *Di* (*Suc m*) **by** *blast*
qed
qed
qed
have *Di 0* ⊆ *D* **using** *b9* **by** *blast*
then have *b13*: *Field s* ⊆ *D* **using** *b7 b8* **by** *simp*
then have *b14*: *s* ⊆ *s'* ∧ *s'* ⊆ *r* **using** *a2 b10* **unfolding** *Field-def* **by** *force*
moreover have *b15*: |*D*| ≤*o* |*s*|
proof −
have |*UNIV*::*nat set*| ≤*o* |*s*| **using** *a3 infinite-iff-card-of-nat* **by** *blast*
then have |⋃ *n*. *Di n*| ≤*o* |*s*| **using** *b11 a3 card-of-UNION-ordLeq-infinite*[*of s UNIV Di*] **by** *blast*
moreover have *D* = (⋃ *n*. *Di n*) **using** *b9* **by** *force*
ultimately show *?thesis* **by** *blast*
qed
moreover have |*s'*| =*o* |*s*|
proof −
have ¬ *finite* (*Field s*) **using** *a3 lem-fin-fl-rel* **by** *blast*
then have ¬ *finite D* **using** *b13 finite-subset* **by** *blast*
then have |*D* × *D*| =*o* |*D*| **by** *simp*

moreover have $s' \subseteq D \times D$ **using** *b10* **by** *blast*
ultimately have $|s'| \leq o\ |s|$ **using** *b15 card-of-mono1 ordLeq-ordIso-trans ordLeq-transitive* **by** *metis*
moreover have $|s| \leq o\ |s'|$ **using** *b14* **by** *simp*
ultimately show *?thesis* **using** *ordIso-iff-ordLeq* **by** *blast*
qed
moreover have $A \subseteq Field\ s'$
proof
fix x
assume *c1*: $x \in A$
obtain $ax\ bx$ **where** *c2*: $ax = fst\ (pt\ x) \wedge bx = snd\ (pt\ x)$ **by** *blast*
have $pt\ x \in Pt\ x$ **using** *c1 p3* **by** *blast*
then have *c3*: $(ax,\ bx) \in r \wedge x \in \{ax,bx\}$ **using** *c2 p1* **by** *simp*
have $\{ax,\ bx\} \subseteq D0$ **using** *b7 c1 c2* **by** *blast*
moreover have $Di\ 0 \subseteq D$ **using** *b9* **by** *blast*
moreover have $Di\ 0 = D0$ **using** *b8* **by** *simp*
ultimately have $\{ax,\ bx\} \subseteq D$ **by** *blast*
then have $(ax,\ bx) \in s'$ **using** *c3 b10* **by** *blast*
then show $x \in Field\ s'$ **using** *c3* **unfolding** *Field-def* **by** *blast*
qed
moreover have $CCR\ s'$
proof −
have $\forall\ a \in Field\ s'.\ \forall\ b \in Field\ s'.\ \exists\ c \in Field\ s'.\ (a,c) \in (s')\hat{}* \wedge (b,c) \in (s')\hat{}*$
proof (*intro ballI*)
fix $a\ b$
assume *d1*: $a \in Field\ s'$ **and** *d2*: $b \in Field\ s'$
then have *d3*: $a \in D \wedge b \in D$ **using** *b10* **unfolding** *Field-def* **by** *blast*
then obtain $ia\ ib$ **where** *d4*: $a \in Di\ ia \wedge b \in Di\ ib$ **using** *b9* **by** *blast*
obtain k **where** *d5*: $k = (max\ ia\ ib)$ **by** *blast*
then have $ia \leq k \wedge ib \leq k$ **by** *simp*
then have *d6*: $a \in Di\ k \wedge b \in Di\ k$ **using** *d4 b12* **by** *blast*
obtain p **where** *d7*: $p = g\ \{a,b\}$ **by** *blast*
have $Field\ p \subseteq H\ (Di\ k)$ **using** *b5 d6 d7* **by** *blast*
moreover have $H\ (Di\ k) = Di\ (Suc\ k)$ **using** *b8* **by** *simp*
moreover have $Di\ (Suc\ k) \subseteq D$ **using** *b9* **by** *blast*
ultimately have *d8*: $Field\ p \subseteq D$ **by** *blast*
have $\{a,\ b\} \subseteq Field\ r$ **using** *d1 d2 b10* **unfolding** *Field-def* **by** *blast*
moreover have $finite\ \{a,\ b\}$ **by** *simp*
ultimately have *d9*: $CCR\ p \wedge p \subseteq r \wedge \{a,b\} \subseteq Field\ p$ **using** *d7 b3* **by** *blast*
then obtain c **where** *d10*: $c \in Field\ p \wedge (a,c) \in p\hat{}* \wedge (b,c) \in p\hat{}*$ **unfolding** *CCR-def* **by** *blast*
have $(p\ ``\ D) \subseteq D$ **using** *d8* **unfolding** *Field-def* **by** *blast*
then have $D \in Inv\ p$ **unfolding** *Inv-def* **by** *blast*
then have $p\hat{}* \cap (D \times (UNIV::'U\ set)) \subseteq (Restr\ p\ D)\hat{}*$ **using** *lem-Inv-restr-rtr*[*of D p*] **by** *blast*
moreover have $Restr\ p\ D \subseteq s'$ **using** *d9 b10* **by** *blast*
moreover have $(a,c) \in p\hat{}* \cap (D \times (UNIV::'U\ set)) \wedge (b,c) \in p\hat{}* \cap (D \times (UNIV::'U\ set))$ **using** *d10 d3* **by** *blast*

ultimately have $(a,c) \in (s')\hat{}* \wedge (b,c) \in (s')\hat{}*$ **using** *rtrancl-mono* **by** *blast*
moreover then have $c \in$ *Field s'* **using** *d1 lem-rtr-field* **by** *metis*
ultimately show $\exists\ c \in$ *Field s'*. $(a,c) \in (s')\hat{}* \wedge (b,c) \in (s')\hat{}*$ **by** *blast*
qed
then show *?thesis* **unfolding** *CCR-def* **by** *blast*
qed
moreover have $\forall\ P{\in}Ps.$ (*Field s'* $\cap$ *P*) $\in$ *SCF s'*
proof −
have $\forall\ P \in Ps.\ \forall\ a{\in}$*Field s'*. $\exists\ b{\in}$(*Field s'* $\cap$ *P*). $(a,\ b) \in s'\hat{}*$
proof (*intro ballI*)
fix *P a*
assume *d0*: $P \in Ps$ **and** *d1*: $a \in$ *Field s'*
then have $a \in D$ **using** *b10* **unfolding** *Field-def* **by** *blast*
then obtain *n* **where** $a \in$ *Di n* **using** *b9* **by** *blast*
then have $\mathfrak{f}$ *r a* (*q P a*) $\subseteq$ *H* (*Di n*) **using** *d0 b5* **by** *blast*
moreover have *H* (*Di n*) = *Di* (*Suc n*) **using** *b8* **by** *simp*
ultimately have *d2*: $\mathfrak{f}$ *r a* (*q P a*) $\subseteq D$ **using** *b9* **by** *blast*
have $a \in$ *Field r* **using** *d1 b10* **unfolding** *Field-def* **by** *blast*
then have *q P a* $\in P \wedge (a,$ *q P a*$) \in r\hat{}*$ **using** *d0 q1* **by** *blast*
moreover have *Restr r* ($\mathfrak{f}$ *r a* (*q P a*)) $\subseteq s'$ **using** *d0 d2 b10* **by** *blast*
ultimately have *q P a* $\in P \wedge (a,$ *q P a*$) \in s'\hat{}*$ **using** *lem-Ccext-fint*[*of r a q P a s'*] **by** *blast*
moreover then have *q P a* $\in$ *Field s'* **using** *d1 lem-rtr-field* **by** *metis*
ultimately show $\exists\ b{\in}$(*Field s'* $\cap$ *P*). $(a,\ b) \in s'\hat{}*$ **by** *blast*
qed
then show *?thesis* **unfolding** *SCF-def* **by** *blast*
qed
moreover have *escl r A0* (*Field s'*) $\subseteq$ *Field s'*
proof
fix *x*
assume *c1*: $x \in$ *escl r A0* (*Field s'*)
then obtain *F a* **where** *c2*: $x \in F \wedge F =$ *dnesc r A0 a* $\wedge\ a \in$ *Field s'* **unfolding** *escl-def* **by** *blast*
obtain *n* **where** $a \in$ *Di n* **using** *c2 b9 b10* **unfolding** *Field-def* **by** *blast*
then have $F \subseteq$ *H* (*Di n*) **using** *c2 b5* **unfolding** *escl-def* **by** *blast*
moreover have *H* (*Di n*) = *Di* (*Suc n*) **using** *b8 b9* **by** *simp*
ultimately have *c3*: $F \subseteq D$ **using** *b9* **by** *blast*
show $x \in$ *Field s'*
proof (*cases dnEsc r A0 a* = {})
assume *dnEsc r A0 a* = {}
then have $x = a$ **using** *c2 lem-dnEsc-emp*[*of r A0*] **by** *blast*
then show *?thesis* **using** *c2* **by** *blast*
next
assume *dnEsc r A0 a* $\neq$ {}
then have $F \in$ *dnEsc r A0 a* **using** *c2 lem-dnEsc-ne*[*of r A0 a*] **by** *blast*
then obtain *b* **where** $F \in \mathcal{F}$ *r a b* **unfolding** *dnEsc-def* **by** *blast*
then obtain *f k* **where** $f \in$ *rpth r a b k* $\wedge F = f`\{i.\ i{\le}k\}$ **unfolding** $\mathcal{F}$*-def* **by** *blast*
moreover then obtain *j* **where** $j{\le}k \wedge x = f\ j$ **using** *c2* **by** *blast*

ultimately have $f \in rpth\ (Restr\ r\ D)\ a\ x\ j$ **using** *c3* **unfolding** *rpth-def* **by** *force*
then have $a \in Field\ s' \wedge (a,x) \in s'\widehat{\ }*$ **using** *c2 b10 lem-ccext-rpth-rtr*[*of - a x*] **by** *blast*
then show *?thesis* **using** *lem-rtr-field* **by** *metis*
qed
qed
moreover have $\exists\ D.\ s' = Restr\ r\ D$ **using** *b10* **by** *blast*
moreover have $\neg\ Conelike\ r \longrightarrow \neg\ Conelike\ s'$
proof
assume $\neg\ Conelike\ r$
then have *c1*: $\forall\ a \in Field\ r.\ Field\ r - dncl\ r\ \{a\} \neq \{\}$ **unfolding** *Conelike-def dncl-def* **by** *blast*
have $\forall\ a \in Field\ s'.\ \exists\ a' \in Field\ s'.\ (a', a) \notin s'\widehat{\ }*$
proof
fix a
assume *d1*: $a \in Field\ s'$
then have *d2*: $a \in Field\ r$ **using** *b10* **unfolding** *Field-def* **by** *blast*
then have *d3*: $w\ a \in Field\ r - dncl\ r\ \{a\}$ **using** *c1 w1* **by** *blast*
then have $(w\ a,\ a) \notin s'\widehat{\ }*$ **unfolding** *dncl-def* **using** *b10 rtrancl-mono*[*of s' r*] **by** *blast*
moreover have $w\ a \in Field\ s'$
proof −
obtain n **where** $a \in Di\ n$ **using** *d1 b9 b10* **unfolding** *Field-def* **by** *blast*
then have $a \in Di\ (Suc\ n) \wedge w\ a \in Di\ (Suc\ n)$ **using** *b5 b8* **by** *simp*
then have *e1*: $Field\ (g\ \{a,\ w\ a\}) \subseteq H\ (Di\ (Suc\ n))$ **using** *b5 b8* **by** *blast*
have *e2*: $\{a,\ w\ a\} \subseteq Field\ r \wedge finite\ \{a,\ w\ a\}$ **using** *d2 d3* **by** *blast*
have $H\ (Di\ (Suc\ n)) = Di\ (Suc\ (Suc\ n))$ **using** *b8* **by** *simp*
moreover have $Di\ (Suc\ (Suc\ n)) \subseteq D$ **using** *b9* **by** *blast*
ultimately have $Field\ (g\ \{a,w\ a\}) \subseteq D$ **using** *e1* **by** *blast*
moreover have $Restr\ (g\ \{a,w\ a\})\ D \subseteq s'$ **using** *e2 b3 b10* **by** *blast*
ultimately have $g\ \{a,w\ a\} \subseteq s'$ **unfolding** *Field-def* **by** *fastforce*
moreover have $w\ a \in Field\ (g\ \{a,\ w\ a\})$ **using** *e2 b3* **by** *blast*
ultimately show $w\ a \in Field\ s'$ **unfolding** *Field-def* **by** *blast*
qed
ultimately show $\exists\ a' \in Field\ s'.\ (a',\ a) \notin s'\widehat{\ }*$ **by** *blast*
qed
moreover have $s' \neq \{\}$ **using** *b14 a3* **by** *force*
ultimately show $\neg\ Conelike\ s'$ **unfolding** *Conelike-def* **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Ccext-infsubccr-pext5-scf3*:
fixes $r::'U\ rel$ **and** $A\ B\ B'::'U\ set$ **and** $x::'U$ **and** $Ps::'U\ set\ set$
assumes *a1*: $CCR\ r$ **and** *a2*: $\neg\ finite\ A$ **and** *a3*: $A \in SF\ r$ **and** *a4*: $Ps \subseteq SCF\ r$
shows $\exists\ A'::('U\ set).\ (x \in Field\ r \longrightarrow x \in A') \wedge A \subseteq A' \wedge CCR\ (Restr\ r\ A') \wedge |A'| =o\ |A|$
$\wedge\ (\forall\, a{\in}A.\ r``\{a\}{\subseteq}B \vee r``\{a\}{\cap}(A'{-}B) \neq \{\}) \wedge A' \in SF\ r$

```
                  ∧ ((∃ y::'U. A'−B' ⊆ {y}) ⟶ Field r ⊆ (A'∪B'))
                  ∧ ( |Ps| ≤o |A| ⟶ (∀ P ∈ Ps. (A' ∩ P) ∈ SCF (Restr r A')) )
                  ∧ (escl r A A' ⊆ A') ∧ clterm (Restr r A') r
proof −
  obtain Ps' where p0: Ps' = (if ( |Ps| ≤o |A| ) then Ps else {}) by blast
  then have p1: Ps' ⊆ SCF r ∧ |Ps'| ≤o |A| using a4 by simp
  have q1: Field (Restr r A) = A using a3 unfolding SF-def by blast
  obtain s where s = (Restr r A) by blast
  then have q2: s ⊆ r and q3: ¬ finite s and q4: A = Field s
    using a2 q1 lem-fin-fl-rel by (blast, metis, blast)
  obtain S where b1: S = (λ a. r``{a} − B ) by blast
  obtain S' where b2: S' = (λ a. if (S a) ≠ {} then (S a) else {a}) by blast
  obtain f where f = (λ a. SOME b. b ∈ S' a) by blast
  moreover have ∀ a. ∃ b. b ∈ (S' a) unfolding b2 by force
  ultimately have ∀ a. f a ∈ S' a by (metis someI-ex)
  then have b3: ∀ a. (S a ≠ {} ⟶ f a ∈ S a) ∧ (S a = {} ⟶ f a = a)
    unfolding b2 by (clarsimp, metis singletonD)
  obtain y1 y2::'U where n1: Field r ≠ {} ⟶ {y1, y2} ⊆ Field r
                       and n2: (¬ (∃ y::'U. Field r − B' ⊆ {y})) ⟶ y1 ∉ B' ∧ y2 ∉
B' ∧ y1 ≠ y2 by blast
  obtain y3 where n3: (¬ (Field r − B' ⊆ {})) ⟶ y3 ∈ Field r − B' by blast
  obtain A1 where b4: A1 = ({x, y1, y2, y3} ∩ Field r) ∪ A ∪ (f ` A) by blast
  have A1 ⊆ Field r
  proof −
    have c1: A ⊆ Field r using q4 q2 unfolding Field-def by blast
    moreover have f ` A ⊆ Field r
    proof
      fix x
      assume x ∈ f ` A
      then obtain a where d2: a ∈ A ∧ x = f a by blast
      show x ∈ Field r
      proof (cases S a = {})
        assume S a = {}
        then have x = a using c1 d2 b3 by blast
        then show x ∈ Field r using d2 c1 by blast
      next
        assume S a ≠ {}
        then have x ∈ S a using d2 b3 by blast
        then show x ∈ Field r using b1 unfolding Field-def by blast
      qed
    qed
    ultimately show A1 ⊆ Field r using b4 by blast
  qed
  moreover have s0: |A1| ≤o |Field s|
  proof −
    obtain C1 where c1: C1 = {x,y1,y2,y3} ∩ Field r by blast
    obtain C2 where c2: C2 = A ∪ f ` A by blast
    have ¬ finite A using q4 q3 lem-fin-fl-rel by blast
    then have |C2| =o |A| using c2 b4 q3 by simp
```

then have $|C2| \leq o\ |Field\ s|$ **unfolding** *q4* **using** *ordIso-iff-ordLeq* **by** *blast*
moreover have *c3*: $\neg$ *finite* (*Field s*) **using** *q3 lem-fin-fl-rel* **by** *blast*
moreover have $|C1| \leq o\ |Field\ s|$
proof −
have $|\{x,y1,y2,y3\}| \leq o\ |Field\ s|$ **using** *c3*
by (*meson card-of-Well-order card-of-ordLeq-finite finite.emptyI finite.insertI ordLeq-total*)
moreover have $|C1| \leq o\ |\{x,y1,y2,y3\}|$ **unfolding** *c1* **by** *simp*
ultimately show *?thesis* **using** *ordLeq-transitive* **by** *blast*
qed
ultimately have $|C1 \cup C2| \leq o\ |Field\ s|$ **unfolding** *b4* **using** *card-of-Un-ordLeq-infinite* **by** *blast*
moreover have $A1 = C1 \cup C2$ **using** *c1 c2 b4* **by** *blast*
ultimately show *?thesis* **by** *blast*
qed
ultimately obtain s' **where** *s1*: $CCR\ s' \wedge s \subseteq s' \wedge s' \subseteq r \wedge |s'| =o\ |s| \wedge A1 \subseteq Field\ s'$
and $s1'$: $(\forall\ P \in Ps'.\ (Field\ s' \cap P) \in SCF\ s')$
and $s1''$: $escl\ r\ A\ (Field\ s') \subseteq Field\ s'$
and $s1'''$: $(\exists\ D.\ s' = Restr\ r\ D) \wedge (Conelike\ s' \longrightarrow Conelike\ r)$
using *p1 a1 q2 q3 q4 lem-Ccext-infsubccr-set-ext-scf3*[*of r s A1 Ps' A*] **by** *blast*
obtain A' **where** *s2*: $A' = Field\ s'$ **by** *blast*
obtain s'' **where** *s3*: $s'' = Restr\ r\ A'$ **by** *blast*
then have *s4*: $s' \subseteq s'' \wedge Field\ s'' = A'$ **using** *s1 s2 lem-Relprop-fld-sat*[*of* s' *r* s''] **by** *blast*
have *s5*: $|Field\ s'| =o\ |Field\ s|$ **using** *s1 q3 lem-cardreleq-cardfldeq-inf*[*of* s' *s*] **by** *blast*
have $A1 \cup (\{x\} \cap Field\ r) \subseteq A'$ **using** *b4 s1 s2* **by** *blast*
moreover have $CCR\ (Restr\ r\ A')$
proof −
have $CCR\ s''$ **using** *s1 s2 s4 lem-Ccext-subccr-eqfld*[*of* s' s''] **by** *blast*
then show *?thesis* **using** *s3* **by** *blast*
qed
moreover have $|A'| =o\ |A1|$
proof −
have $Field\ s \subseteq A1$ **using** *q4 b4* **by** *blast*
then have $|Field\ s| \leq o\ |A1|$ **by** *simp*
then have $|A'| \leq o\ |A1|$ **using** *s2 s5 ordIso-ordLeq-trans* **by** *blast*
moreover have $|A1| \leq o\ |A'|$ **using** *s1 s2* **by** *simp*
ultimately show *?thesis* **using** *ordIso-iff-ordLeq* **by** *blast*
qed
ultimately have *b6*: $A1 \cup (\{x\} \cap Field\ r) \subseteq A' \wedge CCR\ (Restr\ r\ A') \wedge |A'| =o\ |A1|$ **by** *blast*
moreover then have $A \cup (\{x\} \cap Field\ r) \subseteq A'$ **using** *b4* **by** *blast*
moreover have $|A'| =o\ |A|$ **using** *s5 s2 q4* **by** *blast*
moreover have $\forall\, a \in A.\ r``\{a\} \subseteq B \vee r``\{a\} \cap (A' - B) \neq \{\}$
proof
fix *a*
assume *c1*: $a \in A$

have $\neg\ (r``\{a\} \subseteq B) \longrightarrow r``\{a\} \cap (A'{-}B) \neq \{\}$
proof
assume $\neg\ (r``\{a\} \subseteq B)$
then have $S\ a \neq \{\}$ **unfolding** *b1* **by** *blast*
then have $f\ a \in r``\{a\} - B$ **using** *b1 b3* **by** *blast*
moreover have $f\ a \in A'$ **using** *c1 b4 b6* **by** *blast*
ultimately show $r``\{a\} \cap (A'{-}B) \neq \{\}$ **by** *blast*
qed
then show $r``\{a\} \subseteq B \lor r``\{a\} \cap (A'{-}B) \neq \{\}$ **by** *blast*
qed
moreover have $A' \in SF\ r$ **using** *s3 s4* **unfolding** *SF-def* **by** *blast*
moreover have $(\exists\ y{::}'U.\ A' - B' \subseteq \{y\}) \longrightarrow Field\ r \subseteq (A' \cup B')$
proof
assume *c0*: $\exists\ y{::}'U.\ A' - B' \subseteq \{y\}$
show $Field\ r \subseteq (A' \cup B')$
proof (*cases* $\exists\ y{::}'U.\ A' - B' = \{y\}$)
assume *c1*: $\exists\ y{::}'U.\ A' - B' = \{y\}$
moreover have *c2*: $A' \subseteq Field\ r$ **using** *s1 s2* **unfolding** *Field-def* **by** *blast*
ultimately have $Field\ r \neq \{\}$ **by** *blast*
then have $\{y1,\ y2\} \subseteq Field\ r$ **using** *n1* **by** *blast*
then have $\{y1,\ y2\} \subseteq A'$ **using** *b4 s1 s2* **by** *fast*
then have $\neg\ (\exists y.\ Field\ r - B' \subseteq \{y\}) \longrightarrow \{y1,\ y2\} \subseteq A' - B' \land y1 \neq y2$
using *n2* **by** *blast*
moreover have $\neg\ (\{y1,\ y2\} \subseteq A' - B' \land y1 \neq y2)$ **using** *c1* **by** *force*
ultimately have $\exists\ y{::}'U.\ Field\ r - B' \subseteq \{y\}$ **by** *blast*
then show $Field\ r \subseteq A' \cup B'$ **using** *c1 c2* **by** *blast*
next
assume $\neg\ (\exists\ y{::}'U.\ A' - B' = \{y\})$
then have *c1*: $A' - B' = \{\}$ **using** *c0* **by** *blast*
show $Field\ r \subseteq (A' \cup B')$
proof (*cases* $Field\ r = \{\}$)
assume $Field\ r = \{\}$
then show $Field\ r \subseteq (A' \cup B')$ **by** *blast*
next
assume $Field\ r \neq \{\}$
moreover have *c2*: $A' \subseteq Field\ r$ **using** *s1 s2* **unfolding** *Field-def* **by** *blast*
ultimately have $Field\ r \neq \{\}$ **by** *blast*
then have $\neg\ (Field\ r - B' \subseteq \{\}) \longrightarrow \{y3\} \subseteq Field\ r$ **using** *n3* **by** *blast*
then have $\neg\ (Field\ r - B' \subseteq \{\}) \longrightarrow \{y3\} \subseteq A'$ **using** *b4 s1 s2* **by** *fast*
then have $\neg\ (Field\ r - B' \subseteq \{\}) \longrightarrow \{y3\} \subseteq A' - B'$ **using** *n3* **by** *blast*
moreover have $\neg\ (\{y3\} \subseteq A' - B')$ **using** *c1* **by** *force*
ultimately have $Field\ r - B' \subseteq \{\}$ **by** *blast*
then show $Field\ r \subseteq A' \cup B'$ **using** *c1 c2* **by** *blast*
qed
qed
qed
moreover have $(\ |Ps| \leq o\ |A| \longrightarrow (\forall\ P \in Ps.\ (A' \cap P) \in SCF\ (Restr\ r\ A'))\ )$
proof −
have *c1*: $s' \subseteq r$ **using** *s3 s4* **by** *blast*

then have *Field* $s' = $ *Field* (*Restr* r (*Field* s')) **using** *lem-Relprop-fld-sat* **by** *blast*
moreover have $s' \subseteq$ *Restr* r (*Field* s') **using** *c1* **unfolding** *Field-def* **by** *force*
ultimately have *SCF* $s' \subseteq$ *SCF* (*Restr* r (*Field* s')) **using** *lem-ccext-scf-sat*[*of* s' *Restr* r (*Field* s')] **by** *blast*
moreover have $|Ps| \leq o\ |A| \longrightarrow Ps' = Ps$ **using** *p0* **by** *simp*
ultimately show *?thesis* **using** *s1' s2* **by** *blast*
qed
moreover have *escl* r A $A' \subseteq A'$ **using** *s1'' s2* **by** *blast*
moreover have *Conelike* (*Restr* r A') $\longrightarrow$ *Conelike* r
proof
assume *c1*: *Conelike* (*Restr* r A')
obtain D **where** $s' = $ *Restr* r D **using** *s1'''* **by** *blast*
then have $s' = $ *Restr* r (*Field* s') **unfolding** *Field-def* **by** *force*
then have *Conelike* s' **using** *c1 s2* **by** *simp*
then show *Conelike* r **using** *s1'''* **by** *blast*
qed
ultimately show *?thesis* **unfolding** *clterm-def* **by** *blast*
qed

lemma *lem-Ccext-finsubccr-pext5-scf3*:
fixes r::$'U$ *rel* **and** A B B'::$'U$ *set* **and** x::$'U$ **and** Ps::$'U$ *set set*
assumes *a1*: *CCR* r **and** *a2*: *finite* A **and** *a3*: $A \in$ *SF* r **and** *a4*: $Ps \subseteq$ *SCF* r
shows $\exists\ A'$::($'U$ *set*). ($x \in$ *Field* $r \longrightarrow x \in A'$) $\wedge$ $A \subseteq A'$ $\wedge$ *CCR* (*Restr* r A') $\wedge$ *finite* A'
$\wedge$ ($\forall a{\in}A.\ r``\{a\}{\subseteq}B \vee r``\{a\}{\cap}(A'{-}B) \neq \{\}$) $\wedge$ $A' \in$ *SF* r
$\wedge$ (($\exists\ y$::$'U$. $A'{-}B' \subseteq \{y\}$) $\longrightarrow$ *Field* $r \subseteq (A'{\cup}B')$)
$\wedge$ (($\exists\ P.\ Ps = \{P\}$) $\longrightarrow$ ($\forall\ P \in Ps.\ (A' \cap P) \in$ *SCF* (*Restr* r A')))
proof $-$
obtain P **where** *p0*: $P = $ (*if* ($Ps \neq \{\}$) *then* (*SOME* P. $P \in Ps$) *else Field* r) **by** *blast*
moreover have *Field* $r \in$ *SCF* r **unfolding** *SCF-def* **by** *blast*
ultimately have *p1*: $P \in$ *SCF* r **using** *a4* **by** (*metis contra-subsetD some-in-eq*)
have *p2*: ($\exists\ P.\ Ps = \{P\}$) $\longrightarrow Ps = \{P\}$ **using** *p0* **by** *fastforce*
have *q1*: *Field* (*Restr* r A) $= A$ **using** *a3* **unfolding** *SF-def* **by** *blast*
obtain s **where** $s = $ (*Restr* r A) **by** *blast*
then have *q2*: $s \subseteq r$ **and** *q3*: *finite* s **and** *q4*: $A = $ *Field* s
using *a2 q1 lem-fin-fl-rel* **by** (*blast*, *metis*, *blast*)
obtain S **where** *b1*: $S = (\lambda\ a.\ r``\{a\} - B\ )$ **by** *blast*
obtain S' **where** *b2*: $S' = (\lambda\ a.$ *if* $(S\ a) \neq \{\}$ *then* $(S\ a)$ *else* $\{a\})$ **by** *blast*
obtain f **where** $f = (\lambda\ a.$ *SOME* $b.\ b \in S'\ a)$ **by** *blast*
moreover have $\forall\ a.\ \exists\ b.\ b \in (S'\ a)$ **unfolding** *b2* **by** *force*
ultimately have $\forall\ a.\ f\ a \in S'\ a$ **by** (*metis someI-ex*)
then have *b3*: $\forall\ a.\ (S\ a \neq \{\} \longrightarrow f\ a \in S\ a) \wedge (S\ a = \{\} \longrightarrow f\ a = a)$
unfolding *b2* **by** (*clarsimp*, *metis singletonD*)
obtain *y1 y2*::$'U$ **where** *n1*: *Field* $r \neq \{\} \longrightarrow \{y1,\ y2\} \subseteq$ *Field* r
and *n2*: ($\neg$ ($\exists\ y$::$'U$. *Field* $r - B' \subseteq \{y\}$)) $\longrightarrow y1 \notin B' \wedge y2 \notin B' \wedge y1 \neq y2$ **by** *blast*

obtain *y3* **where** *n3*: $(\neg\ (Field\ r - B' \subseteq \{\})) \longrightarrow y3 \in Field\ r - B'$ **by** *blast*
obtain *A1* **where** *b4*: $A1 = (\{x,y1,y2,y3\} \cap Field\ r) \cup A \cup (f\ `\ A)$ **by** *blast*
have $A1 \subseteq Field\ r$
proof −
have *c1*: $A \subseteq Field\ r$ **using** *q4 q2* **unfolding** *Field-def* **by** *blast*
moreover have $f\ `\ A \subseteq Field\ r$
proof
fix *x*
assume $x \in f\ `\ A$
then obtain *a* **where** *d2*: $a \in A \wedge x = f\ a$ **by** *blast*
show $x \in Field\ r$
proof (*cases* $S\ a = \{\}$)
assume $S\ a = \{\}$
then have $x = a$ **using** *c1 d2 b3* **by** *blast*
then show $x \in Field\ r$ **using** *d2 c1* **by** *blast*
next
assume $S\ a \neq \{\}$
then have $x \in S\ a$ **using** *d2 b3* **by** *blast*
then show $x \in Field\ r$ **using** *b1* **unfolding** *Field-def* **by** *blast*
qed
qed
ultimately show $A1 \subseteq Field\ r$ **using** *b4* **by** *blast*
qed
moreover have *s0*: *finite A1* **using** *b4 q3 q4 lem-fin-fl-rel* **by** *blast*
ultimately obtain *s'* **where** *s1*: $CCR\ s' \wedge s \subseteq s' \wedge s' \subseteq r \wedge finite\ s' \wedge A1 \subseteq Field\ s'$
and *s1'*: $(\exists\ P.\ Ps = \{P\}) \longrightarrow (Field\ s' \cap P) \in SCF\ s'$
using *p1 a1 a4 q2 q3 lem-Ccext-finsubccr-set-ext-scf*[*of r s A1 P*] **by** *metis*
obtain *A'* **where** *s2*: $A' = Field\ s'$ **by** *blast*
obtain *s''* **where** *s3*: $s'' = Restr\ r\ A'$ **by** *blast*
then have *s4*: $s' \subseteq s'' \wedge Field\ s'' = A'$ **using** *s1 s2 lem-Relprop-fld-sat*[*of s' r s''*] **by** *blast*
have *s5*: *finite* (*Field s'*) **using** *s1 lem-fin-fl-rel* **by** *blast*
have $A1 \cup (\{x\} \cap Field\ r) \subseteq A'$ **using** *b4 s1 s2* **by** *blast*
moreover have *CCR* (*Restr r A'*)
proof −
have *CCR s''* **using** *s1 s2 s4 lem-Ccext-subccr-eqfld*[*of s' s''*] **by** *blast*
then show *?thesis* **using** *s3* **by** *blast*
qed
ultimately have *b6*: $A1 \cup (\{x\} \cap Field\ r) \subseteq A' \wedge CCR\ (Restr\ r\ A')$ **by** *blast*
moreover then have $A \cup (\{x\} \cap Field\ r) \subseteq A'$ **using** *b4* **by** *blast*
ultimately have $(x \in Field\ r \longrightarrow x \in A') \wedge A \subseteq A' \wedge CCR\ (Restr\ r\ A')$ **by** *blast*
moreover have *finite A'* **using** *s2 s5* **by** *blast*
moreover have $\forall\ a{\in}A.\ r``\{a\} \subseteq B \vee r``\{a\} \cap (A'{-}B) \neq \{\}$
proof
fix *a*
assume *c1*: $a \in A$
have $\neg\ (r``\{a\} \subseteq B) \longrightarrow r``\{a\} \cap (A'{-}B) \neq \{\}$

proof
assume $\neg$ $(r``\{a\} \subseteq B)$
then have $S\ a \neq \{\}$ **unfolding** *b1* **by** *blast*
then have $f\ a \in r``\{a\} - B$ **using** *b1 b3* **by** *blast*
moreover have $f\ a \in A'$ **using** *c1 b4 b6* **by** *blast*
ultimately show $r``\{a\} \cap (A'-B) \neq \{\}$ **by** *blast*
qed
then show $r``\{a\} \subseteq B \vee r``\{a\} \cap (A'-B) \neq \{\}$ **by** *blast*
qed
moreover have $A' \in SF\ r$ **using** *s3 s4* **unfolding** *SF-def* **by** *blast*
moreover have $(\exists\ y::'U.\ A' - B' \subseteq \{y\}) \longrightarrow Field\ r \subseteq (A' \cup B')$
proof
assume *c0*: $\exists\ y::'U.\ A' - B' \subseteq \{y\}$
show $Field\ r \subseteq (A' \cup B')$
proof (*cases* $\exists\ y::'U.\ A' - B' = \{y\}$)
assume *c1*: $\exists\ y::'U.\ A' - B' = \{y\}$
moreover have *c2*: $A' \subseteq Field\ r$ **using** *s1 s2* **unfolding** *Field-def* **by** *blast*
ultimately have $Field\ r \neq \{\}$ **by** *blast*
then have $\{y1,\ y2\} \subseteq Field\ r$ **using** *n1* **by** *blast*
then have $\{y1,\ y2\} \subseteq A'$ **using** *b4 s1 s2* **by** *fast*
then have $\neg$ $(\exists y.\ Field\ r - B' \subseteq \{y\}) \longrightarrow \{y1,\ y2\} \subseteq A' - B' \wedge y1 \neq y2$ **using** *n2* **by** *blast*
moreover have $\neg$ $(\{y1,\ y2\} \subseteq A' - B' \wedge y1 \neq y2)$ **using** *c1* **by** *force*
ultimately have $\exists\ y::'U.\ Field\ r - B' \subseteq \{y\}$ **by** *blast*
then show $Field\ r \subseteq A' \cup B'$ **using** *c1 c2* **by** *blast*
next
assume $\neg$ $(\exists\ y::'U.\ A' - B' = \{y\})$
then have *c1*: $A' - B' = \{\}$ **using** *c0* **by** *blast*
show $Field\ r \subseteq (A' \cup B')$
proof (*cases* $Field\ r = \{\}$)
assume $Field\ r = \{\}$
then show $Field\ r \subseteq (A' \cup B')$ **by** *blast*
next
assume $Field\ r \neq \{\}$
moreover have *c2*: $A' \subseteq Field\ r$ **using** *s1 s2* **unfolding** *Field-def* **by** *blast*
ultimately have $Field\ r \neq \{\}$ **by** *blast*
then have $\neg$ $(Field\ r - B' \subseteq \{\}) \longrightarrow \{y3\} \subseteq Field\ r$ **using** *n3* **by** *blast*
then have $\neg$ $(Field\ r - B' \subseteq \{\}) \longrightarrow \{y3\} \subseteq A'$ **using** *b4 s1 s2* **by** *fast*
then have $\neg$ $(Field\ r - B' \subseteq \{\}) \longrightarrow \{y3\} \subseteq A' - B'$ **using** *n3* **by** *blast*
moreover have $\neg$ $(\{y3\} \subseteq A' - B')$ **using** *c1* **by** *force*
ultimately have $Field\ r - B' \subseteq \{\}$ **by** *blast*
then show $Field\ r \subseteq A' \cup B'$ **using** *c1 c2* **by** *blast*
qed
qed
qed
moreover have $(\exists\ P.\ Ps = \{P\}) \longrightarrow (\forall\ P \in Ps.\ (A' \cap P) \in SCF\ (Restr\ r\ A'))$
proof $-$
have *c1*: $s' \subseteq r$ **using** *s3 s4* **by** *blast*

then have *Field s′* = *Field* (*Restr r* (*Field s′*)) **using** *lem-Relprop-fld-sat* **by** *blast*
moreover have *s′* ⊆ *Restr r* (*Field s′*) **using** *c1* **unfolding** *Field-def* **by** *force*
ultimately have *SCF s′* ⊆ *SCF* (*Restr r* (*Field s′*)) **using** *lem-ccext-scf-sat*[*of s′ Restr r* (*Field s′*)] **by** *blast*
then show *?thesis* **using** *p2 s1′ s2* **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Ccext-subccr-pext5-scf3*:
fixes *r*::′*U rel* **and** *A B B′*::′*U set* **and** *x*::′*U* **and** *Ps*::′*U set set* **and** *C*::′*U set* ⇒ *bool*
assumes *a1*: *CCR r* **and** *a2*: *A* ∈ *SF r* **and** *a3*: *Ps* ⊆ *SCF r*
and *a4*: *C* = (λ *A′*::′*U set*. (*x* ∈ *Field r* ⟶ *x* ∈ *A′*)
∧ *A* ⊆ *A′*
∧ *A′* ∈ *SF r*
∧ (∀ *a*∈*A*. ((*r*“{*a*}⊆*B*) ∨ (*r*“{*a*}∩(*A′*−*B*) ≠ {})))
∧ ((∃ *y*::′*U*. *A′*−*B′* ⊆ {*y*}) ⟶ *Field r* ⊆ (*A′*∪*B′*))
∧ *CCR* (*Restr r A′*)
∧ ((*finite A* ⟶ *finite A′*) ∧ ((¬ *finite A*) ⟶ |*A′*| =*o* |*A*|))
∧ (((∃ *P*. *Ps* = {*P*}) ∨ ((¬ *finite Ps*) ∧ |*Ps*| ≤*o* |*A*|)) ⟶
(∀ *P* ∈ *Ps*. (*A′* ∩ *P*) ∈ *SCF* (*Restr r A′*)))
∧ ((¬ *finite A*) ⟶ ((*escl r A A′* ⊆ *A′*) ∧ (*clterm* (*Restr r A′*) *r*))))
shows ∃ *A′*::(′*U set*). *C A′*
proof (*cases finite A*)
assume *b1*: *finite A*
then obtain *A′*::′*U set* **where** *b2*: (*x* ∈ *Field r* ⟶ *x* ∈ *A′*) ∧ *A* ⊆ *A′* ∧ *CCR* (*Restr r A′*)
∧ (∀ *a*∈*A*. *r*“{*a*}⊆*B* ∨ *r*“{*a*}∩(*A′*−*B*) ≠ {}) ∧ *A′* ∈ *SF r*
∧ ((∃ *y*::′*U*. *A′*−*B′* ⊆ {*y*}) ⟶ *Field r* ⊆ (*A′*∪*B′*))
and *b3*: *finite A′* ∧ ((∃ *P*. *Ps* = {*P*}) ⟶ (∀ *P* ∈ *Ps*. (*A′* ∩ *P*) ∈ *SCF* (*Restr r A′*)))
using *a1 a2 a3 lem-Ccext-finsubccr-pext5-scf3*[*of r A Ps x B B′*] **by** *metis*
have *b4*: ((*finite A* ⟶ *finite A′*) ∧ ((¬ *finite A*) ⟶ |*A′*| =*o* |*A*|))
and *b5*: (((∃ *P*. *Ps* = {*P*}) ∨ ((¬ *finite Ps*) ∧ |*Ps*| ≤*o* |*A*|)) ⟶ (∀ *P* ∈ *Ps*. (*A′* ∩ *P*) ∈ *SCF* (*Restr r A′*)))
using *b1 b3 card-of-ordLeq-finite* **by** *blast+*
show *?thesis*
apply (*rule exI*)
unfolding *a4* **using** *b1 b2 b4 b5* **by** *force*
next
assume *b1*: ¬ *finite A*
then obtain *A′* **where** *b2*: (*x* ∈ *Field r* ⟶ *x* ∈ *A′*) ∧ *A* ⊆ *A′* ∧ *CCR* (*Restr r A′*)
∧ (∀ *a*∈*A*. *r*“{*a*}⊆*B* ∨ *r*“{*a*}∩(*A′*−*B*) ≠ {}) ∧ *A′* ∈ *SF r*
∧ ((∃ *y*::′*U*. *A′*−*B′* ⊆ {*y*}) ⟶ *Field r* ⊆ (*A′*∪*B′*))

and *b3*: $|A'| =o |A| \wedge (|Ps| \leq o |A| \longrightarrow (\forall\ P \in Ps.\ (A' \cap P) \in SCF$ $(Restr\ r\ A')))$
and *b3'*: $(escl\ r\ A\ A' \subseteq A') \wedge clterm\ (Restr\ r\ A')\ r$
using *a1 a2 a3 lem-Ccext-infsubccr-pext5-scf3*[*of r A Ps x B B'*] **by** *metis*
have *b4*: $((finite\ A \longrightarrow finite\ A') \wedge ((\neg\ finite\ A) \longrightarrow |A'| =o |A|))$
using *b1 b3* **by** *metis*
have *b5*: $(((\exists\ P.\ Ps = \{P\}) \vee ((\neg\ finite\ Ps) \wedge |Ps| \leq o |A|)) \longrightarrow (\forall\ P \in Ps.$ $(A' \cap P) \in SCF\ (Restr\ r\ A')))$
using *b1 b3* **by** (*metis card-of-singl-ordLeq finite.simps*)
have *b6*: $((\neg\ finite\ A) \longrightarrow ((escl\ r\ A\ A' \subseteq A') \wedge clterm\ (Restr\ r\ A')\ r))$ **using** *b3'* **by** *blast*
have *C A'* **unfolding** *a4* **using** *b2 b4 b5 b6* **by** *simp*
then show *?thesis* **by** *blast*
qed

lemma *lem-acyc-un-emprd*:
fixes *r s*:: *'U rel*
assumes *a1*: $acyclic\ r \wedge acyclic\ s$ **and** *a2*: $(Range\ r) \cap (Domain\ s) = \{\}$
shows $acyclic\ (r \cup s)$
proof −
have $\bigwedge n.\ (r \cup s)\widehat{\ }\widehat{\ }n \subseteq s\widehat{\ }*\ O\ r\widehat{\ }*$
proof −
fix *n*
show $(r \cup s)\widehat{\ }\widehat{\ }n \subseteq s\widehat{\ }*\ O\ r\widehat{\ }*$
proof (*induct n*)
show $(r \cup s)\widehat{\ }\widehat{\ }0 \subseteq s\widehat{\ }*\ O\ r\widehat{\ }*$ **by** *force*
next
fix *n*
assume $(r \cup s)\widehat{\ }\widehat{\ }n \subseteq s\widehat{\ }*\ O\ r\widehat{\ }*$
moreover then have $(r \cup s)\widehat{\ }\widehat{\ }n\ O\ r \subseteq s\widehat{\ }*\ O\ r\widehat{\ }*$ **by** *force*
moreover have $(s\widehat{\ }*\ O\ r\widehat{\ }*)\ O\ s \subseteq s\widehat{\ }*\ O\ r\widehat{\ }*$
proof −
have $r\widehat{\ }+\ O\ s = r\widehat{\ }*\ O\ (r\ O\ s)$ **by** (*simp add*: *O-assoc trancl-unfold-right*)
moreover have $r\ O\ s = \{\}$ **using** *a2* **by** *force*
ultimately have $s\widehat{\ }*\ O\ (r\widehat{\ }+\ O\ s) = \{\}$ **by** *force*
moreover have $s\widehat{\ }*\ O\ s \subseteq s\widehat{\ }*$ **by** *force*
moreover have $r\widehat{\ }* = Id \cup r\widehat{\ }+$ **by** (*metis rtrancl-unfold trancl-unfold-right*)
moreover then have $(s\widehat{\ }*\ O\ r\widehat{\ }*)\ O\ s = (s\widehat{\ }*\ O\ s) \cup (s\widehat{\ }*\ O\ (r\widehat{\ }+\ O\ s))$ **by** *fastforce*
ultimately show *?thesis* **by** *fastforce*
qed
moreover have $(r \cup s)\widehat{\ }\widehat{\ }(Suc\ n) = ((((r \cup s)\widehat{\ }\widehat{\ }n)\ O\ r) \cup (((r \cup s)\widehat{\ }\widehat{\ }n)\ O\ s))$ **by** *simp*
ultimately show $(r \cup s)\ \widehat{\ }\widehat{\ }\ (Suc\ n) \subseteq s\widehat{\ }*\ O\ r\widehat{\ }*$ **by** *force*
qed
qed
then have *b1*: $(r \cup s)\widehat{\ }* \subseteq s\widehat{\ }*\ O\ r\widehat{\ }*$ **using** *rtrancl-power*[*of - r ∪ s*] **by** *blast*
have $\forall x.\ (x,x) \in (r \cup s)\widehat{\ }+ \longrightarrow False$
proof (*intro allI impI*)

fix x
assume $(x,x) \in (r \cup s)\hat{}+$
then have $(x,x) \in (r \cup s)\hat{}* \; O \; (r \cup s)$ **using** *trancl-unfold-right* **by** *blast*
then have $(x,x) \in ((s\hat{}* \; O \; r\hat{}*) \; O \; r) \cup ((s\hat{}* \; O \; r\hat{}*) \; O \; s)$ **using** *b1* **by** *force*
moreover have $(x,x) \in ((s\hat{}* \; O \; r\hat{}*) \; O \; r) \longrightarrow False$
proof
assume $(x,x) \in ((s\hat{}* \; O \; r\hat{}*) \; O \; r)$
then obtain $u \; v$ **where** *d1*: $(x,u) \in s\hat{}* \wedge (u,v) \in r\hat{}* \wedge (v,x) \in r$ **by** *blast*
moreover then have $x \notin Domain \; s$ **using** *a2* **by** *blast*
ultimately have $x = u$ **by** (*meson Not-Domain-rtrancl*)
then have $(x,x) \in r\hat{}+$ **using** *d1* **by** *force*
then show *False* **using** *a1* **unfolding** *acyclic-def* **by** *blast*
qed
moreover have $(x,x) \in ((s\hat{}* \; O \; r\hat{}*) \; O \; s) \longrightarrow False$
proof
assume $(x,x) \in ((s\hat{}* \; O \; r\hat{}*) \; O \; s)$
then obtain $u \; v$ **where** *d1*: $(x,u) \in s\hat{}* \wedge (u,v) \in r\hat{}* \wedge (v,x) \in s$ **by** *blast*
have $u = v \longrightarrow False$
proof
assume $u = v$
then have $(x,x) \in s\hat{}+$ **using** *d1* **by** *force*
then show *False* **using** *a1* **unfolding** *acyclic-def* **by** *blast*
qed
then have $(u,v) \in r\hat{}+$ **using** *d1* **by** (*meson rtranclD*)
then have $v \in Range \; r$ **using** *trancl-unfold-right*[*of r*] **by** *force*
moreover have $v \in Domain \; s$ **using** *d1* **by** *blast*
ultimately show *False* **using** *a2* **by** *blast*
qed
ultimately show *False* **by** *blast*
qed
then show *?thesis* **using** *a1* **unfolding** *acyclic-def* **by** *blast*
qed

lemma *lem-spthlen-rtr*: $(a,b) \in r\hat{}* \Longrightarrow (a,b) \in r\hat{}\hat{}(spthlen \; r \; a \; b)$
using *rtrancl-power* **unfolding** *spthlen-def* **by** (*metis LeastI-ex*)

lemma *lem-spthlen-tr*: $(a,b) \in r\hat{}* \wedge a \neq b \Longrightarrow (a,b) \in r\hat{}\hat{}(spthlen \; r \; a \; b) \wedge spthlen \; r \; a \; b > 0$
proof −
assume $(a,b) \in r\hat{}* \wedge a \neq b$
moreover then have *b1*: $(a,b) \in r\hat{}\hat{}(spthlen \; r \; a \; b)$ **using** *lem-spthlen-rtr*[*of a b*] **by** *force*
ultimately have $spthlen \; r \; a \; b = 0 \longrightarrow False$ **by** *force*
then show *?thesis* **using** *b1* **by** *blast*
qed

lemma *lem-spthlen-min*: $(a,b) \in r\hat{}\hat{}n \Longrightarrow spthlen \; r \; a \; b \leq n$
unfolding *spthlen-def* **by** (*metis Least-le*)

```
lemma lem-spth-inj:
fixes r::'U rel and a b::'U and f::nat ⇒ 'U and n::nat
assumes a1: f ∈ spth r a b and a2: n = spthlen r a b
shows inj-on f {i. i≤n}
proof −
  have b1: f ∈ rpth r a b n using a1 a2 unfolding spth-def by blast
  have ∀ i j. i ≤ n ∧ j ≤ n ∧ i < j ⟶ f i = f j ⟶ False
  proof (intro allI impI)
    fix i j
    assume c1: i ≤ n ∧ j ≤ n ∧ i < j and c2: f i = f j
    obtain l where c3: l = j − i by blast
    then have c4: l ≠ 0 using c1 by simp
    obtain g where c5: g = (λ k. if (k≤i) then (f k) else (f (k + l))) by blast
    then have g 0 = a using b1 unfolding rpth-def by fastforce
    moreover have g (n − l) = b
    proof (cases j < n)
      assume j < n
      then show ?thesis using c5 c3 b1 unfolding rpth-def by simp
    next
      assume ¬ j < n
      then have j = n using c1 by simp
      then show ?thesis using c5 c2 c3 c4 b1 unfolding rpth-def by simp
    qed
    moreover have ∀ k < n − l. (g k, g (Suc k)) ∈ r
    proof (intro allI impI)
      fix k
      assume d1: k < n − l
      have k ≠ i ⟶ (g k, g (Suc k)) ∈ r using c5 d1 b1 unfolding rpth-def by
fastforce
      moreover have k = i ⟶ (g k, g (Suc k)) ∈ r
      proof
        assume e1: k = i
        then have (g k, g (Suc k)) = (f i, f ((Suc i) + l)) using c5 by simp
        moreover have f i = f (i + l) using c1 c2 c3 by simp
        moreover have i + l < n using d1 e1 by force
         ultimately show (g k, g (Suc k)) ∈ r using b1 unfolding rpth-def by
simp
      qed
      ultimately show (g k, g (Suc k)) ∈ r by force
    qed
    ultimately have g ∈ rpth r a b (n − l) unfolding rpth-def by blast
    then have spthlen r a b ≤ n − l
      using lem-spthlen-min[of a b] lem-ccext-ntr-rpth[of a b] by blast
    then show False using a2 c1 c3 by force
  qed
  moreover then have ∀ i j. i ≤ n ∧ j ≤ n ∧ j < i ⟶ f i = f j ⟶ False by
metis
   ultimately show ?thesis unfolding inj-on-def by (metis linorder-neqE-nat
```

mem-Collect-eq)
qed

lemma *lem-rtn-rpth-inj*: (*a*,*b*) ∈ *r*^^*n* ⟹ *n* = *spthlen r a b* ⟹ ∃ *f* . *f* ∈ *rpth r a b n* ∧ *inj-on f* {*i*. *i* ≤ *n*}
proof –
  assume *a1*: (*a*,*b*) ∈ *r*^^*n* **and** *a2*: *n* = *spthlen r a b*
  then have (*a*,*b*) ∈ *r*^^*n* **using** *lem-spthlen-rtr*[*of a b*] *rtrancl-power* **by** *blast*
  then obtain *f* **where** *b2*: *f* ∈ *rpth r a b n* **using** *lem-ccext-ntr-rpth*[*of a b*] **by** *blast*
  then have *f* ∈ *spth r a b* **using** *a2* **unfolding** *spth-def* **by** *blast*
  then have *inj-on f* {*i*. *i* ≤ *n*} **using** *a2 lem-spth-inj*[*of f*] **by** *blast*
  then show *?thesis* **using** *b2* **by** *blast*
qed

lemma *lem-rtr-rpth-inj*: (*a*,*b*) ∈ *r*^* ⟹ ∃ *f n* . *f* ∈ *rpth r a b n* ∧ *inj-on f* {*i*. *i* ≤ *n*}
  using *lem-spthlen-rtr*[*of a b r*] *lem-rtn-rpth-inj*[*of a b - r*] **by** *blast*

lemma *lem-sum-ind-ex*:
assumes *a1*: *g* = (λ*n*::*nat*. ∑ *i*<*n*. *f i*)
    and *a2*:∀ *i*::*nat*. *f i* > *0*
shows ∃ *n k*. (*m*::*nat*) = *g n* + *k* ∧ *k* < *f n*
proof(*induct m*)
  have *0* = *g 0* + *0* ∧ *0* < *f 0* **using** *a1 a2* **by** *simp*
  then show ∃ *n k*. (*0*::*nat*) = *g n* + *k* ∧ *k* < *f n* **by** *blast*
next
  fix *m*
  assume ∃ *n k*. *m* = *g n* + *k* ∧ *k* < *f n*
  then obtain *n k* **where** *b1*: *m* = *g n* + *k* ∧ *k* < *f n* **by** *blast*
  show ∃ *n′ k′*. *Suc m* = *g n′* + *k′* ∧ *k′* < *f n′*
  proof(*cases Suc k* < *f n*)
    assume *Suc k* < *f n*
    then have *Suc m* = *g n* + (*Suc k*) ∧ (*Suc k*) < *f n* **using** *b1* **by** *simp*
    then show ∃ *n′ k′*. *Suc m* = *g n′* + *k′* ∧ *k′* < *f n′* **by** *blast*
  next
    assume ¬ *Suc k* < *f n*
    then have *Suc m* = *g* (*Suc n*) + *0* ∧ *0* < *f* (*Suc n*) **using** *a1 a2 b1* **by** *simp*
    then show ∃ *n′ k′*. *Suc m* = *g n′* + *k′* ∧ *k′* < *f n′* **by** *blast*
  qed
qed

lemma *lem-sum-ind-un*:
assumes *a1*: *g* = (λ*n*::*nat*. ∑ *i*<*n*. *f i*)
    and *a2*: ∀ *i*::*nat*. *f i* > *0*
    and *a3*: (*m*::*nat*) = *g n* + *k* ∧ *k* < *f n*
    and *a4*: *m* = *g n′* + *k′* ∧ *k′* < *f n′*
shows *n* = *n′* ∧ *k* = *k′*
proof –

have *b1*: $\forall$ *n1 n2*. *n1* $\leq$ *n2* $\longrightarrow$ *g n1* $\leq$ *g n2*
proof(*intro allI impI*)
fix *n1*::*nat* **and** *n2*::*nat*
assume *n1* $\leq$ *n2*
moreover obtain *t* **where** $t = n2 - n1$ **by** *blast*
moreover have *g n1* $\leq$ *g* (*n1* + *t*) **unfolding** *a1* **by** (*induct t*, *simp*+)
ultimately show *g n1* $\leq$ *g n2* **by** *simp*
qed
have $n < n' \longrightarrow$ *False*
proof
assume $n < n'$
then have *g* (*Suc n*) $\leq$ *g* n' **using** *b1* **by** *simp*
then have *g n* + *f n* $\leq$ *g* n' **using** *a1 b1* **by** *simp*
moreover have *g* n' < *g n* + *f n* **using** *a3 a4* **by** *simp*
ultimately show *False* **by** *simp*
qed
moreover have $n' < n \longrightarrow$ *False*
proof
assume $n' < n$
then have *g* (*Suc* n') $\leq$ *g n* **using** *b1* **by** *simp*
then have *g* n' + *f* n' $\leq$ *g n* **using** *a1 b1* **by** *simp*
moreover have *g n* < *g* n' + *f* n' **using** *a3 a4* **by** *simp*
ultimately show *False* **by** *simp*
qed
ultimately show $n = n' \wedge k = k'$ **using** *a3 a4* **by** *simp*
qed

lemma *lem-flatseq*:
fixes *r*::'*U rel* **and** *xi*::*nat* $\Rightarrow$ '*U*
assumes $\forall$ *n*. (*xi n*, *xi* (*Suc n*)) $\in$ *r*^* $\wedge$ (*xi n* $\neq$ *xi* (*Suc n*))
shows $\exists$ *g yi*. ($\forall$ *n*. (*yi n*, *yi* (*Suc n*)) $\in$ *r*)
$\wedge$ ($\forall$ *i*::*nat*. $\forall$ *j*::*nat*. *i* < *j* $\longleftrightarrow$ *g i* < *g j*)
$\wedge$ ($\forall$ *i*::*nat*. *yi* (*g i*) = *xi i*)
$\wedge$ ($\forall$ *i*::*nat*. *inj-on yi* { *k*. *g i* $\leq$ *k* $\wedge$ *k* $\leq$ *g* (*Suc i*) })
$\wedge$ ($\forall$ *k*::*nat*. $\exists$ *i*::*nat*. *g i* $\leq$ *k* $\wedge$ *Suc k* $\leq$ *g* (*Suc i*))
$\wedge$ ($\forall$ *k i i'*. *g i* $\leq$ *k* $\wedge$ *Suc k* $\leq$ *g* (*Suc i*) $\wedge$ *g i'* $\leq$ *k* $\wedge$ *Suc k* $\leq$ *g* (*Suc i'*) $\longrightarrow$ *i* = *i'*)
proof –
obtain *P* **where** *b0*: *P* = (λ *n m*. *m* > *0* $\wedge$ (*xi n*, *xi* (*Suc n*)) $\in$ *r*^^*m* $\wedge$ *m* = *spthlen r* (*xi n*) (*xi* (*Suc n*))) **by** *blast*
then have $\forall$ *n*. $\exists$ *m*. *P n m* **using** *assms lem-spthlen-tr*[*of - - r*] **by** *blast*
then obtain *f* **where** $\forall$ *n*. *P n* (*f n*) **by** *metis*
then have *b1*: $\forall$ *n*. (*f n*) > *0* $\wedge$ (*xi n*, *xi* (*Suc n*)) $\in$ *r*^^(*f n*)
and *b1'*: $\forall$ *n*. (*f n*) = *spthlen r* (*xi n*) (*xi* (*Suc n*)) **using** *b0* **by** *blast*+
have $\forall$ *n*. $\exists$ *yi*. *inj-on yi* {*i*. *i* $\leq$ *f n*} $\wedge$ (*yi 0*) = (*xi n*) $\wedge$
($\forall$ *k*<(*f n*). (*yi k*, *yi* (*Suc k*)) $\in$ *r*) $\wedge$ (*yi* (*f n*)) = (*xi* (*Suc n*))
proof
fix *n*
have (*xi n*, *xi* (*Suc n*)) $\in$ *r*^^(*f n*) **and** (*f n*) = *spthlen r* (*xi n*) (*xi* (*Suc n*))

using *b1 b1′* **by** *blast+*
then obtain *yi* **where** *yi* ∈ *rpth r (xi n) (xi (Suc n)) (f n)* ∧ *inj-on yi* {*i. i* ≤ *f n*}
using *lem-rtn-rpth-inj*[*of xi n xi (Suc n) f n r*] **by** *blast*
then show ∃ *yi. inj-on yi* {*i. i* ≤ *f n*} ∧ (*yi 0*) = (*xi n*) ∧ (∀ *k*<(*f n*). (*yi k*, *yi* (*Suc k*)) ∈ *r*)
∧ (*yi* (*f n*)) = (*xi* (*Suc n*)) **unfolding** *rpth-def* **by** *blast*
qed
then obtain *yin* **where** *b2*: ∀ *n. inj-on* (*yin n*) {*i. i* ≤ *f n*} ∧ ((*yin n*) *0*) = (*xi n*) ∧
(∀ *k* < (*f n*). ((*yin n*) *k*, (*yin n*) (*Suc k*)) ∈ *r*) ∧ ((*yin n*) (*f n*)) = (*xi* (*Suc n*)) **by** *metis*
obtain *g* **where** *b3*: *g* = (λ*n*. ∑ *i*<*n*. *f i*) **by** *blast*
obtain *yi* **where** *b4*: *yi* = (λ*m*. *let p* =
(*SOME p*. *m* = (*g* (*fst p*)) + (*snd p*) ∧ (*snd p*) < (*f* (*fst p*)))
in (*yin* (*fst p*)) (*snd p*)) **by** *blast*
have *b5*: ⋀ *m n k*. *m* = (*g n*) + *k* ∧ *k* < *f n* ⟹ *yi m* = *yin n k*
proof −
fix *m n k*
assume *c0*: *m* = (*g n*) + *k* ∧ *k* < *f n*
have ∃ *p* . (*m* = (*g* (*fst p*)) + (*snd p*)) ∧ ((*snd p*) < (*f* (*fst p*)))
using *b1 b3 lem-sum-ind-ex* **by** *force*
then obtain *n′ k′* **where** *m* = (*g n′*) + *k′* ∧ *k′* < (*f n′*) ∧ *yi m* = (*yin n′*) *k′*
using *b4* **by** (*smt someI-ex*)
moreover then have *n′* = *n* ∧ *k′* = *k* **using** *c0 b1 b3 lem-sum-ind-un*[*of g f m n′ k′ n k*] **by** *blast*
ultimately show *yi m* = *yin n k* **by** *blast*
qed
have ∀ *m*. (*yi m*, *yi* (*Suc m*)) ∈ *r*
proof
fix *m*
have ∃ *p* . (*m* = (*g* (*fst p*)) + (*snd p*)) ∧ ((*snd p*) < (*f* (*fst p*)))
using *b1 b3 lem-sum-ind-ex* **by** *force*
then obtain *n k* **where** *c1*: *m* = (*g n*) + *k* ∧ *k* < (*f n*) ∧ *yi m* = (*yin n*) *k*
using *b4* **by** (*smt someI-ex*)
have ∃ *p* . ((*Suc m*) = (*g* (*fst p*)) + (*snd p*)) ∧ ((*snd p*) < (*f* (*fst p*)))
using *b1 b3 lem-sum-ind-ex* **by** *force*
then obtain *n′ k′* **where** *c2*: (*Suc m*) = (*g n′*) + *k′* ∧ *k′* < (*f n′*) ∧ *yi* (*Suc m*) = (*yin n′*) *k′*
using *b4* **by** (*smt someI-ex*)
show (*yi m*, *yi* (*Suc m*)) ∈ *r*
proof(*cases Suc k* < *f n*)
assume *Suc k* < *f n*
then have *Suc m* = *g n* + (*Suc k*) ∧ (*Suc k*) < *f n* **using** *c1* **by** *simp*
then have *n′* = *n* ∧ *k′* = *Suc k* **using** *b1 b3 c2 lem-sum-ind-un*[*of g*] **by** *blast*
then show (*yi m*, *yi* (*Suc m*)) ∈ *r* **using** *b2 c1 c2* **by** *force*
next
assume *d1*: ¬ *Suc k* < *f n*
then have *Suc m* = *g* (*Suc n*) + *0* ∧ *0* < *f* (*Suc n*) **using** *b1 b3 c1* **by** *simp*

```
   then have n' = Suc n ∧ k' = 0 using b1 b3 c2 lem-sum-ind-un[of g] by blast
    then show (yi m, yi (Suc m)) ∈ r
      using b2 c1 c2 d1 by (metis Suc-le-eq dual-order.antisym not-less)
  qed
qed
moreover have b6: ∀ j::nat. ∀ i::nat. i < j ⟶ g i < g j
proof
  fix j0::nat
  show ∀ i::nat. i < j0 ⟶ g i < g j0
  proof (induct j0)
    show ∀ i<0. g i < g 0 by blast
  next
    fix j::nat
    assume d1: ∀ i<j. g i < g j
    show ∀ i<Suc j. g i < g (Suc j)
    proof (intro allI impI)
      fix i::nat
      assume i < Suc j
      then have i ≤ j by force
      moreover have g j < g (Suc j) using b1 b3 by simp
      moreover then have i < j ⟶ g i < g (Suc j) using d1 by force
      ultimately show g i < g (Suc j) by force
    qed
  qed
qed
moreover have b7: ∀ j::nat. ∀ i::nat. j ≤ i ⟶ g j ≤ g i
proof (intro allI impI)
  fix j::nat and i::nat
  assume j ≤ i
  moreover have j < i ⟶ g j ≤ g i using b6 by force
  moreover have j = i ⟶ g j ≤ g i by blast
  ultimately show g j ≤ g i by force
qed
moreover have b8: ∀ j::nat. ∀ i::nat. g i < g j ⟶ i < j
proof (intro allI impI)
  fix j::nat and i::nat
  assume g i < g j
  moreover have j ≤ i ⟶ g j ≤ g i using b7 by blast
  ultimately show i < j by simp
qed
moreover have b9: ∀ i::nat. yi (g i) = xi i
proof
  fix i::nat
  obtain p where p = (i, 0::nat) by blast
   then have ((g i) = (g (fst p)) + (snd p)) ∧ ((snd p) < (f (fst p))) using b1
by force
   then obtain n k where c1: (g i) = (g n) + k ∧ k < (f n) ∧ yi (g i) = (yin
n) k
    using b4 by (smt someI-ex)
```

then have $g\ n \le g\ i$ **by** *simp*
moreover have $g\ n < g\ i \longrightarrow False$
proof
assume $g\ n < g\ i$
then have $n < i$ **using** *b8* **by** *blast*
then have $g\ (Suc\ n) \le g\ i$ **using** *b7* **by** *simp*
then show *False* **using** *c1 b3 b6* **by** *force*
qed
ultimately have $g\ i = g\ n$ **by** *force*
then have $\neg\ i < n \wedge \neg\ n < i$ **using** *b6* **by** *force*
then have $i = n \wedge k = 0$ **using** *c1* **by** *force*
then have $yi\ (g\ i) = (yin\ i)\ 0$ **using** *c1* **by** *blast*
moreover have $(yin\ i)\ 0 = xi\ i$ **using** *b2* **by** *blast*
ultimately show $yi\ (g\ i) = xi\ i$ **by** *simp*
qed
moreover have $\forall\ i::nat.\ inj\text{-}on\ yi\ \{\ k.\ g\ i \le k \wedge k \le g\ (Suc\ i)\ \}$
proof
fix *i*
have *c1*: $inj\text{-}on\ (yin\ i)\ \{k.\ k \le f\ i\}$ **using** *b2* **by** *blast*
have $\forall\ k1\ k2.\ g\ i \le k1 \wedge k1 \le g\ (Suc\ i) \longrightarrow g\ i \le k2 \wedge k2 \le g\ (Suc\ i) \longrightarrow yi\ k1 = yi\ k2 \longrightarrow k1 = k2$
proof (*intro allI impI*)
fix *k1 k2*
assume *d1*: $g\ i \le k1 \wedge k1 \le g\ (Suc\ i)$
and *d2*: $g\ i \le k2 \wedge k2 \le g\ (Suc\ i)$ **and** *d3*: $yi\ k1 = yi\ k2$
have $g\ i \le k1 \wedge k1 \le g\ i + f\ i$ **using** *d1 b3* **by** *simp*
then have $\exists\ t.\ k1 = g\ i + t \wedge t \le f\ i$ **by** *presburger*
then obtain *t1* **where** *d4*: $k1 = g\ i + t1 \wedge t1 \le f\ i$ **by** *blast*
have $g\ i \le k2 \wedge k2 \le g\ i + f\ i$ **using** *d2 b3* **by** *simp*
then have $\exists\ t.\ k2 = g\ i + t \wedge t \le f\ i$ **by** *presburger*
then obtain *t2* **where** *d5*: $k2 = g\ i + t2 \wedge t2 \le f\ i$ **by** *blast*
have $t1 < f\ i \wedge t2 < f\ i \longrightarrow k1 = k2$
proof
assume $t1 < f\ i \wedge t2 < f\ i$
then have $yi\ k1 = yin\ i\ t1 \wedge yi\ k2 = yin\ i\ t2$ **using** *d4 d5 b5* **by** *blast*
then have $yin\ i\ t1 = yin\ i\ t2$ **using** *d3* **by** *metis*
then show $k1 = k2$ **using** *c1 d4 d5* **unfolding** *inj-on-def* **by** *blast*
qed
moreover have $t1 = f\ i \wedge t2 < f\ i \longrightarrow False$
proof
assume *e1*: $t1 = f\ i \wedge t2 < f\ i$
then have *e2*: $yi\ k2 = yin\ i\ t2$ **using** *d4 d5 b5* **by** *blast*
have *e3*: $k1 = g\ (Suc\ i)$ **using** *e1 d4 b3* **by** *simp*
then have $yi\ k1 = yin\ (Suc\ i)\ 0$ **using** *b1 b5*[*of k1 Suc i 0*] **by** *simp*
moreover have $yi\ k1 = yin\ i\ (f\ i)$ **using** *e3 b9 b2* **by** *simp*
ultimately have $yin\ i\ t2 = yin\ i\ (f\ i)$ **using** *e2 d3* **by** *metis*
then have $t2 = f\ i$ **using** *c1 d5* **unfolding** *inj-on-def* **by** *blast*
then show *False* **using** *e1* **by** *force*
qed

moreover have *t1* $<$ *f i* $\wedge$ *t2* $=$ *f i* $\longrightarrow$ *False*
proof
assume *e1*: *t1* $<$ *f i* $\wedge$ *t2* $=$ *f i*
then have *e2*: *yi k1* $=$ *yin i t1* **using** *d4 d5 b5* **by** *blast*
have *e3*: *k2* $=$ *g* (*Suc i*) **using** *e1 d5 b3* **by** *simp*
then have *yi k2* $=$ *yin* (*Suc i*) *0* **using** *b1 b5*[*of k2 Suc i 0*] **by** *simp*
moreover have *yi k2* $=$ *yin i* (*f i*) **using** *e3 b9 b2* **by** *simp*
ultimately have *yin i t1* $=$ *yin i* (*f i*) **using** *e2 d3* **by** *metis*
then have *t1* $=$ *f i* **using** *c1 d4* **unfolding** *inj-on-def* **by** *blast*
then show *False* **using** *e1* **by** *force*
qed
ultimately show *k1* $=$ *k2* **using** *d4 d5* **by** *force*
qed
then show *inj-on yi* { *k.* *g i* $\leq$ *k* $\wedge$ *k* $\leq$ *g* (*Suc i*) } **unfolding** *inj-on-def* **by** *blast*
qed
moreover have $\forall$ *m.* $\exists$ *n.* *g n* $\leq$ *m* $\wedge$ *Suc m* $\leq$ *g* (*Suc n*)
proof
fix *m*
obtain *n k* **where** *m* $=$ *g n* $+$ *k* $\wedge$ *k* $<$ *f n* **using** *b1 b3 lem-sum-ind-ex*[*of g f m*] **by** *blast*
then have *g n* $\leq$ *m* $\wedge$ *Suc m* $\leq$ *g* (*Suc n*) **using** *b3* **by** *simp*
then show $\exists$ *n.* *g n* $\leq$ *m* $\wedge$ *Suc m* $\leq$ *g* (*Suc n*) **by** *blast*
qed
moreover have $\forall$ *k i i'*. *g i* $\leq$ *k* $\wedge$ *Suc k* $\leq$ *g* (*Suc i*) $\wedge$ *g i'* $\leq$ *k* $\wedge$ *Suc k* $\leq$ *g* (*Suc i'*) $\longrightarrow$ *i* $=$ *i'*
proof (*intro allI impI*)
fix *k i i'*
assume *g i* $\leq$ *k* $\wedge$ *Suc k* $\leq$ *g* (*Suc i*) $\wedge$ *g i'* $\leq$ *k* $\wedge$ *Suc k* $\leq$ *g* (*Suc i'*)
moreover then have *k* $<$ *g i* $+$ *f i* $\wedge$ *k* $<$ *g i'* $+$ *f i'* **using** *b3* **by** *simp*
ultimately have $\exists$ *l1*. *k* $=$ *g i* $+$ *l1* $\wedge$ *l1* $<$ *f i* **and** $\exists$ *l2*. *k* $=$ *g i'* $+$ *l2* $\wedge$ *l2* $<$ *f i'* **by** *presburger+*
then obtain *l1 l2* **where** *k* $=$ *g i* $+$ *l1* $\wedge$ *l1* $<$ *f i* **and** *k* $=$ *g i'* $+$ *l2* $\wedge$ *l2* $<$ *f i'* **by** *blast*
then show *i* $=$ *i'* **using** *b1 b3 lem-sum-ind-un*[*of g f k i l1 i' l2*] **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-sv-un3*:
fixes *r1 r2 r3*::*'U rel*
assumes *single-valued* (*r1* $\cup$ *r3*) **and** *single-valued* (*r2* $\cup$ *r3*) **and** *Field r1* $\cap$ *Field r2* $=$ {}
shows *single-valued* (*r1* $\cup$ *r2* $\cup$ *r3*)
using *assms* **unfolding** *single-valued-def Field-def* **by** *blast*

lemma *lem-cfcomp-d2uset*:
fixes κ::*'U rel* **and** *r*::*'U rel* **and** *W*::*'U rel* $\Rightarrow$ *'U set* **and** *R*::*'U rel* $\Rightarrow$ *'U rel*
and *S*::*'U rel set*

assumes $a1$: $\kappa =o$ *cardSuc* $|UNIV::nat\ set|$
and $a3$: $T = \{\ t::'U\ rel.\ t \neq \{\} \wedge CCR\ t \wedge single\text{-}valued\ t \wedge acyclic\ t \wedge (\forall x{\in}Field\ t.\ t``\{x\} \neq \{\})\ \}$
and $a4$: *Refl r*

and $a5$: $S \subseteq \{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}$
and $a6$: $|\{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}| \leq o\ |S|$
and $a7$: $\forall\ \alpha \in S.\ \exists\ \beta \in S.\ \alpha <o\ \beta$

and $a8$: $Field\ r = (\bigcup\alpha{\in}S.\ W\ \alpha)$ **and** $a9$: $\forall\alpha{\in}S.\ \forall\ \beta{\in}S.\ \alpha \neq \beta \longrightarrow W\ \alpha \cap W\ \beta = \{\}$
and $a10$: $\bigwedge\ \alpha.\ \alpha \in S \Longrightarrow R\ \alpha \in T \wedge R\ \alpha \subseteq r \wedge |W\ \alpha| \leq o\ |UNIV::nat\ set|$
$\wedge\ Field\ (R\ \alpha) = W\ \alpha \wedge \neg\ Conelike\ (Restr\ r\ (W\ \alpha))$
and $a11$: $\bigwedge\ \alpha\ x.\ \alpha \in S \Longrightarrow x \in W\ \alpha \Longrightarrow \exists\ a.$
$((x,a) \in (Restr\ r\ (W\ \alpha))\hat{}* \wedge (\forall\ \beta \in S.\ \alpha <o\ \beta \longrightarrow (r``\{a\} \cap W\ \beta) \neq \{\}))$
shows $\exists\ r'.\ CCR\ r' \wedge DCR\ 2\ r' \wedge r' \subseteq r \wedge (\forall\ a \in Field\ r.\ \exists\ b \in Field\ r'.\ (a,b) \in r\hat{}*)$
proof –
obtain $l :: 'U \Rightarrow 'U\ rel$ **where** $q1$: $l = (\lambda\ a.\ SOME\ \alpha.\ \alpha \in S \wedge a \in W\ \alpha)$ **by** *blast*
have $q2$: $\bigwedge\ a.\ a \in Field\ r \Longrightarrow l\ a \in S \wedge a \in W\ (l\ a)$
proof –
fix a
assume $a \in Field\ r$
then obtain α **where** $\alpha \in S \wedge a \in W\ \alpha$ **using** *q1 a8* **by** *blast*
then show $l\ a \in S \wedge a \in W\ (l\ a)$ **using** *q1 someI-ex*[*of* $\lambda\ \alpha.\ \alpha \in S \wedge a \in W\ \alpha$] **by** *metis*
qed
have $q3$: $\bigwedge\ \alpha\ a.\ \alpha \in S \Longrightarrow a \in W\ \alpha \Longrightarrow l\ a = \alpha$
proof –
fix $\alpha\ a$
assume $\alpha \in S$ **and** $a \in W\ \alpha$
moreover then have $a \in W\ (l\ a) \wedge \alpha \in S \wedge l\ a \in S$ **using** *q2 a8 a10* **by** *fast*
ultimately show $l\ a = \alpha$ **using** *a9* **by** *blast*
qed
have $b1$: $\bigwedge\ \alpha.\ \alpha \in S \Longrightarrow (R\ \alpha) \in T$ **using** *a3 a10* **by** *blast*
have $b4$: $\bigwedge\ \alpha.\ \alpha \in S \Longrightarrow (R\ \alpha) \subseteq r$ **using** *a10* **by** *blast*
have $b7$: $\forall\ \alpha \in S.\ \forall\ \beta \in S.\ \exists\ \gamma{\in}S.\ (\alpha <o\ \gamma \vee \alpha = \gamma) \wedge (\beta <o\ \gamma \vee \beta = \gamma)$
proof (*intro ballI*)
fix $\alpha\ \beta$
assume $\alpha \in S$ **and** $\beta \in S$
then have *Well-order* α $\wedge$ *Well-order* β **and** $\alpha \in S \wedge \beta \in S$
using *a5* **unfolding** *ordLess-def* **by** *blast+*
moreover then have $\alpha <o\ \beta \vee \beta <o\ \alpha \vee \alpha =o\ \beta$
using *ordLeq-iff-ordLess-or-ordIso ordLess-or-ordLeq* **by** *blast*
ultimately show $\exists\ \gamma \in S.\ (\alpha <o\ \gamma \vee \alpha = \gamma) \wedge (\beta <o\ \gamma \vee \beta = \gamma)$
using *a3 a5 lem-Oeq*[*of* $\alpha\ \beta$] **by** *blast*
qed

obtain s :: $'U$ $rel \Rightarrow nat \Rightarrow 'U$ **where** $b8$: $s = (\lambda\ \alpha.\ SOME\ xi.\ cfseq\ (R\ \alpha)\ xi)$ **by** *blast*
moreover have $\forall\ \alpha \in S.\ \exists\ xi.\ cfseq\ (R\ \alpha)\ xi$ **using** *b1 a3 lem-ccrsv-cfseq* **by** *blast*
ultimately have $b9$: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow cfseq\ (R\ \alpha)\ (s\ \alpha)$ **by** (*metis someI-ex*)
obtain *en* **where** *b-en*: $en = (\lambda\ \alpha.\ SOME\ g :: nat \Rightarrow 'U.\ W\ \alpha \subseteq g`UNIV)$ **by** *blast*
obtain *ta* :: $'U \Rightarrow 'U\ rel \Rightarrow 'U$
where $b10$: $ta = (\lambda\ u\ \alpha'.\ SOME\ u'.\ (u,u') \in r \wedge u' \in W\ \alpha')$ **by** *blast*
obtain t :: $('U\ rel) \times 'U \Rightarrow 'U\ rel \Rightarrow 'U$
where $b11$: $t = (\lambda\ (\alpha,a)\ \alpha'.\ ta\ a\ \alpha')$ **by** *blast*
obtain *tm* :: $('U\ rel) \times nat \Rightarrow 'U\ rel \Rightarrow 'U$
where $b12$: $tm = (\lambda\ (\alpha,k)\ \alpha'.\ t\ (\alpha,(en\ \alpha\ k))\ \alpha')$ **by** *blast*
obtain *jnN* :: $'U \Rightarrow 'U \Rightarrow 'U$
where $b13$: $jnN = (\lambda\ u\ u'.\ SOME\ v.\ (u,v) \in (R\ (l\ u))\hat{}* \wedge (u',v) \in (R\ (l\ u))\hat{}*)$ **by** *blast*
obtain h **where** $b20$: $\bigwedge \alpha\ k1\ \beta\ k2.\ \alpha \in S \wedge \beta \in S \Longrightarrow$
$(\exists\ \gamma \in S.\ \alpha <o\ \gamma \wedge \beta <o\ \gamma \wedge h\ \gamma = jnN\ (tm\ (\alpha,k1)\ \gamma)\ (tm\ (\beta,k2)\ \gamma)\ )$
using *a1 a5 a6 a7 lem-jnfix-cardsuc*[*of UNIV*::*nat set* κ *S jnN tm*] **by** *blast*
define *EP* **where** $EP = (\lambda\ \alpha.\ \{\ a \in W\ \alpha.\ \forall\ \beta \in S.\ \alpha <o\ \beta \longrightarrow (r``\{a\} \cap W\ \beta) \neq \{\}\ \})$
have $b24$: $\bigwedge \alpha\ k\ b.\ \alpha \in S \Longrightarrow (s\ \alpha\ k,\ b) \in (R\ \alpha)\hat{}* \Longrightarrow (\exists\ k' \geq k.\ b = s\ \alpha\ k')$
proof –
fix $\alpha\ k\ b$
assume $c1$: $\alpha \in S$ **and** $c2$: $(s\ \alpha\ k,\ b) \in (R\ \alpha)\hat{}*$
moreover then have *single-valued* $(R\ \alpha)$ **using** *b1 a3* **by** *blast*
moreover have $\forall i.\ (s\ \alpha\ i,\ s\ \alpha\ (Suc\ i)) \in R\ \alpha$ **using** *c1 b9* **unfolding** *cfseq-def* **by** *blast*
ultimately show $\exists\ k' \geq k.\ b = s\ \alpha\ k'$
using *lem-rseq-svacyc-inv-rtr*[*of* $R\ \alpha\ s\ \alpha\ k\ b$] **by** *blast*
qed
have $b25$: $\bigwedge \alpha\ k\ b.\ \alpha \in S \Longrightarrow (s\ \alpha\ k,\ b) \in (R\ \alpha)\hat{}+ \Longrightarrow (\exists\ k' > k.\ b = s\ \alpha\ k')$
proof –
fix $\alpha\ k\ b$
assume $c1$: $\alpha \in S$ **and** $c2$: $(s\ \alpha\ k,\ b) \in (R\ \alpha)\hat{}+$
moreover then have *single-valued* $(R\ \alpha)$ **using** *b1 a3* **by** *blast*
moreover have $\forall i.\ (s\ \alpha\ i,\ s\ \alpha\ (Suc\ i)) \in R\ \alpha$ **using** *c1 b9* **unfolding** *cfseq-def* **by** *blast*
ultimately show $\exists\ k' > k.\ b = s\ \alpha\ k'$ **using** *lem-rseq-svacyc-inv-tr*[*of* $R\ \alpha\ s\ \alpha\ k\ b$] **by** *blast*
qed
have $b26$: $\bigwedge \alpha\ a\ b\ c.\ \alpha \in S \Longrightarrow a \in W\ \alpha \Longrightarrow b \in W\ \alpha \Longrightarrow$
$c = jnN\ a\ b \Longrightarrow c \in W\ \alpha \wedge (a,\ c) \in (R\ \alpha)\hat{}* \wedge (b,\ c) \in (R\ \alpha)\hat{}*$
proof –
fix $\alpha\ a\ b\ c$
assume $c1$: $\alpha \in S$ **and** $c2$: $a \in W\ \alpha$ **and** $c3$: $b \in W\ \alpha$ **and** $c4$: $c = jnN\ a\ b$
then have $CCR\ (R\ \alpha) \wedge a \in Field\ (R\ \alpha) \wedge b \in Field\ (R\ \alpha)$ **using** *c1 b1 a3 a10* **by** *blast*
then have $\exists\ c'.\ (a,\ c') \in (R\ \alpha)\hat{}* \wedge (b,\ c') \in (R\ \alpha)\hat{}*$ **unfolding** *CCR-def*

by *blast*
  moreover have $l\ a = \alpha$ **using** *c1 c2 q3* **by** *blast*
  moreover then have $c = (SOME\ c'.\ (a,\ c') \in (R\ \alpha)\hat{}* \wedge (b,\ c') \in (R\ \alpha)\hat{}*)$ **using** *c4 b13* **by** *simp*
  ultimately have *c5*: $(a,\ c) \in (R\ \alpha)\hat{}* \wedge (b,\ c) \in (R\ \alpha)\hat{}*$
    using *someI-ex*[*of* $\lambda\ c'.\ (a,\ c') \in (R\ \alpha)\hat{}* \wedge (b,\ c') \in (R\ \alpha)\hat{}*$] **by** *force*
  moreover have $W\ \alpha \in Inv\ (R\ \alpha)$ **using** *c1 a10*[*of* α] **unfolding** *Field-def Inv-def* **by** *blast*
  moreover then have $c \in W\ \alpha$ **using** *c2 c5 lem-Inv-restr-rtr2*[*of* $W\ \alpha\ R\ \alpha$] **by** *blast*
  ultimately show $c \in W\ \alpha \wedge (a,\ c) \in (R\ \alpha)\hat{}* \wedge (b,\ c) \in (R\ \alpha)\hat{}*$ **by** *blast*
 qed
 have *b-enr*: $\bigwedge\ \alpha.\ \alpha \in S \Longrightarrow W\ \alpha \subseteq (en\ \alpha)\text{`}(UNIV::nat\ set)$
 proof –
  fix α
  assume $\alpha \in S$
  then have $|W\ \alpha| \leq o\ |UNIV::nat\ set|$ **using** *a10* **by** *blast*
  then obtain $g::nat \Rightarrow {'}U$ **where** $W\ \alpha \subseteq g\text{`}UNIV$
    by (*metis card-of-ordLeq2 empty-subsetI order-refl*)
  then show $W\ \alpha \subseteq (en\ \alpha)\text{`}UNIV$ **unfolding** *b-en* **using** *someI-ex* **by** *metis*
 qed
 have *b-h*: $\bigwedge\ \alpha\ a\ \beta\ b.\ \alpha \in S \wedge \beta \in S \Longrightarrow a \in EP\ \alpha \wedge b \in EP\ \beta \Longrightarrow$
   $(\exists\ \gamma \in S.\ \exists\ a' \in W\ \gamma.\ \exists\ b' \in W\ \gamma.\ \alpha <o\ \gamma \wedge \beta <o\ \gamma$
   $\wedge\ (a,a') \in r \wedge (a',\ h\ \gamma) \in (R\ \gamma)\hat{}* \wedge (b,b') \in r \wedge (b',\ h\ \gamma) \in (R\ \gamma)\hat{}*)$
 proof –
  fix $\alpha\ a\ \beta\ b$
  assume *c1*: $\alpha \in S \wedge \beta \in S$ **and** *c2*: $a \in EP\ \alpha \wedge b \in EP\ \beta$
  then have $a \in W\ \alpha \wedge b \in W\ \beta$ **unfolding** *EP-def* **by** *blast*
  moreover then obtain *k1 k2* **where** *c3*: $a = en\ \alpha\ k1 \wedge b = en\ \beta\ k2$ **using** *c1 b-enr* **by** *blast*
  ultimately obtain γ **where** *c4*: $\gamma \in S \wedge \alpha <o\ \gamma \wedge \beta <o\ \gamma$
   and *c5*: $h\ \gamma = jnN\ (tm\ (\alpha,k1)\ \gamma)\ (tm\ (\beta,k2)\ \gamma)$ **using** *c1 b20* **by** *blast*
  have $ta\ a\ \gamma = (SOME\ a'.\ (a,\ a') \in r \wedge a' \in W\ \gamma)$ **using** *b10* **by** *simp*
  moreover have $\exists\ x.\ (a,\ x) \in r \wedge x \in W\ \gamma$ **using** *c2 c4* **unfolding** *EP-def* **by** *blast*
  ultimately have *c6*: $(a,\ ta\ a\ \gamma) \in r \wedge ta\ a\ \gamma \in W\ \gamma$
    using *someI-ex*[*of* $\lambda\ a'.\ (a,\ a') \in r \wedge a' \in W\ \gamma$] **by** *metis*
  have $ta\ b\ \gamma = (SOME\ a'.\ (b,\ a') \in r \wedge a' \in W\ \gamma)$ **using** *b10* **by** *simp*
  moreover have $\exists\ x.\ (b,\ x) \in r \wedge x \in W\ \gamma$ **using** *c2 c4* **unfolding** *EP-def* **by** *blast*
  ultimately have *c7*: $(b,\ ta\ b\ \gamma) \in r \wedge ta\ b\ \gamma \in W\ \gamma$
    using *someI-ex*[*of* $\lambda\ a'.\ (b,\ a') \in r \wedge a' \in W\ \gamma$] **by** *metis*
  have $h\ \gamma = jnN\ (ta\ a\ \gamma)\ (ta\ b\ \gamma)$ **using** *c3 c5 b11 b12* **by** *simp*
  moreover have $ta\ a\ \gamma \in W\ \gamma \wedge ta\ b\ \gamma \in W\ \gamma$ **using** *c6 c7* **by** *blast*
  ultimately have $h\ \gamma \in W\ \gamma \wedge (ta\ a\ \gamma,\ h\ \gamma) \in (R\ \gamma)\hat{}* \wedge (ta\ b\ \gamma,\ h\ \gamma) \in (R\ \gamma)\hat{}*$
    using *c4 b26*[*of* $\gamma\ ta\ a\ \gamma\ ta\ b\ \gamma\ h\ \gamma$] **by** *blast*
  then show $\exists\ \gamma \in S.\ \exists\ a' \in W\ \gamma.\ \exists\ b' \in W\ \gamma.\ \alpha <o\ \gamma \wedge \beta <o\ \gamma$

$\wedge$ $(a,a') \in r \wedge (a', h\ \gamma) \in (R\ \gamma)\hat{}* \wedge (b,b') \in r \wedge (b', h\ \gamma) \in (R\ \gamma)\hat{}*$
using *c4 c6 c7* **by** *blast*
qed
have *p1*: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow R\ \alpha \subseteq$ *Restr* $r\ (W\ \alpha)$ **using** *a10* **unfolding** *Field-def* **by** *fastforce*
have *p2*: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow$ *Field* (*Restr* $r\ (W\ \alpha)) = W\ \alpha$
proof −
fix α
assume $\alpha \in S$
then have $W\ \alpha \subseteq$ *Field* r **using** *a10* **unfolding** *Field-def* **by** *blast*
moreover have *SF* $r = \{A.\ A \subseteq$ *Field* $r\}$ **using** *a4* **unfolding** *SF-def refl-on-def Field-def* **by** *fast*
ultimately have $W\ \alpha \in$ *SF* r **by** *blast*
then show *Field* (*Restr* $r\ (W\ \alpha)) = W\ \alpha$ **unfolding** *SF-def* **by** *blast*
qed
have *p3*: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow \forall n.\ \exists k \geq n.\ (s\ \alpha\ (Suc\ k),\ s\ \alpha\ k) \notin$ (*Restr* $r\ (W\ \alpha))\hat{}*$
proof −
fix α
assume *c1*: $\alpha \in S$
have $\forall a \in$*Field* (*Restr* $r\ (W\ \alpha)).\ \exists i.\ (a,\ s\ \alpha\ i) \in$ (*Restr* $r\ (W\ \alpha))\hat{}*$
proof
fix a
assume $a \in$ *Field* (*Restr* $r\ (W\ \alpha)$)
then have $a \in$ *Field* $(R\ \alpha)$ **using** *c1 a10*[*of* α] **unfolding** *Field-def* **by** *blast*
then obtain i **where** $(a,\ s\ \alpha\ i) \in (R\ \alpha)\hat{}*$ **using** *c1 b9*[*of* α] **unfolding** *cfseq-def* **by** *blast*
moreover have $R\ \alpha \subseteq$ *Restr* $r\ (W\ \alpha)$ **using** *c1 p1* **by** *blast*
ultimately show $\exists i.\ (a,\ s\ \alpha\ i) \in$ (*Restr* $r\ (W\ \alpha))\hat{}*$ **using** *rtrancl-mono* **by** *blast*
qed
moreover have $\forall i.\ (s\ \alpha\ i,\ s\ \alpha\ (Suc\ i)) \in$ *Restr* $r\ (W\ \alpha)$
using *c1 p1 b9*[*of* α] **unfolding** *cfseq-def* **using** *rtrancl-mono* **by** *blast*
ultimately have *cfseq* (*Restr* $r\ (W\ \alpha)$) $(s\ \alpha)$ **unfolding** *cfseq-def* **by** *blast*
then show $\forall n.\ \exists k \geq n.\ (s\ \alpha\ (Suc\ k),\ s\ \alpha\ k) \notin$ (*Restr* $r\ (W\ \alpha))\hat{}*$
using *c1 a10*[*of* α] *lem-cfseq-ncl*[*of Restr* $r\ (W\ \alpha)\ s\ \alpha$] **by** *blast*
qed
obtain E **where** *b27*: $E = (\lambda\ \alpha.\ \{\ k.\ (s\ \alpha\ (Suc\ k),\ s\ \alpha\ k) \notin$ (*Restr* $r\ (W\ \alpha))\hat{}*\ \})$ **by** *blast*
obtain P **where** *b28*: $P = (\lambda\ \alpha.\ (s\ \alpha)`(E\ \alpha)\ )$ **by** *blast*
obtain K **where** *b29*: $K = (\lambda\ \alpha.\ \{\ a \in W\ \alpha.\ (h\ \alpha \in W\ \alpha \longrightarrow (h\ \alpha,\ a) \in (R\ \alpha)\hat{}*)$
$\wedge\ (a,\ h\ \alpha) \notin (R\ \alpha)\hat{}*\ \})$ **by** *blast*
let *?F* $= \lambda\ \alpha.\ P\ \alpha \cap K\ \alpha$
have *b31*: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow P\ \alpha \in$ *SCF* $(R\ \alpha)$
proof −
fix α
assume *c1*: $\alpha \in S$
then have $P\ \alpha \subseteq$ *Field* $(R\ \alpha)$ **using** *b9 b28 lem-cfseq-fld* **by** *blast*
moreover have $\forall\ a \in$ *Field* $(R\ \alpha).\ \exists\ b \in P\ \alpha.\ (a,\ b) \in (R\ \alpha)\hat{}*$

proof
fix *a*
assume $a \in Field\ (R\ \alpha)$
then obtain *i* **where** *d1*: $(a,\ s\ \alpha\ i) \in (R\ \alpha)\hat{}*$ **using** *c1 b9*[*of* α] **unfolding** *cfseq-def* **by** *blast*
then obtain *k* **where** $i \le k \wedge (s\ \alpha\ (Suc\ k),\ s\ \alpha\ k) \notin (Restr\ r\ (W\ \alpha))\hat{}*$ **using** *c1 p3*[*of* α] **by** *blast*
moreover then have *d2*: $(s\ \alpha\ i,\ s\ \alpha\ k) \in (R\ \alpha)\hat{}*$
using *c1 b9*[*of* α] *lem-rseq-rtr* **unfolding** *cfseq-def* **by** *blast*
ultimately have $s\ \alpha\ k \in P\ \alpha$ **using** *b27 b28* **by** *blast*
moreover have $(a,\ s\ \alpha\ k) \in (R\ \alpha)\hat{}*$ **using** *d1 d2* **by** *simp*
ultimately show $\exists\ b \in P\ \alpha.\ (a,\ b) \in (R\ \alpha)\hat{}*$ **by** *blast*
qed
ultimately show $P\ \alpha \in SCF\ (R\ \alpha)$ **unfolding** *SCF-def* **by** *blast*
qed
have *b32*: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow K\ \alpha \in SCF\ (R\ \alpha) \cap Inv\ (R\ \alpha)$
proof
fix α
assume *c1*: $\alpha \in S$
have $\forall a \in Field\ (R\ \alpha).\ \exists b \in K\ \alpha.\ (a,\ b) \in (R\ \alpha)\hat{}*$
proof
fix *a*
assume *d1*: $a \in Field\ (R\ \alpha)$
show $\exists b \in K\ \alpha.\ (a,\ b) \in (R\ \alpha)\hat{}*$
proof (*cases* $h\ \alpha \in Field\ (R\ \alpha)$)
assume $h\ \alpha \in Field\ (R\ \alpha)$
moreover have $CCR\ (R\ \alpha)$ **using** *c1 b1 a3* **by** *blast*
ultimately obtain a' **where** $a' \in Field\ (R\ \alpha)$
and *e1*: $(a,a') \in (R\ \alpha)\hat{}* \wedge (h\ \alpha,\ a') \in (R\ \alpha)\hat{}*$
using *d1* **unfolding** *CCR-def* **by** *blast*
then obtain *b* **where** *e2*: $(a',\ b) \in (R\ \alpha)$ **using** *c1 b1 a3* **by** *blast*
then have $b \in Field\ (R\ \alpha)$ **unfolding** *Field-def* **by** *blast*
moreover have $(h\ \alpha,\ b) \in (R\ \alpha)\hat{}*$ **using** *e1 e2* **by** *force*
moreover have $(b,\ h\ \alpha) \in (R\ \alpha)\hat{}* \longrightarrow False$
proof
assume $(b,\ h\ \alpha) \in (R\ \alpha)\hat{}*$
then have $(b,\ b) \in (R\ \alpha)\hat{}+$ **using** *e1 e2* **by** *fastforce*
then show *False* **using** *c1 b1 a3* **unfolding** *acyclic-def* **by** *blast*
qed
moreover have $(a,\ b) \in (R\ \alpha)\hat{}*$ **using** *e1 e2* **by** *force*
ultimately show *?thesis* **using** *b29 c1 a10* **by** *blast*
next
assume $h\ \alpha \notin Field\ (R\ \alpha)$
then have $(a,\ h\ \alpha) \notin (R\ \alpha)\hat{}* \wedge h\ \alpha \notin W\ \alpha$ **using** *d1 c1 a10 lem-rtr-field*[*of* *a*] **by** *blast*
then have $a \in K\ \alpha$ **using** *d1 b29 c1 a10* **by** *blast*
then show *?thesis* **by** *blast*
qed
qed

then show *K* $\alpha \in$ *SCF* (*R* α) **using** *b29 c1 a10* **unfolding** *SCF-def* **by** *blast*
next
fix α
assume *c1*: $\alpha \in S$
have $\forall$ *a b*. $a \in K\ \alpha \wedge (a,b) \in (R\ \alpha) \longrightarrow b \in K\ \alpha$
proof (*intro allI impI*)
fix *a b*
assume *d1*: $a \in K\ \alpha \wedge (a,b) \in (R\ \alpha)$
then have *d3*: $a \in$ *Field* (*R* α) **and** *d4*: $(a,\ h\ \alpha) \notin (R\ \alpha)^*$ **using** *b29 c1 a10* **by** *blast+*
have $b \in$ *Field* (*R* α) **using** *d1* **unfolding** *Field-def* **by** *blast*
moreover have $h\ \alpha \in W\ \alpha \longrightarrow (h\ \alpha,\ b) \in (R\ \alpha)\widehat{\ }*$ **using** *d1 b29* **by** *force*
moreover have $(b,\ h\ \alpha) \in (R\ \alpha)\widehat{\ }* \longrightarrow$ *False*
proof
assume $(b,\ h\ \alpha) \in (R\ \alpha)\widehat{\ }*$
then have $(a,\ h\ \alpha) \in (R\ \alpha)\widehat{\ }*$ **using** *d1* **by** *force*
then show *False* **using** *d4* **by** *blast*
qed
ultimately show $b \in K\ \alpha$ **using** *b29 c1 a10* **by** *blast*
qed
then show *K* $\alpha \in$ *Inv* (*R* α) **using** *b29* **unfolding** *Inv-def* **by** *blast*
qed
have *b33*: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow$ *?F* $\alpha \in$ *SCF* (*R* α)
proof −
fix α
assume *c1*: $\alpha \in S$
have *K* $\alpha \in$ *SCF* (*R* α) $\cap$ *Inv* (*R* α) **using** *c1 b31 b32* **unfolding** *Inv-def* **by** *blast+*
moreover have *P* $\alpha \in$ *SCF* (*R* α) **using** *c1 b31 b32 lem-scfinv-scf-int* **by** *blast*
ultimately have $K\ \alpha \cap P\ \alpha \in$ *SCF* (*R* α) **using** *lem-scfinv-scf-int* **by** *blast*
moreover have *?F* $\alpha = K\ \alpha \cap P\ \alpha$ **by** *blast*
ultimately show *?F* $\alpha \in$ *SCF* (*R* α) **by** *metis*
qed
define *rei* **where** *rei* = ($\lambda\ \alpha$. *SOME k*. $k \in E\ \alpha \wedge (s\ \alpha\ k) \in$ *?F* α)
define *re0* **where** *re0* = ($\lambda\ \alpha$. *s* α (*rei* α))
define *re1* **where** *re1* = ($\lambda\ \alpha$. *s* α (*Suc* (*rei* α)))
define *ep* **where** *ep* = ($\lambda\ \alpha$. *SOME b*. (*re1* α, *b*) $\in$ (*Restr r* (*W* α))$\widehat{\ }* \wedge b \in$ *EP* α)
define *spl* **where** *spl* = ($\lambda\ \alpha$. *spthlen* (*Restr r* (*W* α)) (*re1* α) (*ep* α))
define *sp* **where** *sp* = ($\lambda\ \alpha$. *SOME f*. $f \in$ *spth* (*Restr r* (*W* α)) (*re1* α) (*ep* α))
define *R0* **where** *R0* = ($\lambda\ \alpha$. { $(a,b) \in R\ \alpha$. (*b*, *re0* α) $\in (R\ \alpha)\widehat{\ }*$ })
define *R2* **where** *R2* = ($\lambda\ \alpha$. { (*a*,*b*). $\exists\ k <$ (*spl* α). $a = sp\ \alpha\ k \wedge b = sp\ \alpha$ (*Suc k*) })
define *R′* **where** *R′* = ($\lambda\ \alpha$. *R0* $\alpha \cup$ *R2* $\alpha \cup$ { (*re0* α, *re1* α) })
define *re′* **where** *re′* = ({ $(a,b) \in r$. $\exists\ \alpha \in S$. $\exists\ \beta \in S$. $\alpha <o\ \beta \wedge a = ep\ \alpha \wedge b \in W\ \beta \wedge (b,\ h\ \beta) \in (R\ \beta)\widehat{\ }*$ })
define *r′* **where** *r′* = (*re′* $\cup$ ($\bigcup\alpha{\in}S$. *R′* α))

have *b-Fne*: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow$ *?F* $\alpha \neq \{\}$

proof −
fix α
assume $\alpha \in S$
then have *?F* $\alpha \in$ *SCF* $(R\ \alpha) \wedge R\ \alpha \neq \{\}$ **using** *b33 a3 a10* **by** *blast*
then show *?F* $\alpha \neq \{\}$ **unfolding** *SCF-def Field-def* **by** *force*
qed
have *b-re0*: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow$ *re0* $\alpha \in$ *?F* $\alpha \wedge$ *rei* $\alpha \in E\ \alpha$
proof −
fix α
assume $\alpha \in S$
then obtain k **where** $k \in E\ \alpha \wedge (s\ \alpha\ k) \in$ *?F* α **using** *b-Fne b28* **by** *force*
then have $(s\ \alpha\ (\mathit{rei}\ \alpha)) \in$ *?F* α **and** *rei* $\alpha \in E\ \alpha$
using *someI-ex*[*of* $\lambda\ k.\ k \in E\ \alpha \wedge s\ \alpha\ k \in P\ \alpha \cap K\ \alpha$] **unfolding** *rei-def* **by** *metis+*
then show *re0* $\alpha \in$ *?F* $\alpha \wedge$ *rei* $\alpha \in E\ \alpha$ **unfolding** *re0-def* **by** *blast*
qed
have *b-rs*: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow s\ \alpha$ ‘ *UNIV* $\subseteq W\ \alpha$
proof −
fix α
assume $\alpha \in S$
then have *cfseq* $(R\ \alpha)\ (s\ \alpha) \wedge$ *Field* $(R\ \alpha) = W\ \alpha$ **using** *b9 a3 a10* **by** *blast*
then show $s\ \alpha$ ‘ *UNIV* $\subseteq W\ \alpha$ **using** *lem-rseq-rtr* **unfolding** *cfseq-def* **by** *blast*
qed
have *b-injs*: $\bigwedge \alpha\ k1\ k2.\ \alpha \in S \Longrightarrow s\ \alpha\ k1 = s\ \alpha\ k2 \Longrightarrow k1 = k2$
proof −
fix α *k1 k2*
assume $\alpha \in S$ **and** $s\ \alpha\ k1 = s\ \alpha\ k2$
moreover then have *cfseq* $(R\ \alpha)\ (s\ \alpha) \wedge$ *acyclic* $(R\ \alpha)$ **using** *b9 a3 a10* **by** *blast*
moreover then have *inj* $(s\ \alpha)$ **using** *lem-cfseq-inj* **by** *blast*
ultimately show $k1 = k2$ **unfolding** *inj-on-def* **by** *blast*
qed
have *b-re1*: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow$ *re1* $\alpha = s\ \alpha\ (\mathit{Suc}\ (\mathit{rei}\ \alpha))$
proof −
fix α
assume *c1*: $\alpha \in S$
then have *re0* $\alpha \in$ *?F* α **using** *b-re0*[*of* α] **by** *blast*
then obtain k **where** *c2*: *re0* $\alpha = s\ \alpha\ k \wedge k \in E\ \alpha$ **unfolding** *b28* **by** *blast*
then have $(s\ \alpha\ (\mathit{Suc}\ k),\ s\ \alpha\ k) \notin (\mathit{Restr}\ r\ (W\ \alpha))\hat{}*$ **unfolding** *b27* **by** *blast*
have *rei* $\alpha = k$ **using** *c1 c2 b-injs* **unfolding** *re0-def* **by** *blast*
moreover have *re1* $\alpha = s\ \alpha\ (\mathit{Suc}\ (\mathit{rei}\ \alpha))$ **unfolding** *re1-def* **by** *blast*
ultimately show *re1* $\alpha = s\ \alpha\ (\mathit{Suc}\ (\mathit{rei}\ \alpha))$ **by** *blast*
qed
have *b-re12*: $\bigwedge \alpha.\ \alpha \in S \Longrightarrow (\mathit{re0}\ \alpha,\ \mathit{re1}\ \alpha) \in R\ \alpha \wedge (\mathit{re1}\ \alpha,\ \mathit{re0}\ \alpha) \notin (\mathit{Restr}\ r\ (W\ \alpha))\hat{}*$
proof −
fix α
assume *c1*: $\alpha \in S$

then have *re0* α = *s* α (*rei* α) **and** *re1* α = *s* α (*Suc* (*rei* α))
and *cfseq* (*R* α) (*s* α) **using** *b9 b-re1 re0-def* **by** *blast+*
then have (*re0* α, *re1* α) $\in$ *R* α **unfolding** *cfseq-def* **by** *simp*
moreover have (*re1* α, *re0* α) $\in$ (*Restr r* (*W* α))^* $\longrightarrow$ *False*
proof
assume (*re1* α, *re0* α) $\in$ (*Restr r* (*W* α))^*
then have (*s* α (*Suc* (*rei* α)), *s* α (*rei* α)) $\in$ (*Restr r* (*W* α))^*
using *c1 b-re1*[*of* α] **unfolding** *re0-def* **by** *metis*
moreover have (*s* α (*Suc* (*rei* α)), *s* α (*rei* α)) $\notin$ (*Restr r* (*W* α))^*
using *c1 b-re0*[*of* α] *b27* **by** *blast*
ultimately show *False* **by** *blast*
qed
ultimately show (*re0* α, *re1* α) $\in$ *R* α $\wedge$ (*re1* α, *re0* α) $\notin$ (*Restr r* (*W* α))^*
by *blast*
qed
have *b-rw*: $\bigwedge$ α *a b*. $\alpha \in S \Longrightarrow a \in W\ \alpha \Longrightarrow (a,b) \in$ (*Restr r* (*W* α))^* $\Longrightarrow b$
$\in W\ \alpha$
proof −
fix α *a b*
assume $\alpha \in S$ **and** $a \in W\ \alpha$ **and** $(a,b) \in$ (*Restr r* (*W* α))^*
then show $b \in W\ \alpha$ **using** *lem-Inv-restr-rtr2*[*of - Restr r* (*W* α)] **unfolding**
Inv-def **by** *blast*
qed
have *b-r0w*: $\bigwedge$ α *a b*. $\alpha \in S \Longrightarrow a \in W\ \alpha \Longrightarrow (a,b) \in (R\ \alpha)$^* $\Longrightarrow b \in W\ \alpha$
using *p1 b-rw rtrancl-mono* **by** *blast*
have *b-ep*: $\bigwedge$ α. $\alpha \in S \Longrightarrow$ (*re1* α, *ep* α) $\in$ (*Restr r* (*W* α))^* $\wedge$ *ep* $\alpha \in$ *EP* α
proof −
fix α
assume *c1*: $\alpha \in S$
moreover then have *c2*: *re1* $\alpha \in W\ \alpha$ **using** *b-rs*[*of* α] *b-re1*[*of* α] **by** *blast*
ultimately obtain *b*
where *c3*: (*re1* α, *b*) $\in$ (*Restr r* (*W* α))^* $\wedge$ ($\forall\beta\in S$. α <*o* $\beta \longrightarrow$ *r*‘‘{*b*} $\cap$
$W\ \beta \neq \{\}$)
using *a11*[*of* α *re1* α] **by** *blast*
then have $b \in W\ \alpha$ **using** *c1 c2 b-rw*[*of* α] **by** *blast*
moreover obtain *L* **where** *c4*: *L* = (λ *b*. (*re1* α, *b*) $\in$ (*Restr r* (*W* α))^* $\wedge$
b $\in$ *EP* α) **by** *blast*
ultimately have *L b* **and** *ep* α = (*SOME b. L b*) **using** *c3* **unfolding** *EP-def*
ep-def **by** *blast+*
then have *L* (*ep* α) **using** *someI-ex* **by** *metis*
then show (*re1* α, *ep* α) $\in$ (*Restr r* (*W* α))^* $\wedge$ *ep* $\alpha \in$ *EP* α **using** *c4* **by**
blast
qed
have *b-sp*: $\bigwedge$ α. $\alpha \in S \Longrightarrow$ *sp* $\alpha \in$ *spth* (*Restr r* (*W* α)) (*re1* α) (*ep* α)
proof −
fix α
assume $\alpha \in S$
then have (*re1* α, *ep* α) $\in$ (*Restr r* (*W* α))^* **using** *b-ep* **by** *blast*
then obtain *f* **where** *f* $\in$ *spth* (*Restr r* (*W* α)) (*re1* α) (*ep* α)

using *lem-spthlen-rtr lem-rtn-rpth-inj* **unfolding** *spth-def* **by** *metis*
then show *sp* $\alpha \in$ *spth* (*Restr r* (*W* α)) (*re1* α) (*ep* α)
unfolding *sp-def* **using** *someI-ex* **by** *metis*
qed
have *b-R0*: $\bigwedge$ α *a*. $\alpha \in S \Longrightarrow$ (*a*,*re0* α) $\in$ (*R* α)^* $\Longrightarrow$ (*a*,*re0* α) $\in$ (*R0* α)^*
proof −
fix α *a*
assume $\alpha \in S$ **and** (*a*,*re0* α) $\in$ (*R* α)^*
then obtain *g n* **where** *g* $\in$ *rpth* (*R* α) *a* (*re0* α) *n* **using** *lem-ccext-rtr-rpth*[*of a re0* α] **by** *blast*
then have *c1*: *g 0* = *a* $\wedge$ *g n* = *re0* α **and** *c2*: $\forall$ *i*<*n*. (*g i*, *g* (*Suc i*)) $\in$ *R* α **unfolding** *rpth-def* **by** *blast*+
then have $\forall$ *i*$\leq$*n*. (*g i*, *re0* α) $\in$ (*R* α)^* **using** *lem-rseq-tl* **by** *metis*
then have $\forall$ *i*<*n*. (*g i*, *g* (*Suc i*)) $\in$ *R0* α **using** *c2* **unfolding** *R0-def* **by** *simp*
then show (*a*, *re0* α) $\in$ (*R0* α)^*
using *c1 lem-ccext-rpth-rtr*[*of R0* α *a re0* α *n*] **unfolding** *rpth-def* **by** *blast*
qed
have *b-hr0*: $\bigwedge$ α. $\alpha \in S \Longrightarrow$ *h* $\alpha \in$ *W* $\alpha \Longrightarrow$ (*h* α, *re0* α) $\in$ (*R0* α)^*
using *b-re0 b-R0 b29* **by** *blast*
have *b-hf*: $\bigwedge$ α. $\alpha \in S \Longrightarrow$ *h* $\alpha \in$ *W* $\alpha \Longrightarrow$ *h* $\alpha \in$ *Field r'*
proof −
fix α
assume *c1*: $\alpha \in S$ **and** *h* $\alpha \in$ *W* α
then have (*h* α, *re0* α) $\in$ (*R0* α)^* **using** *c1 b-hr0* **by** *blast*
moreover have *R0* $\alpha \subseteq$ *R'* α **using** *c1* **unfolding** *R'-def* **by** *blast*
ultimately have (*h* α, *re0* α) $\in$ (*R'* α)^* **using** *rtrancl-mono* **by** *blast*
moreover have *re0* $\alpha \in$ *Field* (*R'* α) **unfolding** *R'-def Field-def* **by** *blast*
ultimately have *h* $\alpha \in$ *Field* (*R'* α) **using** *lem-rtr-field*[*of h* α *re0* α] **by** *force*
moreover have *R'* $\alpha \subseteq$ *r'* **using** *c1* **unfolding** *r'-def* **by** *blast*
ultimately show *h* $\alpha \in$ *Field r'* **unfolding** *Field-def* **by** *blast*
qed
have *b-fR'*: $\bigwedge$ α. $\alpha \in S \Longrightarrow$ *Field* (*R'* α) $\subseteq$ *W* α
proof −
fix α
assume *c1*: $\alpha \in S$
then have *Field* (*R0* α) $\subseteq$ *W* α **using** *a10* **unfolding** *R0-def Field-def* **by** *blast*
moreover have *Field* (*R2* α) $\subseteq$ *W* α
proof
fix *a*
assume *a* $\in$ *Field* (*R2* α)
then obtain *x y* **where** *d1*: (*x*,*y*) $\in$ *R2* α $\wedge$ (*a* = *x* $\vee$ *a* = *y*) **unfolding** *Field-def* **by** *blast*
then obtain *k* **where** *k* < *spl* α $\wedge$ (*x*,*y*) = (*sp* α *k*, *sp* α (*Suc k*)) **unfolding** *R2-def* **by** *blast*
then show *a* $\in$ *W* α **using** *d1 c1 b-sp*[*of* α] **unfolding** *spth-def rpth-def spl-def* **by** *blast*
qed

moreover have $re0\ \alpha \in W\ \alpha$ **using** *c1 b-re0*[*of* α] *b29* **by** *blast*
moreover have $re1\ \alpha \in W\ \alpha$ **using** *c1 b-re12*[*of* α] *a10*[*of* α] **unfolding** *Field-def* **by** *blast*
ultimately show $Field\ (R'\ \alpha) \subseteq W\ \alpha$ **unfolding** *R′-def Field-def* **by** *fast*
qed
have *b-fR2*: $\bigwedge\ \alpha\ a.\ \alpha \in S \Longrightarrow a \in Field\ (R2\ \alpha) \Longrightarrow \exists\ k.\ k \leq spl\ \alpha \wedge a = sp\ \alpha\ k$
proof −
fix α a
assume $\alpha \in S$ **and** $a \in Field\ (R2\ \alpha)$
then obtain x y **where** $(x,y) \in R2\ \alpha \wedge (a = x \vee a = y)$ **unfolding** *Field-def* **by** *blast*
moreover then obtain k' **where** $k' < spl\ \alpha \wedge x = sp\ \alpha\ k' \wedge y = sp\ \alpha\ (Suc\ k')$
unfolding *R2-def* **by** *blast*
ultimately show $\exists\ k.\ k \leq spl\ \alpha \wedge a = sp\ \alpha\ k$ **by** (*metis Suc-leI less-or-eq-imp-le*)
qed
have *b-bhf*: $\bigwedge\ \alpha\ a.\ \alpha \in S \Longrightarrow a \in W\ \alpha \Longrightarrow (a,\ h\ \alpha) \in (R\ \alpha)\hat{\ }* \Longrightarrow a \in Field\ (R'\ \alpha)$
proof −
fix α a
assume *c1*: $\alpha \in S$ **and** *c2*: $a \in W\ \alpha$ **and** *c3*: $(a,\ h\ \alpha) \in (R\ \alpha)\hat{\ }*$
then have $(h\ \alpha,\ re0\ \alpha) \in (R0\ \alpha)\hat{\ }*$ **using** *b-hr0*[*of* α] *b-r0w*[*of* α] **by** *blast*
moreover have $R0\ \alpha \subseteq R\ \alpha$ **unfolding** *R0-def* **by** *blast*
ultimately have $(h\ \alpha,\ re0\ \alpha) \in (R\ \alpha)\hat{\ }*$ **using** *c3 rtrancl-mono* **by** *blast*
then have $(a,\ re0\ \alpha) \in (R\ \alpha)\hat{\ }*$ **using** *c3* **by** *force*
then have $(a,\ re0\ \alpha) \in (R0\ \alpha)\hat{\ }*$ **using** *c1 c3 b-R0*[*of* α] **by** *blast*
moreover have $R0\ \alpha \subseteq R'\ \alpha$ **unfolding** *R′-def* **by** *blast*
ultimately have $(a,\ re0\ \alpha) \in (R'\ \alpha)\hat{\ }*$ **using** *rtrancl-mono* **by** *blast*
moreover have $re0\ \alpha \in Field\ (R'\ \alpha)$ **unfolding** *R′-def Field-def* **by** *blast*
ultimately show $a \in Field\ (R'\ \alpha)$ **using** *lem-rtr-field*[*of a re0* α] **by** *blast*
qed
have *b-clR′*: $\bigwedge\ \alpha\ a.\ \alpha \in S \Longrightarrow a \in Field\ (R'\ \alpha) \Longrightarrow (a,\ ep\ \alpha) \in (R'\ \alpha)\hat{\ }*$
proof −
fix α a
assume *c1*: $\alpha \in S$ **and** *c2*: $a \in Field\ (R'\ \alpha)$
have *c3*: $sp\ \alpha\ 0 = re1\ \alpha$ **using** *c1 b-sp*[*of* α] **unfolding** *spth-def spl-def rpth-def* **by** *blast*
then have $a \in Field\ (R2\ \alpha) \vee a = re1\ \alpha \longrightarrow (\exists\ k.\ k \leq spl\ \alpha \wedge a = sp\ \alpha\ k)$ **using** *c1 b-fR2* **by** *force*
moreover have $a \in Field\ (R0\ \alpha) \vee a = re0\ \alpha \longrightarrow (a,\ re0\ \alpha) \in (R\ \alpha)\hat{\ }*$
unfolding *R0-def Field-def* **by** *fastforce*
moreover have $a \in Field\ (R0\ \alpha) \vee a \in Field\ (R2\ \alpha) \vee a = re0\ \alpha \vee a = re1\ \alpha$
using *c1 c2* **unfolding** *R′-def Field-def* **by** *blast*
moreover have *c4*: $\forall\ k.\ (k \leq spl\ \alpha \longrightarrow (sp\ \alpha\ k,\ ep\ \alpha) \in (R'\ \alpha)\hat{\ }*)$
proof (*intro allI impI*)
fix k
assume $k \leq spl\ \alpha$

moreover have *sp* α (*spl* α) = *ep* α
using *c1 b-sp*[*of* α] **unfolding** *spth-def spl-def rpth-def* **by** *blast*
moreover have $\forall$ $i <$ *spl* α. (*sp* α *i*, *sp* α (*Suc i*)) $\in R' \alpha$
unfolding *R'-def R2-def* **by** *blast*
ultimately show (*sp* α *k*, *ep* α) $\in (R' \alpha)$^* **using** *lem-rseq-tl* **by** *metis*
qed
moreover have (*a*, *re0* α) $\in (R\ \alpha)$^* $\longrightarrow$ (*a*, *ep* α) $\in (R' \alpha)$^*
proof
assume (*a*, *re0* α) $\in (R\ \alpha)$^*
then have (*a*, *re0* α) $\in$ (*R0* α)^* **using** *c1 b-R0* **by** *blast*
moreover have *R0* $\alpha \subseteq R' \alpha$ **using** *c1* **unfolding** *R'-def* **by** *blast*
ultimately have (*a*, *re0* α) $\in (R' \alpha)$^* **using** *rtrancl-mono* **by** *blast*
moreover have (*re0* α, *re1* α) $\in (R' \alpha)$ **using** *c1* **unfolding** *R'-def* **by** *blast*
moreover have (*re1* α, *ep* α) $\in (R' \alpha)$^* **using** *c3 c4* **by** *force*
ultimately show (*a*, *ep* α) $\in (R' \alpha)$^* **by** *simp*
qed
ultimately show (*a*, *ep* α) $\in (R' \alpha)$^* **by** *blast*
qed
have *b-epr'*: $\bigwedge$ *a*. $a \in$ *Field r'* $\Longrightarrow \exists\ \alpha \in S$. (*a*, *ep* α) $\in (R' \alpha)$^*
proof −
fix *a*
assume $a \in$ *Field r'*
then have $a \in$ *Field re'* $\vee$ ($\exists\ \alpha \in S$. $a \in$ *Field* $(R' \alpha)$) **unfolding** *r'-def Field-def* **by** *blast*
moreover have $a \in$ *Field re'* $\longrightarrow$ ($\exists\ \alpha \in S$. (*a*, *ep* α) $\in (R' \alpha)$^*)
proof
assume $a \in$ *Field re'*
then obtain $x\ y\ \alpha\ \beta$ **where** *d1*: $a = x \vee a = y$ **and** *d2*: $\alpha \in S \wedge \beta \in S \wedge \alpha <o\ \beta$
and *d3*: $x =$ *ep* $\alpha \wedge y \in W\ \beta \wedge (y, h\ \beta) \in (R\ \beta)$^*
unfolding *re'-def Field-def* **by** *blast*
have (*x*, *ep* α) $\in (R' \alpha)$^* **using** *d3* **by** *blast*
moreover have (*y*, *ep* β) $\in (R' \beta)$^* **using** *d2 d3 b-bhf*[*of* β *y*] *b-clR'*[*of* β] **by** *blast*
ultimately show $\exists\ \alpha \in S$. (*a*, *ep* α) $\in (R' \alpha)$^* **using** *d1 d2* **by** *blast*
qed
ultimately show $\exists\ \alpha \in S$. (*a*, *ep* α) $\in (R' \alpha)$^* **using** *b-clR'* **by** *blast*
qed
have *b-svR'*: $\bigwedge \alpha$. $\alpha \in S \Longrightarrow$ *single-valued* $(R' \alpha)$
proof −
fix α
assume *c1*: $\alpha \in S$
have *c2*: *re0* $\alpha \in$ *Domain* (*R0* α) $\longrightarrow$ *False*
proof
assume *re0* $\alpha \in$ *Domain* (*R0* α)
then obtain *b* **where** (*re0* α, *b*) $\in$ *R0* α **by** *blast*
then have (*re0* α, *b*) $\in R\ \alpha \wedge$ (*b*, *re0* α) $\in (R\ \alpha)$^* **unfolding** *R0-def* **by** *blast*
then have (*re0* α, *re0* α) $\in (R\ \alpha)$^+ **by** *force*

moreover have *acyclic* (*R* α) **using** *c1 a10 a3* **by** *blast*
ultimately show *False* **unfolding** *acyclic-def* **by** *blast*
qed
have *c3*: *re0* α ∈ *Domain* (*R2* α) ⟶ *False*
proof
assume *re0* α ∈ *Domain* (*R2* α)
then obtain *b* **where** (*re0* α, *b*) ∈ *R2* α **by** *blast*
then obtain *k* **where** *d1*: *k* ≤ *spl* α ∧ *re0* α = *sp* α *k* ∧ *b* = *sp* α (*Suc k*)
unfolding *R2-def* **by** *force*
have *sp* α ∈ *spth* (*Restr r* (*W* α)) (*re1* α) (*ep* α) **using** *c1 b-sp* **by** *blast*
then have *sp* α *0* = *re1* α **and** ∀ *i*<*spl* α. (*sp* α *i*, *sp* α (*Suc i*)) ∈ *Restr r* (*W* α)
unfolding *spth-def spl-def rpth-def* **by** *blast*+
then have (*re1* α, *re0* α) ∈ (*Restr r* (*W* α))^* **using** *d1 lem-rseq-hd* **by** *metis*
then show *False* **using** *c1 b-re12*[*of* α] **by** *blast*
qed
have *c4*: ∀ *a* ∈ *Field* (*R0* α) ∩ *Field* (*R2* α). *False*
proof
fix *a*
assume *d1*: *a* ∈ *Field* (*R0* α) ∩ *Field* (*R2* α)
obtain *k* **where** *d2*: *k* ≤ *spl* α ∧ *a* = *sp* α *k* **using** *d1 c1 b-fR2*[*of* α *a*] **by** *blast*
have *sp* α ∈ *spth* (*Restr r* (*W* α)) (*re1* α) (*ep* α) **using** *c1 b-sp* **by** *blast*
then have *sp* α *0* = *re1* α **and** ∀ *i*<*spl* α. (*sp* α *i*, *sp* α (*Suc i*)) ∈ *Restr r* (*W* α)
unfolding *spth-def spl-def rpth-def* **by** *blast*+
then have *d3*: (*re1* α, *a*) ∈ (*Restr r* (*W* α))^*
using *d2 lem-rseq-hd* **unfolding** *spth-def rpth-def* **by** *metis*
have (*a*, *re0* α) ∈ (*R* α)^* **using** *d1* **unfolding** *R0-def Field-def* **by** *force*
moreover have *R* α ⊆ *Restr r* (*W* α) **using** *c1 a10* **unfolding** *Field-def* **by** *fastforce*
ultimately have (*a*, *re0* α) ∈ (*Restr r* (*W* α))^* **using** *rtrancl-mono* **by** *blast*
then have (*re1* α, *re0* α) ∈ (*Restr r* (*W* α))^* **using** *d3* **by** *force*
then show *False* **using** *c1 b-re12*[*of* α] **by** *blast*
qed
have *R0* α ⊆ *R* α **unfolding** *R0-def* **by** *blast*
then have *c5*: *single-valued* (*R0* α) **using** *c1 a3 a10*[*of* α] **unfolding** *single-valued-def* **by** *blast*
have *c6*: ∀ *a b c*. (*a*,*b*) ∈ *R2* α ∧ (*a*,*c*) ∈ *R2* α ⟶ *b* = *c*
proof (*intro allI impI*)
fix *a b c*
assume (*a*,*b*) ∈ *R2* α ∧ (*a*,*c*) ∈ *R2* α
then obtain *k1 k2* **where** *d1*: *k1* < *spl* α ∧ *a* = *sp* α *k1* ∧ *b* = *sp* α (*Suc k1*)
and *d2*: *k2* < *spl* α ∧ *a* = *sp* α *k2* ∧ *c* = *sp* α (*Suc k2*)
unfolding *R2-def* **by** *blast*
then have *sp* α *k1* = *sp* α *k2* ∧ *k1* ≤ *spl* α ∧ *k2* ≤ *spl* α **by** *force*

moreover have *inj-on* (*sp* α) {*i*. *i*≤*spl* α}
using *c1 b-sp*[*of* α] *lem-spth-inj*[*of sp* α] **unfolding** *spl-def* **by** *blast*
ultimately have *k1* = *k2* **unfolding** *inj-on-def* **by** *blast*
then show *b* = *c* **using** *d1 d2* **by** *blast*
qed
have *single-valued* (*R0* α ∪ {(*re0* α, *re1* α)})
using *c2 c5* **unfolding** *single-valued-def* **by** *blast*
moreover have *single-valued* (*R2* α ∪ {(*re0* α, *re1* α)})
using *c3 c6* **unfolding** *single-valued-def* **by** *blast*
ultimately show *single-valued* (*R′* α) **using** *c4 lem-sv-un3* **unfolding** *R′-def* **by** *blast*
qed
have *b-acR′*: ⋀ α. α ∈ *S* ⟹ *acyclic* (*R′* α)
proof −
fix α
assume *c1*: α ∈ *S*
obtain *s* **where** *c2*: *s* = *R0* α ∪ {(*re0* α, *re1* α)} **by** *blast*
then have *s* ⊆ *R* α **using** *c1 b-re12*[*of* α] **unfolding** *R0-def* **by** *blast*
moreover have *acyclic* (*R* α) **using** *c1 a3 a10* **by** *blast*
ultimately have *acyclic s* **using** *acyclic-subset* **by** *blast*
moreover have *acyclic* (*R2* α)
proof −
have ∀ *a*. (*a*,*a*) ∈ (*R2* α)^+ ⟶ *False*
proof (*intro allI impI*)
fix *a*
assume (*a*,*a*) ∈ (*R2* α)^+
then obtain *n* **where** *e1*: *n* > *0* ∧ (*a*,*a*) ∈ (*R2* α)^^*n* **using** *trancl-power* **by** *blast*
then obtain *g* **where** *e2*: *g 0* = *a* ∧ *g n* = *a* **and** *e3*: ∀ *i*<*n*. (*g i*, *g* (*Suc i*)) ∈ *R2* α
using *relpow-fun-conv*[*of a a n R2* α] **by** *blast*
then have (*g 0*, *g* (*Suc 0*)) ∈ *R2* α **using** *e1* **by** *force*
then obtain *k0* **where** *e4*: *k0* < *spl* α ∧ *g 0* = *sp* α *k0* **unfolding** *R2-def* **by** *blast*
have *e5*: *inj-on* (*sp* α) {*i*. *i*≤*spl* α}
using *c1 b-sp*[*of* α] *lem-spth-inj*[*of sp* α] **unfolding** *spl-def* **by** *blast*
have ∀ *i*≤*n*. *k0* + *i* ≤ *spl* α ∧ *g i* = *sp* α (*k0* + *i*)
proof
fix *i*
show *i* ≤ *n* ⟶ *k0* + *i* ≤ *spl* α ∧ *g i* = *sp* α (*k0* + *i*)
proof (*induct i*)
show *0* ≤ *n* ⟶ *k0* + *0* ≤ *spl* α ∧ *g 0* = *sp* α (*k0* + *0*) **using** *e4* **by** *simp*
next
fix *i*
assume *g1*: *i* ≤ *n* ⟶ *k0* + *i* ≤ *spl* α ∧ *g i* = *sp* α (*k0* + *i*)
show *Suc i* ≤ *n* ⟶ *k0* + *Suc i* ≤ *spl* α ∧ *g* (*Suc i*) = *sp* α (*k0* + *Suc i*)
proof

assume *h1*: *Suc i* ≤ *n*
then have *h2*: *k0* + *i* ≤ *spl* α ∧ *g i* = *sp* α (*k0* + *i*) **using** *g1* **by** *simp*
moreover have (*g i*, *g* (*Suc i*)) ∈ *R2* α **using** *h1 e3* **by** *simp*
ultimately obtain *k* **where**
h3: *k* < *spl* α ∧ *sp* α (*k0* + *i*) = *sp* α *k* ∧ *g* (*Suc i*) = *sp* α (*Suc k*)
unfolding *R2-def* **by** *fastforce*
then have *h4*: *k0* + *i* = *k* **using** *h2 h3 e5* **unfolding** *inj-on-def* **by** *simp*
then have *k0* + *Suc i* ≤ *spl* α **using** *h3* **by** *simp*
moreover have *g* (*Suc i*) = *sp* α (*k0* + *Suc i*) **using** *h3 h4* **by** *simp*
ultimately show *k0* + *Suc i* ≤ *spl* α ∧ *g* (*Suc i*) = *sp* α (*k0* + *Suc i*) **by** *blast*
qed
qed
qed
then have *k0* + *n* ≤ *spl* α ∧ *a* = *sp* α (*k0* + *n*) **using** *e2* **by** *simp*
moreover have *k0* ≤ *spl* α ∧ *a* = *sp* α *k0* **using** *e2 e4* **by** *simp*
ultimately have *k0* + *n* = *k0* **using** *e5* **unfolding** *inj-on-def* **by** *blast*
then show *False* **using** *e1* **by** *simp*
qed
then show *?thesis* **unfolding** *acyclic-def* **by** *blast*
qed
moreover have ∀ *a* ∈ (*Range* (*R2* α)) ∩ (*Domain s*). *False*
proof
fix *a*
assume *e1*: *a* ∈ (*Range* (*R2* α)) ∩ (*Domain s*)
then have *e2*: *a* ∈ *Field* (*R0* α) ∨ *a* = *re0* α **using** *c2* **unfolding** *Field-def* **by** *blast*
obtain *k* **where** *e3*: *k* ≤ *spl* α ∧ *a* = *sp* α *k* **using** *e1 c1 b-fR2*[*of* α *a*] **unfolding** *Field-def* **by** *blast*
have *sp* α ∈ *spth* (*Restr r* (*W* α)) (*re1* α) (*ep* α) **using** *c1 b-sp* **by** *blast*
then have *sp* α *0* = *re1* α **and** ∀ *i*<*spl* α. (*sp* α *i*, *sp* α (*Suc i*)) ∈ *Restr r* (*W* α)
unfolding *spth-def spl-def rpth-def* **by** *blast+*
then have *e4*: (*re1* α, *a*) ∈ (*Restr r* (*W* α))^*
using *e3 lem-rseq-hd* **unfolding** *spth-def rpth-def* **by** *metis*
have (*a*, *re0* α) ∈ (*R* α)^* **using** *e2* **unfolding** *R0-def Field-def* **by** *force*
moreover have *R* α ⊆ *Restr r* (*W* α) **using** *c1 a10* **unfolding** *Field-def* **by** *fastforce*
ultimately have (*a*, *re0* α) ∈ (*Restr r* (*W* α))^* **using** *rtrancl-mono* **by** *blast*
then have (*re1* α, *re0* α) ∈ (*Restr r* (*W* α))^* **using** *e4* **by** *force*
then show *False* **using** *c1 b-re12*[*of* α] **by** *blast*
qed
moreover have *R′* α = *R2* α ∪ *s* **using** *c2* **unfolding** *R′-def* **by** *blast*
ultimately show *acyclic* (*R′* α) **using** *lem-acyc-un-emprd*[*of R2* α *s*] **by** *force*
qed
have *b-dr′*: ⋀ α. α ∈ *S* ⟹ *Domain* (*R′* α) ∩ *Domain re′* = {}

proof −
fix α
assume *c1*: $\alpha \in S$
have $\forall$ *a b c*. $(a,b) \in (R' \alpha) \wedge (a,c) \in re' \longrightarrow$ *False*
proof (*intro allI impI*)
fix *a b c*
assume *d1*: $(a,b) \in (R' \alpha) \wedge (a,c) \in re'$
then obtain α' **where** *d2*: $\alpha' \in S \wedge a = ep\ \alpha'$ **unfolding** *re′-def* **by** *blast*
then have $a \in W\ \alpha'$ **using** *b-ep*[*of* α'] **unfolding** *EP-def* **by** *blast*
moreover have $a \in W\ \alpha$ **using** *d1 c1 b-fR′*[*of* α] **unfolding** *Field-def* **by** *blast*
ultimately have $\alpha' = \alpha$ **using** *d2 c1 a9* **by** *blast*
then have $a = ep\ \alpha$ **using** *d2* **by** *blast*
moreover have $(b,\ ep\ \alpha) \in (R' \alpha)\hat{}*$ **using** *d1 c1 b-clR′* **unfolding** *Field-def* **by** *blast*
ultimately have $(a,\ a) \in (R' \alpha)\hat{}+$ **using** *d1* **by** *force*
then show *False* **using** *c1 b-acR′* **unfolding** *acyclic-def* **by** *blast*
qed
then show *Domain* $(R' \alpha) \cap$ *Domain* $re' = \{\}$ **by** *blast*
qed
have *b-pkr′*: $\bigwedge$ *a b1 b2*. $(a,b1) \in r' \wedge (a,b2) \in r' \wedge b1 \neq b2 \Longrightarrow \forall$ *b*. $(a,b) \in r' \longrightarrow (a,b) \in re'$
proof −
fix *a b1 b2*
assume *c1*: $(a,b1) \in r' \wedge (a,b2) \in r' \wedge b1 \neq b2$
moreover have $\forall \alpha{\in}S.\ \forall \beta{\in}S.\ (a,b1) \in R' \alpha \wedge (a,b2) \in R' \beta \longrightarrow$ *False*
proof (*intro ballI impI*)
fix $\alpha\ \beta$
assume $\alpha \in S$ **and** $\beta \in S$ **and** $(a,b1) \in R' \alpha \wedge (a,b2) \in R' \beta$
moreover then have $\alpha = \beta$ **using** *b-fR′*[*of* α] *b-fR′*[*of* β] *a9* **unfolding** *Field-def* **by** *blast*
ultimately show *False* **using** *c1 b-svR′*[*of* α] **unfolding** *single-valued-def* **by** *blast*
qed
ultimately have $(a,b1) \in re' \vee (a,b2) \in re'$ **unfolding** *r′-def* **by** *blast*
then have $\forall\ \alpha{\in}S.\ a \notin$ *Domain* $(R' \alpha)$ **using** *b-dr′* **by** *blast*
then show $\forall$ *b*. $(a,b) \in r' \longrightarrow (a,b) \in re'$ **using** *c1* **unfolding** *r′-def* **by** *blast*
qed
have $r' \subseteq r$
proof
fix *p*
assume $p \in r'$
moreover have $\forall\ \alpha \in S.\ p \in R' \alpha \longrightarrow p \in r$
proof (*intro ballI impI*)
fix α
assume *d1*: $\alpha \in S$ **and** $p \in R' \alpha$
moreover have $p \in R0\ \alpha \longrightarrow p \in r$ **unfolding** *R0-def* **using** *d1 a10* **by** *blast*
moreover have $p \in R2\ \alpha \longrightarrow p \in r$

proof
assume $p \in R2\ \alpha$
then obtain k **where** $k{<}spl\ \alpha \wedge p = (sp\ \alpha\ k,\ sp\ \alpha\ (Suc\ k))$ **unfolding** *R2-def* **by** *blast*
then have $p \in Restr\ r\ (W\ \alpha)$ **using** *d1 b-sp*[*of* α] **unfolding** *spth-def rpth-def spl-def* **by** *blast*
then show $p \in r$ **by** *blast*
qed
moreover have $(re0\ \alpha,\ re1\ \alpha) \in r$ **using** *d1 b-re12 a10* **by** *blast*
ultimately show $p \in r$ **unfolding** *R′-def* **by** *blast*
qed
ultimately show $p \in r$ **unfolding** *r′-def re′-def* **by** *blast*
qed
moreover have $\forall a{\in}Field\ r.\ \exists b{\in}Field\ r'.\ (a,\ b) \in r\hat{}*$
proof
fix a
assume $a \in Field\ r$
then obtain α **where** $c1$: $\alpha \in S \wedge a \in W\ \alpha$ **using** *a8* **by** *blast*
then obtain a' **where** $c2$: $(a,\ a') \in (Restr\ r\ (W\ \alpha))\hat{}*$
and $c3$: $\forall \beta{\in}S.\ \alpha <o\ \beta \longrightarrow r``\{a'\} \cap W\ \beta \neq \{\}$ **using** *a11*[*of* α a] **by** *blast*
have $a' \in W\ \alpha$ **using** *c1 c2 lem-rtr-field*[*of a a′*] **unfolding** *Field-def* **by** *blast*
then have $a' \in EP\ \alpha$ **using** *c3* **unfolding** *EP-def* **by** *blast*
then obtain $\gamma\ a''$ **where** $c4$: $\gamma \in S$ **and** $c5$: $a'' \in W\ \gamma \wedge (a',\ a'') \in r \wedge (a'',\ h\ \gamma) \in (R\ \gamma)\hat{}*$
using *c1 b-h*[*of* α α a' a'] **by** *blast*
moreover then have $(a'',\ h\ \gamma) \in r\hat{}*$ **using** *p1 rtrancl-mono*[*of* $R\ \gamma\ r$] **by** *blast*
moreover have $(a,\ a') \in r\hat{}*$ **using** *c2 rtrancl-mono*[*of* $Restr\ r\ (W\ \alpha)\ r$] **by** *blast*
ultimately have $(a,\ h\ \gamma) \in r\hat{}*$ **by** *force*
moreover have $h\ \gamma \in W\ \gamma$ **using** *c4 c5 b-r0w* **by** *blast*
moreover then have $h\ \gamma \in Field\ r'$ **using** *c4 b-hf* **by** *blast*
ultimately show $\exists b{\in}Field\ r'.\ (a,\ b) \in r\hat{}*$ **by** *blast*
qed
moreover have $DCR\ 2\ r' \wedge CCR\ r'$
proof −
obtain $g0$ **where** $c1$: $g0 = \{\ (u,v) \in r'.\ r'``\{u\} = \{v\}\ \}$ **by** *blast*
obtain $g1$ **where** $c2$: $g1 = r' - g0$ **by** *blast*
obtain g **where** $c3$: $g = (\lambda n{::}nat.\ (if\ (n{=}0)\ then\ g0\ else\ (if\ (n{=}1)\ then\ g1\ else\ \{\})))$ **by** *blast*
have $c4$: $\forall\ \beta \in S.\ R'\ \beta \subseteq g0$
proof
fix β
assume $d1$: $\beta \in S$
then have $R'\ \beta \subseteq r'$ **unfolding** *r′-def* **by** *blast*
moreover have $\forall\ a\ b\ c.\ (a,b) \in R'\ \beta \wedge (a,c) \in r' \longrightarrow b = c$
proof (*intro allI impI*)
fix $a\ b\ c$

assume $e1$: $(a, b) \in R'\ \beta \wedge (a, c) \in r'$
moreover then have $(a,b) \in r'$ **using** *d1* **unfolding** *r′-def* **by** *blast*
ultimately have $b = c \vee (a, b) \in re'$ **using** *b-pkr′[of a b c]* **by** *blast*
moreover have $(a,b) \in re' \longrightarrow False$ **using** *e1 d1 b-dr′[of β]* **by** *blast*
ultimately show $b = c$ **by** *blast*
qed
ultimately show $R'\ \beta \subseteq g0$ **using** *c1* **by** *blast*
qed
have $c5$: $re' \subseteq g1$
proof −
have $re' \subseteq r'$ **unfolding** *r′-def* **by** *blast*
moreover have $\forall\ a\ b.\ (a,b) \in re' \wedge (a,b) \in g0 \longrightarrow False$
proof (*intro allI impI*)
fix *a b*
assume $e1$: $(a,b) \in re' \wedge (a,b) \in g0$
then obtain α **where** $e2$: $\alpha \in S \wedge a = ep\ \alpha$ **unfolding** *re′-def* **by** *blast*
then have $e3$: $a \in EP\ \alpha$ **using** *b-ep* **by** *blast*
obtain $\gamma 1\ a1$ **where** $e4$: $\gamma 1 \in S \wedge \alpha <o\ \gamma 1 \wedge a1 \in W\ \gamma 1 \wedge (a,a1) \in re'$
using *e2 e3 b-h[of α α a a] b-bhf re′-def* **by** *blast*
then have $\gamma 1 \in S \wedge ep\ \gamma 1 \in EP\ \gamma 1$ **using** *b-ep* **by** *blast*
then obtain $\gamma 2\ a2$ **where** $e5$: $\gamma 2 \in S \wedge \gamma 1 <o\ \gamma 2 \wedge a2 \in W\ \gamma 2 \wedge (a,a2) \in re'$
using *e2 e3 b-h[of α γ1 a ep γ1] re′-def* **by** *blast*
then have $\gamma 1 \neq \gamma 2$ **using** *ordLess-irrefl* **unfolding** *irrefl-def* **by** *blast*
then have $a1 \neq a2$ **using** *e4 e5 a9* **by** *blast*
moreover have $a1 \in r'``\{a\} \wedge a2 \in r'``\{a\}$ **using** *e4 e5* **unfolding** *r′-def* **by** *blast*
moreover have $r'``\{a\} = \{b\}$ **using** *e1 c1* **by** *blast*
ultimately have $a1 \in \{b\} \wedge a2 \in \{b\} \wedge a1 \neq a2$ **by** *blast*
then show *False* **by** *blast*
qed
ultimately show *?thesis* **using** *c2* **by** *force*
qed
have $r' = \bigcup\{r'.\ \exists \alpha'<2.\ r' = g\ \alpha'\}$
proof
have $r' \subseteq g0 \cup g1$ **using** *c1 c2* **by** *blast*
moreover have $g0 = g\ 0 \wedge g1 = g\ 1 \wedge (0::nat) < 2 \wedge (1::nat) < 2$ **using** *c3* **by** *simp*
ultimately show $r' \subseteq \bigcup\{r'.\ \exists \alpha'<2.\ r' = g\ \alpha'\}$ **by** *blast*
next
have $\bigwedge \alpha.\ g\ \alpha \subseteq g0 \cup g1$ **unfolding** *c3* **by** *simp*
then show $\bigcup\{r'.\ \exists \alpha'<2.\ r' = g\ \alpha'\} \subseteq r'$ **using** *c1 c2* **by** *blast*
qed
moreover have $\forall l1\ l2\ u\ v\ w.\ l1 \leq l2 \longrightarrow (u, v) \in g\ l1 \wedge (u, w) \in g\ l2 \longrightarrow$
$(\exists v'\ v''\ w'\ w''\ d.\ (v, v', v'', d) \in \mathfrak{D}\ g\ l1\ l2 \wedge (w, w', w'', d) \in \mathfrak{D}\ g\ l2\ l1)$
proof (*intro allI impI*)
fix *l1 l2 u v w*
assume $d1$: $l1 \leq l2$ **and** $d2$: $(u, v) \in g\ l1 \wedge (u, w) \in g\ l2$
have $d3$: $g0 = g\ 0 \wedge g1 = g\ 1$

and $d4$: $\forall\ \alpha.\ g\ \alpha \neq \{\} \longrightarrow \alpha = 0 \vee \alpha = 1$ **unfolding** *c3* **by** *simp+*
have $d5$: $\mathfrak{L}1\ g\ 1 = g0$ **and** $d6$: $\mathfrak{L}v\ g\ 1\ 1 = g0$
and $d7$: $\mathfrak{L}v\ g\ 1\ 0 = g0$ **and** $d8$: $\mathfrak{L}v\ g\ 0\ 1 = g0$ **using** *d3* **unfolding** *𝔏1-def 𝔏v-def* **by** *blast+*
show $\exists\, v'\ v''\ w'\ w''\ d.\ (v,\ v',\ v'',\ d) \in \mathfrak{D}\ g\ l1\ l2 \wedge (w,\ w',\ w'',\ d) \in \mathfrak{D}\ g\ l2\ l1$
proof −
have $l1 = 0 \wedge l2 = 0 \Longrightarrow$ *?thesis*
proof −
assume $l1 = 0 \wedge l2 = 0$
then have $r'``\{u\} = \{v\} \wedge r'``\{u\} = \{w\}$ **using** *c1 d2 d3* **by** *blast*
then have $v = w$ **by** *blast*
then show *?thesis* **unfolding** *𝔇-def* **by** *fastforce*
qed
moreover have $l1 = 0 \wedge l2 = 1 \Longrightarrow$ *False*
proof −
assume $l1 = 0 \wedge l2 = 1$
then have $(u,\ v) \in r' \wedge (u,\ w) \in r'$
and $r'``\{u\} = \{v\} \wedge r'``\{u\} \neq \{w\}$ **using** *c1 c2 d2 d3* **by** *blast+*
then show *False* **by** *force*
qed
moreover have $l1 = 1 \wedge l2 = 1 \Longrightarrow$ *?thesis*
proof −
assume $f1$: $l1 = 1 \wedge l2 = 1$
then have $(u,v) \in g1 \wedge (u,w) \in g1$ **using** *d2 d3* **by** *blast*
then have $(u,v) \in re' \wedge (u,w) \in re'$ **using** *c1 c2 b-pkr′* **by** *blast*
then obtain $\beta 1\ \beta 2$ **where** $f2$: $\beta 1 \in S \wedge \beta 2 \in S$
and $v \in W\ \beta 1 \wedge (v,\ h\ \beta 1) \in (R\ \beta 1)\hat{}*$
and $w \in W\ \beta 2 \wedge (w,\ h\ \beta 2) \in (R\ \beta 2)\hat{}*$ **unfolding** *re′-def* **by** *blast*
then have $v \in Field\ (R'\ \beta 1) \wedge w \in Field\ (R'\ \beta 2)$ **using** *b-bhf* **by** *blast*
then have $f3$: $(v,\ ep\ \beta 1) \in (R'\ \beta 1)\hat{}* \wedge (w,\ ep\ \beta 2) \in (R'\ \beta 2)\hat{}*$ **using** *f2 b-clR′* **by** *blast*
then have $ep\ \beta 1 \in EP\ \beta 1 \wedge ep\ \beta 2 \in EP\ \beta 2$ **using** *f2 b-ep* **by** *blast*
then obtain $\gamma\ v''\ w''$ **where** $f4$: $\gamma \in S \wedge \beta 1 <o\ \gamma \wedge \beta 2 <o\ \gamma$
and $v'' \in W\ \gamma \wedge (ep\ \beta 1,\ v'') \in r \wedge (v'',\ h\ \gamma) \in (R\ \gamma)\hat{}*$
and $w'' \in W\ \gamma \wedge (ep\ \beta 2,\ w'') \in r \wedge (w'',\ h\ \gamma) \in (R\ \gamma)\hat{}*$
using *f2 b-h[of β1 β2 ep β1 ep β2]* **by** *blast*
then have $(ep\ \beta 1,\ v'') \in re' \wedge (ep\ \beta 2,\ w'') \in re'$
and $(v'',\ ep\ \gamma) \in (R'\ \gamma)\hat{}* \wedge (w'',\ ep\ \gamma) \in (R'\ \gamma)\hat{}*$
using *f2 b-bhf b-clR′* **unfolding** *re′-def* **by** *blast+*
moreover obtain $v'\ w'\ d$ **where** $v' = ep\ \beta 1 \wedge w' = ep\ \beta 2 \wedge d = ep\ \gamma$ **by** *blast*
ultimately have $f5$: $(v,\ v') \in (R'\ \beta 1)\hat{}* \wedge (v',\ v'') \in re' \wedge (v'',\ d) \in (R'\ \gamma)\hat{}*$
and $f6$: $(w,\ w') \in (R'\ \beta 2)\hat{}* \wedge (w',\ w'') \in re' \wedge (w'',\ d) \in (R'\ \gamma)\hat{}*$
using *f3* **by** *blast+*
have $(R'\ \beta 1)\hat{}* \subseteq (\mathfrak{L}1\ g\ l1)\hat{}*$ **using** *f1 f2 d5 c4 rtrancl-mono* **by** *blast*
moreover have $re' \subseteq g\ l2$ **using** *f1 d3 c5* **by** *blast*

moreover have $(R'\ \gamma)\widehat{\ }{*} \subseteq (\mathfrak{L}v\ g\ l1\ l2)\widehat{\ }{*}$ **using** *f1 f4 d6 c4 rtrancl-mono* **by** *blast*
moreover have $(R'\ \beta 2)\widehat{\ }{*} \subseteq (\mathfrak{L}1\ g\ l2)\widehat{\ }{*}$ **using** *f1 f2 d5 c4 rtrancl-mono* **by** *blast*
moreover have $re' \subseteq g\ l1$ **using** *f1 d3 c5* **by** *blast*
moreover have $(R'\ \gamma)\widehat{\ }{*} \subseteq (\mathfrak{L}v\ g\ l2\ l1)\widehat{\ }{*}$ **using** *f1 f4 d6 c4 rtrancl-mono* **by** *blast*
ultimately have $(v, v', v'', d) \in \mathfrak{D}\ g\ l1\ l2 \wedge (w, w', w'', d) \in \mathfrak{D}\ g\ l2\ l1$
using *f5 f6* **unfolding** $\mathfrak{D}$*-def* **by** *blast*
then show *?thesis* **by** *blast*
qed
moreover have $(l1 = 0 \vee l1 = 1) \wedge (l2 = 0 \vee l2 = 1)$ **using** *d2 d4* **by** *blast*
ultimately show *?thesis* **using** *d1* **by** *fastforce*
qed
qed
ultimately have *c9*: *DCR 2 r′* **using** *lem-Ldo-ldogen-ord* **unfolding** *DCR-def* **by** *blast*
have $\forall\, a{\in}Field\ r'.\ \forall\, b{\in}Field\ r'.\ \exists\, c \in Field\ r'.\ (a,c) \in r'\widehat{\ }{*} \wedge (b,c) \in r'\widehat{\ }{*}$
proof (*intro ballI impI*)
fix *a b*
assume *d1*: $a \in Field\ r'$ **and** *d2*: $b \in Field\ r'$
obtain $\alpha\ \beta$ **where** *d3*: $\alpha \in S \wedge \beta \in S$
and *d4*: $(a,\ ep\ \alpha) \in (R'\ \alpha)\widehat{\ }{*} \wedge (b,\ ep\ \beta) \in (R'\ \beta)\widehat{\ }{*}$ **using** *d1 d2 b-epr′* **by** *blast*
then have $ep\ \alpha \in EP\ \alpha \wedge ep\ \beta \in EP\ \beta$ **using** *b-ep* **by** *blast*
then obtain $\gamma\ a'\ b'$ **where** *d5*: $\gamma \in S \wedge \alpha <o\ \gamma \wedge \beta <o\ \gamma$
and *d6*: $a' \in W\ \gamma \wedge (ep\ \alpha,\ a') \in r \wedge (a',\ h\ \gamma) \in (R\ \gamma)\widehat{\ }{*}$
and *d7*: $b' \in W\ \gamma \wedge (ep\ \beta,\ b') \in r \wedge (b',\ h\ \gamma) \in (R\ \gamma)\widehat{\ }{*}$
using *d3 b-h*[*of* $\alpha\ \beta$ *ep* α *ep* β] **by** *blast*
then have $(a',\ ep\ \gamma) \in (R'\ \gamma)\widehat{\ }{*} \wedge (b',\ ep\ \gamma) \in (R'\ \gamma)\widehat{\ }{*}$ **using** *b-bhf b-clR′* **by** *blast*
moreover have $R'\ \alpha \subseteq r' \wedge R'\ \beta \subseteq r' \wedge R'\ \gamma \subseteq r'$ **using** *d3 d5* **unfolding** *r′-def* **by** *blast*
ultimately have $(a,\ ep\ \alpha) \in r'\widehat{\ }{*} \wedge (b,\ ep\ \beta) \in r'\widehat{\ }{*}$
and $(a',\ ep\ \gamma) \in r'\widehat{\ }{*} \wedge (b',\ ep\ \gamma) \in r'\widehat{\ }{*}$ **using** *d4 rtrancl-mono* **by** *blast+*
moreover have $(ep\ \alpha,\ a') \in r'$ **using** *d3 d5 d6* **unfolding** *r′-def re′-def* **by** *blast*
moreover have $(ep\ \beta,\ b') \in r'$ **using** *d3 d5 d7* **unfolding** *r′-def re′-def* **by** *blast*
ultimately have $(a,\ ep\ \gamma) \in r'\widehat{\ }{*} \wedge (b,\ ep\ \gamma) \in r'\widehat{\ }{*}$ **by** *force*
moreover then have $ep\ \gamma \in Field\ r'$ **using** *d1 lem-rtr-field* **by** *metis*
ultimately show $\exists\, c \in Field\ r'.\ (a,c) \in r'\widehat{\ }{*} \wedge (b,c) \in r'\widehat{\ }{*}$ **by** *blast*
qed
then have *CCR r′* **unfolding** *CCR-def* **by** *blast*
then show *?thesis* **using** *c9* **by** *blast*
qed
ultimately show *?thesis* **by** *blast*

qed

lemma *lem-uset-cl-ext*:
fixes *r*::*'U rel* **and** *s*::*'U rel*
assumes *s* ∈ 𝔘 *r* **and** *Conelike s*
shows *Conelike r*
proof (*cases s* = {})
  assume *s* = {}
  then have *r* = {} **using** *assms* **unfolding** 𝔘-*def Field-def* **by** *fast*
  then show *Conelike r* **unfolding** *Conelike-def* **by** *blast*
next
  assume *s* ≠ {}
  then obtain *m* **where** *m* ∈ *Field s* ∧ (∀ *a* ∈ *Field s*. (*a*,*m*) ∈ *s*^*) **using** *assms* **unfolding** *Conelike-def* **by** *blast*
  moreover have *s* ⊆ *r* ∧ (∀ *a* ∈ *Field r*. ∃ *b* ∈ *Field s*. (*a*,*b*) ∈ *r*^*) **using** *assms* **unfolding** 𝔘-*def* **by** *blast*
  moreover then have *Field s* ⊆ *Field r* ∧ *s*^* ⊆ *r*^* **unfolding** *Field-def* **using** *rtrancl-mono* **by** *blast*
  ultimately have (*m* ∈ *Field r*) ∧ (∀ *a* ∈ *Field r*. (*a*,*m*) ∈ *r*^*) **by** (*meson rtrancl-trans subsetCE*)
  then show *Conelike r* **unfolding** *Conelike-def* **by** *blast*
qed

lemma *lem-uset-cl-singleton*:
fixes *r*::*'U rel*
assumes *Conelike r* **and** *r* ≠ {}
shows ∃ *m*::*'U*. ∃ *m'*::*'U*. {(*m'*,*m*)} ∈ 𝔘 *r*
proof −
  obtain *m* **where** *b1*: *m* ∈ *Field r* ∧ (∀ *a* ∈ *Field r*. (*a*,*m*) ∈ *r*^*) **using** *assms* **unfolding** *Conelike-def* **by** *blast*
  then obtain *x* **where** *b2*: (*m*,*x*) ∈ *r* ∨ (*x*,*m*) ∈ *r* **unfolding** *Field-def* **by** *blast*
  then have (*x*,*m*) ∈ *r*^* **using** *b1* **unfolding** *Field-def* **by** *blast*
  then obtain *m'* **where** *b3*: (*m'*,*m*) ∈ *r* **using** *b2* **by** (*metis rtranclE*)
  have *CCR* {(*m'*,*m*)} **unfolding** *CCR-def Field-def* **by** *force*
  moreover have ∀ *a*∈*Field r*. ∃ *b*∈*Field* {(*m'*,*m*)}. (*a*, *b*) ∈ *r*^* **using** *b1* **unfolding** *Field-def* **by** *blast*
  ultimately show *?thesis* **using** *b3* **unfolding** 𝔘-*def* **by** *blast*
qed

lemma *lem-rcc-emp*: ‖{}‖ = {}
  unfolding *RCC-def RCC-rel-def* 𝔘-*def* **apply** *simp*
  unfolding *CCR-def* **apply** *simp*
  using *lem-card-emprel* **by** (*smt iso-ozero-empty ordIso-symmetric ozero-def someI-ex*)

lemma *lem-rcc-rccrel*:
fixes *r*::*'U rel*
shows *RCC-rel r* ‖*r*‖
proof −
  have ∃ α. *RCC-rel r* α

proof (*cases* $\mathfrak{U}$ *r* = {})
assume $\mathfrak{U}$ *r* = {}
then show ∃ α. *RCC-rel r* α **unfolding** *RCC-rel-def* **by** *blast*
next
assume *b1*: $\mathfrak{U}$ *r* ≠ {}
obtain *Q* **where** *b2*: *Q* = { α::'*U rel*. ∃ *s* ∈ $\mathfrak{U}$ *r*. α =*o* |*s*| } **by** *blast*
have *b3*: ∀ *s* ∈ $\mathfrak{U}$ *r*. ∃ α ∈ *Q*. α ≤*o* |*s*|
proof
fix *s*
assume *c1*: *s* ∈ $\mathfrak{U}$ *r*
then have *c2*: *s* ⊆ (*UNIV*::'*U set*) × (*UNIV*::'*U set*) **unfolding** $\mathfrak{U}$-*def* **by** *simp*
then have *c3*: |*s*| ≤*o* |(*UNIV*::'*U set*) × (*UNIV*::'*U set*)| **by** *simp*
show ∃ α ∈ *Q*. α ≤*o* |*s*|
proof (*cases finite* (*UNIV*::'*U set*))
assume *finite* (*UNIV*::'*U set*)
then have *finite s* **using** *c2 finite-subset* **by** *blast*
moreover have *CCR s* **using** *c1* **unfolding** $\mathfrak{U}$-*def* **by** *blast*
ultimately have *Conelike s* **using** *lem-Relprop-fin-ccr* **by** *blast*
then have *d1*: *Conelike r* **using** *c1 lem-uset-cl-ext* **by** *blast*
show ∃ α ∈ *Q*. α ≤*o* |*s*|
proof (*cases r* = {})
assume *e1*: *r* = {}
obtain α **where** *e2*: α = ({}::'*U rel*) **by** *blast*
then have α ∈ $\mathfrak{U}$ *r* **using** *e1* **unfolding** $\mathfrak{U}$-*def CCR-def Field-def* **by** *blast*
moreover have *e3*: α =*o* |({}::'*U rel*)| **using** *e2 lem-card-emprel ordIso-symmetric* **by** *blast*
ultimately have α ∈ *Q* **using** *b2 e2* **by** *blast*
moreover have α ≤*o* |*s*| **using** *e3 card-of-empty ordIso-ordLeq-trans* **by** *blast*
ultimately show ∃ α ∈ *Q*. α ≤*o* |*s*| **by** *blast*
next
assume *e1*: *r* ≠ {}
then obtain *m m'* **where** *e2*: {(*m'*,*m*)} ∈ $\mathfrak{U}$ *r* **using** *d1 lem-uset-cl-singleton* **by** *blast*
obtain α **where** *e3*: α = |{*m*}| **by** *blast*
then have α =*o* |{(*m'*,*m*)}| **by** (*simp add*: *ordIso-iff-ordLeq*)
then have α ∈ *Q* **using** *b2 e2* **by** *blast*
moreover have *s* ≠ {} **using** *c1 e1* **unfolding** $\mathfrak{U}$-*def Field-def* **by** *force*
moreover then have α ≤*o* |*s*| **using** *e3* **by** *simp*
ultimately show ∃ α ∈ *Q*. α ≤*o* |*s*| **by** *blast*
qed
next
assume ¬ *finite* (*UNIV*::'*U set*)
then have |(*UNIV*::'*U set*) × (*UNIV*::'*U set*)| =*o* |*UNIV*::'*U set*| **using** *card-of-Times-same-infinite* **by** *blast*
then have |*s*| ≤*o* |*UNIV*::'*U set*| **using** *c3* **using** *ordLeq-ordIso-trans* **by** *blast*
then obtain *A*::'*U set* **where** |*s*| =*o* |*A*| **using** *internalize-card-of-ordLeq2*

by *fast*
moreover then obtain α::$'U$ *rel* **where** $\alpha = |A|$ **by** *blast*
ultimately have $\alpha \in Q \wedge \alpha =o\ |s|$ **using** *b2 c1 ordIso-symmetric* **by** *blast*
then show $\exists\ \alpha \in Q.\ \alpha \leq o\ |s|$ **using** *ordIso-iff-ordLeq* **by** *blast*
qed
qed
then have $Q \neq \{\}$ **using** *b1* **by** *blast*
then obtain α **where** *b4*: $\alpha \in Q \wedge (\forall \alpha'.\ \alpha' <o\ \alpha \longrightarrow \alpha' \notin Q)$ **using** *wf-ordLess wf-eq-minimal*[*of ordLess*] **by** *blast*
moreover have $\forall\ \alpha' \in Q.$ *Card-order* α' **using** *b2* **using** *ordIso-card-of-imp-Card-order* **by** *blast*
ultimately have $\forall\ \alpha' \in Q.\ \neg\ (\alpha' <o\ \alpha) \longrightarrow \alpha \leq o\ \alpha'$ **by** *simp*
then have *b5*: $\alpha \in Q \wedge (\forall\ \alpha' \in Q.\ \alpha \leq o\ \alpha')$ **using** *b4* **by** *blast*
then obtain s **where** *b6*: $s \in \mathfrak{U}\ r \wedge |s| =o\ \alpha$ **using** *b2 ordIso-symmetric* **by** *blast*
moreover have $\forall\ s' \in \mathfrak{U}\ r.\ |s| \leq o\ |s'|$
proof
fix s'
assume $s' \in \mathfrak{U}\ r$
then obtain α' **where** $\alpha' \in Q \wedge \alpha' \leq o\ |s'|$ **using** *b3* **by** *blast*
moreover then have $|s| =o\ \alpha \wedge \alpha \leq o\ \alpha'$ **using** *b5 b6* **by** *blast*
ultimately show $|s| \leq o\ |s'|$ **using** *ordIso-ordLeq-trans ordLeq-transitive* **by** *blast*
qed
ultimately have *RCC-rel r* α **unfolding** *RCC-rel-def* **by** *blast*
then show $\exists\ \alpha.$ *RCC-rel r* α **by** *blast*
qed
then show *?thesis* **unfolding** *RCC-def* **by** (*metis someI2*)
qed

lemma *lem-rcc-uset-ne*:
assumes $\mathfrak{U}\ r \neq \{\}$
shows $\exists\ s \in \mathfrak{U}\ r.\ |s| =o\ \|r\| \wedge (\ \forall\ s' \in \mathfrak{U}\ r.\ |s| \leq o\ |s'|\ )$
using *assms lem-rcc-rccrel* **unfolding** *RCC-rel-def* **by** *blast*

lemma *lem-rcc-uset-emp*:
assumes $\mathfrak{U}\ r = \{\}$
shows $\|r\| = \{\}$
using *assms lem-rcc-rccrel* **unfolding** *RCC-rel-def* **by** *blast*

lemma *lem-rcc-uset-mem-bnd*:
assumes $s \in \mathfrak{U}\ r$
shows $\|r\| \leq o\ |s|$
proof –
obtain *s0* **where** $s0 \in \mathfrak{U}\ r \wedge |s0| =o\ \|r\| \wedge (\ \forall\ s' \in \mathfrak{U}\ r.\ |s0| \leq o\ |s'|\ )$ **using** *assms lem-rcc-uset-ne* **by** *blast*
moreover then have $|s0| \leq o\ |s|$ **using** *assms* **by** *blast*
ultimately show $\|r\| \leq o\ |s|$ **by** (*metis ordIso-iff-ordLeq ordLeq-transitive*)
qed

lemma *lem-rcc-cardord*: *Card-order* $\|r\|$
proof (*cases* $\mathfrak{U}$ $r = \{\}$)
  assume $\mathfrak{U}$ $r = \{\}$
  then have $\|r\| = \{\}$ **using** *lem-rcc-uset-emp* **by** *blast*
  then show *Card-order* $\|r\|$ **using** *lem-cardord-emp* **by** *simp*
next
  assume $\mathfrak{U}$ $r \neq \{\}$
  then obtain s **where** $s \in \mathfrak{U}\ r \wedge |s| =o\ \|r\|$ **using** *lem-rcc-uset-ne* **by** *blast*
  then show *Card-order* $\|r\|$ **using** *Card-order-ordIso2 card-of-Card-order* **by** *blast*
qed

lemma *lem-uset-ne-rcc-inf*:
fixes r::$'U$ *rel*
assumes $\neg$ ($\|r\|$ $<o$ ω-*ord*)
shows $\mathfrak{U}$ $r \neq \{\}$
proof −
  have $\|r\| = \{\} \longrightarrow \|r\| <o\ |UNIV :: nat\ set|$
   by (*metis card-of-Well-order finite.emptyI infinite-iff-card-of-nat ordIso-ordLeq-trans ordIso-symmetric ordLeq-iff-ordLess-or-ordIso ozero-def ozero-ordLeq*)
  then have $\|r\| = \{\} \longrightarrow \|r\| <o\ \omega$-*ord* **using** *card-of-nat ordLess-ordIso-trans* **by** *blast*
  then show $\mathfrak{U}$ $r \neq \{\}$ **using** *assms lem-rcc-uset-emp* **by** *blast*
qed

lemma *lem-rcc-inf*: (ω-*ord* $\leq o$ $\|r\|$) = ($\neg$ ($\|r\|$ $<o$ ω-*ord*))
  using *lem-rcc-cardord lem-cord-lin* **by** (*metis Field-natLeq natLeq-card-order*)

lemma *lem-Rcc-eq1-12*:
fixes r::$'U$ *rel*
shows *CCR* $r \Longrightarrow r \in \mathfrak{U}\ r$
  unfolding $\mathfrak{U}$-*def CCR-def* **by** *blast*

lemma *lem-Rcc-eq1-23*:
fixes r::$'U$ *rel*
assumes $r \in \mathfrak{U}\ r$
shows (r = ({}::$'U$ *rel*)) $\vee$ (({}::$'U$ *rel*) $<o$ $\|r\|$)
proof −
  obtain *s0* **where** *a2*: $s0 \in \mathfrak{U}\ r$ **and** *a3*: $|s0| =o\ \|r\|$ **using** *assms lem-rcc-uset-ne* **by** *blast*
  have $s0 = \{\} \longrightarrow r = \{\}$ **using** *a2* **unfolding** $\mathfrak{U}$-*def Field-def* **by** *force*
  moreover have $s0 \neq \{\} \longrightarrow$ ({}::$'U$ *rel*) $<o$ $\|r\|$
    using *a3 lem-rcc-cardord lem-cardord-emp*
      by (*metis* (*no-types*, *lifting*) *Card-order-iff-ordIso-card-of Field-empty card-of-empty3 card-order-on-well-order-on not-ordLeq-iff-ordLess ordLeq-iff-ordLess-or-ordIso ordLeq-ordIso-trans ozero-def ozero-ordLeq*)
  ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Rcc-eq1-31*:
fixes $r::'U\ rel$
assumes $(r = (\{\}::'U\ rel)) \vee ((\{\}::'U\ rel) <o \|r\|)$
shows *CCR r*
proof (*cases* $r = \{\}$)
assume $r = \{\}$
then show *CCR r* **unfolding** *CCR-def Field-def* **by** *blast*
next
assume *b1*: $r \neq \{\}$
then have *b2*: $(\{\}::'U\ rel) <o \|r\|$ **using** *assms* **by** *blast*
then have $\|r\| \neq (\{\}::'U\ rel)$ **using** *ordLess-irreflexive* **by** *fastforce*
then have $\mathfrak{U}\ r \neq \{\}$ **using** *lem-rcc-uset-emp* **by** *blast*
then obtain *s* **where** *b3*: $s \in \mathfrak{U}\ r$ **and** *b4*: $|s| =o \|r\|$ **and**
b5: $\forall\ s' \in \mathfrak{U}\ r.\ |s| \leq o\ |s'|$ **using** *lem-rcc-uset-ne* **by** *blast*
have $s \neq \{\}$ **using** *assms b1 b4 lem-card-emprel not-ordLess-ordIso ordIso-ordLess-trans* **by** *blast*
have $s \subseteq r$ **using** *b3* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
then have $Field\ s \subseteq Field\ r \wedge s\hat{\ }* \subseteq r\hat{\ }*$ **unfolding** *Field-def* **using** *rtrancl-mono* **by** *blast*
have $\forall\ a{\in}Field\ r.\ \forall\ b{\in}Field\ r.\ \exists\ c{\in}Field\ r.\ (a,\ c) \in r\hat{\ }* \wedge (b,\ c) \in r\hat{\ }*$
proof (*intro ballI*)
fix *a b*
assume *c1*: $a \in Field\ r$ **and** *c2*: $b \in Field\ r$
then obtain a' b' **where** *c3*: $a' \in Field\ s \wedge b' \in Field\ s \wedge (a,a') \in r\hat{\ }* \wedge (b,b') \in r\hat{\ }*$
using *b3* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
then obtain *c* **where** *c4*: $c \in Field\ s \wedge (a',c) \in s\hat{\ }* \wedge (b',c) \in s\hat{\ }*$ **using** *b3* **unfolding** $\mathfrak{U}$*-def CCR-def* **by** *blast*
have $a' \in Field\ r \wedge b' \in Field\ r \wedge c \in Field\ r$ **using** *b3 c3 c4* **unfolding** $\mathfrak{U}$*-def Field-def* **by** *blast*
moreover have $(a',c) \in r\hat{\ }* \wedge (b',c) \in r\hat{\ }*$ **using** *b3 c4* **unfolding** $\mathfrak{U}$*-def* **using** *rtrancl-mono* **by** *blast*
ultimately have $c \in Field\ r \wedge (a,\ c) \in r\hat{\ }* \wedge (b,\ c) \in r\hat{\ }*$ **using** *c3* **by** *force*
then show $\exists\ c{\in}Field\ r.\ (a,\ c) \in r\hat{\ }* \wedge (b,\ c) \in r\hat{\ }*$ **by** *blast*
qed
then show *CCR r* **unfolding** *CCR-def* **by** *blast*
qed

lemma *lem-Rcc-eq2-12*:
fixes $r::'U\ rel$ **and** $a::'a$
assumes *Conelike r*
shows $\|r\| \leq o\ |\{a\}|$
proof (*cases* $r = \{\}$)
assume $r = \{\}$
then have $\|r\| = \{\}$ **using** *lem-rcc-emp* **by** *blast*
then show $\|r\| \leq o\ |\{a\}|$ **by** (*metis card-of-Well-order ozero-def ozero-ordLeq*)
next
assume $r \neq \{\}$
then obtain *m* **where** *b1*: $m \in Field\ r \wedge (\forall\ a \in Field\ r.\ (a,m) \in r\hat{\ }*)$ **using**

assms **unfolding** *Conelike-def* **by** *blast*
then obtain m' **where** *b2*: $(m,m') \in r \vee (m',m) \in r$ **unfolding** *Field-def* **by** *blast*
then have $(m',m) \in r\hat{}*$ **using** *b1* **by** (*meson FieldI2 r-into-rtrancl*)
then obtain x **where** $(x,m) \in r$ **using** *b2* **by** (*metis rtranclE*)
moreover have *CCR* $\{(x,m)\}$ **unfolding** *CCR-def Field-def* **by** *blast*
ultimately have $\{(x,m)\} \in \mathfrak{U}\ r$ **using** *b1* **unfolding** $\mathfrak{U}$*-def* **by** *simp*
then have $\|r\| \leq o\ |\{(x,m)\}|$ **using** *lem-rcc-uset-mem-bnd* **by** *blast*
moreover have $|\{(x,m)\}| \leq o\ |\{a\}|$ **by** *simp*
ultimately show $\|r\| \leq o\ |\{a\}|$ **using** *ordLeq-transitive* **by** *blast*
qed

lemma *lem-Rcc-eq2-23*:
fixes $r::'U\ rel$ **and** $a::'a$
assumes $\|r\| \leq o\ |\{a\}|$
shows $\|r\| <o\ \omega\text{-}ord$
proof −
have $|\{a\}| <o\ |UNIV :: nat\ set|$ **using** *finite-iff-cardOf-nat* **by** *blast*
then show $\|r\| <o\ \omega\text{-}ord$ **using** *assms ordLeq-ordLess-trans card-of-nat ord-Less-ordIso-trans* **by** *blast*
qed

lemma *lem-Rcc-eq2-31*:
fixes $r::'U\ rel$
assumes *CCR r* **and** $\|r\| <o\ \omega\text{-}ord$
shows *Conelike r*
proof −
have $r \in \mathfrak{U}\ r$ **using** *assms lem-Rcc-eq1-12* **by** *blast*
then obtain s **where** *b1*: $s \in \mathfrak{U}\ r$ **and** *b2*: $|s| =o\ \|r\|$ **using** *lem-rcc-uset-ne* **by** *blast*
have $|s| <o\ \omega\text{-}ord$ **using** *assms b2* **using** *ordIso-imp-ordLeq ordLeq-ordLess-trans* **by** *blast*
then have *finite s* **using** *finite-iff-ordLess-natLeq* **by** *blast*
moreover have *CCR s* **using** *b1* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
ultimately have *Conelike s* **using** *lem-Relprop-fin-ccr* **by** *blast*
then show *Conelike r* **using** *b1 lem-uset-cl-ext* **by** *blast*
qed

lemma *lem-Rcc-range*:
fixes $r::'U\ rel$
shows $\|r\| \leq o\ |UNIV::('U\ set)|$
by (*simp add*: *lem-rcc-cardord*)

lemma *lem-rcc-nccr*:
fixes $r::'U\ rel$
assumes $\neg$ (*CCR r*)
shows $\|r\| = \{\}$
proof −
have $\neg$ $((\{\}::'U\ rel) <o\ \|r\|)$ **using** *assms lem-Rcc-eq1-31*[*of r*] **by** *blast*

moreover have *Well-order* ({}::′*U rel*) **using** *Well-order-empty* **by** *blast*
moreover have *Well-order* $\|r\|$ **using** *lem-rcc-cardord* **unfolding** *card-order-on-def* **by** *blast*
ultimately have $\|r\| \leq o$ ({}::′*U rel*) **by** *simp*
then show $\|r\| = \{\}$ **using** *lem-ord-subemp* **by** *blast*
qed

lemma *lem-Rcc-relcard-bnd*:
fixes *r*::′*U rel*
shows $\|r\| \leq o\ |r|$
proof(*cases CCR r*)
assume *CCR r*
then show $\|r\| \leq o\ |r|$ **using** *lem-Rcc-eq1-12 lem-rcc-uset-mem-bnd* **by** *blast*
next
assume ¬ *CCR r*
then have $\|r\| = \{\}$ **using** *lem-rcc-nccr* **by** *blast*
then have $\|r\| \leq o$ ({}::′*U rel*) **by** (*metis card-of-empty ordLeq-Well-order-simp ozero-def ozero-ordLeq*)
moreover have ({}::′*U rel*) $\leq o\ |r|$ **by** (*metis card-of-Well-order ozero-def ozero-ordLeq*)
ultimately show $\|r\| \leq o\ |r|$ **using** *ordLeq-transitive* **by** *blast*
qed

lemma *lem-Rcc-inf-lim*:
fixes *r*::′*U rel*
assumes ω-*ord* $\leq o\ \|r\|$
shows ¬($\|r\| = \{\}$ ∨ *isSuccOrd* $\|r\|$)
using *assms lem-card-inf-lim lem-rcc-cardord* **by** *blast*

lemma *lem-rcc-uset-ne-ccr*:
fixes *r*::′*U rel*
assumes $\mathfrak{U}\ r \neq \{\}$
shows *CCR r*
proof −
obtain *s* **where** *b1*: $s \in \mathfrak{U}\ r$ **using** *assms* **by** *blast*
have ∀ *a*∈*Field r*. ∀ *b*∈*Field r*. ∃ *c*∈*Field r*. (*a*, *c*) ∈ *r*^∗ ∧ (*b*, *c*) ∈ *r*^∗
proof (*intro ballI impI*)
fix *a b*
assume *a*∈*Field r* **and** *b*∈*Field r*
then obtain *a′ b′* **where** *c1*: *a′* ∈ *Field s* ∧ *b′* ∈ *Field s* ∧ (*a*,*a′*) ∈ *r*^∗ ∧ (*b*,*b′*) ∈ *r*^∗
using *b1* **unfolding** $\mathfrak{U}$-*def* **by** *blast*
then obtain *c* **where** *c* ∈ *Field s* ∧ (*a′*,*c*) ∈ *s*^∗ ∧ (*b′*,*c*) ∈ *s*^∗ **using** *b1* **unfolding** $\mathfrak{U}$-*def CCR-def* **by** *blast*
moreover have *s* ⊆ *r* **using** *b1* **unfolding** $\mathfrak{U}$-*def* **by** *blast*
ultimately have *c* ∈ *Field r* ∧ (*a′*,*c*) ∈ *r*^∗ ∧ (*b′*,*c*) ∈ *r*^∗ **using** *rtrancl-mono* **unfolding** *Field-def* **by** *blast*
moreover then have (*a*,*c*) ∈ *r*^∗ ∧ (*b*,*c*) ∈ *r*^∗ **using** *c1* **by** *force*
ultimately show ∃ *c*∈*Field r*. (*a*, *c*) ∈ *r*^∗ ∧ (*b*, *c*) ∈ *r*^∗ **by** *blast*
qed

then show *?thesis* **unfolding** *CCR-def* **by** *blast*
qed

lemma *lem-rcc-uset-tr*:
fixes *r s t*::*'U rel*
assumes *a1*: $s \in \mathfrak{U}\ r$ **and** *a2*: $t \in \mathfrak{U}\ s$
shows $t \in \mathfrak{U}\ r$
proof −
have $\forall\, a{\in}Field\ r.\ \exists\, b{\in}Field\ t.\ (a,\ b) \in r\hat{}*$
proof
fix *a*
assume $a \in Field\ r$
then obtain b' **where** $b' \in Field\ s \wedge (a,b') \in r\hat{}*$ **using** *a1* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
moreover then obtain *b* **where** $b \in Field\ t \wedge (b',b) \in s\hat{}*$ **using** *a2* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
moreover have $s \subseteq r$ **using** *a1* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
ultimately have $b \in Field\ t \wedge (a,b') \in r\hat{}* \wedge (b',b) \in r\hat{}*$ **using** *rtrancl-mono* **by** *blast*
then have $b \in Field\ t \wedge (a,b) \in r\hat{}*$ **by** *force*
then show $\exists\, b{\in}Field\ t.\ (a,\ b) \in r\hat{}*$ **by** *blast*
qed
then show *?thesis* **using** *a1 a2* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
qed

lemma *lem-scf-emp*: *scf* $\{\} = \{\}$
unfolding *scf-def scf-rel-def SCF-def* **apply** *simp*
using *lem-card-emprel* **by** (*smt card-of-empty-ordIso iso-ozero-empty ordIso-symmetric ozero-def someI-ex*)

lemma *lem-scf-scfrel*:
fixes *r*::*'U rel*
shows *scf-rel r* (*scf r*)
proof −
have *b1*: $SCF\ r \neq \{\}$ **unfolding** *SCF-def* **by** *blast*
obtain *Q* **where** *b2*: $Q = \{\ \alpha{::}'U\ rel.\ \exists\ A \in SCF\ r.\ \alpha =o\ |A|\ \}$ **by** *blast*
have *b3*: $\forall\ A \in SCF\ r.\ \exists\ \alpha \in Q.\ \alpha \leq o\ |A|$
proof
fix *A*
assume $A \in SCF\ r$
then have $|A| \in Q \wedge |A| =o\ |A|$ **using** *b2 ordIso-symmetric* **by** *force*
then show $\exists\ \alpha \in Q.\ \alpha \leq o\ |A|$ **using** *ordIso-iff-ordLeq* **by** *blast*
qed
then have $Q \neq \{\}$ **using** *b1* **by** *blast*
then obtain α **where** *b4*: $\alpha \in Q \wedge (\forall \alpha'.\ \alpha' <o\ \alpha \longrightarrow \alpha' \notin Q)$ **using** *wf-ordLess wf-eq-minimal*[*of ordLess*] **by** *blast*
moreover have $\forall\ \alpha' \in Q.$ *Card-order* α' **using** *b2* **using** *ordIso-card-of-imp-Card-order* **by** *blast*
ultimately have $\forall\ \alpha' \in Q.\ \neg\ (\alpha' <o\ \alpha) \longrightarrow \alpha \leq o\ \alpha'$ **by** *simp*

then have *b5*: $\alpha \in Q \wedge (\forall\ \alpha' \in Q.\ \alpha \le o\ \alpha')$ **using** *b4* **by** *blast*
then obtain *A* **where** *b6*: $A \in SCF\ r \wedge |A| =o\ \alpha$ **using** *b2 ordIso-symmetric* **by** *blast*
moreover have $\forall\ B \in SCF\ r.\ |A| \le o\ |B|$
proof
fix *B*
assume $B \in SCF\ r$
then obtain α' **where** $\alpha' \in Q \wedge \alpha' \le o\ |B|$ **using** *b3* **by** *blast*
moreover then have $|A| =o\ \alpha \wedge \alpha \le o\ \alpha'$ **using** *b5 b6* **by** *blast*
ultimately show $|A| \le o\ |B|$ **using** *ordIso-ordLeq-trans ordLeq-transitive* **by** *blast*
qed
ultimately have *scf-rel r* α **unfolding** *scf-rel-def* **by** *blast*
then show *?thesis* **unfolding** *scf-def* **by** (*metis someI2*)
qed

lemma *lem-scf-uset*:
shows $\exists\ A \in SCF\ r.\ |A| =o\ scf\ r \wedge (\ \forall\ B \in SCF\ r.\ |A| \le o\ |B|\ )$
using *lem-scf-scfrel* **unfolding** *scf-rel-def* **by** *blast*

lemma *lem-scf-uset-mem-bnd*:
assumes $B \in SCF\ r$
shows *scf r* $\le o\ |B|$
proof −
obtain *A* **where** $A \in SCF\ r \wedge |A| =o\ scf\ r \wedge (\ \forall\ A' \in SCF\ r.\ |A| \le o\ |A'|\ )$ **using** *assms lem-scf-uset* **by** *blast*
moreover then have $|A| \le o\ |B|$ **using** *assms* **by** *blast*
ultimately show *?thesis* **by** (*metis ordIso-iff-ordLeq ordLeq-transitive*)
qed

lemma *lem-scf-cardord*: *Card-order* (*scf r*)
proof −
obtain *A* **where** $A \in SCF\ r \wedge |A| =o\ scf\ r$ **using** *lem-scf-uset* **by** *blast*
then show *Card-order* (*scf r*) **using** *Card-order-ordIso2 card-of-Card-order* **by** *blast*
qed

lemma *lem-scf-inf*: $(\ \omega\text{-}ord \le o\ (scf\ r)\ ) = (\ \neg\ (\ (scf\ r) < o\ \omega\text{-}ord\ )\ )$
using *lem-scf-cardord lem-cord-lin* **by** (*metis Field-natLeq natLeq-card-order*)

lemma *lem-scf-eq1-12*:
fixes *r*::*'U rel*
shows *Field r* $\in SCF\ r$
unfolding *SCF-def* **by** *blast*

lemma *lem-scf-range*:
fixes *r*::*'U rel*
shows (*scf r*) $\le o\ |UNIV::('U\ set)|$
by (*simp add*: *lem-scf-cardord*)

```
lemma lem-scf-relfldcard-bnd:
fixes r::'U rel
shows (scf r) ≤o |Field r|
  using lem-scf-eq1-12 lem-scf-uset-mem-bnd by blast

lemma lem-scf-ccr-scf-rcc-eq:
fixes r::'U rel
assumes CCR r
shows ‖r‖ =o (scf r)
proof −
   obtain B where b1: B ∈ SCF r ∧ |B| =o scf r using lem-scf-scfrel[of r] unfolding scf-rel-def by blast
  have B ⊆ Field r using b1 unfolding SCF-def by blast
  then obtain A where  b2: B ⊆ A ∧ A ∈ SF r
                  and b3: (finite B ⟶ finite A) ∧ ((¬ finite B) ⟶ |A| =o |B| )
                 using lem-inv-sf-ext[of B r] by blast
  then obtain A' where b4: A ⊆ A' ∧ A' ∈ SF r ∧ CCR (Restr r A')
                  and b5: (finite A ⟶ finite A') ∧ ((¬ finite A) ⟶ |A'| =o |A| )
    using assms lem-Ccext-subccr-pext5[of r A - {}] by metis
  have Restr r A' ∈ 𝔘 r
  proof −
    have ∀ a∈Field r. ∃ b∈Field (Restr r A'). (a, b) ∈ r^*
    proof
      fix a
      assume a ∈ Field r
     then obtain b where b ∈ B ∧ (a,b) ∈ r^* using b1 unfolding SCF-def by blast
      moreover then have b ∈ Field (Restr r A') using b2 b4 unfolding SF-def by blast
      ultimately show ∃ b∈Field (Restr r A'). (a, b) ∈ r^* by blast
    qed
    then show Restr r A' ∈ 𝔘 r unfolding 𝔘-def using b4 by blast
  qed
  then have b6: ‖r‖ ≤o |Restr r A'| using lem-rcc-uset-mem-bnd by blast
  obtain x0::'U where True by blast
  have b7: ‖r‖ ≤o (scf r)
  proof (cases finite B)
    assume finite B
    then have finite (Restr r A') using b3 b5 by blast
    then have Conelike r
       using assms b6 lem-Rcc-eq2-31[of r] finite-iff-ordLess-natLeq[of Restr r A'] ordLeq-ordLess-trans by blast
    then have c1: ‖r‖ ≤o |{x0}| using lem-Rcc-eq2-12[of r x0] by blast
    show ?thesis
    proof (cases r = {})
      assume r = {}
      then have scf r = {} ∧ ‖r‖ = {} using lem-scf-emp lem-rcc-emp by blast
       then show ‖r‖ ≤o (scf r) using b1 lem-ord-subemp ordIso-iff-ordLeq by
```

metis
next
assume $r \neq \{\}$
then have $B \neq \{\}$ **using** *b1* **unfolding** *SCF-def Field-def* **by** *force*
then have $|\{x0\}| \leq o\ |B|$ **using** *card-of-singl-ordLeq* **by** *metis*
then show *?thesis* **using** *c1 b1 ordLeq-transitive ordIso-imp-ordLeq* **by** *metis*
qed
next
assume *c1*: $\neg$ *finite B*
then have $|A| =o\ |B| \wedge |A'| =o\ |A|$ **using** *b3 b5 finite-subset* **by** *simp*
then have $|A'| =o$ *scf r* **using** *b1* **using** *ordIso-transitive* **by** *blast*
moreover have ω*-ord* $\leq o$ *scf r* **using** *c1 b1 infinite-iff-natLeq-ordLeq ordLeq-ordIso-trans* **by** *blast*
ultimately have $|Restr\ r\ A'| \leq o$ *scf r* **using** *lem-restr-ordbnd*[*of scf r A′ r*] *ordIso-imp-ordLeq* **by** *blast*
then show $\|r\| \leq o$ (*scf r*) **using** *b6 ordLeq-transitive* **by** *blast*
qed
moreover have (*scf r*) $\leq o\ \|r\|$
proof −
obtain *s* **where** *b1*: $s \in \mathfrak{U}\ r \wedge |s| =o\ \|r\| \wedge (\forall s' \in \mathfrak{U}\ r.\ |s| \leq o\ |s'|\ )$
using *assms lem-Rcc-eq1-12*[*of r*] *lem-rcc-uset-ne*[*of r*] **by** *blast*
then have *Field s* $\subseteq$ *Field r* $\wedge$ ($\forall$ *a*$\in$*Field r*. $\exists$ *b*$\in$*Field s*. (*a*, *b*) $\in r\hat{}*$)
unfolding $\mathfrak{U}$*-def Field-def* **by** *blast*
then have *Field s* $\in$ *SCF r* **unfolding** *SCF-def* **by** *blast*
then have *b2*: *scf r* $\leq o$ |*Field s*| **using** *lem-scf-uset-mem-bnd* **by** *blast*
show *?thesis*
proof (*cases finite s*)
assume *finite s*
then have $\|r\| <o\ \omega$*-ord*
using *b1 finite-iff-ordLess-natLeq not-ordLeq-ordLess ordIso-iff-ordLeq ordIso-transitive ordLeq-iff-ordLess-or-ordIso ordLeq-transitive* **by** *metis*
then have *c1*: *Conelike r* **using** *assms lem-Rcc-eq2-31* **by** *blast*
show *?thesis*
proof (*cases* $r = \{\}$)
assume $r = \{\}$
then have *scf r* $= \{\} \wedge \|r\| = \{\}$ **using** *lem-scf-emp lem-rcc-emp* **by** *blast*
then show *?thesis* **using** *b7* **by** *simp*
next
assume *d1*: $r \neq \{\}$
then obtain *m* **where** $m \in$ *Field r* $\wedge$ ($\forall$ $a \in$ *Field r*. $(a,m) \in r\hat{}*$) **using** *c1* **unfolding** *Conelike-def* **by** *blast*
then have $\{m\} \in$ *SCF r* **unfolding** *SCF-def* **by** *blast*
then have *d2*: *scf r* $\leq o\ |\{m\}|$ **using** *lem-scf-uset-mem-bnd* **by** *blast*
have ($\{\}$::′*U rel*) $<o\ \|r\|$ **using** *d1 assms lem-Rcc-eq1-23 lem-Rcc-eq1-12* **by** *blast*
then have $|\{m\}| \leq o\ \|r\|$ **using** *lem-co-one-ne-min* **by** (*metis card-of-empty3 card-of-empty4 insert-not-empty ordLess-Well-order-simp*)
then show *?thesis* **using** *d2 ordLeq-transitive* **by** *blast*
qed

next
assume ¬ *finite s*
then have |*Field s*| =*o* |*s*| **using** *lem-rel-inf-fld-card* **by** *blast*
then show *?thesis* **using** *b1 b2 ordIso-iff-ordLeq ordLeq-transitive* **by** *metis*
qed
qed
ultimately show *?thesis* **using** *not-ordLeq-ordLess ordLeq-iff-ordLess-or-ordIso* **by** *blast*
qed

lemma *lem-scf-ccr-scf-uset*:
fixes *r*::*'U rel*
assumes *CCR r* **and** ¬ *Conelike r*
shows ∃ *s* ∈ $\mathfrak{U}$ *r*. (¬ *finite s*) ∧ |*Field s*| =*o* (*scf r*)
proof −
have ‖*r*‖ =*o* (*scf r*) **using** *assms lem-scf-ccr-scf-rcc-eq* **by** *blast*
moreover then obtain *s* **where** *b1*: *s* ∈ $\mathfrak{U}$ *r* ∧ |*s*| =*o* ‖*r*‖ **using** *assms lem-Rcc-eq1-12 lem-rcc-uset-ne*[*of r*] **by** *blast*
moreover have (¬ *finite s*) ⟶ |*Field s*| =*o* |*s*| **using** *lem-rel-inf-fld-card* **by** *blast*
moreover have *finite s* ⟶ *False*
proof
assume *finite s*
then have |*s*| <*o* ω*-ord* **using** *finite-iff-ordLess-natLeq* **by** *blast*
then have ‖*r*‖ <*o* ω*-ord* **using** *b1*
by (*meson not-ordLess-ordIso ordIso-iff-ordLeq ordIso-transitive ordLeq-iff-ordLess-or-ordIso ordLeq-transitive*)
then show *False* **using** *assms lem-Rcc-eq2-31* **by** *blast*
qed
ultimately show *?thesis* **using** *ordIso-transitive* **by** *metis*
qed

lemma *lem-Scf-scfprops*:
fixes *r*::*'U rel*
shows ((*scf r*) ≤*o* |*UNIV*::(*'U set*)|) ∧ ((*scf r*) ≤*o* |*Field r*|)
using *lem-scf-range lem-scf-relfldcard-bnd* **by** *blast*

lemma *lem-scf-ccr-finscf-cl*:
assumes *CCR r*
shows *finite* (*Field* (*scf r*)) = *Conelike r*
proof
assume *finite* (*Field* (*scf r*))
then have *finite* ‖*r*‖ **using** *assms lem-scf-ccr-scf-rcc-eq lem-fin-fl-rel ordIso-finite-Field* **by** *blast*
then have ‖*r*‖ <*o* ω*-ord* **using** *lem-rcc-cardord lem-fin-fl-rel*
by (*metis card-of-Field-ordIso finite-iff-ordLess-natLeq ordIso-iff-ordLeq ordLeq-ordLess-trans*)
then show *Conelike r* **using** *assms lem-Rcc-eq2-31* **by** *blast*
next

assume *Conelike r*
then have *finite* (*Field* $\|r\|$) **using** *lem-Rcc-eq2-12*[*of r*] **by** (*metis Field-card-of finite.emptyI finite-insert ordLeq-finite-Field*)
then show *finite* (*Field* (*scf r*)) **using** *assms lem-scf-ccr-scf-rcc-eq ordIso-finite-Field* **by** *blast*
qed

lemma *lem-sv-uset-sv-span*:
fixes *r s*::*'U rel*
assumes *a1*: $s \in \mathfrak{U}\ r$ **and** *a2*: *single-valued s*
shows $\exists$ *r1*. $r1 \in$ *Span r* $\wedge$ *CCR r1* $\wedge$ *single-valued r1* $\wedge$ $s \subseteq r1$ $\wedge$ (*acyclic s* $\longrightarrow$ *acyclic r1*)
proof −
have *b0*: $s \subseteq r$ **using** *a1* **unfolding** $\mathfrak{U}$-*def* **by** *blast*
obtain *isd* **where** *b3*: *isd* = (λ *a i*. $\exists$ *b* $\in$ *Field s*. (*a*, *b*) $\in$ $r^{\frown\frown}i$ $\wedge$ ($\forall$ *i'*. ($\exists$ *b* $\in$ *Field s*. (*a*, *b*) $\in$ $r^{\frown\frown}(i')$) $\longrightarrow$ $i \leq i'$)) **by** *blast*
obtain *d* **where** *b4*: *d* = (λ *a*. *SOME i*. *isd a i*) **by** *blast*
obtain *B* **where** *b5*: *B* = (λ *a*. { *a'*. (*a*, *a'*) $\in$ *r* }) **by** *blast*
obtain *H* **where** *b6*: *H* = (λ *a*. { $a' \in B\ a$. $\forall$ $a'' \in B\ a$. (*d a'*) $\leq$ (*d a''*) }) **by** *blast*
obtain *D* **where** *b7*: *D* = { *a* $\in$ *Field r* − *Field s*. *H a* $\neq$ {}} **by** *blast*
obtain *h* **where** *h* = (λ *a*. *SOME a'*. $a' \in H\ a$) **by** *blast*
then have *b8*: $\forall$ *a* $\in$ *D*. *h a* $\in$ *H a* **using** *b7 someI-ex*[*of* λ *a'*. $a' \in H$ -] **by** *force*
have *q1*: $\bigwedge$ *a*. *a* $\in$ *Field r* $\Longrightarrow$ *isd a* (*d a*)
proof −
fix *a*
assume *c1*: *a* $\in$ *Field r*
then obtain *b* **where** *c2*: *b* $\in$ *Field s* $\wedge$ (*a*,*b*) $\in$ $r^{\frown}*$ **using** *a1* **unfolding** $\mathfrak{U}$-*def* **by** *blast*
moreover obtain *N* **where** *c3*: *N* = {*i*. $\exists$ *b* $\in$ *Field s*. (*a*, *b*) $\in$ $r^{\frown\frown}i$} **by** *blast*
ultimately have *N* $\neq$ {} **using** *rtrancl-imp-relpow* **by** *blast*
then obtain *m* **where** *m* $\in$ *N* $\wedge$ ($\forall$ *i* $\in$ *N*. $m \leq i$)
using *LeastI*[*of* λ *x*. *x* $\in$ *N*] *Least-le*[*of* λ *x*. *x* $\in$ *N*] **by** *blast*
then have *isd a m* **using** *c2 c3* **unfolding** *b3* **by** *blast*
then show *isd a* (*d a*) **using** *b4 someI-ex* **by** *metis*
qed
have *q2*: $\bigwedge$ *a*. *B a* $\neq$ {} $\Longrightarrow$ *H a* $\neq$ {}
proof −
fix *a*
assume *B a* $\neq$ {}
moreover obtain *N* **where** *c1*: *N* = *d* ` (*B a*) **by** *blast*
ultimately have *N* $\neq$ {} **by** *blast*
then obtain *m* **where** *c2*: *m* $\in$ *N* $\wedge$ ($\forall$ *i* $\in$ *N*. $m \leq i$)
using *LeastI*[*of* λ *x*. *x* $\in$ *N*] *Least-le*[*of* λ *x*. *x* $\in$ *N*] **by** *blast*
then obtain *a'* **where** *c3*: *m* = *d a'* $\wedge$ $a' \in B\ a$ **using** *c1* **by** *blast*
moreover then have $\forall$ $a'' \in B\ a$. *d a'* $\leq$ *d a''* **using** *c1 c2* **by** *force*
ultimately have $a' \in H\ a$ **unfolding** *b6* **by** *blast*
then show *H a* $\neq$ {} **by** *blast*

qed
have *q3*: ∀ *a* ∈ *Field r* − *Field s*. *d a* = *1* ∨ *d a* > *1*
proof
fix *a*
assume *c1*: *a* ∈ *Field r* − *Field s*
then have *isd a* (*d a*) **using** *q1* **by** *blast*
then obtain *b* **where** *b* ∈ *Field s* ∧ (*a*, *b*) ∈ *r*^^(*d a*) **using** *b3* **by** *blast*
then have *d a* = *0* ⟶ *False* **using** *c1* **by** *force*
then show *d a* = *1* ∨ *d a* > *1* **by** *force*
qed
have *Field r* − *Field s* ⊆ *D*
proof
fix *a*
assume *c1*: *a* ∈ *Field r* − *Field s*
moreover have *H a* = {} ⟶ *False*
proof
assume *H a* = {}
then have *B a* = {} **using** *q2* **by** *blast*
moreover obtain *b* **where** *b* ∈ *Field s* ∧ (*a*, *b*) ∈ *r*^* **using** *a1 c1* **unfolding** 𝔘*-def* **by** *blast*
ultimately have *a* ∈ *Field s* **unfolding** *b5* **by** (*metis Collect-empty-eq converse-rtranclE*)
then show *False* **using** *c1* **by** *blast*
qed
ultimately show *a* ∈ *D* **using** *b7* **by** *blast*
qed
then have *q4*: *D* = *Field r* − *Field s* **using** *b5 b6 b7* **by** *blast*
have *q5*: ∀ *a* ∈ *D*. *d a* > *1* ⟶ *d a* = *Suc* (*d* (*h a*)) ∧ (*d* (*h a*) > *1* ⟶ *h a* ∈ *D*)
proof (*intro ballI impI*)
fix *a*
assume *c1*: *a* ∈ *D* **and** *c2*: *d a* > *1*
then obtain *b* **where** *c3*: *b* ∈ *Field s* **and** *c4*: (*a*, *b*) ∈ *r*^^(*d a*)
and *c5*: ∀ *i'*. (∃ *b* ∈ *Field s*. (*a*, *b*) ∈ *r*^^(*i'*)) ⟶ (*d a*) ≤ *i'*
using *b3 b7 q1* **by** *blast*
have *c6*: *d a* ≥ *1* **using** *c1 c4 b7 q3* **by** *force*
then have *d a* = *Suc* ((*d a*) − *1*) **by** *simp*
then obtain *a'* **where** *c7*: (*a*,*a'*) ∈ *r* ∧ (*a'*,*b*) ∈ *r*^^((*d a*) − *1*)
using *c4 relpow-Suc-D2*[*of a b d a* − *1 r*] **by** *metis*
moreover then have *a'* ∉ *Field s* **using** *c2 c5* **by** (*metis less-Suc-eq-le not-less-eq relpow-1*)
ultimately have (*a*,*a'*) ∈ *r* ∧ *a'* ∈ *Field r* − *Field s* **unfolding** *Field-def* **by** *blast*
then have *a'* ∈ *B a* **unfolding** *b5* **by** *blast*
moreover have *h a* ∈ *H a* **using** *c1 b8* **by** *blast*
ultimately have *d* (*h a*) ≤ *d a'* **unfolding** *b6* **by** *blast*
moreover have *Suc* (*d a'*) ≤ *d a*
proof −
have *d a'* ≤ *d a* − *1* **using** *q1 b3 c7 c3* **unfolding** *Field-def* **by** *blast*

then show *?thesis* **using** *c6* **by** *force*
qed
moreover have *d a* ≤ (*Suc* (*d* (*h a*)))
proof −
have *d1*: (*a*, *h a*) ∈ *r* **using** *c1 b5 b6 b8* **by** *blast*
then have *h a* ∈ *Field r* **unfolding** *Field-def* **by** *blast*
then obtain *b*′ **where** *b*′ ∈ *Field s* ∧ ((*h a*), *b*′) ∈ *r*^^(*d* (*h a*)) **using** *b3 q1* **by** *blast*
moreover then have (*a*,*b*′) ∈ *r*^^(*Suc* (*d* (*h a*))) **using** *d1 c7* **by** (*meson relpow-Suc-I2*)
ultimately show *d a* ≤ (*Suc* (*d* (*h a*))) **using** *c5* **by** *blast*
qed
ultimately have *d a* = *Suc* (*d* (*h a*)) **by** *force*
moreover have *d* (*h a*) > *1* ⟶ *h a* ∈ *D*
proof
assume *d1*: *d* (*h a*) > *1*
then have *d2*: (*a*, *h a*) ∈ *r* **using** *c1 b5 b6 b8* **by** *simp*
then have *isd* (*h a*) (*d* (*h a*)) **using** *d1 q1* **unfolding** *Field-def* **by** *force*
then have (*h a*) ∉ *Field s* **using** *d1 b3* **by** *force*
then show *h a* ∈ *D* **using** *d2 q4* **unfolding** *Field-def* **by** *blast*
qed
ultimately show *d a* = *Suc* (*d* (*h a*)) ∧ (*d* (*h a*) > *1* ⟶ *h a* ∈ *D*) **by** *blast*
qed
obtain *g1* **where** *b9*: *g1* = { (*a*, *b*). *a* ∈ *D* ∧ *b* = *h a* } **by** *blast*
have *q6*: ∀ *a* ∈ *D*. ∃ *a*′ ∈ *D*. *d a*′ = *1* ∧ (*a*,*a*′) ∈ *g1*^*
proof −
have ∀ *n*. ∀ *a* ∈ *D*. *d a* = *Suc n* ⟶ ((*h*^^*n*) *a*) ∈ *D* ∧ *d* ((*h*^^*n*) *a*) = *1*
proof
fix *n0*
show ∀ *a* ∈ *D*. *d a* = *Suc n0* ⟶ ((*h*^^*n0*) *a*) ∈ *D* ∧ *d* ((*h*^^*n0*) *a*) = *1*
proof (*induct n0*)
show ∀ *a*∈*D*. *d a* = *Suc 0* ⟶ ((*h*^^*0*) *a*) ∈ *D* ∧ *d* ((*h* ^^ *0*) *a*) = *1*
using *q4* **by** *force*
next
fix *n*
assume *d1*: ∀ *a*∈*D*. *d a* = *Suc n* ⟶ ((*h*^^*n*) *a*) ∈ *D* ∧ *d* ((*h* ^^ *n*) *a*) = *1*
show ∀ *a*∈*D*. *d a* = *Suc* (*Suc n*) ⟶ ((*h*^^(*Suc n*)) *a*) ∈ *D* ∧ *d* ((*h* ^^ *Suc n*) *a*) = *1*
proof (*intro ballI impI*)
fix *a*
assume *e1*: *a* ∈ *D* **and** *e2*: *d a* = *Suc* (*Suc n*)
then have *d a* = *Suc* (*d* (*h a*)) ∧ (*d* (*h a*) > *1* ⟶ *h a* ∈ *D*) **using** *q5* **by** *simp*
moreover then have *e3*: *d* (*h a*) = *Suc n* **using** *e2* **by** *simp*
ultimately have *d* (*h a*) > *1* ⟶ ((*h*^^*n*) (*h a*)) ∈ *D* ∧ *d* ((*h*^^*n*) (*h a*)) = *1* **using** *d1* **by** *blast*
moreover have (*h*^^*n*) (*h a*) = (*h*^^(*Suc n*)) *a* **by** (*metis comp-apply funpow-Suc-right*)
moreover have *e4*: *d* (*h a*) = *1* ⟶ *d* ((*h*^^(*Suc n*)) *a*) = *1* **using** *e3*

by *simp*
moreover have *d* (*h a*) = *1* ⟶ ((*h*^^(*Suc n*)) *a*) ∈ *D*
proof
assume *f1*: *d* (*h a*) = *1*
then have *f2*: *n* = *0* ∧ (*a*, *h a*) ∈ *r* **using** *e1 e3 b5 b6 b8* **by** *simp*
then have *isd* (*h a*) *1* **using** *f1 q1* **unfolding** *Field-def* **by** *force*
then have (*h a*) ∉ *Field s* **using** *b3* **by** *force*
then have (*h a*) ∈ *D* **using** *q4 f2* **unfolding** *Field-def* **by** *blast*
then show ((*h*^^(*Suc n*)) *a*) ∈ *D* **using** *f2* **by** *simp*
qed
moreover have *d* (*h a*) = *1* ∨ *d* (*h a*) > *1* **using** *e3* **by** *force*
ultimately show ((*h*^^(*Suc n*)) *a*) ∈ *D* ∧ *d* ((*h* ^^ (*Suc n*)) *a*) = *1* **by** *force*
qed
qed
qed
moreover have ∀ *i*. ∀ *a* ∈ *D*. *d a* > *i* ⟶ (*a*, (*h*^^*i*) *a*) ∈ *g1*^*
proof
fix *i0*
show ∀ *a* ∈ *D*. *d a* > *i0* ⟶ (*a*, (*h*^^*i0*) *a*) ∈ *g1*^*
proof (*induct i0*)
show ∀ *a*∈*D*. *d a* > *0* ⟶ (*a*, (*h*^^*0*) *a*) ∈ *g1*^* **by** *force*
next
fix *i*
assume *d1*: ∀ *a*∈*D*. *d a* > *i* ⟶ (*a*, (*h*^^*i*) *a*) ∈ *g1*^*
show ∀ *a*∈*D*. *d a* > (*Suc i*) ⟶ (*a*, (*h*^^(*Suc i*)) *a*) ∈ *g1*^*
proof (*intro ballI impI*)
fix *a*
assume *e1*: *a* ∈ *D* **and** *e2*: *d a* > (*Suc i*)
then have *e3*: *d a* = *Suc* (*d* (*h a*)) ∧ (*d* (*h a*) > *1* ⟶ *h a* ∈ *D*) **using** *q5* **by** *simp*
moreover then have *e4*: *d* (*h a*) > *i* **using** *e2* **by** *simp*
ultimately have *d* (*h a*) > *1* ⟶ (*h a*, (*h*^^*i*) (*h a*)) ∈ *g1*^* **using** *d1* **by** *simp*
moreover have (*h*^^*i*) (*h a*) = (*h*^^(*Suc i*)) *a* **by** (*metis comp-apply funpow-Suc-right*)
moreover have *d* (*h a*) = *1* ⟶ (*h*^^(*Suc i*)) *a* = (*h a*) **using** *e4* **by** *force*
moreover have *d* (*h a*) = *1* ∨ *d* (*h a*) > *1* **using** *e4* **by** *force*
moreover then have (*a*, *h a*) ∈ *g1* **using** *e1 e3* **unfolding** *b9* **by** *simp*
ultimately show (*a*, (*h*^^(*Suc i*)) *a*) ∈ *g1*^*
by (*metis converse-rtrancl-into-rtrancl r-into-rtrancl*)
qed
qed
qed
ultimately have ∀ *n*. ∀ *a*∈*D*. *d a* = *Suc n* ⟶ (*h*^^*n*) *a* ∈ *D* ∧ *d* ((*h*^^*n*) *a*) = *1* ∧ (*a*, (*h* ^^ *n*) *a*) ∈ *g1*^*
by *simp*
then have ∀ *n*. ∀ *a*∈*D*. *d a* = *Suc n* ⟶ (∃ *a'* ∈ *D*. *d a'* = *1* ∧ (*a*,*a'*) ∈ *g1*^*)

by *blast*
moreover have ∀ a ∈ D. ∃ n. d a = Suc n **using** *q3 q4 q5* **by** *force*
ultimately show *?thesis* **by** *blast*
qed
obtain *r1* **where** *b19*: r1 = s ∪ g1 **by** *blast*
have *t1*: g1 ⊆ r1 **using** *b19* **by** *blast*
have *b20*: s ⊆ r1 **using** *b19* **by** *blast*
have *b21*: r1 ⊆ r
proof −
have ∀ a ∈ D. (a, h a) ∈ r **using** *b5 b6 b8* **by** *blast*
then have g1 ⊆ r **using** *b9* **by** *blast*
then show *?thesis* **using** *b0 b19* **by** *blast*
qed
have *b22*: ∀ a ∈ Field r1 − Field s. ∃ b ∈ Field s. (a, b) ∈ r1^*
proof
fix *a*
assume *d1*: a ∈ Field r1 − Field s
then have a ∈ D **using** *q4 b21* **unfolding** *Field-def* **by** *blast*
then obtain a′ **where** *d2*: a′ ∈ D ∧ d a′ = 1 ∧ (a, a′) ∈ g1^* **using** *q6* **by** *blast*
then have *d3*: (a′, h a′) ∈ r1 ∧ h a′ ∈ H a′ **using** *b8 b9 t1* **by** *blast*
obtain *b* **where** b ∈ Field s ∧ (a′,b) ∈ r **using** *d2 q1 q4 b3* **by** *force*
moreover then have isd b (d b) **using** *q1* **unfolding** *Field-def* **by** *blast*
ultimately have b ∈ B a′ ∧ d b = 0 **using** *b3 b5* **by** *force*
then have d (h a′) = 0 **using** *d3 b6* **by** *force*
then have isd (h a′) 0 **using** *q1 d3 b21* **unfolding** *Field-def* **by** *force*
then have h a′ ∈ Field s **using** *b3* **by** *force*
moreover have (a, a′) ∈ r1^* **using** *d2 t1 rtrancl-mono[of g1 r1]* **by** *blast*
ultimately have (h a′) ∈ Field s ∧ (a, h a′) ∈ r1^* **using** *d3* **by** *force*
then show ∃ b ∈ Field s. (a, b) ∈ r1^* **by** *blast*
qed
have *b23*: Field r ⊆ Field r1
proof −
have (Field r − Field s) ⊆ Field r1 **using** *q4 b9 t1* **unfolding** *Field-def* **by** *blast*
moreover have Field s ⊆ Field r1 **using** *b20* **unfolding** *Field-def* **by** *blast*
ultimately show Field r ⊆ Field r1 **by** *blast*
qed
have Field r1 ⊆ Field r **using** *b21* **unfolding** *Field-def* **by** *blast*
then have r1 ∈ Span r **using** *b21 b23* **unfolding** *Span-def* **by** *blast*
moreover have CCR r1
proof −
have s ∈ 𝔘 r1 **using** *b20 b22 a1* **unfolding** *𝔘-def* **by** *blast*
then show CCR r1 **using** *lem-rcc-uset-ne-ccr* **by** *blast*
qed
moreover have single-valued r1
proof −
have ∀ a b c. (a,b) ∈ r1 ∧ (a,c) ∈ r1 ⟶ b = c
proof (*intro allI impI*)

```
    fix a b c
    assume (a,b) ∈ r1 ∧ (a,c) ∈ r1
    moreover have (a,b) ∈ s ∧ (a,c) ∈ s ⟶ b = c using a2 unfolding single-valued-def by blast
    moreover have (a,b) ∈ s ∧ (a,c) ∈ g1 ⟶ False using b9 b7 unfolding Field-def by blast
    moreover have (a,b) ∈ g1 ∧ (a,c) ∈ s ⟶ b = c using b9 b7 unfolding Field-def by blast
    moreover have (a,b) ∈ g1 ∧ (a,c) ∈ g1 ⟶ b = c using b9 by blast
    ultimately show b = c using b19 by blast
  qed
  then show ?thesis unfolding single-valued-def by blast
qed
moreover have acyclic s ⟶ acyclic r1
proof
  assume c1: acyclic s
  have c2: ∀ a′ ∈ D. d a′ = 1 ⟶ d (h a′) = 0
  proof (intro ballI impI)
    fix a′
    assume d1: a′ ∈ D and d2: d a′ = 1
    then have d3: (a′, h a′) ∈ r1 ∧ h a′ ∈ H a′ using b8 b9 t1 by blast
    obtain b where b ∈ Field s ∧ (a′,b) ∈ r using d1 d2 q1 q4 b3 by force
    moreover then have isd b (d b) using q1 unfolding Field-def by blast
    ultimately have b ∈ B a′ ∧ d b = 0 using b3 b5 by force
    then show d (h a′) = 0 using d3 b6 by force
  qed
  have c3: ∀ a b. (a,b) ∈ g1 ⟶ d b < d a
  proof (intro allI impI)
    fix a b
    assume (a,b) ∈ g1
    then have d1: a ∈ D ∧ b = h a using b9 by blast
    then have d a > 1 ∨ d a = 1 and d a > 1 ⟶ d b < d a using q3 q4 q5 by force+
    moreover have d a = 1 ⟶ d b < d a using d1 c2 by force
    ultimately show d b < d a by blast
  qed
  have c4: ∀ n. ∀ a b. (a,b) ∈ g1^^(Suc n) ⟶ d b < d a
  proof
    fix n
    show ∀ a b. (a,b) ∈ g1^^(Suc n) ⟶ d b < d a
    proof (induct n)
      show ∀ a b. (a, b) ∈ g1 ^^ (Suc 0) ⟶ d b < d a using c3 by force
    next
      fix n
      assume e1: ∀ a b. (a, b) ∈ g1 ^^ (Suc n) ⟶ d b < d a
      show ∀ a b. (a, b) ∈ g1 ^^ (Suc (Suc n)) ⟶ d b < d a
      proof (intro allI impI)
        fix a b
        assume (a, b) ∈ g1 ^^ (Suc (Suc n))
```

then obtain *c* **where** (*a*,*c*) ∈ *g1*^^(*Suc n*) ∧ (*c*,*b*) ∈ *g1* **by** *force*
then have *d c* < *d a* ∧ *d b* < *d c* **using** *e1 c3* **by** *blast*
then show *d b* < *d a* **by** *simp*
qed
qed
qed
have ∀ *x*. (*x*,*x*) ∈ *g1*^+ ⟶ *False*
proof (*intro allI impI*)
fix *x*
assume (*x*,*x*) ∈ *g1*^+
then obtain *m*::*nat* **where** *m* > *0* ∧ (*x*,*x*) ∈ *g1*^^*m* **using** *trancl-power* **by** *blast*
moreover then obtain *n* **where** *m* = *Suc n* **using** *less-imp-Suc-add* **by** *blast*
ultimately have *d x* < *d x* **using** *c4* **by** *blast*
then show *False* **by** *blast*
qed
then have *acyclic g1* **unfolding** *acyclic-def* **by** *blast*
moreover have ∀ *a b c*. (*a*,*b*) ∈ *s* ∧ (*b*,*c*) ∈ *g1* ⟶ *False* **using** *b9 b7* **unfolding** *Field-def* **by** *blast*
moreover have *r1* = *s* ∪ *g1* **using** *b19* **by** *blast*
ultimately show *acyclic r1* **using** *c1 lem-acyc-un-emprd* **by** *blast*
qed
ultimately show *?thesis* **using** *b20* **by** *blast*
qed

lemma *lem-incrfun-nat*: ∀ *i*::*nat*. *f i* < *f* (*Suc i*) ⟹ ∀ *i j*. *i* ≤ *j* ⟶ *f i* + (*j*−*i*) ≤ *f j*
proof −
assume *a1*: ∀ *i*::*nat*. *f i* < *f* (*Suc i*)
have ∀ *j*. ∀ *i*. *i*≤*j* ⟶ *f i* + (*j*−*i*) ≤ *f j*
proof
fix *j0*
show ∀ *i*. *i*≤*j0* ⟶ *f i* + (*j0*−*i*) ≤ *f j0*
proof (*induct j0*)
show ∀ *i*≤*0*. *f i* + (*0* − *i*) ≤ *f 0* **by** *simp*
next
fix *j*
assume *c1*: ∀ *i*≤*j*. *f i* + (*j* − *i*) ≤ *f j*
show ∀ *i*≤*Suc j*. *f i* + (*Suc j* − *i*) ≤ *f* (*Suc j*)
proof (*intro allI impI*)
fix *i*
assume *d1*: *i* ≤ *Suc j*
show *f i* + (*Suc j* − *i*) ≤ *f* (*Suc j*)
proof (*cases i* ≤ *j*)
assume *i* ≤ *j*
moreover then have *f i* + (*j* − *i*) ≤ *f j* **using** *c1* **by** *blast*
ultimately show *?thesis* **using** *a1*
by (*metis Suc-diff-le Suc-le-eq add-Suc-right not-le order-trans*)

```
        next
          assume ¬ i ≤ j
          then have i = Suc j using d1 by simp
          then show ?thesis by simp
        qed
      qed
    qed
  qed
  then show ∀ i j. i ≤ j ⟶ f i + (j−i) ≤ f j by blast
qed

lemma lem-sv-uset-rcceqw:
fixes r::'U rel
assumes a1: ‖r‖ =o ω-ord
shows ∃ r1 ∈ 𝔘 r. single-valued r1 ∧ acyclic r1 ∧ (∀ x ∈ Field r1. r1``{x} ≠ {})
proof −
  have ¬ ( ‖r‖ <o ω-ord ) using a1 by (metis not-ordLess-ordIso)
 then obtain s where b1: s ∈ 𝔘 r ∧ |s| =o ‖r‖ using lem-rcc-uset-ne lem-uset-ne-rcc-inf
by blast
  then have |Field s| =o ω-ord
   using a1 lem-rel-inf-fld-card[of s] by (metis ordIso-natLeq-infinite1 ordIso-transitive)
 then obtain ai where b2: Field s = ai ` (UNIV::nat set) using lem-cntset-enum
by blast
  obtain f where b3: f = (λ x. SOME y. (x,y) ∈ r^* ∧ y ∈ Field s ) by blast
  obtain g where b4: g = (λ A. SOME y. y ∈ Field r ∧ A ⊆ dncl r {y}) by blast
  obtain h where b5: h = (λ A. SOME y. y ∈ Field r − dncl r A) by blast
  have b6: ⋀ x. x ∈ Field r ⟹ (x, f x) ∈ r^* ∧ f x ∈ Field s
  proof −
    fix x
    assume x ∈ Field r
   then have ∃ y. (x,y) ∈ r^* ∧ y ∈ Field s using b1 unfolding 𝔘-def by blast
    then show (x,f x) ∈ r^* ∧ f x ∈ Field s
      using b3 someI-ex[of λ y. (x,y) ∈ r^* ∧ y ∈ Field s ] by blast
  qed
  have b7: ⋀ A. finite A ∧ A ⊆ Field r ⟹ g A ∈ Field r ∧ A ⊆ dncl r {g A}
  proof −
    fix A::'U set
    assume c1: finite A ∧ A ⊆ Field r
    moreover have CCR r using b1 lem-rcc-uset-ne-ccr by blast
    ultimately obtain s where c2: finite s ∧ CCR s ∧ s ⊆ r ∧ A ⊆ Field s
      using lem-Ccext-finsubccr-dext[of r A] by blast
    then have c3: Conelike s using lem-Relprop-fin-ccr by blast
    have ∃ y. y ∈ Field r ∧ A ⊆ dncl r {y}
    proof (cases A = {})
      assume A = {}
       moreover have r ≠ {} using a1 lem-rcc-emp lem-Rcc-inf-lim by (metis
ordIso-iff-ordLeq)
      moreover then have Field r ≠ {} unfolding Field-def by force
      ultimately show ?thesis unfolding dncl-def by blast
```

next
assume *d1*: $A \neq \{\}$
then have $s \neq \{\}$ **using** *c2* **unfolding** *Field-def* **by** *blast*
then obtain y **where** $\forall x \in A.\ (x, y) \in s\hat{}*$ **using** *c2 c3* **unfolding** *Conelike-def* **by** *blast*
then have *d2*: $\forall\ x \in A.\ (x,y) \in r\hat{}*$ **using** *c2 rtrancl-mono* **by** *blast*
obtain *x0* **where** $x0 \in A \cap \mathit{Field}\ r$ **using** *d1 c1 c2* **by** *blast*
moreover then have $(x0,\ y) \in r\hat{}*$ **using** *d2* **by** *blast*
ultimately have $y \in \mathit{Field}\ r$ **using** *lem-rtr-field*[*of x0 y r*] **by** *blast*
then show *?thesis* **using** *d2* **unfolding** *dncl-def* **by** *blast*
qed
then show $g\ A \in \mathit{Field}\ r \wedge A \subseteq \mathit{dncl}\ r\ \{g\ A\}$
using *b4 someI-ex*[*of* $\lambda\ y.\ y \in \mathit{Field}\ r \wedge A \subseteq \mathit{dncl}\ r\ \{y\}$] **by** *blast*
qed
have *b8*: $\bigwedge\ A::'U\ set.\ \mathit{finite}\ A \Longrightarrow (h\ A) \in \mathit{Field}\ r - \mathit{dncl}\ r\ A$
proof $-$
fix $A::'U\ set$
assume *c1*: *finite A*
have $\mathit{Field}\ r - \mathit{dncl}\ r\ A = \{\} \longrightarrow \mathit{False}$
proof
assume $\mathit{Field}\ r - \mathit{dncl}\ r\ A = \{\}$
then have $\forall\ x \in \mathit{Field}\ r.\ \exists\ y \in A \cap \mathit{Field}\ r.\ (x,y) \in r\hat{}*$
using *lem-rtr-field*[*of - - r*] **unfolding** *dncl-def* **by** *blast*
then have $A \cap \mathit{Field}\ r \in \mathit{SCF}\ r$ **unfolding** *SCF-def* **by** *blast*
then have $\mathit{scf}\ r \leq o\ |A \cap \mathit{Field}\ r|$ **using** *lem-scf-uset-mem-bnd* **by** *blast*
moreover have $|A \cap \mathit{Field}\ r| < o\ \omega\text{-}\mathit{ord}$ **using** *c1 finite-iff-ordLess-natLeq* **by** *blast*
ultimately have $\mathit{scf}\ r < o\ \omega\text{-}\mathit{ord}$ **by** (*metis ordLeq-ordLess-trans*)
moreover have $\|r\| = o\ \mathit{scf}\ r$ **using** *b1 lem-scf-ccr-scf-rcc-eq*[*of r*] *lem-rcc-uset-ne-ccr*[*of r*] **by** *blast*
ultimately show *False* **using** *a1*
by (*meson not-ordLeq-ordLess ordIso-iff-ordLeq ordLess-ordLeq-trans*)
qed
then show $(h\ A) \in \mathit{Field}\ r - \mathit{dncl}\ r\ A$
using *b5 someI-ex*[*of* $\lambda\ y.\ y \in \mathit{Field}\ r - \mathit{dncl}\ r\ A$] **by** *blast*
qed
obtain *Ci* **where** *b9*: $Ci = \mathit{rec\text{-}nat}\ \{\ ai\ 0\ \}\ (\lambda\ n\ B.\ B \cup \{f(g(\{(h\ B)\} \cup B \cup ai`\{k.\ k \leq n\}))\})$ **by** *blast*
then have *b10*: $Ci\ 0 = \{ai\ 0\}$
and *b11*: $\bigwedge\ n.\ Ci\ (Suc\ n) = Ci\ n \cup \{f(g(\{(h\ (Ci\ n))\} \cup Ci\ n \cup ai`\{k.\ k \leq n\}))\}$ **by** *simp+*
have *b12*: $\mathit{Field}\ s \subseteq \mathit{Field}\ r$ **using** *b1* **unfolding** $\mathfrak{U}$*-def Field-def* **by** *blast*
have *b13*: $\bigwedge\ n.\ Ci\ n \subseteq \mathit{Field}\ s \wedge \mathit{finite}\ (Ci\ n)$
proof $-$
fix n
show $Ci\ n \subseteq \mathit{Field}\ s \wedge \mathit{finite}\ (Ci\ n)$
proof (*induct n*)
show $Ci\ 0 \subseteq \mathit{Field}\ s \wedge \mathit{finite}\ (Ci\ 0)$ **using** *b2 b10* **by** *simp*
next

fix *n*
assume *Ci n* ⊆ *Field s* ∧ *finite* (*Ci n*)
moreover then have {*h* (*Ci n*)} ∪ *Ci n* ∪ *ai* ' {*k*. *k* ≤ *n*} ⊆ *Field r* **using** *b2 b8 b12* **by** *blast*
ultimately show *Ci* (*Suc n*) ⊆ *Field s* ∧ *finite* (*Ci* (*Suc n*)) **using** *b6 b7 b11* **by** *simp*
qed
qed
have *b14*: ⋀ *n*. ∃ *m*∈(*Ci n*). *Ci n* ∪ *ai*'{*k*. *k*≤*n*−*1*} ⊆ *dncl r* {*m*}
proof −
fix *n*
show ∃ *m*∈(*Ci n*). *Ci n* ∪ *ai*'{*k*. *k*≤*n*−*1*} ⊆ *dncl r* {*m*}
proof (*induct n*)
show ∃*m*∈*Ci 0*. *Ci 0* ∪ *ai*'{*k*. *k*≤*0*−*1*} ⊆ *dncl r* {*m*} **using** *b10* **unfolding** *dncl-def* **by** *simp*
next
fix *n*
assume ∃*m*∈*Ci n*. *Ci n* ∪ *ai*'{*k*. *k*≤*n*−*1*} ⊆ *dncl r* {*m*}
obtain *A* **where** *d1*: *A* = {(*h* (*Ci n*))} ∪ *Ci n* ∪ *ai*'{*k*. *k*≤*n*} **by** *blast*
obtain *m* **where** *d2*: *m* = *f*(*g*(*A*)) **by** *blast*
have *finite A* ∧ *A* ⊆ *Field r* **using** *d1 b2 b8 b12 b13* **by** *force*
then have *d3*: *g A* ∈ *Field r* ∧ *A* ⊆ *dncl r* {*g A*} **using** *b7* **by** *blast*
then have *d4*: (*g A*, *m*) ∈ *r*^∗ ∧ *m* ∈ *Field s* **using** *d2 b6* **by** *blast*
have *m* ∈ *Ci* (*Suc n*) **using** *d1 d2 b11* **by** *blast*
moreover have *ai*'{*k*. *k*≤*n*} ⊆ *dncl r* {*m*} **using** *d1 d3 d4* **unfolding** *dncl-def* **by** *force*
moreover have *Ci n* ⊆ *dncl r* {*m*} **using** *d1 d3 d4* **unfolding** *dncl-def* **by** *force*
moreover then have *Ci* (*Suc n*) ⊆ *dncl r* {*m*} **using** *d1 d2 b11* **unfolding** *dncl-def* **by** *blast*
ultimately show ∃*m*∈*Ci* (*Suc n*). *Ci* (*Suc n*) ∪ *ai*'{*k*. *k*≤(*Suc n*)−*1*} ⊆ *dncl r* {*m*} **by** *force*
qed
qed
obtain *ci* **where** *b15*: *ci* = (λ *n*. *SOME m*. *m* ∈ *Ci n* ∧ *Ci n* ⊆ *dncl r* {*m*}) **by** *blast*
have *b16*: ⋀ *n*. (*ci n*) ∈ *Ci n* ∧ *Ci n* ⊆ *dncl r* {*ci n*}
proof −
fix *n*
have ∃ *m*∈(*Ci n*). *Ci n* ⊆ *dncl r* {*m*} **using** *b14* **by** *blast*
then show (*ci n*) ∈ *Ci n* ∧ *Ci n* ⊆ *dncl r* {*ci n*}
using *b15 someI-ex*[*of* λ *m*. *m* ∈ *Ci n* ∧ *Ci n* ⊆ *dncl r* {*m*}] **by** *blast*
qed
have *b17*: ⋀ *n*. *ci* (*Suc n*) ∉ *dncl r* (*Ci n*)
proof −
fix *n*
obtain *A* **where** *c1*: *A* = {(*h* (*Ci n*))} ∪ *Ci n* ∪ *ai*'{*k*. *k*≤*n*} **by** *blast*
then have *c2*: *finite A* ∧ *A* ⊆ *Field r* **using** *b2 b8*[*of Ci n*] *b13*[*of n*] *b12* **by** *blast*

then have *c3*: *g A* ∈ *Field r* ∧ *A* ⊆ *dncl r* {*g A*} **using** *b7* **by** *simp*
then have (*h* (*Ci n*), *g A*) ∈ *r*^* **using** *c1* **unfolding** *dncl-def* **by** *blast*
moreover have (*g A*, *f* (*g A*)) ∈ *r*^* **using** *c3 b6*[*of g A*] **by** *blast*
moreover have (*f* (*g A*), *ci* (*Suc n*)) ∈ *r*^* **using** *c1 b11 b16* **unfolding** *dncl-def* **by** *blast*
ultimately have (*h* (*Ci n*), *ci* (*Suc n*)) ∈ *r*^* **by** *force*
moreover have *h* (*Ci n*) ∉ *dncl r* (*Ci n*) **using** *b8*[*of Ci n*] *b13*[*of n*] **by** *blast*
ultimately show *ci* (*Suc n*) ∉ *dncl r* (*Ci n*) **unfolding** *dncl-def*
by (*meson Image-iff converse-iff rtrancl-trans*)
qed
have ∀ *n*. (*ci n*, *ci* (*Suc n*)) ∈ *r*^* ∧ *ci n* ≠ *ci* (*Suc n*)
proof
fix *n*
have (*ci n*, *ci* (*Suc n*)) ∈ *r*^* **using** *b11 b16* **unfolding** *dncl-def* **by** *blast*
moreover have *ci n* ≠ *ci* (*Suc n*) **using** *b16*[*of n*] *b17*[*of n*] **unfolding** *dncl-def* **by** *fastforce*
ultimately show (*ci n*, *ci* (*Suc n*)) ∈ *r*^* ∧ *ci n* ≠ *ci* (*Suc n*) **by** *blast*
qed
then obtain *l yi* **where**
b18: ∀ *n*. (*yi n*, *yi* (*Suc n*)) ∈ *r*
and *b19*: ∀ *i j*. (*i* < *j*) = (*l i* < *l j*)
and *b20*: ∀ *i*. *yi* (*l i*) = *ci i*
and *b21*: ∀ *i*. *inj-on yi* {*k*. *l i* ≤ *k* ∧ *k* ≤ *l* (*Suc i*)}
and *b22*: ∀ *k*. ∃ *i*. *l i* ≤ *k* ∧ *Suc k* ≤ *l* (*Suc i*)
using *lem-flatseq*[*of ci r*] **by** *blast*
obtain *r′* **where** *b23*: *r′* = { (*x*,*y*). ∃ *i*. *x* = *yi i* ∧ *y* = *yi* (*Suc i*) } **by** *blast*
have *b24*: ∀ *j*. ∀ *i*. *i* ≤ *j* ⟶ (*yi i*, *yi j*) ∈ *r′*^*
proof
fix *j*
show ∀ *i*. *i* ≤ *j* ⟶ (*yi i*, *yi j*) ∈ *r′*^*
proof (*induct j*)
show ∀ *i* ≤ *0*. (*yi i*, *yi 0*) ∈ *r′*^* **by** *blast*
next
fix *j*
assume *d1*: ∀ *i* ≤ *j*. (*yi i*, *yi j*) ∈ *r′*^*
show ∀ *i* ≤ *Suc j*. (*yi i*, *yi* (*Suc j*)) ∈ *r′*^*
proof (*intro allI impI*)
fix *i*
assume *e1*: *i* ≤ *Suc j*
show (*yi i*, *yi* (*Suc j*)) ∈ *r′*^*
proof (*cases i* ≤ *j*)
assume *i* ≤ *j*
then have (*yi i*, *yi j*) ∈ *r′*^* **using** *d1* **by** *blast*
moreover have (*yi j*, *yi* (*Suc j*)) ∈ *r′* **using** *b23* **by** *blast*
ultimately show *?thesis* **by** *simp*
next
assume ¬ *i* ≤ *j*
then have *i* = *Suc j* **using** *e1* **by** *simp*
then show *?thesis* **using** *e1* **by** *blast*

qed
qed
qed
qed
have *b25*: $\forall$ *j*. ($\forall$ *i*. $i \leq j \longrightarrow Ci\ i \subseteq Ci\ j$)
proof
fix *j*
show $\forall$ *i*. $i \leq j \longrightarrow Ci\ i \subseteq Ci\ j$
proof (*induct j*)
show $\forall i \leq 0.\ Ci\ i \subseteq Ci\ 0$ **by** *force*
next
fix *j*
assume $\forall i \leq j.\ Ci\ i \subseteq Ci\ j$
moreover have $Ci\ j \subseteq Ci\ (Suc\ j)$ **using** *b11* **by** *blast*
ultimately show $\forall i \leq Suc\ j.\ Ci\ i \subseteq Ci\ (Suc\ j)$ **using** *le-Suc-eq* **by** *fastforce*
qed
qed
have *b26*: $\forall$ *k1 k2*. $k1 < k2 \longrightarrow yi\ k1 = yi\ k2 \longrightarrow (\exists\ i.\ l\ i \leq k1 \wedge k2 \leq l\ (i+2))$
proof (*intro allI impI*)
fix *k1::nat* **and** *k2::nat*
assume *d1*: $k1 < k2$ **and** *d2*: $yi\ k1 = yi\ k2$
obtain *i1 i2* **where** *d3*: $l\ i1 \leq k1 \wedge Suc\ k1 \leq l\ (Suc\ i1)$
and *d4*: $l\ i2 \leq k2 \wedge Suc\ k2 \leq l\ (Suc\ i2)$ **using** *b22* **by** *blast*
have $i1 = i2 \longrightarrow False$
proof
assume $i1 = i2$
then have $l\ i1 \leq k2 \wedge k2 \leq l\ (Suc\ i1)$ **using** *d4* **by** *simp*
moreover have $l\ i1 \leq k1 \wedge k1 \leq l\ (Suc\ i1)$ **using** *d3* **by** *simp*
ultimately show *False* **using** *d1 d2 b21* **unfolding** *inj-on-def* **by** *blast*
qed
moreover have $i2 < i1 \longrightarrow False$
proof
assume $i2 < i1$
then have $Suc\ i2 = i1 \vee Suc\ i2 < i1$ **by** *fastforce*
then have $l\ (Suc\ i2) = l\ i1 \vee l\ (Suc\ i2) < l\ i1$ **using** *b19* **by** *blast*
then have $l\ (Suc\ i2) \leq l\ i1$ **by** *fastforce*
moreover have $l\ i1 < l\ (Suc\ i2)$ **using** *d1 d3 d4* **by** *simp*
ultimately show *False* **by** *simp*
qed
moreover have $Suc\ i1 < i2 \longrightarrow False$
proof
assume *e1*: $Suc\ i1 < i2$
have $k1 \leq l\ (Suc\ i1) \wedge l\ i2 \leq k2$ **using** *d3 d4* **by** *force*
then have $(yi\ k1,\ yi\ (l\ (Suc\ i1))) \in r\hat{}*$ **and** $(yi\ (l\ i2),\ yi\ k2) \in r\hat{}*$
using *b18 b23 b24 rtrancl-mono*[*of r' r*] **by** *blast+*
then have *e2*: $(yi\ k1,\ ci\ (Suc\ i1)) \in r\hat{}*$ **and** *e3*: $(ci\ i2,\ yi\ k1) \in r\hat{}*$ **using** *d2 b20* **by** *force+*
have $Suc\ i1 \leq i2-1 \wedge i2-1 \leq i2$ **and** $Suc\ (i2-1) = i2$ **using** *e1* **by** *simp+*

then have *e4*: $ci\ i2 \notin dncl\ r\ (Ci\ (i2\ -\ 1))$ **and** *e5*: $ci\ (Suc\ i1) \in Ci\ (i2-1)$

using *b16*[*of Suc i1*] *b17*[*of i2* − *1*] *b25* **by** *fastforce+*
have $yi\ k1 \notin dncl\ r\ (Ci\ (i2-1))$ **using** *e3 e4* **unfolding** *dncl-def*
by (*meson Image-iff converse-iff rtrancl-trans*)
moreover have $yi\ k1 \in dncl\ r\ (Ci\ (i2-1))$ **using** *e2 e5* **unfolding** *dncl-def*
by *blast*
ultimately show *False* **by** *blast*
qed
ultimately have $Suc\ i1 = i2$ **by** *simp*
moreover then have $l\ (Suc\ i1) = l\ i2$ **using** *b19* **by** *blast*
ultimately have $l\ i1 \leq k1 \wedge k2 \leq l\ (i1 + 2)$ **using** *d3 d4* **by** *simp*
then show $\exists\ i.\ l\ i \leq k1 \wedge k2 \leq l\ (i+2)$ **by** *blast*
qed
obtain *w* **where** *b27*: $w = (\lambda\ k.\ k + l\ ((GREATEST\ j.\ l\ j \leq k) + 2))$ **by** *blast*
have *b28*: $\bigwedge k.\ \forall\ k'.\ yi\ k = yi\ k' \longrightarrow k' < Suc\ (w\ k)$
proof −
fix *k*
show $\forall\ k'.\ yi\ k = yi\ k' \longrightarrow k' < Suc\ (w\ k)$
proof (*cases* $\exists\ k' > k.\ yi\ k' = yi\ k$)
assume *d1*: $\exists\ k' > k.\ yi\ k' = yi\ k$
have *d2*: $\forall\ k'.\ k < k' \longrightarrow yi\ k = yi\ k' \longrightarrow (\exists\ i.\ l\ i \leq k \wedge k' \leq l\ (i+2))$
using *b26* **by** *blast*
have *d3*: $\forall\ i.\ i \leq l\ i$
proof
fix *i*
show $i \leq l\ i$
proof (*induct i*)
show $0 \leq l\ 0$ **by** *blast*
next
fix *i*
assume $i \leq l\ i$
moreover have $l\ i < l\ (Suc\ i)$ **using** *b19* **by** *blast*
ultimately show $Suc\ i \leq l\ (Suc\ i)$ **by** *simp*
qed
qed
obtain *i0* **where** *d4*: $i0 = (GREATEST\ j.\ l\ j \leq k)$ **by** *blast*
obtain *t* **where** *d5*: $t = k + l\ (i0+2)$ **by** *blast*
then have $t \geq k$ **by** *force*
moreover have $\forall\ k'.\ yi\ k' = yi\ k \longrightarrow k' \leq t$
proof (*intro allI impI*)
fix k'
assume *e1*: $yi\ k' = yi\ k$
have $k < k' \longrightarrow k' \leq t$
proof
assume $k < k'$
then obtain *i* **where** *f1*: $l\ i \leq k \wedge k' \leq l\ (i+2)$ **using** *e1 d2* **by** *metis*
moreover have $\forall y.\ l\ y \leq k \longrightarrow y < Suc\ k$ **using** *d3 less-Suc-eq-le*
order-trans **by** *blast*

ultimately have $i \leq i0$ **using** *d4 Greatest-le-nat*[*of* λ *j. l j* $\leq$ *k i Suc k*] **by** *force*
then have *l* (*i*+*2*) $\leq$ *l*(*i0*+*2*) **using** *b19* **by** (*metis Suc-less-eq add-2-eq-Suc′ not-le*)
then show $k' \leq t$ **using** *f1 d5* **by** *fastforce*
qed
then show $k' \leq t$ **using** *d5* **by** *fastforce*
qed
ultimately show *?thesis* **using** *d4 d5 b27* **by** *fastforce*
next
assume ¬ (∃ *k′* > *k*. *yi k′* = *yi k*)
then have ∀ *k′*. *yi k′* = *yi k* ⟶ *k′* ≤ *k* **using** *leI* **by** *blast*
then show *?thesis* **using** *b27* **by** *fastforce*
qed
qed
obtain *q* **where** *b29*: *q* = (λ *k*. *GREATEST k′*. *yi k* = *yi k′*) **by** *blast*
have *b30*: ⋀ *k*. *yi k* = *yi* (*q k*)
proof −
fix *k*
show *yi k* = *yi* (*q k*) **using** *b28*[*of k*] *b29 GreatestI-nat*[*of* λ *k′*. *yi k* = *yi k′ k Suc* (*w k*)] **by** *force*
qed
have *b31*: ⋀ *k k′*. *yi k′* = *yi* (*q k*) ⟶ *k′* ≤ *q k*
proof
fix *k k′*
assume *yi k′* = *yi* (*q k*)
then show *k′* ≤ *q k* **using** *b28*[*of k*] *b29 b30 Greatest-le-nat*[*of* λ *k′*. *yi k* = *yi k′ k′ Suc* (*w k*)] **by** *force*
qed
obtain *p* **where** *b32*: *p* = *rec-nat* (*q 0*) (λ *n y*. *q* (*Suc y*)) **by** *blast*
obtain *r1* **where** *b33*: *r1* = { (*x*,*y*). ∃ *i*. *x* = *yi* (*p i*) ∧ *y* = *yi* (*Suc* (*p i*)) } **by** *blast*
have *b34*: ⋀ *i*. *p i* = *q* (*p i*)
proof −
fix *i*
show *p i* = *q* (*p i*)
proof (*induct i*)
show *p 0* = *q* (*p 0*) **using** *b29 b30 b32* **by** *simp*
next
fix *i*
assume *p i* = *q* (*p i*)
then show *p* (*Suc i*) = *q* (*p* (*Suc i*)) **using** *b29 b30 b32* **by** *simp*
qed
qed
have *b35*: ⋀ *i j*. *i*≤*j* ⟶ *p i* + (*j*−*i*) ≤ *p j*
proof −
fix *i j*
have ⋀ *k*. *q k* = *k* ⟶ *q k* < *q* (*Suc k*) **using** *b30 b31* **by** (*metis less-eq-Suc-le*)
then have ∀ *i*. *p i* < *p* (*Suc i*) **using** *b32 b34* **by** *simp*

```
    then show i≤j ⟶ p i + (j−i) ≤ p j using lem-incrfun-nat[of p] by blast
  qed
  have b36: ∀ i j. p i = p j ⟶ i = j
  proof (intro allI impI)
    fix i j
    assume p i = p j
    then have i≤j ⟶ i = j and j≤i ⟶ j = i using b35 by fastforce+
    then show i = j by fastforce
  qed
  have b37: ∀ i j. yi (p i) = yi (p j) ⟶ i = j using b29 b34 b36 by metis
  have b38: ∀ x ∈ Field r1. ∃ i. x = yi (p i)
  proof
    fix x
    assume x ∈ Field r1
    moreover have ∀ i. yi (Suc (p i)) = yi (p (Suc i)) using b30 b32 by simp
    ultimately show ∃ i. x = yi (p i) using b33 unfolding Field-def by force
  qed
  have b39: ⋀ i. (yi (p i), yi (p (Suc i))) ∈ r1 using b30 b32 b33 by fastforce
  have b40: ∀ j. ∀ i. i ≤ j ⟶ (yi (p i), yi (p j)) ∈ r1^*
  proof
    fix j0
    show ∀ i. i ≤ j0 ⟶ (yi (p i), yi (p j0)) ∈ r1^*
    proof (induct j0)
      show ∀i≤0. (yi (p i), yi (p 0)) ∈ r1^* by blast
    next
      fix j
      assume d1: ∀i≤j. (yi (p i), yi (p j)) ∈ r1^*
      show ∀i≤Suc j. (yi (p i), yi (p (Suc j))) ∈ r1^*
      proof (intro allI impI)
        fix i
        assume e1: i≤Suc j
        show (yi (p i), yi (p (Suc j))) ∈ r1^*
        proof (cases i = Suc j)
          assume i = Suc j
          then show ?thesis by force
        next
          assume i ≠ Suc j
          then have (yi (p i), yi (p j)) ∈ r1^* using e1 d1 by simp
          then show ?thesis using e1 d1 b39[of j] by simp
        qed
      qed
    qed
  qed
  have r1 ⊆ r' using b23 b33 by blast
  moreover have ∀ a ∈ Field r'. ∃ b ∈ Field r1. (a, b) ∈ r'^*
  proof
    fix a
    assume a ∈ Field r'
    then obtain k where a = yi k using b23 unfolding Field-def by blast
```

moreover have *k* ≤ *p k* **using** *b35*[*of 0 k*] **by** *fastforce*
ultimately have (*a*, *yi* (*p k*)) ∈ *r′^** **using** *b24* **by** *blast*
moreover have *yi* (*p k*) ∈ *Field r1* **using** *b33* **unfolding** *Field-def* **by** *blast*
ultimately show ∃ *b* ∈ *Field r1*. (*a*, *b*) ∈ *r′^** **by** *blast*
qed
moreover have *CCR r1*
proof −
have ∀ *a*∈*Field r1*. ∀ *b*∈*Field r1*. ∃ *c*∈*Field r1*. (*a*, *c*) ∈ *r1^** ∧ (*b*, *c*) ∈ *r1^**
proof (*intro ballI*)
fix *a b*
assume *d1*: *a* ∈ *Field r1* **and** *d2*: *b* ∈ *Field r1*
then obtain *i j* **where** *a* = *yi* (*p i*) ∧ *b* = *yi* (*p j*) **using** *b38* **by** *blast*
then have *i* ≤ *j* ⟶ (*a*,*b*) ∈ *r1^** **and** *j* ≤ *i* ⟶ (*b*,*a*) ∈ *r1^** **using** *b40* **by** *blast+*
then show ∃ *c*∈*Field r1*. (*a*, *c*) ∈ *r1^** ∧ (*b*, *c*) ∈ *r1^** **using** *d1 d2* **by** *fastforce*
qed
then show *CCR r1* **unfolding** *CCR-def* **by** *blast*
qed
ultimately have *b41*: *r1* ∈ 𝔘 *r′* **unfolding** 𝔘-*def* **by** *blast*
then have *CCR r′* **using** *lem-rcc-uset-ne-ccr* **by** *blast*
moreover have *r′* ⊆ *r* **using** *b18 b23* **by** *blast*
moreover have ∀ *x* ∈ *Field r*. ∃ *y* ∈ *Field r′*. (*x*, *y*) ∈ *r^**
proof
fix *x*
assume *c1*: *x* ∈ *Field r*
then obtain *y* **where** *c2*: *y* ∈ *Field s* ∧ (*x*,*y*) ∈ *r^** **using** *b1* **unfolding** 𝔘-*def* **by** *blast*
then obtain *n* **where** *y* = *ai n* **using** *b2* **by** *blast*
then obtain *m* **where** *y* ∈ *dncl r* {*m*} ∧ *m* ∈ *Ci* (*Suc n*) **using** *b14*[*of Suc n*] **by** *force*
then have (*y*, *m*) ∈ *r^** ∧ (*m*, *ci* (*Suc n*)) ∈ *r^** **using** *b16* **unfolding** *dncl-def* **by** *blast*
then have (*x*, *ci* (*Suc n*)) ∈ *r^** **using** *c2* **by** *force*
moreover obtain *y′* **where** *c2*: *y′* = *yi* (*l* (*Suc n*)) **by** *blast*
ultimately have *c3*: (*x*,*y′*) ∈ *r^** **using** *b20* **by** *metis*
have (*y′*, *yi* (*Suc* (*l* (*Suc n*)))) ∈ *r′* **using** *c2 b23* **by** *blast*
then have *y′* ∈ *Field r′* **unfolding** *Field-def* **by** *blast*
then show ∃ *y* ∈ *Field r′*. (*x*, *y*) ∈ *r^** **using** *c3* **by** *blast*
qed
ultimately have *r′* ∈ 𝔘 *r* **unfolding** 𝔘-*def* **by** *blast*
then have *r1* ∈ 𝔘 *r* **using** *b41 lem-rcc-uset-tr* **by** *blast*
moreover have *single-valued r1* **using** *b33 b37* **unfolding** *single-valued-def* **by** *blast*
moreover have *acyclic r1*
proof −
have *c1*: ∀ *n*. ∀ *i j*. (*yi* (*p i*), *yi* (*p j*)) ∈ *r1^^*(*Suc n*) ⟶ *i* < *j*
proof
fix *n0*

show $\forall$ *i j*. (*yi* (*p i*), *yi* (*p j*)) $\in$ *r1*^^(*Suc n0*) $\longrightarrow$ *i* < *j*
proof (*induct n0*)
show $\forall$ *i j*. (*yi* (*p i*), *yi* (*p j*)) $\in$ *r1* ^^ (*Suc 0*) $\longrightarrow$ *i* < *j*
proof (*intro allI impI*)
fix *i j*
assume (*yi* (*p i*), *yi* (*p j*)) $\in$ *r1*^^(*Suc 0*)
then obtain *i' j'*::*nat* **where** *yi* (*p i*) = *yi* (*p i'*) $\wedge$ *yi* (*p j*) = *yi* (*Suc* (*p i'*)) **using** *b33* **by** *force*
then have *i* = *i'* $\wedge$ *j* = *Suc i'* **using** *b30 b32 b37* **by** *simp*
then show *i* < *j* **by** *blast*
qed
next
fix *n*
assume *d1*: $\forall$ *i j*. (*yi* (*p i*), *yi* (*p j*)) $\in$ *r1* ^^ (*Suc n*) $\longrightarrow$ *i* < *j*
show $\forall$ *i j*. (*yi* (*p i*), *yi* (*p j*)) $\in$ *r1* ^^ *Suc* (*Suc n*) $\longrightarrow$ *i* < *j*
proof (*intro allI impI*)
fix *i j*
assume (*yi* (*p i*), *yi* (*p j*)) $\in$ *r1* ^^ *Suc* (*Suc n*)
then obtain *x* **where** (*yi* (*p i*), *x*) $\in$ *r1* ^^ (*Suc n*) $\wedge$ (*x*, *yi* (*p j*)) $\in$ *r1* **by** *force*
moreover then obtain *k* **where** *x* = *yi* (*p k*) **using** *b38* **unfolding** *Field-def* **by** *blast*
ultimately have *e1*: *i* < *k* $\wedge$ (*yi* (*p k*), *yi* (*p j*)) $\in$ *r1* **using** *d1* **by** *blast*
then obtain *i' j'*::*nat* **where** *yi* (*p k*) = *yi* (*p i'*) $\wedge$ *yi* (*p j*) = *yi* (*Suc* (*p i'*)) **using** *b33* **by** *force*
then have *k* = *i'* $\wedge$ *j* = *Suc i'* **using** *b30 b32 b37* **by** *simp*
then have *k* < *j* **by** *blast*
then show *i* < *j* **using** *e1* **by** *simp*
qed
qed
qed
have $\forall$ *x*. (*x*,*x*) $\in$ *r1*^+ $\longrightarrow$ *False*
proof (*intro allI impI*)
fix *x*
assume *d1*: (*x*,*x*) $\in$ *r1*^+
then have *x* $\in$ *Field r1* **by** (*metis FieldI2 Field-def trancl-domain trancl-range*)
then obtain *i* **where** *x* = *yi* (*p i*) **using** *b38* **by** *blast*
moreover obtain *m*::*nat* **where** *m* > *0* $\wedge$ (*x*,*x*) $\in$ *r1*^^*m* **using** *d1 trancl-power* **by** *blast*
moreover then obtain *n* **where** *m* = *Suc n* **using** *less-imp-Suc-add* **by** *blast*
ultimately have *n* < *n* **using** *c1* **by** *blast*
then show *False* **by** *blast*
qed
then show *?thesis* **unfolding** *acyclic-def* **by** *blast*
qed
moreover have $\forall$ *x* $\in$ *Field r1*. *r1*``{*x*} $\neq$ {}
proof
fix *x*

assume *x* ∈ *Field r1*
then obtain *i* **where** *x* = *yi* (*p i*) **using** *b38* **by** *blast*
moreover then obtain *y* **where** *y* = *yi* (*Suc* (*p i*)) **by** *blast*
ultimately have (*x*,*y*) ∈ *r1* **using** *b33* **by** *blast*
then show *r1*‘‘{*x*} ≠ {} **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-sv-span-scflew*:
fixes *r*::′*U rel*
assumes *CCR r* **and** *scf r* ≤*o* *ω-ord*
shows ∃ *r1*. *r1* ∈ *Span r* ∧ *CCR r1* ∧ *single-valued r1*
proof (*cases* ‖*r*‖ =*o* *ω-ord*)
assume ‖*r*‖ =*o* *ω-ord*
then obtain *s* **where** *s* ∈ 𝔘 *r* ∧ *single-valued s* **using** *lem-sv-uset-rcceqw* **by** *blast*
then show *?thesis* **using** *lem-sv-uset-sv-span* **by** *blast*
next
assume ¬ (‖*r*‖ =*o* *ω-ord*)
then have ‖*r*‖ <*o* *ω-ord* **using** *assms lem-scf-ccr-scf-rcc-eq*[*of r*]
by (*metis ordIso-ordLess-trans ordIso-transitive ordLeq-iff-ordLess-or-ordIso*)
then have *b1*: *Conelike r* **using** *assms lem-Rcc-eq2-31* **by** *blast*
have ∃ *s*. *s* ∈ 𝔘 *r* ∧ *single-valued s*
proof (*cases r* = {})
assume *r* = {}
then have {} ∈ 𝔘 *r* **unfolding** 𝔘*-def CCR-def Field-def* **by** *blast*
moreover have *single-valued* {} **unfolding** *single-valued-def* **by** *blast*
ultimately show *?thesis* **by** *blast*
next
assume *r* ≠ {}
then obtain *m* **where** *c1*: *m* ∈ *Field r* ∧ (∀ *a* ∈ *Field r*. (*a*, *m*) ∈ *r*^*) **using** *b1* **unfolding** *Conelike-def* **by** *blast*
then obtain *u v* **where** *c2*: (*u*, *v*) ∈ *r* ∧ (*u* = *m* ∨ *v* = *m*) **unfolding** *Field-def* **by** *blast*
obtain *s* **where** *c3*: *s* = {(*u*,*v*)} **by** *blast*
have *s* ⊆ *r* **using** *c2 c3* **by** *blast*
moreover have *CCR s* **using** *c3* **unfolding** *CCR-def* **by** *fastforce*
moreover have ∀ *a*∈*Field r*. ∃ *b*∈*Field s*. (*a*, *b*) ∈ *r*^*
proof
fix *a*
assume *a* ∈ *Field r*
moreover have *m* ∈ *Field s* **using** *c2 c3* **unfolding** *Field-def* **by** *fastforce*
ultimately show ∃ *b*∈*Field s*. (*a*, *b*) ∈ *r*^* **using** *c1* **by** *blast*
qed
ultimately have *s* ∈ 𝔘 *r* **unfolding** 𝔘*-def* **by** *blast*
moreover have *single-valued s* **using** *c3* **unfolding** *single-valued-def* **by** *blast*
ultimately show *?thesis* **by** *blast*
qed

then show *?thesis* **using** *lem-sv-uset-sv-span* **by** *blast*
qed

lemma *lem-sv-span-scfeqw*:
fixes *r*::*'U rel*
assumes *CCR r* **and** *scf r =o ω-ord*
shows ∃ *r1*. *r1* ∈ *Span r* ∧ *r1* ≠ {} ∧ *CCR r1* ∧ *single-valued r1* ∧ *acyclic r1* ∧ (∀ *x*∈*Field r1*. *r1*‘‘{*x*} ≠ {})
proof –
have *b1*: ‖*r*‖ =*o ω-ord* **using** *assms lem-scf-ccr-scf-rcc-eq*[*of r*] **by** (*metis ordIso-transitive*)
then obtain *s* **where** *s* ∈ 𝔘 *r* ∧ *single-valued s* ∧ *acyclic s* ∧ (∀ *x*∈*Field s*. *s*‘‘{*x*} ≠ {})
using *lem-sv-uset-rcceqw* **by** *blast*
then obtain *r1* **where** *b2*: *r1* ∈ *Span r* ∧ *CCR r1* ∧ *single-valued r1* ∧ *s* ⊆ *r1* ∧ *acyclic r1*
using *lem-sv-uset-sv-span*[*of s r*] **by** *blast*
moreover have *r1* = {} ⟶ *False*
proof
assume *r1* = {}
then have *r* = {} **using** *b2* **unfolding** *Span-def Field-def* **by** *force*
then show *False* **using** *b1 lem-Rcc-inf-lim lem-rcc-emp lem-rcc-inf* **by** (*metis not-ordLess-ordIso*)
qed
moreover have ∀ *x*∈*Field r1*. *r1*‘‘{*x*} = {} ⟶ *False*
proof (*intro ballI impI*)
fix *x*
assume *c1*: *x* ∈ *Field r1* **and** *c2*: *r1*‘‘{*x*} = {}
have ∀ *a*∈*Field r1*. (*a*, *x*) ∈ *r1*^*
proof
fix *a*
assume *a* ∈ *Field r1*
then obtain *t* **where** (*x*,*t*) ∈ *r1*^* ∧ (*a*,*t*) ∈ *r1*^* **using** *c1 b2* **unfolding** *CCR-def* **by** *blast*
moreover then have *x* = *t* **using** *c2* **by** (*metis Image-singleton-iff converse-rtranclE empty-iff*)
ultimately show (*a*,*x*) ∈ *r1*^* **by** *blast*
qed
then have *Conelike r1* **using** *c1* **unfolding** *Conelike-def* **by** *blast*
moreover have *r1* ∈ 𝔘 *r* **using** *b2* **unfolding** 𝔘*-def Span-def* **by** *blast*
ultimately have *Conelike r* **using** *lem-uset-cl-ext*[*of r1 r*] **by** *blast*
then show *False* **using** *b1 lem-Rcc-eq2-12*[*of r*] *lem-Rcc-eq2-23*[*of r*] **by** (*metis not-ordLess-ordIso*)
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Ldo-den-ccr-uset*:
fixes *r s*::*'U rel*

assumes $CCR\ s$ **and** $s \subseteq r \wedge Field\ s \in Den\ r$
shows $s \in \mathfrak{U}\ r$
using *assms* **unfolding** *Den-def* $\mathfrak{U}$*-def* **by** *blast*

lemma *lem-Ldo-ds-reduc*:
fixes $r\ s{::}'U\ rel$ **and** $n0{::}nat$
assumes $a1$: $CCR\ s \wedge DCR\ n0\ s$ **and** $a2$: $s \subseteq r$ **and** $a3$: $Field\ s \in Den\ r$ **and** $a4$: $Field\ s \in Inv\ (r - s)$
shows $CCR\ r \wedge DCR\ (Suc\ n0)\ r$
proof −
obtain $g0$ **where** $b1$: $DCR\text{-}generating\ g0$
and $b2$: $s = \bigcup\ \{r'.\ \exists\alpha'.\ \alpha' < n0 \wedge r' = g0\ \alpha'\}$
using $a1$ **unfolding** *DCR-def* **by** *blast*
obtain $g :: nat \Rightarrow 'U\ rel$
where $b8$: $g = (\lambda\ \alpha.\ if\ (\alpha < n0)\ then\ (g0\ \alpha)\ else\ (r-\ s))$ **by** *blast*
obtain $n :: nat$ **where** $b9$: $n = (Suc\ n0)$ **by** *blast*
have $b11$: $\bigwedge\ \alpha.\ \alpha < n0 \Longrightarrow g\ \alpha = (g0\ \alpha)$ **using** $b8$ **by** *simp*
have $b12$: $\bigwedge\ \alpha.\ \neg\ (\alpha < n0) \Longrightarrow g\ \alpha = (r-\ s)$ **using** $b8$ **by** *force*
have $\forall\,\alpha\ \beta\ a\ b\ c.$
$\alpha \leq \beta \longrightarrow (a,\ b) \in g\ \alpha \wedge (a,\ c) \in g\ \beta \longrightarrow$
$(\exists\, b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ \beta\ \alpha)$
proof (*intro allI impI*)
fix $\alpha\ \beta\ a\ b\ c$
assume $c0$: $\alpha \leq \beta$ **and** $c1$: $(a,\ b) \in g\ \alpha \wedge (a,\ c) \in g\ \beta$
have $\alpha < n0 \wedge \beta < n0$
$\longrightarrow (\exists\, b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ \beta\ \alpha)$
proof
assume $d1$: $\alpha < n0 \wedge \beta < n0$
moreover then have $(a,\ b) \in g0\ \alpha \wedge (a,\ c) \in g0\ \beta$ **using** $c1\ b11$ **by** *blast*
then obtain $b'\ b''\ c'\ c''\ d$ **where** $d2$: $(b,\ b',\ b'',\ d) \in \mathfrak{D}\ g0\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g0\ \beta\ \alpha$
using $b1$ **unfolding** *DCR-generating-def* **by** *blast*
have $(b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta$
proof −
have $(b,\ b') \in (\mathfrak{L}1\ g\ \alpha)\hat{}*$
proof −
have $\forall\ \alpha'.\ \alpha' < \alpha \longrightarrow g\ \alpha' = g0\ \alpha'$ **using** $d1\ b11$ **by** *force*
then have $\mathfrak{L}1\ g\ \alpha = \mathfrak{L}1\ g0\ \alpha$ **unfolding** $\mathfrak{L}1$*-def* **by** *blast*
moreover have $(b,b') \in (\mathfrak{L}1\ g0\ \alpha)\hat{}*$ **using** $d2$ **unfolding** $\mathfrak{D}$*-def* **by** *blast*
ultimately show *?thesis* **by** *metis*
qed
moreover have $(b',\ b'') \in (g\ \beta)\hat{}=$
proof −
have $g\ \beta = g0\ \beta$ **using** $d1\ b11$ **by** *blast*
moreover have $(b',b'') \in (g0\ \beta)\hat{}=$ **using** $d2$ **unfolding** $\mathfrak{D}$*-def* **by** *blast*
ultimately show *?thesis* **by** *metis*
qed
moreover have $(b'',\ d) \in (\mathfrak{L}v\ g\ \alpha\ \beta)\hat{}*$
proof −

have ∀ α'. $\alpha' < \alpha \vee \alpha' < \beta \longrightarrow g\ \alpha' = g0\ \alpha'$ **using** *d1 b11* **by** *force*
then have $\mathfrak{L}v\ g\ \alpha\ \beta = \mathfrak{L}v\ g0\ \alpha\ \beta$ **unfolding** *𝔏v-def* **by** *blast*
moreover have $(b'',d) \in (\mathfrak{L}v\ g0\ \alpha\ \beta)\hat{}*$ **using** *d2* **unfolding** *𝔇-def* **by** *blast*
ultimately show *?thesis* **by** *metis*
qed
ultimately show *?thesis* **unfolding** *𝔇-def* **by** *blast*
qed
moreover have $(c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ \beta\ \alpha$
proof −
have $(c,\ c') \in (\mathfrak{L}1\ g\ \beta)\hat{}*$
proof −
have ∀ α'. $\alpha' < \beta \longrightarrow g\ \alpha' = g0\ \alpha'$ **using** *d1 b11* **by** *force*
then have $\mathfrak{L}1\ g\ \beta = \mathfrak{L}1\ g0\ \beta$ **unfolding** *𝔏1-def* **by** *blast*
moreover have $(c,c') \in (\mathfrak{L}1\ g0\ \beta)\hat{}*$ **using** *d2* **unfolding** *𝔇-def* **by** *blast*
ultimately show *?thesis* **by** *metis*
qed
moreover have $(c',\ c'') \in (g\ \alpha)\hat{}=$
proof −
have $g\ \alpha = g0\ \alpha$ **using** *d1 b11* **by** *blast*
moreover have $(c',c'') \in (g0\ \alpha)\hat{}=$ **using** *d2* **unfolding** *𝔇-def* **by** *blast*
ultimately show *?thesis* **by** *metis*
qed
moreover have $(c'',\ d) \in (\mathfrak{L}v\ g\ \beta\ \alpha)\hat{}*$
proof −
have ∀ α'. $\alpha' < \alpha \vee \alpha' < \beta \longrightarrow g\ \alpha' = g0\ \alpha'$ **using** *d1 b11* **by** *force*
then have $\mathfrak{L}v\ g\ \beta\ \alpha = \mathfrak{L}v\ g0\ \beta\ \alpha$ **unfolding** *𝔏v-def* **by** *blast*
moreover have $(c'',d) \in (\mathfrak{L}v\ g0\ \beta\ \alpha)\hat{}*$ **using** *d2* **unfolding** *𝔇-def* **by** *blast*
ultimately show *?thesis* **by** *metis*
qed
ultimately show *?thesis* **unfolding** *𝔇-def* **by** *blast*
qed
ultimately show $\exists\, b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ \beta\ \alpha$ **by** *blast*
qed
moreover have $\alpha < n0 \wedge \neg\ (\beta < n0)$
$\longrightarrow (\exists\, b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ \beta\ \alpha)$
proof
assume *d1*: $\alpha < n0 \wedge \neg\ (\beta < n0)$
then have *d2*: $(a,\ b) \in g0\ \alpha \wedge (g\ \beta) = (r - s)$ **using** *c1 b11 b12* **by** *blast*
have *d3*: $(a,b) \in s \wedge (a,c) \in r - s$ **using** *d1 d2 c1 b2* **unfolding** *Field-def* **by** *blast*
then have $b \in Field\ s \wedge c \in Field\ s$ **using** *a4* **unfolding** *Field-def Inv-def* **by** *blast*
then obtain d **where** *d6*: $d \in Field\ s \wedge (b,d) \in s\hat{}* \wedge (c,d) \in s\hat{}*$
using *a1* **unfolding** *CCR-def* **by** *blast*
have ∀ α'. $\alpha' < n0 \longrightarrow \alpha' < \beta$ **using** *d1* **by** *force*
then have $s \subseteq \mathfrak{L}v\ g\ \alpha\ \beta \wedge s \subseteq \mathfrak{L}v\ g\ \beta\ \alpha$ **using** *b2 b11* **unfolding** *𝔏v-def*

by *blast*
then have $(b,d) \in (\mathfrak{L}v\ g\ \alpha\ \beta)\widehat{\ }*$ $\wedge$ $(c,d) \in (\mathfrak{L}v\ g\ \beta\ \alpha)\widehat{\ }*$ **using** *d6 rtrancl-mono*
by *blast*
then show $\exists\, b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ \beta$
α
unfolding *$\mathfrak{D}$-def* **by** *blast*
qed
moreover have $\neg\ (\alpha < n0) \wedge (\beta < n0) \longrightarrow False$ **using** *c0* **by** *force*
moreover have $\neg\ (\alpha < n0) \wedge \neg\ (\beta < n0)$
$\longrightarrow (\exists\, b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ \beta\ \alpha)$
proof
assume *d1*: $\neg\ (\alpha < n0) \wedge \neg\ (\beta < n0)$
then have *d2*: $(g\ \alpha) = (r - s) \wedge (g\ \beta) = (r - s)$ **using** *b12* **by** *blast*
then have *d3*: $b \in Field\ r \wedge c \in Field\ r$ **using** *c1* **unfolding** *Field-def* **by**
blast
obtain b'' **where** *d4*: $b'' \in Field\ s \wedge (b,b'') \in r\widehat{\ }= \wedge ((b,b'') \in s \longrightarrow b = b'')$
using *a3 d3* **unfolding** *Den-def*
by (*cases* $\exists\ b''.\ (b,b'') \in s$, *metis Domain.DomainI Field-def UnCI pair-in-Id-conv*,
blast)
obtain c'' **where** *d5*: $c'' \in Field\ s \wedge (c,c'') \in r\widehat{\ }= \wedge ((c,c'') \in s \longrightarrow c = c'')$
using *a3 d3* **unfolding** *Den-def*
by (*cases* $\exists\ c''.\ (c,c'') \in s$, *metis Domain.DomainI Field-def UnCI*
pair-in-Id-conv, *blast*)
obtain d **where** *d6*: $d \in Field\ s \wedge (b'',d) \in s\widehat{\ }* \wedge (c'',d) \in s\widehat{\ }*$
using *d4 d5 a1* **unfolding** *CCR-def* **by** *blast*
have $\forall\ \alpha'.\ \alpha' < n0 \longrightarrow \alpha' < \alpha$ **using** *d1* **by** *force*
then have $s \subseteq \mathfrak{L}v\ g\ \alpha\ \beta \wedge s \subseteq \mathfrak{L}v\ g\ \beta\ \alpha$ **using** *b2 b11* **unfolding** *$\mathfrak{L}v$-def*
by *blast*
then have $(b'',d) \in (\mathfrak{L}v\ g\ \alpha\ \beta)\widehat{\ }* \wedge (c'',d) \in (\mathfrak{L}v\ g\ \beta\ \alpha)\widehat{\ }*$ **using** *d6*
rtrancl-mono **by** *blast*
moreover have $(b,b'') \in (g\ \beta)\widehat{\ }=$ **using** *d2 d4* **by** *blast*
moreover have $(c,c'') \in (g\ \alpha)\widehat{\ }=$ **using** *d2 d5* **by** *blast*
ultimately show $\exists\, b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in$
$\mathfrak{D}\ g\ \beta\ \alpha$
unfolding *$\mathfrak{D}$-def* **by** *blast*
qed
ultimately show $\exists\, b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in$
$\mathfrak{D}\ g\ \beta\ \alpha$ **by** *blast*
qed
then have *DCR-generating g* **using** *lem-Ldo-ldogen-ord* **by** *blast*
moreover have $r = \bigcup\ \{r'.\ \exists \alpha'.\ \alpha' < n \wedge r' = g\ \alpha'\}$
proof −
have $r \subseteq \bigcup\ \{r'.\ \exists \alpha'.\ \alpha' < n \wedge r' = g\ \alpha'\}$
proof
fix p
assume *c1*: $p \in r$
have $\exists\ \alpha'.\ \alpha' < n \wedge p \in g\ \alpha'$
proof (*cases* $p \in s$)
assume $p \in s$

then obtain α' **where** $\alpha' < n0 \land p \in g\ \alpha'$ **using** *b2 b11* **by** *blast*
moreover then have $\alpha' < n$ **using** *b9* **by** *force*
ultimately show $\exists\ \alpha'.\ \alpha' < n \land p \in g\ \alpha'$ **by** *blast*
next
assume $p \notin s$
moreover have $\neg\ (\ n < n0)$ **using** *b9* **by** *simp*
ultimately have $p \in g\ n0$ **using** *c1 b12* **by** *blast*
then show $\exists\ \alpha'.\ \alpha' < n \land p \in g\ \alpha'$ **using** *b9* **by** *blast*
qed
then show $p \in \bigcup\ \{r'.\ \exists \alpha'.\ \alpha' < n \land r' = g\ \alpha'\}$ **by** *blast*
qed
moreover have $\forall\ \alpha'.\ g\ \alpha' \subseteq r$
proof
fix α'
have $\alpha' < n0 \longrightarrow g0\ \alpha' \subseteq r$ **using** *a2 b2* **by** *blast*
then show $g\ \alpha' \subseteq r$ **using** *b8* **by** (*cases* $\alpha' < n0$, *force+*)
qed
ultimately show *?thesis* **by** *force*
qed
moreover have *CCR r* **using** *a1 a2 a3 lem-Ldo-den-ccr-uset lem-rcc-uset-ne-ccr* **by** *blast*
ultimately show *?thesis* **unfolding** *b9 DCR-def* **by** *blast*
qed

lemma *lem-Ldo-sat-reduc*:
fixes *r s*::*'U rel* **and** *n*::*nat*
assumes *a1*: $s \in$ *Span r* **and** *a2*: *CCR s* $\land$ *DCR n s*
shows *CCR r* $\land$ *DCR* (*Suc n*) *r*
proof −
have *Field s* $\in$ *Inv* $(r - s)$ **using** *a1* **unfolding** *Span-def Inv-def Field-def* **by** *blast*
moreover have $s \subseteq r$ **and** *Field s* $\in$ *Den r* **using** *a1* **unfolding** *Span-def Den-def* **by** *blast+*
ultimately show *?thesis* **using** *a2 lem-Ldo-ds-reduc* **by** *blast*
qed

lemma *lem-Ldo-uset-reduc*:
fixes *r s*::*'U rel* **and** *n0*::*nat*
assumes *a1*: $s \in \mathfrak{U}\ r$ **and** *a2*: *DCR n0 s* **and** *a3*: $n0 \neq 0$
shows *DCR* (*Suc n0*) *r*
proof −
have *b0*: $s \subseteq r$ **using** *a1* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
obtain *g0* **where** *b1*: *DCR-generating g0*
and *b2*: $s = \bigcup\ \{r'.\ \exists \alpha'.\ \alpha' < n0 \land r' = g0\ \alpha'\}$
using *a2* **unfolding** *DCR-def* **by** *blast*
obtain *isd* **where** *b3*: $isd = (\lambda\ a\ i.\ \exists\ b \in Field\ s.\ (a, b) \in r^{\frown\frown}i \land (\forall\ i'.\ (\exists\ b \in Field\ s.\ (a, b) \in r^{\frown\frown}(i')) \longrightarrow i \leq i'))$ **by** *blast*
obtain *d* **where** *b4*: $d = (\lambda\ a.\ SOME\ i.\ isd\ a\ i)$ **by** *blast*
obtain *B* **where** *b5*: $B = (\lambda\ a.\ \{\ a'.\ (a, a') \in r\ \})$ **by** *blast*

```
  obtain H where b6: H = (λ a. { a′ ∈ B a. ∀ a′′ ∈ B a. (d a′) ≤ (d a′′) }) by
blast
  obtain D where b7: D = { a ∈ Field r − Field s. H a ≠ {}} by blast
  obtain h where h = (λ a. SOME a′. a′ ∈ H a) by blast
   then have b8: ∀ a ∈ D. h a ∈ H a using b7 someI-ex[of λ a′. a′ ∈ H -] by
force
  have q1: ⋀ a. a ∈ Field r ⟹ isd a (d a)
  proof −
    fix a
    assume c1: a ∈ Field r
   then obtain b where c2: b ∈ Field s ∧ (a,b) ∈ r^* using a1 unfolding 𝔘-def
by blast
   moreover obtain N where c3: N = {i. ∃ b ∈ Field s. (a, b) ∈ r^^i} by blast
    ultimately have N ≠ {} using rtrancl-imp-relpow by blast
    then obtain m where m ∈ N ∧ (∀ i ∈ N. m ≤ i)
      using LeastI[of λ x. x ∈ N] Least-le[of λ x. x ∈ N] by blast
    then have isd a m using c2 c3 unfolding b3 by blast
    then show isd a (d a) using b4 someI-ex by metis
  qed
  have q2: ⋀ a. B a ≠ {} ⟹ H a ≠ {}
  proof −
    fix a
    assume B a ≠ {}
    moreover obtain N where c1: N = d ‘ (B a) by blast
    ultimately have N ≠ {}  by blast
    then obtain m where c2: m ∈ N ∧ (∀ i ∈ N. m ≤ i)
      using LeastI[of λ x. x ∈ N] Least-le[of λ x. x ∈ N] by blast
    then obtain a′ where c3: m = d a′ ∧ a′ ∈ B a using c1 by blast
    moreover then have ∀ a′′ ∈ B a. d a′ ≤ d a′′ using c1 c2 by force
    ultimately have a′ ∈ H a unfolding b6 by blast
    then show H a ≠ {} by blast
  qed
  have q3: ∀ a ∈ Field r − Field s. d a = 1 ∨ d a > 1
  proof
    fix a
    assume c1: a ∈ Field r − Field s
    then have isd a (d a) using q1 by blast
    then obtain b where b ∈ Field s ∧ (a, b) ∈ r^^(d a) using b3 by blast
    then have d a = 0 ⟶ False using c1 by force
    then show d a = 1 ∨ d a > 1 by force
  qed
  have Field r − Field s ⊆ D
  proof
    fix a
    assume c1: a ∈ Field r − Field s
    moreover have H a = {} ⟶ False
    proof
      assume H a = {}
      then have B a = {} using q2 by blast
```

moreover obtain *b* **where** *b* ∈ *Field s* ∧ (*a*, *b*) ∈ *r*^* **using** *a1 c1* **unfolding** 𝔘*-def* **by** *blast*
ultimately have *a* ∈ *Field s* **unfolding** *b5* **by** (*metis Collect-empty-eq converse-rtranclE*)
then show *False* **using** *c1* **by** *blast*
qed
ultimately show *a* ∈ *D* **using** *b7* **by** *blast*
qed
then have *q4*: *D* = *Field r* − *Field s* **using** *b5 b6 b7* **by** *blast*
have *q5*: ∀ *a* ∈ *D*. *d a* > *1* ⟶ *d a* = *Suc* (*d* (*h a*)) ∧ (*d* (*h a*) > *1* ⟶ *h a* ∈ *D*)
proof (*intro ballI impI*)
fix *a*
assume *c1*: *a* ∈ *D* **and** *c2*: *d a* > *1*
then obtain *b* **where** *c3*: *b* ∈ *Field s* **and** *c4*: (*a*, *b*) ∈ *r*^^(*d a*)
and *c5*: ∀ *i′*. (∃ *b* ∈ *Field s*. (*a*, *b*) ∈ *r*^^(*i′*)) ⟶ (*d a*) ≤ *i′*
using *b3 b7 q1* **by** *blast*
have *c6*: *d a* ≥ *1* **using** *c1 c4 b7 q3* **by** *force*
then have *d a* = *Suc* ((*d a*) − *1*) **by** *simp*
then obtain *a′* **where** *c7*: (*a*,*a′*) ∈ *r* ∧ (*a′*,*b*) ∈ *r*^^((*d a*) − *1*)
using *c4 relpow-Suc-D2*[*of a b d a* − *1 r*] **by** *metis*
moreover then have *a′* ∉ *Field s* **using** *c2 c5* **by** (*metis less-Suc-eq-le not-less-eq relpow-1*)
ultimately have (*a*,*a′*) ∈ *r* ∧ *a′* ∈ *Field r* − *Field s* **unfolding** *Field-def* **by** *blast*
then have *a′* ∈ *B a* **unfolding** *b5* **by** *blast*
moreover have *h a* ∈ *H a* **using** *c1 b8* **by** *blast*
ultimately have *d* (*h a*) ≤ *d a′* **unfolding** *b6* **by** *blast*
moreover have *Suc* (*d a′*) ≤ *d a*
proof −
have *d a′* ≤ *d a* − *1* **using** *q1 b3 c7 c3* **unfolding** *Field-def* **by** *blast*
then show *?thesis* **using** *c6* **by** *force*
qed
moreover have *d a* ≤ (*Suc* (*d* (*h a*)))
proof −
have *d1*: (*a*, *h a*) ∈ *r* **using** *c1 b5 b6 b8* **by** *blast*
then have *h a* ∈ *Field r* **unfolding** *Field-def* **by** *blast*
then obtain *b′* **where** *b′* ∈ *Field s* ∧ ((*h a*), *b′*) ∈ *r*^^(*d* (*h a*)) **using** *b3 q1* **by** *blast*
moreover then have (*a*,*b′*) ∈ *r*^^(*Suc* (*d* (*h a*))) **using** *d1 c7* **by** (*meson relpow-Suc-I2*)
ultimately show *d a* ≤ (*Suc* (*d* (*h a*))) **using** *c5* **by** *blast*
qed
ultimately have *d a* = *Suc* (*d* (*h a*)) **by** *force*
moreover have *d* (*h a*) > *1* ⟶ *h a* ∈ *D*
proof
assume *d1*: *d* (*h a*) > *1*
then have *d2*: (*a*, *h a*) ∈ *r* **using** *c1 b5 b6 b8* **by** *simp*
then have *isd* (*h a*) (*d* (*h a*)) **using** *d1 q1* **unfolding** *Field-def* **by** *force*

then have $(h\ a) \notin$ *Field s* **using** *d1 b3* **by** *force*
then show $h\ a \in D$ **using** *d2 q4* **unfolding** *Field-def* **by** *blast*
qed
ultimately show $d\ a = Suc\ (d\ (h\ a)) \wedge (d\ (h\ a) > 1 \longrightarrow h\ a \in D)$ **by** *blast*
qed
obtain *g1* **where** *b9*: $g1 = \{ (a, b).\ a \in D \wedge b = h\ a \}$ **by** *blast*
have *q6*: $\forall\ a \in D.\ \exists\ a' \in D.\ d\ a' = 1 \wedge (a,a') \in g1\hat{}*$
proof −
have $\forall\ n.\ \forall\ a \in D.\ d\ a = Suc\ n \longrightarrow ((h\hat{}\hat{}n)\ a) \in D \wedge d\ ((h\hat{}\hat{}n)\ a) = 1$
proof
fix *n0*
show $\forall\ a \in D.\ d\ a = Suc\ n0 \longrightarrow ((h\hat{}\hat{}n0)\ a) \in D \wedge d\ ((h\hat{}\hat{}n0)\ a) = 1$
proof (*induct n0*)
show $\forall a \in D.\ d\ a = Suc\ 0 \longrightarrow ((h\hat{}\hat{}0)\ a) \in D \wedge d\ ((h\ \hat{}\hat{}\ 0)\ a) = 1$
using *q4* **by** *force*
next
fix *n*
assume *d1*: $\forall a \in D.\ d\ a = Suc\ n \longrightarrow ((h\hat{}\hat{}n)\ a) \in D \wedge d\ ((h\ \hat{}\hat{}\ n)\ a) = 1$
show $\forall a \in D.\ d\ a = Suc\ (Suc\ n) \longrightarrow ((h\hat{}\hat{}(Suc\ n))\ a) \in D \wedge d\ ((h\ \hat{}\hat{}\ Suc\ n)\ a) = 1$
proof (*intro ballI impI*)
fix *a*
assume *e1*: $a \in D$ **and** *e2*: $d\ a = Suc\ (Suc\ n)$
then have $d\ a = Suc\ (d\ (h\ a)) \wedge (d\ (h\ a) > 1 \longrightarrow h\ a \in D)$ **using** *q5* **by** *simp*
moreover then have *e3*: $d\ (h\ a) = Suc\ n$ **using** *e2* **by** *simp*
ultimately have $d\ (h\ a) > 1 \longrightarrow ((h\hat{}\hat{}n)\ (h\ a)) \in D \wedge d\ ((h\hat{}\hat{}n)\ (h\ a)) = 1$ **using** *d1* **by** *blast*
moreover have $(h\hat{}\hat{}n)\ (h\ a) = (h\hat{}\hat{}(Suc\ n))\ a$ **by** (*metis comp-apply funpow-Suc-right*)
moreover have *e4*: $d\ (h\ a) = 1 \longrightarrow d\ ((h\hat{}\hat{}(Suc\ n))\ a) = 1$ **using** *e3* **by** *simp*
moreover have $d\ (h\ a) = 1 \longrightarrow ((h\hat{}\hat{}(Suc\ n))\ a) \in D$
proof
assume *f1*: $d\ (h\ a) = 1$
then have *f2*: $n = 0 \wedge (a,\ h\ a) \in r$ **using** *e1 e3 b5 b6 b8* **by** *simp*
then have *isd* $(h\ a)\ 1$ **using** *f1 q1* **unfolding** *Field-def* **by** *force*
then have $(h\ a) \notin$ *Field s* **using** *b3* **by** *force*
then have $(h\ a) \in D$ **using** *q4 f2* **unfolding** *Field-def* **by** *blast*
then show $((h\hat{}\hat{}(Suc\ n))\ a) \in D$ **using** *f2* **by** *simp*
qed
moreover have $d\ (h\ a) = 1 \vee d\ (h\ a) > 1$ **using** *e3* **by** *force*
ultimately show $((h\hat{}\hat{}(Suc\ n))\ a) \in D \wedge d\ ((h\ \hat{}\hat{}\ (Suc\ n))\ a) = 1$ **by** *force*
qed
qed
qed
moreover have $\forall\ i.\ \forall\ a \in D.\ d\ a > i \longrightarrow (a,\ (h\hat{}\hat{}i)\ a) \in g1\hat{}*$
proof

fix *i0*
show ∀ a ∈ D. d a > i0 ⟶ (a, (h^^i0) a) ∈ g1^*
proof (*induct i0*)
show ∀ a∈D. d a > 0 ⟶ (a, (h^^0) a) ∈ g1^* **by** *force*
next
fix *i*
assume *d1*: ∀ a∈D. d a > i ⟶ (a, (h^^i) a) ∈ g1^*
show ∀ a∈D. d a > (Suc i) ⟶ (a, (h^^(Suc i)) a) ∈ g1^*
proof (*intro ballI impI*)
fix *a*
assume *e1*: a ∈ D **and** *e2*: d a > (Suc i)
then have *e3*: d a = Suc (d (h a)) ∧ (d (h a) > 1 ⟶ h a ∈ D) **using** *q5* **by** *simp*
moreover then have *e4*: d (h a) > i **using** *e2* **by** *simp*
ultimately have d (h a) > 1 ⟶ (h a, (h^^i) (h a)) ∈ g1^* **using** *d1* **by** *simp*
moreover have (h^^i) (h a) = (h^^(Suc i)) a **by** (*metis comp-apply funpow-Suc-right*)
moreover have d (h a) = 1 ⟶ (h^^(Suc i)) a = (h a) **using** *e4* **by** *force*
moreover have d (h a) = 1 ∨ d (h a) > 1 **using** *e4* **by** *force*
moreover then have (a, h a) ∈ g1 **using** *e1 e3* **unfolding** *b9* **by** *simp*
ultimately show (a, (h^^(Suc i)) a) ∈ g1^*
by (*metis converse-rtrancl-into-rtrancl r-into-rtrancl*)
qed
qed
qed
ultimately have ∀ n. ∀ a∈D. d a = Suc n ⟶ (h^^n) a ∈ D ∧ d ((h^^n) a) = 1 ∧ (a, (h ^^ n) a) ∈ g1^*
by *simp*
then have ∀ n. ∀ a∈D. d a = Suc n ⟶ (∃ a′ ∈ D. d a′ = 1 ∧ (a,a′) ∈ g1^*)
by *blast*
moreover have ∀ a ∈ D. ∃ n. d a = Suc n **using** *q3 q4 q5* **by** *force*
ultimately show *?thesis* **by** *blast*
qed
let ?cond1 = λ α. α = 0
let ?cond3 = λ α. (1 ≤ α ∧ α < n0)
obtain g :: nat ⇒ ′U rel
where *b12*: g = (λ α. if (?cond1 α) then (g0 α) ∪ g1
else (if (?cond3 α) then (g0 α)
else {})) **by** *blast*
obtain n :: nat **where** *b13*: n = n0 **by** *blast*
then have *b14*: ⋀ α. α < n ⟹ (?cond1 α ∨ ?cond3 α) **by** *force*
have *b15*: ⋀ α. ?cond1 α ⟹ g α = (g0 α) ∪ g1 **using** *b12* **by** *simp*
have *b17*: ⋀ α. ?cond3 α ⟹ g α = (g0 α) **using** *b12* **by** *force*
obtain *r1* **where** *b19*: r1 = ⋃ {r′. ∃ α′. α′ < n ∧ r′ = g α′} **by** *blast*
have *t1*: g1 ⊆ r1 **using** *b15 b19 b13 a3* **by** *blast*
have *b20*: s ⊆ r1
proof

fix *p*
assume $p \in s$
then obtain α' **where** *c1*: $\alpha' < n0 \wedge p \in g0\ \alpha'$ **using** *b2* **by** *blast*
then have *c2*: $\alpha' < n$ **unfolding** *b13* **by** *fastforce*
then have *?cond1* $\alpha' \vee$ *?cond3* α' **using** *b14* **by** *blast*
then have $g0\ \alpha' \subseteq g\ \alpha'$ **using** *b12* **by** *fastforce*
then show $p \in r1$ **using** *c1 c2 b19* **by** *blast*
qed
have *b21*: $r1 \subseteq r$
proof −
have $\forall\ r'\ \alpha'.\ \alpha' < n \longrightarrow g\ \alpha' \subseteq r$
proof (*intro allI impI*)
fix $r'\ \alpha'$
assume *d1*: $\alpha' < n$
have $\forall\ a \in D.\ (a,\ h\ a) \in r$ **using** *b5 b6 b8* **by** *blast*
then have *d2*: $g1 \subseteq r$ **using** *b9* **by** *blast*
have $(\alpha' = 0) \longrightarrow g\ \alpha' \subseteq r$ **using** *d2 b0 b2 b15*[*of* α'] *a3* **by** *blast*
moreover have $1 \leq \alpha' \longrightarrow g\ \alpha' \subseteq r$ **using** *b17 b0 b2 b13 d1* **by** *blast*
ultimately show $g\ \alpha' \subseteq r$ **using** *d1 b14* **by** *blast*
qed
then show $r1 \subseteq r$ **unfolding** *b19* **by** *fast*
qed
have *b22*: $\forall a \in Field\ r1 - Field\ s.\ \exists b \in Field\ s.\ (a,\ b) \in r1\hat{}*$
proof
fix *a*
assume *d1*: $a \in Field\ r1 - Field\ s$
then have $a \in D$ **using** *q4 b21* **unfolding** *Field-def* **by** *blast*
then obtain a' **where** *d2*: $a' \in D \wedge d\ a' = 1 \wedge (a,\ a') \in g1\hat{}*$ **using** *q6* **by** *blast*
then have *d3*: $(a',\ h\ a') \in r1 \wedge h\ a' \in H\ a'$ **using** *q4 b8 b9 t1 a3* **by** *blast*
obtain *b* **where** $b \in Field\ s \wedge (a',b) \in r$ **using** *d2 q1 q4 b3* **by** *force*
moreover then have *isd* $b\ (d\ b)$ **using** *q1* **unfolding** *Field-def* **by** *blast*
ultimately have $b \in B\ a' \wedge d\ b = 0$ **using** *b3 b5* **by** *force*
then have $d\ (h\ a') = 0$ **using** *d3 b6* **by** *force*
then have *isd* $(h\ a')\ 0$ **using** *q1 d3 b21 a3* **unfolding** *Field-def* **by** *force*
then have $h\ a' \in Field\ s$ **using** *b3* **by** *force*
moreover have $(a,\ a') \in r1\hat{}*$ **using** *d2 t1 rtrancl-mono*[*of g1 r1*] *a3* **by** *blast*
ultimately have $(h\ a') \in Field\ s \wedge (a,\ h\ a') \in r1\hat{}*$ **using** *d3* **by** *force*
then show $\exists b \in Field\ s.\ (a,\ b) \in r1\hat{}*$ **by** *blast*
qed
have *b23*: $Field\ r \subseteq Field\ r1$
proof −
have $(Field\ r - Field\ s) \subseteq Field\ r1$ **using** *q4 b9 t1* **unfolding** *Field-def* **by** *blast*
moreover have $Field\ s \subseteq Field\ r1$ **using** *b20* **unfolding** *Field-def* **by** *blast*
ultimately show $Field\ r \subseteq Field\ r1$ **by** *blast*
qed
have $\forall \alpha\ \beta\ a\ b\ c.\ \alpha \leq \beta \longrightarrow (a,b) \in g\ \alpha \wedge (a,c) \in g\ \beta \longrightarrow$
$(\exists b'\ b''\ c'\ c''\ d.\ (b,b',b'',d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,c',c'',d) \in \mathfrak{D}\ g\ \beta\ \alpha)$

proof (*intro allI impI*)
fix $\alpha\ \beta\ a\ b\ c$
assume *c1*: $\alpha \leq \beta$ **and** *c2*: $(a,b) \in g\ \alpha \wedge (a,c) \in g\ \beta$
obtain *c123* **where** *c0*: $c123 = (\lambda\ \alpha{::}nat.\ ?cond1\ \alpha \vee ?cond3\ \alpha)$ **by** *blast*
have *c3*: $\bigwedge \alpha'.\ c123\ \alpha' \Longrightarrow g0\ \alpha' \subseteq s$
proof −
fix α'
assume $c123\ \alpha'$
moreover have $?cond1\ \alpha' \longrightarrow g0\ \alpha' \subseteq s$ **using** *a3* **unfolding** *b2* **by** *force*
moreover have $?cond3\ \alpha' \longrightarrow g0\ \alpha' \subseteq s$ **using** *b2* **by** *force*
ultimately show $g0\ \alpha' \subseteq s$ **using** *c0* **by** *blast*
qed
have *c4*: $\bigwedge\alpha'.\ \bigwedge p.\ p \in g\ \alpha' \longrightarrow (?cond1\ \alpha' \wedge p \in (g0\ \alpha' \cup g1)) \vee (?cond3\ \alpha' \wedge p \in (g0\ \alpha'))$
proof (*intro impI*)
fix $\alpha'\ p$
assume $p \in g\ \alpha'$
then show $(?cond1\ \alpha' \wedge p \in (g0\ \alpha' \cup g1)) \vee (?cond3\ \alpha' \wedge p \in (g0\ \alpha'))$
using *b12* **by** (*cases* $?cond1\ \alpha'$, *simp*, *cases* $?cond3\ \alpha'$, *force+*)
qed
have *c5*: $\bigwedge \alpha'\ \beta'.\ \alpha' \leq \beta' \Longrightarrow c123\ \beta' \Longrightarrow c123\ \alpha'$ **unfolding** *c0* **using** *b14* **by** *force*
have *c6*: $(a,b) \in g0\ \alpha \wedge (a,c) \notin g0\ \beta \longrightarrow \neg\ c123\ \beta$
proof
assume *d1*: $(a,b) \in g0\ \alpha \wedge (a,c) \notin g0\ \beta$
then have $(a,c) \in g1$ **using** *c2 c4* **by** *blast*
then have $a \in Field\ r - Field\ s$ **using** *b7 b9* **by** *blast*
then have $\neg\ c123\ \alpha$ **using** *d1 c3* **unfolding** *Field-def* **by** *blast*
then show $\neg\ c123\ \beta$ **using** *c1 c5* **by** *blast*
qed
have *c7*: $(a,b) \notin g0\ \alpha \wedge (a,c) \in g0\ \beta \longrightarrow \neg\ c123\ \beta$
proof
assume *d1*: $(a,b) \notin g0\ \alpha \wedge (a,c) \in g0\ \beta$
then have $(a,b) \in g1$ **using** *c2 c4* **by** *blast*
then have $a \in Field\ r - Field\ s$ **using** *b7 b9* **by** *blast*
then show $\neg\ c123\ \beta$ **using** *d1 c3* **unfolding** *Field-def* **by** *blast*
qed
have *c8*: $\bigwedge \alpha'.\ c123\ \alpha' \Longrightarrow g0\ \alpha' \subseteq g\ \alpha'$
proof −
fix α'
assume $c123\ \alpha'$
then show $g0\ \alpha' \subseteq g\ \alpha'$ **unfolding** *c0* **using** *b15*[*of* α'] *b17*[*of* α'] **by** *blast*
qed
then have *c9*: $\bigwedge \alpha'\ \alpha''.\ c123\ \alpha' \Longrightarrow \alpha'' < \alpha' \Longrightarrow g0\ \alpha'' \subseteq g\ \alpha''$
using *c5 less-or-eq-imp-le* **by** *blast*
have *c10*: $\bigwedge \alpha'\ \beta'.\ c123\ \alpha' \Longrightarrow c123\ \beta' \Longrightarrow \mathfrak{D}\ g0\ \alpha'\ \beta' \subseteq \mathfrak{D}\ g\ \alpha'\ \beta'$
proof −
fix $\alpha'\ \beta'$
assume *d1*: $c123\ \alpha'$ **and** *d2*: $c123\ \beta'$

have $\mathfrak{L}1\ g0\ \alpha' \subseteq \mathfrak{L}1\ g\ \alpha'$ **using** *d1 c9* **unfolding** *𝔏1-def* **by** *blast*
moreover have $\mathfrak{L}v\ g0\ \alpha'\ \beta' \subseteq \mathfrak{L}v\ g\ \alpha'\ \beta'$ **using** *d1 d2 c9* **unfolding** *𝔏v-def* **by** *blast*
ultimately have $(\mathfrak{L}1\ g0\ \alpha')\hat{}* \subseteq (\mathfrak{L}1\ g\ \alpha')\hat{}* \wedge (\mathfrak{L}v\ g0\ \alpha'\ \beta')\hat{}* \subseteq (\mathfrak{L}v\ g\ \alpha'\ \beta')\hat{}*$
using *rtrancl-mono* **by** *blast*
moreover have $g0\ \beta' \subseteq g\ \beta'$ **using** *d2 c8* **by** *blast*
ultimately show $\mathfrak{D}\ g0\ \alpha'\ \beta' \subseteq \mathfrak{D}\ g\ \alpha'\ \beta'$ **unfolding** *𝔇-def* **by** *blast*
qed
show $\exists\, b'\ b''\ c'\ c''\ d'.\ (b,b',b'',d') \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,c',c'',d') \in \mathfrak{D}\ g\ \beta\ \alpha$
proof (*cases c123* β)
assume *d1*: *c123* β
show *?thesis*
proof (*cases* $(a,b) \in g0\ \alpha \wedge (a,c) \in g0\ \beta$)
assume *e1*: $(a,b) \in g0\ \alpha \wedge (a,c) \in g0\ \beta$
then obtain $b'\ b''\ c'\ c''\ d'$ **where** $(b,\ b',\ b'',\ d') \in \mathfrak{D}\ g0\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d') \in \mathfrak{D}\ g0\ \beta\ \alpha$
using *b1* **unfolding** *DCR-generating-def* **by** *blast*
moreover have *c123* α **using** *d1 c1 c5* **by** *blast*
ultimately have $(b,\ b',\ b'',\ d') \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d') \in \mathfrak{D}\ g\ \beta\ \alpha$ **using** *d1 c10* **by** *blast*
then show *?thesis* **by** *blast*
next
assume $\neg\ ((a,b) \in g0\ \alpha \wedge (a,c) \in g0\ \beta)$
then have $(a,b) \notin g0\ \alpha \wedge (a,c) \notin g0\ \beta$ **using** *d1 c6 c7* **by** *blast*
moreover have *c123* α **using** *d1 c1 c5* **by** *blast*
ultimately have $(a,b) \in g1 \wedge (a,c) \in g1$ **using** *d1 c0 c2 c4* **by** *blast*
then have $b = c$ **using** *b9* **by** *blast*
then show *?thesis* **unfolding** *𝔇-def* **by** *blast*
qed
next
assume *d1*: $\neg$ *c123* β
then have *d2*: *False* **using** *c2 c4* **unfolding** *c0* **by** *blast*
then show *?thesis* **by** *blast*
qed
qed
then have *b24*: *DCR-generating g* **using** *a3 lem-Ldo-ldogen-ord* **by** *blast*
moreover then have *Field r1* $\subseteq$ *Field r* **using** *b21* **unfolding** *Field-def* **by** *blast*
ultimately have *r1* $\in$ *Span r* **using** *b21 b23* **unfolding** *Span-def* **by** *blast*
moreover have *DCR n r1* **using** *b19 b24* **unfolding** *DCR-def* **by** *blast*
moreover have *CCR r1*
proof −
have $s \in \mathfrak{U}\ r1$ **using** *b20 b22 a1* **unfolding** *𝔘-def* **by** *blast*
then show *CCR r1* **using** *lem-rcc-uset-ne-ccr* **by** *blast*
qed
ultimately show *DCR* (*Suc n0*) *r* **using** *b13 a3 lem-Ldo-sat-reduc* **by** *blast*
qed

lemma *lem-Ldo-addid*:
fixes r::$'U$ *rel* **and** r'::$'U$ *rel* **and** *n0*::*nat* **and** A::$'U$ *set*
assumes *a1*: *DCR n0 r* **and** *a2*: $r' = r \cup \{(a,b).\ a = b \wedge a \in A\}$ **and** *a3*: $n0 \neq 0$
shows *DCR n0 r'*
proof −
obtain *g0* **where** *b1*: *DCR-generating g0* **and** *b2*: $r = \bigcup\{r'.\ \exists\alpha'<n0.\ r' = g0\ \alpha'\}$ **using** *a1* **unfolding** *DCR-def* **by** *blast*
obtain g :: *nat* $\Rightarrow$ $'U$ *rel* **where** *b3*: $g = (\lambda\ \alpha.\ (g0\ \alpha) \cup \{(a,b).\ a = b \wedge a \in A\})$ **by** *blast*
have $\forall\alpha\ \beta\ a\ b\ c.\ \alpha \leq \beta \longrightarrow (a,b) \in g\ \alpha \wedge (a,c) \in g\ \beta \longrightarrow$
$(\exists b'\ b''\ c'\ c''\ d.\ (b,b',b'',d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,c',c'',d) \in \mathfrak{D}\ g\ \beta\ \alpha)$
proof (*intro allI impI*)
fix $\alpha\ \beta\ a\ b\ c$
assume *c1*: $\alpha \leq \beta$ **and** *c2*: $(a,b) \in g\ \alpha \wedge (a,c) \in g\ \beta$
have *c3*: $\bigwedge \alpha'\ \beta'.\ \mathfrak{D}\ g0\ \alpha'\ \beta' \subseteq \mathfrak{D}\ g\ \alpha'\ \beta'$
proof −
fix $\alpha'\ \beta'$
have $\mathfrak{L}1\ g0\ \alpha' \subseteq (\mathfrak{L}1\ g\ \alpha')^{=}$ **unfolding** *𝔏1-def b3* **by** (*clarsimp*, *auto*)
moreover have $\mathfrak{L}v\ g0\ \alpha'\ \beta' \subseteq (\mathfrak{L}v\ g\ \alpha'\ \beta')^{=}$ **unfolding** *𝔏v-def b3* **by** (*clarsimp*, *auto*)
ultimately have $(\mathfrak{L}1\ g0\ \alpha')^{*} \subseteq (\mathfrak{L}1\ g\ \alpha')^{*} \wedge (\mathfrak{L}v\ g0\ \alpha'\ \beta')^{*} \subseteq (\mathfrak{L}v\ g\ \alpha'\ \beta')^{*}$ **using** *rtrancl-reflcl rtrancl-mono* **by** *blast*
moreover have $(g0\ \beta')^{=} \subseteq (g\ \beta')^{=}$ **unfolding** *b3* **by** *force*
ultimately show $\mathfrak{D}\ g0\ \alpha'\ \beta' \subseteq \mathfrak{D}\ g\ \alpha'\ \beta'$ **unfolding** *𝔇-def* **by** *blast*
qed
have *c4*: $((a,b) \in g0\ \alpha \vee a = b) \wedge ((a,c) \in g0\ \beta \vee a = c)$ **using** *c1 c2 b3* **by** *blast*
moreover then have $a = b \vee a = c \longrightarrow (\exists b'\ b''\ c'\ c''\ d.\ (b,b',b'',d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,c',c'',d) \in \mathfrak{D}\ g\ \beta\ \alpha)$
using *b3* **unfolding** *𝔇-def* **by** *blast*
moreover have $(a,b) \in g0\ \alpha \wedge (a,c) \in g0\ \beta \longrightarrow (\exists b'\ b''\ c'\ c''\ d.\ (b,b',b'',d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,c',c'',d) \in \mathfrak{D}\ g\ \beta\ \alpha)$
proof
assume $(a,b) \in g0\ \alpha \wedge (a,c) \in g0\ \beta$
then obtain $b'\ b''\ c'\ c''\ d'$ **where** $(b, b', b'', d') \in \mathfrak{D}\ g0\ \alpha\ \beta \wedge (c, c', c'', d') \in \mathfrak{D}\ g0\ \beta\ \alpha$
using *b1* **unfolding** *DCR-generating-def* **by** *blast*
then have $(b, b', b'', d') \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c, c', c'', d') \in \mathfrak{D}\ g\ \beta\ \alpha$ **using** *c3* **by** *blast*
then show $\exists b'\ b''\ c'\ c''\ d'.\ (b,b',b'',d') \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,c',c'',d') \in \mathfrak{D}\ g\ \beta\ \alpha$ **by** *blast*
qed
ultimately show $\exists b'\ b''\ c'\ c''\ d.\ (b,b',b'',d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,c',c'',d) \in \mathfrak{D}\ g\ \beta\ \alpha$ **by** *blast*
qed
then have *DCR-generating g* **using** *lem-Ldo-ldogen-ord* **by** *blast*
moreover have $r' = \bigcup\{s.\ \exists\alpha'<n0.\ s = g\ \alpha'\}$ **unfolding** *b2 b3 a2* **using** *a3* **by** *blast*

ultimately show *DCR n0 r'* **unfolding** *DCR-def* **by** *blast*
qed

lemma *lem-Ldo-removeid*:
fixes *r*::*'U rel* **and** *r'*::*'U rel* **and** *n0*::*nat*
assumes *a1*: *DCR n0 r* **and** *a2*: $r' = r - \{(a,b).\ a = b\}$
shows *DCR n0 r'*
proof −
obtain *g0* **where** *b1*: *DCR-generating g0* **and** *b2*: $r = \bigcup\{r'.\ \exists\alpha'<n0.\ r' = g0\ \alpha'\}$ **using** *a1* **unfolding** *DCR-def* **by** *blast*
obtain *g* :: *nat* ⇒ *'U rel* **where** *b3*: $g = (\lambda\ \alpha.\ (g0\ \alpha) - \{(a,b).\ a = b\ \})$ **by** *blast*
have $\forall\alpha\ \beta\ a\ b\ c.\ \alpha \le \beta \longrightarrow (a,b) \in g\ \alpha \land (a,c) \in g\ \beta \longrightarrow$
$(\exists\, b'\ b''\ c'\ c''\ d.\ (b,b',b'',d) \in \mathfrak{D}\ g\ \alpha\ \beta \land (c,c',c'',d) \in \mathfrak{D}\ g\ \beta\ \alpha)$
proof (*intro allI impI*)
fix $\alpha\ \beta\ a\ b\ c$
assume *c1*: $\alpha \le \beta$ **and** *c2*: $(a,b) \in g\ \alpha \land (a,c) \in g\ \beta$
have *c3*: $\bigwedge\alpha'\ \beta'.\ \mathfrak{D}\ g0\ \alpha'\ \beta' \subseteq \mathfrak{D}\ g\ \alpha'\ \beta'$
proof −
fix $\alpha'\ \beta'$
have $\mathfrak{L}1\ g0\ \alpha' \subseteq (\mathfrak{L}1\ g\ \alpha')^{=}$ **unfolding** *𝔏1-def b3* **by** (*clarsimp*, *auto*)
moreover have $\mathfrak{L}v\ g0\ \alpha'\ \beta' \subseteq (\mathfrak{L}v\ g\ \alpha'\ \beta')^{=}$ **unfolding** *𝔏v-def b3* **by** (*clarsimp*, *auto*)
ultimately have $(\mathfrak{L}1\ g0\ \alpha')^{*} \subseteq (\mathfrak{L}1\ g\ \alpha')^{*} \land (\mathfrak{L}v\ g0\ \alpha'\ \beta')^{*} \subseteq (\mathfrak{L}v\ g\ \alpha'\ \beta')^{*}$ **using** *rtrancl-reflcl rtrancl-mono* **by** *blast*
moreover have $(g0\ \beta')^{=} \subseteq (g\ \beta')^{=}$ **unfolding** *b3* **by** *force*
ultimately show $\mathfrak{D}\ g0\ \alpha'\ \beta' \subseteq \mathfrak{D}\ g\ \alpha'\ \beta'$ **unfolding** *𝔇-def* **by** *blast*
qed
have $(a,b) \in g0\ \alpha \land (a,c) \in g0\ \beta$ **using** *c1 c2 b3* **by** *blast*
then obtain $b'\ b''\ c'\ c''\ d'$ **where** $(b,\ b',\ b'',\ d') \in \mathfrak{D}\ g0\ \alpha\ \beta \land (c,\ c',\ c'',\ d') \in \mathfrak{D}\ g0\ \beta\ \alpha$
using *b1* **unfolding** *DCR-generating-def* **by** *blast*
then have $(b,\ b',\ b'',\ d') \in \mathfrak{D}\ g\ \alpha\ \beta \land (c,\ c',\ c'',\ d') \in \mathfrak{D}\ g\ \beta\ \alpha$ **using** *c3* **by** *blast*
then show $\exists\, b'\ b''\ c'\ c''\ d'.\ (b,b',b'',d') \in \mathfrak{D}\ g\ \alpha\ \beta \land (c,c',c'',d') \in \mathfrak{D}\ g\ \beta\ \alpha$ **by** *blast*
qed
then have *DCR-generating g* **using** *lem-Ldo-ldogen-ord* **by** *blast*
moreover have $r' = \bigcup\{s.\ \exists\alpha'<n0.\ s = g\ \alpha'\}$ **unfolding** *b2 b3 a2* **by** *blast*
ultimately show *DCR n0 r'* **unfolding** *DCR-def* **by** *blast*
qed

lemma *lem-Ldo-eqid*:
fixes *r*::*'U rel* **and** *r'*::*'U rel* **and** *n*::*nat*
assumes *a1*: *DCR n r* **and** *a2*: $r' - \{(a,b).\ a = b\} = r - \{(a,b).\ a = b\}$ **and** *a3*: $n \ne 0$
shows *DCR n r'*
proof −
obtain *r''* **where** *b1*: $r'' = r' - \{(a,b).\ a = b\}$ **by** *blast*

then have *DCR n r''* **using** *a1 a2 lem-Ldo-removeid* **by** *blast*
moreover have $r' = r'' \cup \{(a,b).\ a = b \wedge (a,a) \in r'\}$ **using** *b1* **by** *blast*
ultimately show *DCR n r'* **using** *lem-Ldo-addid*[*of n r'' r'* $\{a\ .\ (a,a) \in r'\}$] *a3* **by** *blast*
qed

lemma *lem-wdn-range-lb*: $A \subseteq$ *w-dncl r A*
unfolding *w-dncl-def dncl-def* $\mathcal{F}$*-def rpth-def* **by** *fastforce*

lemma *lem-wdn-range-ub*: *w-dncl r A* $\subseteq$ *dncl r A* **unfolding** *w-dncl-def* **by** *blast*

lemma *lem-wdn-mon*: $A \subseteq A' \Longrightarrow$ *w-dncl r A* $\subseteq$ *w-dncl r A'* **unfolding** *w-dncl-def dncl-def* **by** *blast*

lemma *lem-wdn-compl*:
fixes *r*::*'U rel* **and** *A*::*'U set*
shows *UNIV* $-$ *w-dncl r A* $= \{a.\ \exists\ b.\ b \notin$ *dncl r A* $\wedge (a,b) \in$ (*Restr r* (*UNIV*$-$*A*))^*}
proof
show *UNIV* $-$ *w-dncl r A* $\subseteq \{a.\ \exists\ b.\ b \notin$ *dncl r A* $\wedge (a,b) \in$ (*Restr r* (*UNIV*$-$*A*))^*}
proof
fix *x*
assume *c1*: $x \in$ *UNIV* $-$ *w-dncl r A*
show $x \in \{a.\ \exists\ b.\ b \notin$ *dncl r A* $\wedge (a,b) \in$ (*Restr r* (*UNIV*$-$*A*))^*}
proof (*cases* $x \in$ *dncl r A*)
assume $x \in$ *dncl r A*
then obtain *b F* **where** *d1*: $F \in \mathcal{F}$ *r x b* $\wedge\ b \notin$ *dncl r A* $\wedge\ F \cap A = \{\}$
using *c1* **unfolding** *w-dncl-def* **by** *blast*
then obtain *f n* **where** $f \in$ *rpth r x b n* $\wedge\ F = f$ ‘ $\{i.\ i \le n\}$ **unfolding** $\mathcal{F}$*-def* **by** *blast*
moreover then have $\forall\, i \le n.\ f\ i \notin A$ **using** *d1* **unfolding** *rpth-def* **by** *blast*
ultimately have $f \in$ *rpth* (*Restr r* (*UNIV*$-$*A*)) *x b n* **unfolding** *rpth-def* **by** *force*
then have $(x,b) \in$ (*Restr r* (*UNIV*$-$*A*))^* **using** *lem-ccext-rpth-rtr*[*of Restr r* (*UNIV*$-$*A*)] **by** *blast*
then show *?thesis* **using** *d1* **by** *blast*
next
assume $x \notin$ *dncl r A*
then show *?thesis* **unfolding** *w-dncl-def* **by** *blast*
qed
qed
next
show $\{a.\ \exists\ b.\ b \notin$ *dncl r A* $\wedge (a,b) \in$ (*Restr r* (*UNIV*$-$*A*))^*} $\subseteq$ *UNIV* $-$ *w-dncl r A*
proof
fix *x*
assume $x \in \{a.\ \exists\ b.\ b \notin$ *dncl r A* $\wedge (a,b) \in$ (*Restr r* (*UNIV*$-$*A*))^*}
then obtain *y* **where** *c1*: $y \notin$ *dncl r A* $\wedge (x,y) \in$ (*Restr r* (*UNIV*$-$*A*))^* **by** *blast*
obtain *f n* **where** *c2*: $f \in$ *rpth* (*Restr r* (*UNIV*$-$*A*)) *x y n* **using** *c1 lem-ccext-rtr-rpth*[*of*

x y] **by** *blast*
then have *c3*: $f \in$ *rpth r x y n* **unfolding** *rpth-def* **by** *blast*
obtain *F* **where** *c4*: $F = f`\{i.\ i \le n\}$ **by** *blast*
have $n = 0 \longrightarrow f\ 0 \notin A$ **using** *c1 c3* **unfolding** *rpth-def dncl-def* **by** *blast*
moreover have $\forall\ i<n.\ f\ i \notin A \wedge f\ (Suc\ i) \notin A$ **using** *c2* **unfolding** *rpth-def* **by** *blast*
moreover have $\forall\ i \le n.\ (n = 0 \vee (\exists\ j<n.\ (j=i \vee i=Suc\ j)))$
by (*metis le-eq-less-or-eq lessI less-Suc-eq-0-disj*)
ultimately have $\forall\ i \le n.\ f\ i \notin A$ **by** *blast*
then have $F \cap A = \{\}$ **using** *c4* **by** *blast*
moreover have $F \in \mathcal{F}\ r\ x\ y$ **using** *c3 c4* **unfolding** *$\mathcal{F}$-def* **by** *blast*
ultimately show $x \in UNIV - $ *w-dncl r A* **using** *c1* **unfolding** *w-dncl-def* **by** *blast*
qed
qed

lemma *lem-cowdn-uset*:
fixes *r*::*'U rel* **and** *A A' W*::*'U set*
assumes *a1*: *CCR* (*Restr r A'*) **and** *a2*: *escl r A A'* $\subseteq$ *A'*
and *a3*: $Q = A' -$ *dncl r A* **and** *a4*: $W = A' -$ *w-dncl r A* **and** *a5*: $Q \in SF\ r$
shows *Restr r Q* $\in \mathfrak{U}$ (*Restr r W*)
proof −
have *CCR* (*Restr r Q*) **using** *a1 a3 lem-Inv-ccr-restr-invdiff lem-Inv-dncl-invbk* **by** *blast*
moreover have *Restr r Q* $\subseteq$ *Restr r W* **using** *a3 a4 lem-wdn-range-ub*[*of r*] **by** *blast*
moreover have $\forall a \in Field$ (*Restr r W*). $\exists b \in Field$ (*Restr r Q*). $(a, b) \in$ (*Restr r W*)^*
proof
fix *a*
assume $a \in Field$ (*Restr r W*)
then have *c1*: $a \in W$ **unfolding** *Field-def* **by** *blast*
show $\exists b \in Field$ (*Restr r Q*). $(a, b) \in$ (*Restr r W*)^*
proof (*cases* $a \in Q$)
assume $a \in Q$
then show *?thesis* **using** *a5* **unfolding** *SF-def* **by** *blast*
next
assume $a \notin Q$
then obtain *b F* **where** *d1*: $a \in A' \wedge F \in \mathcal{F}\ r\ a\ b \wedge b \notin$ *dncl r A* $\wedge F \cap A = \{\}$
using *c1 a3 a4* **unfolding** *w-dncl-def* **by** *blast*
then have *d2*: *dnesc r A a* $\subseteq$ *escl r A A'* **unfolding** *escl-def* **by** *blast*
obtain *E* **where** *d3*: $E =$ *dnesc r A a* **by** *blast*
have *dnEsc r A a* $\neq \{\}$ **using** *d1* **unfolding** *dnEsc-def* **by** *blast*
then have $E \in$ *dnEsc r A a* **using** *d3 lem-dnEsc-ne*[*of r A*] **by** *blast*
then obtain *b'* **where** *d4*: $b' \notin$ *dncl r A* $\wedge E \in \mathcal{F}\ r\ a\ b' \wedge E \cap A = \{\}$
unfolding *dnEsc-def* **by** *blast*
have *d5*: $E \subseteq A'$ **using** *d2 d3 a2* **by** *blast*
have $b' \in E$ **using** *d4* **unfolding** *$\mathcal{F}$-def rpth-def* **by** *blast*

then have $b' \in$ *Field* (*Restr r Q*) **using** *d4 d5 a3 a5* **unfolding** *SF-def* **by** *blast*
moreover have $(a, b') \in$ (*Restr r W*)^*
proof −
obtain *f n* **where** *e1*: $f \in$ *rpth r a* b' *n* **and** *e2*: $E = f$ ‘ $\{i.\ i \leq n\}$
using *d4* **unfolding** $\mathcal{F}$*-def* **by** *blast*
have *e3*: $\forall\ i \leq n.\ f\ i \in W$
proof (*intro allI impI*)
fix *i*
assume *f1*: $i \leq n$
obtain *g* **where** *f2*: $g = (\lambda\ k.\ f\ (k + i))$ **by** *blast*
have $g\ 0 = f\ i$ **using** *f2* **by** *simp*
moreover have $g\ (n - i) = b'$ **using** *f1 f2 e1* **unfolding** *rpth-def* **by** *simp*
moreover have $\forall k<n-i.\ (g\ k,\ g\ (Suc\ k)) \in$ *Restr r* (*UNIV* − *A*)
proof (*intro allI impI*)
fix *k*
assume $k < n-i$
then have $(g\ k,\ g\ (Suc\ k)) \in$ (*Restr r E*) **using** *f2 e1 e2* **unfolding** *rpth-def* **by** *simp*
then show $(g\ k,\ g\ (Suc\ k)) \in$ *Restr r* (*UNIV* − *A*) **using** *d4* **by** *blast*
qed
ultimately have $g \in$ *rpth* (*Restr r* (*UNIV*−*A*)) (*f i*) b' $(n-i)$ **unfolding** *rpth-def* **by** *blast*
then have $(f\ i,\ b') \in$ (*Restr r* (*UNIV*−*A*))^* **using** *lem-ccext-rpth-rtr*[*of* - *f i b'*] **by** *blast*
then have $f\ i \notin$ *w-dncl r A* **using** *d4 lem-wdn-compl*[*of r A*] **by** *blast*
then show $f\ i \in W$ **using** *f1 e2 d5 a4* **by** *blast*
qed
have $\forall i<n.\ (f\ i,\ f\ (Suc\ i)) \in$ *Restr r W*
proof (*intro allI impI*)
fix *i*
assume $i < n$
moreover then have $f\ i \in W \wedge f\ (Suc\ i) \in W$ **using** *e2 e3* **by** *force*
ultimately show $(f\ i,\ f\ (Suc\ i)) \in$ *Restr r W* **using** *e1* **unfolding** *rpth-def* **by** *blast*
qed
then have $E \in \mathcal{F}$ (*Restr r W*) *a* b' **using** *e1 e2* **unfolding** *rpth-def* $\mathcal{F}$*-def* **by** *blast*
then show *?thesis* **using** *lem-ccext-rtr-Fne*[*of a b'*] **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed
qed
ultimately show *?thesis* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
qed

lemma *lem-shrel-L-eq*:
fixes *f*::*'U rel* ⇒ *'U set* **and** α::*'U rel* **and** β::*'U rel*

assumes α =o β
shows $\mathfrak{L}$ f α = $\mathfrak{L}$ f β
proof
  show $\mathfrak{L}$ f α ⊆ $\mathfrak{L}$ f β **using** *assms ordLess-ordIso-trans* **unfolding** $\mathfrak{L}$*-def* **by** *fastforce*
next
  have β =o α **using** *assms ordIso-symmetric* **by** *blast*
  then show $\mathfrak{L}$ f β ⊆ $\mathfrak{L}$ f α **using** *ordLess-ordIso-trans* **unfolding** $\mathfrak{L}$*-def* **by** *fastforce*
qed

lemma *lem-shrel-dbk-eq*:
fixes f::'U *rel* ⇒ 'U *set* **and** *Ps*::'U *set set* **and** α::'U *rel* **and** β::'U *rel*
assumes $f \in \mathcal{N}$ r *Ps* **and** α =o β **and** α ≤o |*Field r*| **and** β ≤o |*Field r*|
shows (∇ f α) = (∇ f β)
proof −
  have α ≤o β ∧ β ≤o α **using** *assms ordIso-iff-ordLeq* **by** *blast*
  then have f α = f β **using** *assms* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*1-def* **by** *blast*
  moreover have $\mathfrak{L}$ f α = $\mathfrak{L}$ f β **using** *assms lem-shrel-L-eq* **by** *blast*
  ultimately show *?thesis* **unfolding** *Dbk-def* **by** *blast*
qed

lemma *lem-L-emp*: α =o ({}::'U *rel*) ⟹ $\mathfrak{L}$ f α = {}
proof −
  assume α =o ({}::'U *rel*)
  then have ∀ α'. α' <o α ⟶ *False* **using** *lem-ord-subemp*
    by (*metis iso-ozero-empty not-ordLess-ordIso ordLess-imp-ordLeq ozero-def*)
  then show $\mathfrak{L}$ f α = {} **unfolding** $\mathfrak{L}$*-def* **by** *blast*
qed

lemma *lem-der-qinv1*:
fixes r::'U *rel* **and** α::'U *rel* **and** x y::'U
assumes *a1*: $x \in \mathcal{Q}$ r f α **and** *a2*: $(x,y) \in r$^* **and** *a3*: $y \in (f\ \alpha)$
shows $y \in \mathcal{Q}$ r f α
proof −
  obtain A **where** *b1*: A = ($\mathfrak{L}$ f α) **by** *blast*
  have ∀ x y. y ∈ *dncl r A* ⟶ $(x,y) \in r$ ⟶ x ∈ *dncl r A*
  proof (*intro allI impI*)
    fix x y
    assume y ∈ *dncl r A* **and** $(x,y) \in r$
    moreover then obtain a **where** $a \in A \wedge (y,a) \in r$^* **unfolding** *dncl-def* **by** *blast*
    ultimately have $a \in A \wedge (x,a) \in r$^* **by** *force*
    then show x ∈ *dncl r A* **unfolding** *dncl-def* **by** *blast*
  qed
  then have (*UNIV* − *dncl r A*) ∈ *Inv r* **unfolding** *Inv-def* **by** *blast*
  moreover have x ∈ *UNIV* − (*dncl r A*) **using** *b1 a1* **unfolding** $\mathcal{Q}$*-def* **by** *blast*
  ultimately have y ∈ *UNIV* − (*dncl r A*) **using** *a2 lem-Inv-restr-rtr2*[*of UNIV* − *dncl r A r*] **by** *blast*

then show *?thesis* **using** *b1 a3* **unfolding** *𝒬-def* **by** *blast*
qed

lemma *lem-der-qinv2*:
fixes *r*::*′U rel* **and** *α*::*′U rel* **and** *x y*::*′U*
assumes *a1*: *x* ∈ *𝒬 r f α* **and** *a2*: (*x*,*y*) ∈ (*Restr r* (*f α*))^* **and** *a3*: *y* ∈ (*f α*)
shows (*x*,*y*) ∈ (*Restr r* (*𝒬 r f α*))^*
proof −
 obtain *Q* **where** *b1*: *Q* = *𝒬 r f α* **by** *blast*
 have ∀ *a b*. *a* ∈ *Q* ⟶ (*a*,*b*) ∈ *Restr r* (*f α*) ⟶ *b* ∈ *Q*
  using *lem-der-qinv1*[*of - r f α -*] **unfolding** *b1* **by** *blast*
 then have *Q* ∈ *Inv* (*Restr r* (*f α*)) **unfolding** *Inv-def* **by** *blast*
 moreover have *x* ∈ *Q* **using** *b1 a1* **by** *blast*
 ultimately have (*x*,*y*) ∈ (*Restr* (*Restr r* (*f α*)) *Q*)^*
  using *a2 lem-Inv-restr-rtr*[*of Q Restr r* (*f α*)] **by** *blast*
 moreover have *Restr* (*Restr r* (*f α*)) *Q* ⊆ *Restr r* (*𝒬 r f α*) **using** *b1* **by** *blast*
 ultimately show *?thesis* **using** *rtrancl-mono* **by** *blast*
qed

lemma *lem-der-qinv3*:
fixes *r*::*′U rel* **and** *α*::*′U rel*
assumes *a1*: *A* ⊆ (*f α*) **and** *a2*: ∀ *x* ∈ (*f α*). ∃ *y* ∈ *A*. (*x*,*y*) ∈ (*Restr r* (*f α*))^*
shows ∀ *x* ∈ (*𝒬 r f α*). ∃ *y* ∈ (*A* ∩ (*𝒬 r f α*)). (*x*,*y*) ∈ (*Restr r* (*𝒬 r f α*))^*
proof
 fix *x*
 assume *b1*: *x* ∈ (*𝒬 r f α*)
 then have *b2*: *x* ∈ (*f α*) **unfolding** *𝒬-def* **by** *blast*
 then obtain *y* **where** *b3*: *y* ∈ *A* ∧ (*x*,*y*) ∈ (*Restr r* (*f α*))^* **using** *a2* **by** *blast*
 then have (*x*, *y*) ∈ (*Restr r* (*𝒬 r f α*))^* **using** *a1 b1 lem-der-qinv2*[*of x r f α y*] **by** *blast*
 moreover then have *y* ∈ (*𝒬 r f α*) **using** *b1 IntE mem-Sigma-iff rtranclE*[*of x y*] **by** *metis*
 ultimately show ∃ *y* ∈ (*A* ∩ (*𝒬 r f α*)). (*x*,*y*) ∈ (*Restr r* (*𝒬 r f α*))^* **using** *b3* **by** *blast*
qed

lemma *lem-der-inf-qrestr-ccr1*:
fixes *r*::*′U rel* **and** *Ps*::*′U set set* **and** *α*::*′U rel*
assumes *f* ∈ *𝒩 r Ps* **and** *α* ≤*o* |*Field r*|
shows *CCR* (*Restr r* (*𝒬 r f α*))
proof −
 have *CCR* (*Restr r* (*f α*)) **using** *assms* **unfolding** *𝒩-def 𝒩6-def* **by** *blast*
 moreover have *dncl r* (*𝔏 f α*) ∈ *Inv* (*r*^−*1*) **using** *lem-Inv-dncl-invbk* **by** *blast*
 ultimately show *?thesis* **unfolding** *𝒬-def* **using** *lem-Inv-ccr-restr-invdiff* **by** *blast*
qed

lemma *lem-Nfdn-aemp*:
fixes *r*::*′U rel* **and** *Ps*::*′U set set* **and** *f*::*′U rel* ⇒ *′U set* **and** *α*::*′U rel*

assumes *a1*: *CCR r* **and** *a2*: $f \in \mathcal{N}$ *r Ps* **and** *a3*: α *<o scf r* **and** *a4*: *Field r* $\subseteq$ *dncl r* (*f* α)
shows $\alpha = \{\}$
proof (*cases finite r*)
  assume *finite r*
  then have *scf r <o* ω*-ord* **using** *lem-scf-relfldcard-bnd lem-fin-fl-rel*
    by (*metis finite-iff-ordLess-natLeq ordLeq-ordLess-trans*)
  then have *finite* (*Field* (*scf r*)) **using** *finite-iff-ordLess-natLeq* **by** *force*
  then have *Conelike r* **using** *a1 lem-scf-ccr-finscf-cl* **by** *blast*
  moreover obtain *a*::*'U* **where** *True* **by** *blast*
  ultimately have α *<o* |{*a*}| **using** *a1 a3 lem-Rcc-eq2-12 lem-scf-ccr-scf-rcc-eq*
    by (*metis ordIso-iff-ordLeq ordLess-ordLeq-trans*)
  then have *b1*: α *=o* |{}::*'U set*| **using** *lem-co-one-ne-min*
    by (*metis card-of-card-order-on card-of-empty3 card-of-unique insert-not-empty not-ordLeq-ordLess ordIso-Well-order-simp ordLess-Well-order-simp*)
  then have α $\leq$*o* |*Field r*| **using** *card-of-empty ordIso-ordLeq-trans* **by** *blast*
  then have *b2*: *f* $\alpha \in$ *SF r* **using** *a2* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*5-def* **by** *blast*
  have ¬ (∃ α'::*'U rel*. α' *<o* α) **using** *b1*
    by (*metis BNF-Cardinal-Order-Relation.ordLess-Field card-of-empty5 ordLess-ordIso-trans*)
  then show $\alpha = \{\}$ **using** *a3 b1* **using** *lem-co-one-ne-min*
    by (*metis card-of-empty card-of-empty3 insert-not-empty ordIso-ordLeq-trans ordLeq-transitive ordLess-Well-order-simp*)
next
  assume *q0*: ¬ *finite r*
  have *b0*: α *<o* ‖*r*‖ **using** *a1 a3 lem-scf-ccr-scf-rcc-eq* **by** (*metis ordIso-iff-ordLeq ordLess-ordLeq-trans*)
  obtain *A'* **where** *b1*: *A'* = $\mathcal{Q}$ *r f* α **by** *blast*
  have ‖*r*‖ $\leq$*o* |*r*| **using** *lem-Rcc-relcard-bnd* **by** *blast*
  moreover have |*Field r*| *=o* |*r*| **using** *q0 lem-rel-inf-fld-card* **by** *blast*
  ultimately have ‖*r*‖ $\leq$*o* |*Field r*| **using** *ordIso-symmetric ordLeq-ordIso-trans* **by** *blast*
  then have *b2*: α $\leq$*o* |*Field r*| **using** *b0 ordLeq-transitive ordLess-imp-ordLeq* **by** *blast*
  then have *b3*: *f* $\alpha \in$ *SF r* ∧ *CCR* (*Restr r* (*f* α))
      using *b1 a2* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*5-def* $\mathcal{N}$*10-def* $\mathcal{N}$*6-def* **by** *blast+*
  have *b5*: (*A'* ∈ *SF r*) ∨ (∃ *y*::*'U*. *A'* = {*y*})
    using *b1 b3* **unfolding** $\mathcal{Q}$*-def* **using** *lem-Inv-ccr-sf-dn-diff*[*of f* α *r A'* $\mathfrak{L}$ *f* α] **by** *blast*
  have ∀ *a*∈*Field r*. ∃ *b*∈*Field* (*Restr r* (*f* α)). (*a*, *b*) ∈ *r*^*
  proof
    fix *a*
    assume *a* ∈ *Field r*
    then have *a* ∈ *dncl r* (*f* α) **using** *a4* **by** *blast*
    then obtain *b*::*'U* **where** (*a*, *b*) ∈ *r*^* ∧ *b* ∈ *f* α **unfolding** *dncl-def* **by** *blast*
    moreover have (*f* α) ∈ *SF r* **using** *b3* **by** *blast*
    ultimately have *b* ∈ *Field* (*Restr r* (*f* α)) ∧ (*a*, *b*) ∈ *r*^* **unfolding** *SF-def* **by** *blast*
    then show ∃ *b*∈*Field* (*Restr r* (*f* α)). (*a*, *b*) ∈ *r*^* **by** *blast*
  qed

moreover have *CCR* (*Restr r* (*f* α)) **using** *b3* **by** *blast*
ultimately have *Restr r* (*f* α) $\in \mathfrak{U}$ *r* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
then have *d3*: $\|r\| \leq o$ |*Restr r* (*f* α)| **using** *lem-rcc-uset-mem-bnd* **by** *blast*
obtain *x*::'*U* **where** *d4*: *True* **by** *blast*
have ω*-ord* $\leq o$ $\alpha \longrightarrow$ *False*
proof
assume *e1*: ω*-ord* $\leq o$ α
then have |*f* α| $\leq o$ α **using** *b2 a2* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*7-def* **by** *blast*
moreover then have |*Restr r* (*f* α)| $\leq o$ α **using** *e1 lem-restr-ordbnd* **by** *blast*
ultimately have $\|r\| \leq o$ α **using** *d3 ordLeq-transitive* **by** *blast*
then show *False* **using** *b0 not-ordLess-iff-ordLeq ordLess-Well-order-simp* **by** *blast*
qed
then have $\alpha <o$ ω*-ord* **using** *b0 natLeq-Well-order not-ordLess-iff-ordLeq ordLess-Well-order-simp* **by** *blast*
then have |*f* α| $<o$ ω*-ord* **using** *b2 a2* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*7-def* **by** *blast*
then have *finite* (*f* α) **using** *finite-iff-ordLess-natLeq* **by** *blast*
then have *finite* (*Restr r* (*f* α)) **by** *blast*
then have |*Restr r* (*f* α)| $<o$ ω*-ord* **using** *finite-iff-ordLess-natLeq* **by** *blast*
then have *d5*: $\|r\| <o$ ω*-ord* **using** *d3 ordLeq-ordLess-trans* **by** *blast*
have $\|r\| \leq o$ $|\{x\}|$
proof (*cases CCR r*)
assume *CCR r*
then show $\|r\| \leq o$ $|\{x\}|$ **using** *d5 lem-Rcc-eq2-31*[*of r*] *lem-Rcc-eq2-12*[*of r x*] **by** *blast*
next
assume ¬ *CCR r*
moreover then have $\|r\| = \{\}$ **using** *lem-rcc-nccr* **by** *blast*
moreover have $\{\} \leq o$ $|\{x\}|$ **by** (*metis card-of-Well-order ozero-def ozero-ordLeq*)
ultimately show $\|r\| \leq o$ $|\{x\}|$ **by** *metis*
qed
then have $\alpha <o$ $|\{x\}|$ **using** *b0 ordLess-ordLeq-trans* **by** *blast*
then show $\alpha = \{\}$ **by** (*meson lem-co-one-ne-min not-ordLeq-ordLess ordLess-Well-order-simp*)
qed

lemma *lem-der-qccr-lscf-sf*:
fixes *r*::'*U rel* **and** *Ps*::'*U set set* **and** *f*::'*U rel* $\Rightarrow$ '*U set* **and** α::'*U rel*
assumes *a1*: *CCR r* **and** *a2*: *f* $\in \mathcal{N}$ *r Ps* **and** *a3*: $\alpha <o$ *scf r*
shows ($\mathcal{Q}$ *r f* α) $\in$ *SF r*
proof (*cases finite r*)
assume *finite r*
then have *scf r* $<o$ ω*-ord* **using** *lem-scf-relfldcard-bnd lem-fin-fl-rel*
by (*metis finite-iff-ordLess-natLeq ordLeq-ordLess-trans*)
then have *finite* (*Field* (*scf r*)) **using** *finite-iff-ordLess-natLeq* **by** *force*
then have *Conelike r* **using** *a1 lem-scf-ccr-finscf-cl* **by** *blast*
moreover obtain *a*::'*U* **where** *True* **by** *blast*
ultimately have $\alpha <o$ $|\{a\}|$ **using** *a1 a3 lem-Rcc-eq2-12 lem-scf-ccr-scf-rcc-eq*
by (*metis ordIso-iff-ordLeq ordLess-ordLeq-trans*)
then have *b1*: $\alpha =o$ $|\{\}$::'*U set*| **using** *lem-co-one-ne-min*

```
    by (metis card-of-card-order-on card-of-empty3 card-of-unique insert-not-empty
          not-ordLeq-ordLess ordIso-Well-order-simp ordLess-Well-order-simp)
  then have α ≤o |Field r| using card-of-empty ordIso-ordLeq-trans by blast
  then have b2: f α ∈ SF r using a2 unfolding 𝒩-def 𝒩5-def by blast
  have ¬ (∃ α'::'U rel. α' <o α) using b1
   by (metis BNF-Cardinal-Order-Relation.ordLess-Field card-of-empty5 ordLess-ordIso-trans)
  then have 𝔏 f α = {} unfolding 𝔏-def by blast
  then have 𝒬 r f α = f α unfolding 𝒬-def dncl-def by blast
  then show ?thesis using b2 by metis
next
  assume q0: ¬ finite r
  have b0: α <o ‖r‖ using a1 a3 lem-scf-ccr-scf-rcc-eq by (metis ordIso-iff-ordLeq
ordLess-ordLeq-trans)
  obtain A' where b1: A' = 𝒬 r f α by blast
  have ‖r‖ ≤o |r| using lem-Rcc-relcard-bnd by blast
  moreover have |Field r| =o |r| using q0 lem-rel-inf-fld-card by blast
  ultimately have ‖r‖ ≤o |Field r| using ordIso-symmetric ordLeq-ordIso-trans
by blast
  then have b2: α ≤o |Field r| using b0 ordLeq-transitive ordLess-imp-ordLeq by
blast
  then have b3: f α ∈ SF r ∧ CCR (Restr r (f α))
       and b4: (∃ y::'U. A' = {y}) ⟶ Field r ⊆ dncl r (f α)
       using b1 a2 unfolding 𝒩-def 𝒩5-def 𝒩10-def 𝒩6-def by blast+
  have b5: (A' ∈ SF r ) ∨ (∃ y::'U. A' = {y})
    using b1 b3 unfolding 𝒬-def using lem-Inv-ccr-sf-dn-diff[of f α r A' 𝔏 f α]
by blast
  show (𝒬 r f α) ∈ SF r
  proof (cases Field r ⊆ dncl r (f α))
    assume c1: Field r ⊆ dncl r (f α)
    have ∀ a∈Field r. ∃ b∈Field (Restr r (f α)). (a, b) ∈ r^*
    proof
      fix a
      assume a ∈ Field r
      then have a ∈ dncl r (f α) using c1 by blast
       then obtain b::'U where (a, b) ∈ r^* ∧ b ∈ f α unfolding dncl-def by
blast
      moreover have (f α) ∈ SF r using b3 by blast
      ultimately have b ∈ Field (Restr r (f α)) ∧ (a, b) ∈ r^* unfolding SF-def
by blast
      then show ∃ b∈Field (Restr r (f α)). (a, b) ∈ r^* by blast
    qed
    moreover have CCR (Restr r (f α)) using b3 by blast
    ultimately have Restr r (f α) ∈ 𝔘 r unfolding 𝔘-def by blast
    then have d3: ‖r‖ ≤o |Restr r (f α)| using lem-rcc-uset-mem-bnd by blast
    obtain x::'U where d4: True by blast
    have ω-ord ≤o α ⟶ False
    proof
      assume e1: ω-ord ≤o α
      then have |f α| ≤o α using b2 a2 unfolding 𝒩-def 𝒩7-def by blast
```

moreover then have $|Restr\ r\ (f\ \alpha)| \leq o\ \alpha$ **using** *e1 lem-restr-ordbnd* **by** *blast*
ultimately have $\|r\| \leq o\ \alpha$ **using** *d3 ordLeq-transitive* **by** *blast*
then show *False* **using** *b0 not-ordLess-iff-ordLeq ordLess-Well-order-simp* **by** *blast*
qed
then have $\alpha <o\ \omega\text{-}ord$ **using** *b0 natLeq-Well-order not-ordLess-iff-ordLeq ordLess-Well-order-simp* **by** *blast*
then have $|f\ \alpha| <o\ \omega\text{-}ord$ **using** *b2 a2* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*7-def* **by** *blast*
then have *finite* $(f\ \alpha)$ **using** *finite-iff-ordLess-natLeq* **by** *blast*
then have *finite* $(Restr\ r\ (f\ \alpha))$ **by** *blast*
then have $|Restr\ r\ (f\ \alpha)| <o\ \omega\text{-}ord$ **using** *finite-iff-ordLess-natLeq* **by** *blast*
then have *d5*: $\|r\| <o\ \omega\text{-}ord$ **using** *d3 ordLeq-ordLess-trans* **by** *blast*
have $\|r\| \leq o\ |\{x\}|$
proof (*cases CCR r*)
assume *CCR r*
then show $\|r\| \leq o\ |\{x\}|$ **using** *d5 lem-Rcc-eq2-31*[*of r*] *lem-Rcc-eq2-12*[*of r x*] **by** *blast*
next
assume ¬ *CCR r*
moreover then have $\|r\| = \{\}$ **using** *lem-rcc-nccr* **by** *blast*
moreover have $\{\} \leq o\ |\{x\}|$ **by** (*metis card-of-Well-order ozero-def ozero-ordLeq*)
ultimately show $\|r\| \leq o\ |\{x\}|$ **by** *metis*
qed
then have $\alpha <o\ |\{x\}|$ **using** *b0 ordLess-ordLeq-trans* **by** *blast*
then have $\alpha = \{\}$ **by** (*meson lem-co-one-ne-min not-ordLeq-ordLess ordLess-Well-order-simp*)
then have $\forall\ \alpha'.\ \alpha' <o\ \alpha \longrightarrow$ *False* **using** *lem-ord-subemp* **by** (*metis iso-ozero-empty not-ordLess-ordIso ordLess-imp-ordLeq ozero-def*)
then have *dncl r* $(\mathfrak{L}\ f\ \alpha) = \{\}$ **unfolding** *dncl-def* $\mathfrak{L}$*-def* **by** *blast*
then have $\mathcal{Q}\ r\ f\ \alpha = f\ \alpha$ **unfolding** $\mathcal{Q}$*-def* **by** *blast*
then show $(\mathcal{Q}\ r\ f\ \alpha) \in SF\ r$ **using** *b3* **by** *metis*
next
assume ¬ $(Field\ r \subseteq dncl\ r\ (f\ \alpha))$
then have $A' \in SF\ r$ **using** *b4 b5* **by** *blast*
then show $(\mathcal{Q}\ r\ f\ \alpha) \in SF\ r$ **using** *b1* **by** *blast*
qed
qed

lemma *lem-der-q-uset*:
fixes r::$'U$ *rel* **and** Ps::$'U$ *set set* **and** α::$'U$ *rel*
assumes *a1*: *CCR r* **and** *a2*: $f \in \mathcal{N}\ r\ Ps$ **and** *a3*: $\alpha <o\ scf\ r$ **and** *a4*: *isSuccOrd* α
shows $Restr\ r\ (\mathcal{Q}\ r\ f\ \alpha) \in \mathfrak{U}\ (Restr\ r\ (f\ \alpha))$
proof −
have *b1*: $\alpha \leq o\ |Field\ r|$ **using** *a3 lem-scf-relfldcard-bnd*
by (*metis ordLess-ordLeq-trans ordLess-imp-ordLeq*)
have *a4*: $\mathcal{Q}\ r\ f\ \alpha = \{\} \longrightarrow$ *False*
proof

assume $\mathcal{Q}\ r\ f\ \alpha = \{\}$
then have *Field r* $\subseteq$ *dncl r* $(f\ \alpha)$ **using** *b1 a2 a4* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*11-def* **by** *blast*
then have $\alpha = \{\}$ **using** *a1 a2 a3 lem-Nfdn-aemp* **by** *blast*
then show *False* **using** *a4* **using** *wo-rel-def wo-rel.isSuccOrd-def* **unfolding** *Field-def* **by** *force*
qed
have $(\mathcal{Q}\ r\ f\ \alpha) \in$ *SF r* **using** *a1 a2 a3 lem-der-qccr-lscf-sf* **by** *blast*
then have *b2*: *Field* (*Restr r* $(\mathcal{Q}\ r\ f\ \alpha)) \neq \{\}$ **using** *a4* **unfolding** *SF-def* **by** *blast*
have *Restr r* $(\mathcal{Q}\ r\ f\ \alpha) \subseteq$ *Restr r* $(f\ \alpha)$ **unfolding** $\mathcal{Q}$*-def* **by** *blast*
moreover have *CCR* (*Restr r* $(\mathcal{Q}\ r\ f\ \alpha))$ **using** *b1 a2 lem-der-inf-qrestr-ccr1* **by** *blast*
moreover have $\forall$ *a*$\in$*Field* (*Restr r* $(f\ \alpha))$. $\exists$ *b*$\in$*Field* (*Restr r* $(\mathcal{Q}\ r\ f\ \alpha))$. $(a,b) \in$ (*Restr r* $(f\ \alpha))\hat{}*$
proof
fix *a*
assume *c1*: $a \in$ *Field* (*Restr r* $(f\ \alpha))$
obtain *b* **where** *c2*: $b \in$ *Field* (*Restr r* $(\mathcal{Q}\ r\ f\ \alpha))$ **using** *b2* **by** *blast*
then have *c3*: $b \in f\ \alpha \wedge b \in \mathcal{Q}\ r\ f\ \alpha$ **unfolding** $\mathcal{Q}$*-def Field-def* **by** *blast*
have $f\ \alpha \in$ *SF r* **using** *b1 a2* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*5-def* **by** *blast*
then have $b \in$ *Field* (*Restr r* $(f\ \alpha))$ **using** *c3* **unfolding** *SF-def* **by** *blast*
moreover have *CCR* (*Restr r* $(f\ \alpha))$ **using** *b1 a2* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*6-def* **by** *blast*
ultimately obtain *c* **where** $c \in$ *Field* (*Restr r* $(f\ \alpha))$
and *c4*: $(a,c) \in$ (*Restr r* $(f\ \alpha))\hat{}* \wedge (b,c) \in$ (*Restr r* $(f\ \alpha))\hat{}*$
using *c1* **unfolding** *CCR-def* **by** *blast*
moreover then have $c \in f\ \alpha$ **unfolding** *Field-def* **by** *blast*
ultimately have $(b,\ c) \in$ (*Restr r* $(\mathcal{Q}\ r\ f\ \alpha))\hat{}*$ **using** *c3 lem-der-qinv2*[*of b r f* α *c*] **by** *blast*
moreover have *Field* (*Restr r* $(\mathcal{Q}\ r\ f\ \alpha)) \in$ *Inv* (*Restr r* $(\mathcal{Q}\ r\ f\ \alpha))$
unfolding *Inv-def Field-def* **by** *blast*
ultimately have $c \in$ *Field* (*Restr r* $(\mathcal{Q}\ r\ f\ \alpha))$
using *c2 lem-Inv-restr-rtr2*[*of Field* (*Restr r* $(\mathcal{Q}\ r\ f\ \alpha))$] **by** *blast*
then show $\exists$ *b*$\in$*Field* (*Restr r* $(\mathcal{Q}\ r\ f\ \alpha))$. $(a,\ b) \in$ (*Restr r* $(f\ \alpha))\hat{}*$ **using** *c4* **by** *blast*
qed
ultimately show *Restr r* $(\mathcal{Q}\ r\ f\ \alpha) \in \mathfrak{U}$ (*Restr r* $(f\ \alpha))$ **unfolding** $\mathfrak{U}$*-def* **by** *blast*
qed

lemma *lem-qw-range*: $f \in \mathcal{N}\ r\ Ps \Longrightarrow \alpha \leq o\ |Field\ r| \Longrightarrow \mathcal{W}\ r\ f\ \alpha \subseteq Field\ r$
unfolding $\mathcal{N}$*-def* $\mathcal{N}$*5-def SF-def Field-def* $\mathcal{W}$*-def* **by** *blast*

lemma *lem-der-qw-eq*:
fixes *r*::*'U rel* **and** *Ps*::*'U set set* **and** α β::*'U rel*
assumes $f \in \mathcal{N}\ r\ Ps$ **and** $\alpha =o\ \beta$
shows $\mathcal{W}\ r\ f\ \alpha = \mathcal{W}\ r\ f\ \beta$
proof –

have *f* α = *f* β **using** *assms* **unfolding** $\mathcal{N}$*-def* **by** *blast*
moreover have $\mathfrak{L}$ *f* α = $\mathfrak{L}$ *f* β **using** *assms lem-shrel-L-eq* **by** *blast*
ultimately show *?thesis* **unfolding** $\mathcal{W}$*-def* **by** *simp*
qed

lemma *lem-Der-inf-qw-disj*:
fixes *r*::*'U rel* **and** α β::*'U rel*
assumes *Well-order* α **and** *Well-order* β
shows $(\neg\ (\alpha =o\ \beta)) \longrightarrow (\mathcal{W}\ r\ f\ \alpha) \cap (\mathcal{W}\ r\ f\ \beta) = \{\}$
proof
assume *b1*: $\neg\ (\alpha =o\ \beta)$
obtain *W* **where** *b2*: $W = (\lambda\ \alpha.\ \mathcal{W}\ r\ f\ \alpha)$ **by** *blast*
have $\alpha <o\ \beta \lor \beta <o\ \alpha$ **using** *b1 assms* **by** (*meson not-ordLeq-iff-ordLess ordLeq-iff-ordLess-or-ordIso*)
moreover have $\forall\ \alpha'\ \beta'.\ \alpha' <o\ \beta' \longrightarrow (W\ \alpha' \cap W\ \beta' \neq \{\}) \longrightarrow False$
proof (*intro allI impI*)
fix α' β'::*'U rel*
assume *d1*: $\alpha' <o\ \beta'$ **and** $W\ \alpha' \cap W\ \beta' \neq \{\}$
then obtain *a* **where** *d2*: $a \in W\ \alpha' \cap W\ \beta'$ **by** *blast*
then have $a \in f\ \alpha'$ **using** *b2* **unfolding** $\mathcal{W}$*-def* **by** *blast*
then have $a \in \mathfrak{L}\ f\ \beta'$ **using** *d1* **unfolding** $\mathfrak{L}$*-def* **by** *blast*
then have $a \notin W\ \beta'$ **using** *b2 lem-wdn-range-lb*[*of - r*] **unfolding** $\mathcal{W}$*-def* **by** *blast*
then show *False* **using** *d2* **by** *blast*
qed
ultimately show $(\mathcal{W}\ r\ f\ \alpha) \cap (\mathcal{W}\ r\ f\ \beta) = \{\}$ **unfolding** *b2* **by** *blast*
qed

lemma *lem-der-inf-qw-restr-card*:
fixes *r*::*'U rel* **and** *Ps*::*'U set set* **and** α::*'U rel*
assumes *a1*: $\neg$ *finite r* **and** *a2*: $f \in \mathcal{N}\ r\ Ps$ **and** *a3*: $\alpha <o\ |Field\ r|$
shows $|Restr\ r\ (\mathcal{W}\ r\ f\ \alpha)| <o\ |Field\ r|$
proof −
have *b0*: $|Field\ r| =o\ |r|$ **using** *a1 lem-rel-inf-fld-card* **by** *blast*
obtain *W* **where** *b2*: $W = (\lambda\ \alpha.\ \mathcal{W}\ r\ f\ \alpha)$ **by** *blast*
have $\alpha \leq o\ |Field\ r|$ **using** *a3 b0 ordLess-imp-ordLeq ordIso-iff-ordLeq ordLeq-transitive* **by** *blast*
then have $(\alpha <o\ \omega\text{-}ord \longrightarrow |f\ \alpha| <o\ \omega\text{-}ord) \land (\omega\text{-}ord \leq o\ \alpha \longrightarrow |f\ \alpha| \leq o\ \alpha)$
using *a2* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*7-def* **by** *blast*
moreover have *c2*: $\alpha <o\ \omega\text{-}ord \lor \omega\text{-}ord \leq o\ \alpha$ **using** *a3 Field-natLeq natLeq-well-order-on* **by** *force*
moreover have *c3*: $|f\ \alpha| <o\ \omega\text{-}ord \longrightarrow |Restr\ r\ (W\ \alpha)| <o\ |Field\ r|$
proof
assume $|f\ \alpha| <o\ \omega\text{-}ord$
then have *finite* $(f\ \alpha)$ **using** *finite-iff-ordLess-natLeq* **by** *blast*
then have *finite* $(Restr\ r\ (W\ \alpha))$ **unfolding** *b2* $\mathcal{W}$*-def* **by** *blast*
then have $|Restr\ r\ (W\ \alpha)| <o\ \omega\text{-}ord$ **using** *finite-iff-ordLess-natLeq* **by** *blast*
moreover have $\omega\text{-}ord \leq o\ |r|$ **using** *a1 infinite-iff-natLeq-ordLeq* **by** *blast*
moreover then have $\omega\text{-}ord \leq o\ |Field\ r|$ **using** *lem-rel-inf-fld-card*

by (*metis card-of-ordIso-finite infinite-iff-natLeq-ordLeq*)
ultimately show $|Restr\ r\ (W\ \alpha)| <o\ |Field\ r|$ **using** *ordLess-ordLeq-trans* **by** *blast*
qed
moreover have $\omega\text{-}ord \leq o\ \alpha \wedge |f\ \alpha| \leq o\ \alpha \longrightarrow |Restr\ r\ (W\ \alpha)| <o\ |Field\ r|$
proof
assume *d1*: $\omega\text{-}ord \leq o\ \alpha \wedge |f\ \alpha| \leq o\ \alpha$
moreover have $|W\ \alpha| \leq o\ |f\ \alpha|$ **unfolding** *b2* $\mathcal{W}$*-def* **by** *simp*
ultimately have $|W\ \alpha| \leq o\ \alpha$ **using** *ordLeq-transitive* **by** *blast*
then have $|Restr\ r\ (W\ \alpha)| \leq o\ \alpha$ **using** *d1 lem-restr-ordbnd*[*of* α $W\ \alpha$ r] **by** *blast*
then show $|Restr\ r\ (W\ \alpha)| <o\ |Field\ r|$ **using** *a3 ordLeq-ordLess-trans* **by** *blast*
qed
ultimately show *?thesis* **using** *b2* **by** *blast*
qed

lemma *lem-QS-subs-WS*: $\mathcal{Q}\ r\ f\ \alpha \subseteq \mathcal{W}\ r\ f\ \alpha$
unfolding $\mathcal{Q}$*-def* $\mathcal{W}$*-def* **using** *lem-wdn-range-ub* **by** *force*

lemma *lem-WS-limord*:
fixes *r*::'*U rel* **and** *Ps*::'*U set set* **and** *f*::'*U rel* ⇒ '*U set* **and** α::'*U rel*
assumes *a1*: ¬ *finite r* **and** *a2*: $f \in \mathcal{N}\ r\ Ps$ **and** *a3*: $\alpha <o\ |Field\ r|$
and *a4*: $\neg\ (\alpha = \{\} \vee isSuccOrd\ \alpha)$
shows $\mathcal{W}\ r\ f\ \alpha = \{\}$
proof −
have $\alpha \leq o\ |Field\ r|$ **using** *a3 ordLess-imp-ordLeq* **by** *blast*
then have $f\ \alpha \subseteq \mathfrak{L}\ f\ \alpha$ **using** *a2 a4* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*2-def Dbk-def* **by** *blast*
then have $w\text{-}dncl\ r\ (f\ \alpha) \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha)$ **using** *lem-wdn-mon* **by** *blast*
moreover have $f\ \alpha \subseteq w\text{-}dncl\ r\ (f\ \alpha)$ **using** *lem-wdn-range-lb*[*of f* α *r*] **by** *metis*
ultimately have $f\ \alpha \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha)$ **by** *blast*
then show *?thesis* **unfolding** $\mathcal{W}$*-def* **by** *blast*
qed

lemma *lem-der-inf-qw-restr-uset*:
fixes *r*::'*U rel* **and** *Ps*::'*U set set* **and** *f*::'*U rel* ⇒ '*U set* **and** α::'*U rel*
assumes *a1*: $Refl\ r \wedge \neg\ finite\ r$ **and** *a2*: $f \in \mathcal{N}\ r\ Ps$
and *a3*: $\alpha <o\ |Field\ r|$ **and** *a4*: $\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha|$
shows $Restr\ r\ (\mathcal{Q}\ r\ f\ \alpha) \in \mathfrak{U}\ (Restr\ r\ (\mathcal{W}\ r\ f\ \alpha))$
proof (*cases* $\alpha = \{\} \vee isSuccOrd\ \alpha$)
assume $\alpha = \{\} \vee isSuccOrd\ \alpha$
moreover have $|Field\ r| =o\ |r|$ **using** *a1 lem-rel-inf-fld-card* **by** *blast*
then have *b1*: $\alpha \leq o\ |Field\ r|$ **using** *a3 ordLess-imp-ordLeq ordIso-iff-ordLeq ordLeq-transitive* **by** *blast*
ultimately have *b2*: $escl\ r\ (\mathfrak{L}\ f\ \alpha)\ (f\ \alpha) \subseteq f\ \alpha$ **using** *a2 a4* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*3-def* **by** *blast*
moreover have *b3*: $CCR\ (Restr\ r\ (f\ \alpha))$ **using** *b1 a2* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*6-def* **by** *blast*
moreover have $SF\ r = \{A.\ A \subseteq Field\ r\}$ **using** *a1* **unfolding** *SF-def refl-on-def*

Field-def **by** *fast*
moreover then have $\mathcal{W}\ r\ f\ \alpha \in SF\ r$ **and** $\mathcal{Q}\ r\ f\ \alpha \in SF\ r$
using *a2 a3 lem-qw-range*[*of f r Ps α*] *lem-QS-subs-WS*[*of r f α*] *ordLess-imp-ordLeq*
by *fast+*
ultimately show *?thesis*
using *a1 lem-cowdn-uset*[*of r f α* $\mathfrak{L}$ *f α*] $\mathcal{Q}$*-def*[*of r f α*] $\mathcal{W}$*-def*[*of r f α*] **by**
blast
next
assume $\neg\ (\alpha = \{\} \vee isSuccOrd\ \alpha)$
then have $\mathcal{W}\ r\ f\ \alpha = \{\} \wedge \mathcal{Q}\ r\ f\ \alpha = \{\}$
using *assms lem-WS-limord lem-QS-subs-WS*[*of r f α*] **by** *blast*
then show *?thesis* **unfolding** $\mathfrak{U}$*-def CCR-def Field-def* **by** *blast*
qed

lemma *lem-der-inf-qw-restr-ccr*:
fixes r::*'U rel* **and** Ps::*'U set set* **and** f::*'U rel* ⇒ *'U set* **and** α::*'U rel*
assumes *a1*: $Refl\ r \wedge \neg\ finite\ r$ **and** *a2*: $f \in \mathcal{N}\ r\ Ps$
and *a3*: $\alpha <o\ |Field\ r|$ **and** *a4*: $\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha|$
shows $CCR\ (Restr\ r\ (\mathcal{W}\ r\ f\ \alpha))$
using *assms lem-der-inf-qw-restr-uset lem-rcc-uset-ne-ccr* **by** *blast*

lemma *lem-der-qw-uset*:
fixes r::*'U rel* **and** Ps::*'U set set* **and** f::*'U rel* ⇒ *'U set* **and** α::*'U rel*
assumes *a1*: $CCR\ r \wedge Refl\ r \wedge \neg\ finite\ r$ **and** *a2*: $f \in \mathcal{N}\ r\ Ps$
and *a3*: $\alpha <o\ scf\ r$ **and** *a4*: $\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha|$ **and** *a5*: $isSuccOrd\ \alpha$
shows $Restr\ r\ (\mathcal{W}\ r\ f\ \alpha) \in \mathfrak{U}\ (Restr\ r\ (f\ \alpha))$
proof −
have *b1*: $\alpha <o\ |Field\ r|$ **using** *a3 lem-scf-relfldcard-bnd* **by** (*metis ordLess-ordLeq-trans*)
have $\mathcal{Q}\ r\ f\ \alpha \subseteq \mathcal{W}\ r\ f\ \alpha$ **using** *lem-QS-subs-WS*[*of r f α*] **by** *blast*
then have $Field\ (Restr\ r\ (\mathcal{Q}\ r\ f\ \alpha)) \subseteq Field\ (Restr\ r\ (\mathcal{W}\ r\ f\ \alpha))$ **unfolding**
Field-def **by** *blast*
moreover have $Restr\ r\ (\mathcal{Q}\ r\ f\ \alpha) \in \mathfrak{U}\ (Restr\ r\ (f\ \alpha))$
using *a1 a2 a3 a5 lem-der-q-uset ordLess-imp-ordLeq* **by** *blast*
ultimately have $\forall\ a{\in}Field\ (Restr\ r\ (f\ \alpha)).\ \exists\ b{\in}Field\ (Restr\ r\ (\mathcal{W}\ r\ f\ \alpha)).$
$(a,b) \in (Restr\ r\ (f\ \alpha))\hat{}*$ **unfolding** $\mathfrak{U}$*-def* **by** *blast*
moreover have $Restr\ r\ (\mathcal{W}\ r\ f\ \alpha) \subseteq Restr\ r\ (f\ \alpha)$ **unfolding** $\mathcal{W}$*-def* **by** *blast*
moreover have $CCR\ (Restr\ r\ (\mathcal{W}\ r\ f\ \alpha))$ **using** *assms b1 lem-der-inf-qw-restr-ccr*
by *blast*
ultimately show *?thesis* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
qed

lemma *lem-Shinf-N1*:
fixes r::*'U rel* **and** F::*'U rel* ⇒ *'U set* ⇒ *'U set* **and** f::*'U rel* ⇒ *'U set*
assumes *a0*: $f \in \mathcal{T}\ F$
and *a1*: $\forall\ \alpha\ A.\ Well\text{-}order\ \alpha \longrightarrow A \subseteq F\ \alpha\ A$
shows $\forall\alpha.\ Well\text{-}order\ \alpha \longrightarrow f \in \mathcal{N}1\ r\ \alpha$
proof −
have *b2*: $f\ \{\} = \{\}$
and *b3*: $\forall\ \alpha 0\ \alpha$::*'U rel*. $(sc\text{-}ord\ \alpha 0\ \alpha \longrightarrow f\ \alpha = F\ \alpha 0\ (f\ \alpha 0))$

and *b4*: $\forall\ \alpha.\ (lm\text{-}ord\ \alpha \longrightarrow f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \})$
and *b5*: $\forall \alpha\ \beta.\ \alpha =o\ \beta \longrightarrow f\ \alpha = f\ \beta$ **using** *a0* **unfolding** *$\mathcal{T}$-def* **by** *blast+*
have $f \in \mathcal{N}1\ r\ \{\}$ **using** *b2* **unfolding** *$\mathcal{N}$1-def* **by** (*clarsimp*, *metis lem-ord-subemp*)
moreover have $\forall \alpha 0\ \alpha.\ sc\text{-}ord\ \alpha 0\ \alpha \wedge f \in \mathcal{N}1\ r\ \alpha 0 \longrightarrow f \in \mathcal{N}1\ r\ \alpha$
proof (*intro allI impI*)
fix $\alpha 0\ \alpha$::*'U rel*
assume *c1*: $sc\text{-}ord\ \alpha 0\ \alpha \wedge f \in \mathcal{N}1\ r\ \alpha 0$
then have *c2*: $f\ \alpha = F\ \alpha 0\ (f\ \alpha 0)$ **using** *b3* **by** *blast*
have $\forall \alpha'\ \alpha''.\ \alpha' \leq o\ \alpha \wedge \alpha'' \leq o\ \alpha' \longrightarrow f\ \alpha'' \subseteq f\ \alpha'$
proof (*intro allI impI*)
fix $\alpha'\ \alpha''$::*'U rel*
assume *d1*: $\alpha' \leq o\ \alpha \wedge \alpha'' \leq o\ \alpha'$
moreover then have $\alpha'' \leq o\ \alpha$ **using** *ordLeq-transitive* **by** *blast*
ultimately have $(\alpha'' \leq o\ \alpha 0 \vee \alpha'' =o\ \alpha) \wedge (\alpha' \leq o\ \alpha 0 \vee \alpha' =o\ \alpha)$ **using** *c1*
unfolding *sc-ord-def*
by (*meson not-ordLess-iff-ordLeq ordLeq-iff-ordLess-or-ordIso ordLess-Well-order-simp*)
moreover have $\alpha' \leq o\ \alpha 0 \longrightarrow f\ \alpha'' \subseteq f\ \alpha'$ **using** *d1 c1* **unfolding** *$\mathcal{N}$1-def*
by *blast*
moreover have $\alpha' =o\ \alpha \wedge \alpha'' =o\ \alpha \longrightarrow f\ \alpha'' \subseteq f\ \alpha'$ **using** *b5* **by** *blast*
moreover have $\alpha' =o\ \alpha \wedge \alpha'' \leq o\ \alpha 0 \longrightarrow f\ \alpha'' \subseteq f\ \alpha'$
proof
assume *e1*: $\alpha' =o\ \alpha \wedge \alpha'' \leq o\ \alpha 0$
moreover then have $\alpha 0\ \leq o\ \alpha 0$ **using** *ordLeq-Well-order-simp or-*
dLeq-reflexive **by** *blast*
ultimately have $f\ \alpha'' \subseteq f\ \alpha 0$ **using** *c1* **unfolding** *$\mathcal{N}$1-def* **by** *blast*
moreover have $f\ \alpha 0 \subseteq f\ \alpha$ **using** *a1 c2 e1 ordLeq-Well-order-simp* **by**
blast
ultimately show $f\ \alpha'' \subseteq f\ \alpha'$ **using** *b5 e1* **by** *blast*
qed
ultimately show $f\ \alpha'' \subseteq f\ \alpha'$ **by** *blast*
qed
then show $f \in \mathcal{N}1\ r\ \alpha$ **unfolding** *$\mathcal{N}$1-def* **by** *blast*
qed
moreover have $\forall \alpha.\ lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}1\ r\ \beta) \longrightarrow f \in \mathcal{N}1\ r$
α
proof (*intro allI impI*)
fix α::*'U rel*
assume *c1*: $lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}1\ r\ \beta)$
then have *c2*: $f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \}$ **using** *b4* **by** *blast*
have $\forall \alpha'\ \alpha''.\ \alpha' \leq o\ \alpha \wedge \alpha'' \leq o\ \alpha' \longrightarrow f\ \alpha'' \subseteq f\ \alpha'$
proof (*intro allI impI*)
fix $\alpha'\ \alpha''$::*'U rel*
assume *d1*: $\alpha' \leq o\ \alpha \wedge \alpha'' \leq o\ \alpha'$
then have $(\alpha' <o\ \alpha \vee \alpha' =o\ \alpha) \wedge (\alpha'' <o\ \alpha' \vee \alpha'' =o\ \alpha')$ **using** *or-*
dLeq-iff-ordLess-or-ordIso **by** *blast*
moreover have $\alpha' <o\ \alpha \longrightarrow f\ \alpha'' \subseteq f\ \alpha'$
using *d1 c1 ordLeq-Well-order-simp ordLeq-reflexive* **unfolding** *$\mathcal{N}$1-def* **by**
blast
moreover have $\alpha' =o\ \alpha \wedge \alpha'' <o\ \alpha' \longrightarrow f\ \alpha'' \subseteq f\ \alpha'$

using *c2 b5 ordLess-ordIso-trans* **by** *blast*
moreover have $\alpha' =o\ \alpha \wedge \alpha'' =o\ \alpha' \longrightarrow f\ \alpha'' \subseteq f\ \alpha'$ **using** *b5* **by** *blast*
ultimately show $f\ \alpha'' \subseteq f\ \alpha'$ **by** *blast*
qed
then show $f \in \mathcal{N}1\ r\ \alpha$ **unfolding** *𝒩1-def* **by** *blast*
qed
ultimately show *?thesis* **using** *lem-sclm-ordind*[*of* $\lambda\ \alpha.\ f \in \mathcal{N}1\ r\ \alpha$] **by** *blast*
qed

lemma *lem-Shinf-N2*:
fixes *r*::*'U rel* **and** *F*::*'U rel* ⇒ *'U set* ⇒ *'U set* **and** *f*::*'U rel* ⇒ *'U set*
assumes *a0*: $f \in \mathcal{T}\ F$
shows $\forall \alpha.$ *Well-order* $\alpha \longrightarrow f \in \mathcal{N}2\ r\ \alpha$
proof −
have *b4*: $\forall\ \alpha.$ (*lm-ord* $\alpha \longrightarrow f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \})$
and *b5*: $\forall \alpha\ \beta.\ \alpha =o\ \beta \longrightarrow f\ \alpha = f\ \beta$ **using** *a0* **unfolding** *𝒯-def* **by** *blast+*
have $f \in \mathcal{N}2\ r\ \{\}$ **using** *lem-ord-subemp* **unfolding** *𝒩2-def* **by** *blast*
moreover have $\forall \alpha0\ \alpha.$ *sc-ord* $\alpha0\ \alpha \wedge f \in \mathcal{N}2\ r\ \alpha0 \longrightarrow f \in \mathcal{N}2\ r\ \alpha$
proof (*intro allI impI*)
fix $\alpha0\ \alpha$::*'U rel*
assume *c1*: *sc-ord* $\alpha0\ \alpha \wedge f \in \mathcal{N}2\ r\ \alpha0$
have $\forall \alpha'$::*'U rel*. $\alpha' \leq o\ \alpha \wedge \neg\ (\alpha' = \{\} \vee$ *isSuccOrd* $\alpha') \longrightarrow (\nabla\ f\ \alpha') = \{\}$
proof (*intro allI impI*)
fix α'::*'U rel*
assume *d1*: $\alpha' \leq o\ \alpha \wedge \neg\ (\alpha' = \{\} \vee$ *isSuccOrd* $\alpha')$
then have $\alpha0 <o\ \alpha' \vee \alpha' \leq o\ \alpha0$ **using** *c1* **unfolding** *sc-ord-def*
using *not-ordLeq-iff-ordLess ordLeq-Well-order-simp ordLess-Well-order-simp*
by *blast*
moreover have $\alpha' \leq o\ \alpha0 \longrightarrow (\nabla\ f\ \alpha') = \{\}$ **using** *d1 c1* **unfolding** *𝒩2-def*
by *blast*
moreover have $\alpha0 <o\ \alpha' \longrightarrow \alpha =o\ \alpha'$ **using** *d1 c1* **unfolding** *sc-ord-def*
using *ordIso-iff-ordLeq* **by** *blast*
moreover have $\alpha =o\ \alpha' \longrightarrow$ *False*
proof
assume $\alpha =o\ \alpha'$
moreover have *isSuccOrd* α **using** *c1 lem-ordint-sucord*[*of* $\alpha0\ \alpha$] **unfolding**
sc-ord-def **by** *blast*
ultimately have *isSuccOrd* α' **using** *lem-osucc-eq* **by** *blast*
then show *False* **using** *d1* **by** *blast*
qed
ultimately show $(\nabla\ f\ \alpha') = \{\}$ **by** *blast*
qed
then show $f \in \mathcal{N}2\ r\ \alpha$ **unfolding** *𝒩2-def* **by** *blast*
qed
moreover have $\forall \alpha.$ *lm-ord* $\alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}2\ r\ \beta) \longrightarrow f \in \mathcal{N}2\ r$
α
proof (*intro allI impI*)
fix α::*'U rel*
assume *c1*: *lm-ord* $\alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}2\ r\ \beta)$

then have *c2*: $f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \}$ **using** *b4* **by** *blast*
have $\forall\alpha'::'U\ rel.\ \alpha' \leq o\ \alpha \wedge \neg\ (\alpha' = \{\} \vee isSuccOrd\ \alpha') \longrightarrow (\nabla\ f\ \alpha') = \{\}$
proof (*intro allI impI*)
fix $\alpha'::'U\ rel$
assume *d1*: $\alpha' \leq o\ \alpha \wedge \neg\ (\alpha' = \{\} \vee isSuccOrd\ \alpha')$
then have $\alpha' <o\ \alpha \vee \alpha' =o\ \alpha$ **using** *ordLeq-iff-ordLess-or-ordIso* **by** *blast*
moreover have $\alpha' <o\ \alpha \longrightarrow (\nabla\ f\ \alpha') = \{\}$
proof
assume $\alpha' <o\ \alpha$
moreover then have $\alpha' \leq o\ \alpha'$ **using** *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
ultimately show $(\nabla\ f\ \alpha') = \{\}$ **using** *c1 d1* **unfolding** $\mathcal{N}$*2-def* **by** *blast*
qed
moreover have $\alpha' =o\ \alpha \longrightarrow (\nabla\ f\ \alpha') = \{\}$
proof
assume $\alpha' =o\ \alpha$
moreover have $(\nabla\ f\ \alpha) = \{\}$ **using** *c2* **unfolding** *Dbk-def* $\mathfrak{L}$*-def* **by** *blast*
ultimately show $(\nabla\ f\ \alpha') = \{\}$ **using** *b5 lem-shrel-L-eq* **unfolding** *Dbk-def* **by** *blast*
qed
ultimately show $(\nabla\ f\ \alpha') = \{\}$ **by** *blast*
qed
then show $f \in \mathcal{N}2\ r\ \alpha$ **unfolding** $\mathcal{N}$*2-def* **by** *blast*
qed
ultimately show *?thesis* **using** *lem-sclm-ordind*[*of* $\lambda\ \alpha.\ f \in \mathcal{N}2\ r\ \alpha$] **by** *blast*
qed

lemma *lem-Shinf-N3*:
fixes $r::'U\ rel$ **and** $F::'U\ rel \Rightarrow 'U\ set \Rightarrow 'U\ set$ **and** $f::'U\ rel \Rightarrow 'U\ set$
assumes *a0*: $f \in \mathcal{T}\ F$
and *a1*: $\forall\ \alpha\ A.\ Well\text{-}order\ \alpha \longrightarrow A \subseteq F\ \alpha\ A$
and *a5*: $\forall\alpha.\ Well\text{-}order\ \alpha \longrightarrow f \in \mathcal{N}5\ r\ \alpha$
and *a3*: $\forall\ \alpha\ A.\ Well\text{-}order\ \alpha \longrightarrow A \in SF\ r \longrightarrow$
$(\omega\text{-}ord \leq o\ |A| \longrightarrow escl\ r\ A\ (F\ \alpha\ A) \subseteq (F\ \alpha\ A) \wedge clterm\ (Restr\ r\ (F\ \alpha\ A))\ r)$
shows $\forall\alpha.\ Well\text{-}order\ \alpha \longrightarrow f \in \mathcal{N}3\ r\ \alpha$
proof –
have *b2*: $f\ \{\} = \{\}$
and *b3*: $\forall\ \alpha0\ \alpha::'U\ rel.\ (sc\text{-}ord\ \alpha0\ \alpha \longrightarrow f\ \alpha = F\ \alpha0\ (f\ \alpha0))$
and *b4*: $\forall\ \alpha.\ (lm\text{-}ord\ \alpha \longrightarrow f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \})$
and *b5*: $\forall\alpha\ \beta.\ \alpha =o\ \beta \longrightarrow f\ \alpha = f\ \beta$ **using** *a0* **unfolding** $\mathcal{T}$*-def* **by** *blast+*
have $\mathfrak{L}\ f\ \{\} = \{\}$ **unfolding** $\mathfrak{L}$*-def* **using** *b2 lem-ord-subemp ordLess-imp-ordLeq* **by** *blast*
then have $\neg\ \omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \{\}|$ **using** *ctwo-ordLess-natLeq finite-iff-ordLess-natLeq ordLeq-transitive* **by** *auto*
then have $f \in \mathcal{N}3\ r\ \{\}$ **using** *b2 lem-ord-subemp* **unfolding** $\mathcal{N}$*3-def Field-def* **by** *blast*
moreover have $\forall\alpha0\ \alpha.\ sc\text{-}ord\ \alpha0\ \alpha \wedge f \in \mathcal{N}3\ r\ \alpha0 \longrightarrow f \in \mathcal{N}3\ r\ \alpha$
proof (*intro allI impI*)

fix $\alpha 0$ α::$'U$ *rel*
assume *c1*: *sc-ord* $\alpha 0$ $\alpha \wedge f \in \mathcal{N}3\ r\ \alpha 0$
have $\forall \alpha'$::$'U$ *rel*. $\alpha' \leq o\ \alpha \wedge (\alpha' = \{\} \vee$ *isSuccOrd* $\alpha') \longrightarrow (\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha'|$ $\longrightarrow$
escl $r\ (\mathfrak{L}\ f\ \alpha')\ (f\ \alpha') \subseteq f\ \alpha' \wedge$ *clterm* (*Restr* $r\ (f\ \alpha'))\ r)$
proof (*intro allI impI*)
fix α'::$'U$ *rel*
assume *d1*: $\alpha' \leq o\ \alpha \wedge (\alpha' = \{\} \vee$ *isSuccOrd* $\alpha')$ **and** *d2*: $\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha'|$
then have $\alpha 0 < o\ \alpha' \vee \alpha' \leq o\ \alpha 0$ **using** *c1* **unfolding** *sc-ord-def*
using *not-ordLeq-iff-ordLess ordLeq-Well-order-simp ordLess-Well-order-simp* **by** *blast*
moreover have $\alpha' \leq o\ \alpha 0 \longrightarrow (\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha'| \longrightarrow$
escl $r\ (\mathfrak{L}\ f\ \alpha')\ (f\ \alpha') \subseteq f\ \alpha' \wedge$ *clterm* (*Restr* $r\ (f\ \alpha'))\ r)$
using *d1 c1* **unfolding** $\mathcal{N}3$*-def* **by** *blast*
moreover have $\alpha 0 < o\ \alpha' \longrightarrow \alpha = o\ \alpha'$ **using** *d1 c1* **unfolding** *sc-ord-def* **using** *ordIso-iff-ordLeq* **by** *blast*
moreover have $\alpha = o\ \alpha' \longrightarrow (\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha'| \longrightarrow$
escl $r\ (\mathfrak{L}\ f\ \alpha')\ (f\ \alpha') \subseteq f\ \alpha' \wedge$ *clterm* (*Restr* $r\ (f\ \alpha'))\ r)$
proof (*intro impI*)
assume *e1*: $\alpha = o\ \alpha'$ **and** *e2*: $\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha'|$
have $\mathfrak{L}\ f\ \alpha \subseteq f\ \alpha 0$
proof
fix p
assume $p \in \mathfrak{L}\ f\ \alpha$
then obtain β::$'U$ *rel* **where** $\beta < o\ \alpha \wedge p \in f\ \beta$ **unfolding** $\mathfrak{L}$*-def* **by** *blast*
moreover then have $\beta \leq o\ \alpha 0 \wedge \alpha 0 \leq o\ \alpha 0$ **using** *c1* **unfolding** *sc-ord-def*
using *not-ordLess-iff-ordLeq ordLess-Well-order-simp* **by** *blast*
moreover then have $f \in \mathcal{N}1\ r\ \alpha 0$ **using** *a0 a1 lem-Shinf-N1*[*of f F*] *ordLeq-Well-order-simp* **by** *metis*
ultimately show $p \in f\ \alpha 0$ **unfolding** $\mathcal{N}1$*-def* **by** *blast*
qed
moreover have $f\ \alpha 0 \subseteq \mathfrak{L}\ f\ \alpha$ **using** *c1* **unfolding** *sc-ord-def* $\mathfrak{L}$*-def* **by** *blast*
ultimately have *e3*: $\mathfrak{L}\ f\ \alpha = f\ \alpha 0$ **by** *blast*
then have $\omega\text{-}ord \leq o\ |f\ \alpha 0|$ **using** *e1 e2 lem-shrel-L-eq* **by** *metis*
moreover have *Well-order* $\alpha 0$ **using** *c1* **unfolding** *sc-ord-def ordLess-def* **by** *blast*
moreover then have $(f\ \alpha 0) \in SF\ r$
using *a5* **unfolding** $\mathcal{N}5$*-def* **using** *ordLeq-reflexive* **by** *blast*
moreover have $f\ \alpha = F\ \alpha 0\ (f\ \alpha 0)$ **using** *c1 b3* **by** *blast*
ultimately have *e4*: *escl* $r\ (f\ \alpha 0)\ (f\ \alpha) \subseteq f\ \alpha \wedge$ *clterm* (*Restr* $r\ (f\ \alpha))\ r$ **using** *a3* **by** *metis*
then have *escl* $r\ (\mathfrak{L}\ f\ \alpha)\ (f\ \alpha) \subseteq f\ \alpha$ **using** *e3* **by** *simp*
then have *escl* $r\ (\mathfrak{L}\ f\ \alpha')\ (f\ \alpha') \subseteq f\ \alpha'$ **using** *e1 b5 lem-shrel-L-eq* **by** *metis*
moreover have *clterm* (*Restr* $r\ (f\ \alpha'))\ r$ **using** *e1 e4 b5* **by** *metis*
ultimately show *escl* $r\ (\mathfrak{L}\ f\ \alpha')\ (f\ \alpha') \subseteq f\ \alpha' \wedge$ *clterm* (*Restr* $r\ (f\ \alpha'))\ r$ **by** *blast*
qed
ultimately show *escl* $r\ (\mathfrak{L}\ f\ \alpha')\ (f\ \alpha') \subseteq f\ \alpha' \wedge$ *clterm* (*Restr* $r\ (f\ \alpha'))\ r$

using *d2* **by** *blast*
  qed
  then show $f \in \mathcal{N}3\ r\ \alpha$ **unfolding** *$\mathcal{N}$3-def* **by** *blast*
 qed
 moreover have $\forall \alpha.\ lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}3\ r\ \beta) \longrightarrow f \in \mathcal{N}3\ r\ \alpha$
 proof (*intro allI impI*)
  fix α::*'U rel*
  assume *c1*: $lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}3\ r\ \beta)$
  then have *c2*: $f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \}$ **using** *b4* **by** *blast*
  have $\forall \alpha'$::*'U rel*. $\alpha' \leq o\ \alpha \wedge (\alpha' = \{\} \vee isSuccOrd\ \alpha') \longrightarrow (\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha'|$
$\longrightarrow$
    $escl\ r\ (\mathfrak{L}\ f\ \alpha')\ (f\ \alpha') \subseteq f\ \alpha' \wedge clterm\ (Restr\ r\ (f\ \alpha'))\ r)$
  proof (*intro allI impI*)
   fix α'::*'U rel*
   assume *d1*: $\alpha' \leq o\ \alpha \wedge (\alpha' = \{\} \vee isSuccOrd\ \alpha')$ **and** *d2*: $\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha'|$
   then have $\alpha' <o\ \alpha \vee \alpha' =o\ \alpha$ **using** *ordLeq-iff-ordLess-or-ordIso* **by** *blast*
   moreover have $\alpha' <o\ \alpha \longrightarrow (\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha'| \longrightarrow$
     $escl\ r\ (\mathfrak{L}\ f\ \alpha')\ (f\ \alpha') \subseteq f\ \alpha' \wedge clterm\ (Restr\ r\ (f\ \alpha'))\ r)$
   proof
    assume $\alpha' <o\ \alpha$
   moreover then have $\alpha' \leq o\ \alpha'$ **using** *ordLess-Well-order-simp ordLeq-reflexive*
by *blast*
    ultimately show $(\omega\text{-}ord \leq o\ |\mathfrak{L}\ f\ \alpha'| \longrightarrow escl\ r\ (\mathfrak{L}\ f\ \alpha')\ (f\ \alpha') \subseteq f\ \alpha' \wedge$
$clterm\ (Restr\ r\ (f\ \alpha'))\ r)$
     using *c1 d1* **unfolding** *$\mathcal{N}$3-def* **by** *blast*
   qed
   moreover have $\alpha' =o\ \alpha \longrightarrow False$
   proof
    assume $\alpha' =o\ \alpha$
    moreover then have $\alpha' = \{\} \vee isSuccOrd\ \alpha$ **using** *d1 lem-osucc-eq* **by**
blast
    moreover have $\neg\ (\alpha = \{\} \vee isSuccOrd\ \alpha)$ **using** *c1* **unfolding** *lm-ord-def*
by *blast*
    ultimately have $\alpha' =o\ \alpha \wedge \alpha' = \{\} \wedge \alpha \neq \{\}$ **by** *blast*
    then show *False* **by** (*metis iso-ozero-empty ordIso-symmetric ozero-def*)
   qed
   ultimately show $escl\ r\ (\mathfrak{L}\ f\ \alpha')\ (f\ \alpha') \subseteq f\ \alpha' \wedge clterm\ (Restr\ r\ (f\ \alpha'))\ r$
using *d2* **by** *blast*
  qed
  then show $f \in \mathcal{N}3\ r\ \alpha$ **unfolding** *$\mathcal{N}$3-def* **by** *blast*
 qed
 ultimately show *?thesis* **using** *lem-sclm-ordind*[*of* $\lambda\ \alpha.\ f \in \mathcal{N}3\ r\ \alpha$] **by** *blast*
qed

lemma *lem-Shinf-N4*:
fixes *r*::*'U rel* **and** *F*::*'U rel* $\Rightarrow$ *'U set* $\Rightarrow$ *'U set* **and** *f*::*'U rel* $\Rightarrow$ *'U set*
assumes *a0*: $f \in \mathcal{T}\ F$
  and *a1*: $\forall\ \alpha\ A.\ Well\text{-}order\ \alpha \longrightarrow A \subseteq F\ \alpha\ A$
  and *a5*: $\forall \alpha.\ Well\text{-}order\ \alpha \longrightarrow f \in \mathcal{N}5\ r\ \alpha$

and $a4$: $\forall\ \alpha\ A.$ *Well-order* $\alpha \longrightarrow A \in SF\ r \longrightarrow (\forall\ a{\in}A.\ r``\{a\} \subseteq w\text{-}dncl\ r\ A$
$\vee\ r``\{a\} \cap (F\ \alpha\ A - w\text{-}dncl\ r\ A) \neq \{\})$
shows $\forall\alpha.$ *Well-order* $\alpha \longrightarrow f \in \mathcal{N}4\ r\ \alpha$
proof −
have $b2$: $f\ \{\} = \{\}$
and $b3$: $\forall\ \alpha0\ \alpha{::}'U\ rel.\ (sc\text{-}ord\ \alpha0\ \alpha \longrightarrow f\ \alpha = F\ \alpha0\ (f\ \alpha0))$
and $b4$: $\forall\ \alpha.\ (lm\text{-}ord\ \alpha \longrightarrow f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \})$
and $b5$: $\forall\alpha\ \beta.\ \alpha =o\ \beta \longrightarrow f\ \alpha = f\ \beta$ **using** *a0* **unfolding** $\mathcal{T}$*-def* **by** *blast+*
have $\mathfrak{L}\ f\ \{\} = \{\}$ **unfolding** $\mathfrak{L}$*-def* **using** *lem-ord-subemp ordLeq-iff-ordLess-or-ordIso ordLess-irreflexive* **by** *blast*
then have $f \in \mathcal{N}4\ r\ \{\}$ **using** *lem-ord-subemp* **unfolding** $\mathcal{N}4$*-def* **by** *blast*
moreover have $\forall\alpha0\ \alpha.\ sc\text{-}ord\ \alpha0\ \alpha \wedge f \in \mathcal{N}4\ r\ \alpha0 \longrightarrow f \in \mathcal{N}4\ r\ \alpha$
proof (*intro allI impI*)
fix $\alpha0\ \alpha{::}'U\ rel$
assume $c1$: $sc\text{-}ord\ \alpha0\ \alpha \wedge f \in \mathcal{N}4\ r\ \alpha0$
have $\forall\alpha'{::}'U\ rel.\ \alpha' \leq o\ \alpha \wedge (\alpha' = \{\} \vee isSuccOrd\ \alpha') \longrightarrow$
$(\ \forall a \in (\mathfrak{L}\ f\ \alpha').\ r``\{a\} \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha') \vee r``\{a\}{\cap}(f\ \alpha' - w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha')){\neq}\{\}\ )$
proof (*intro allI impI*)
fix $\alpha'{::}'U\ rel$
assume $d1$: $\alpha' \leq o\ \alpha \wedge (\alpha' = \{\} \vee isSuccOrd\ \alpha')$
then have $\alpha0 <o\ \alpha' \vee \alpha' \leq o\ \alpha0$ **using** *c1* **unfolding** *sc-ord-def* **using** *not-ordLeq-iff-ordLess ordLeq-Well-order-simp ordLess-Well-order-simp* **by** *blast*
moreover have $\alpha' \leq o\ \alpha0 \longrightarrow (\ \forall a \in (\mathfrak{L}\ f\ \alpha').\ r``\{a\} \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha') \vee r``\{a\}{\cap}(f\ \alpha' - w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha')){\neq}\{\}\ )$
using *d1 c1* **unfolding** $\mathcal{N}4$*-def Dbk-def* $\mathcal{W}$*-def* **by** *blast*
moreover have $\alpha0 <o\ \alpha' \longrightarrow \alpha =o\ \alpha'$ **using** *d1 c1* **unfolding** *sc-ord-def* **using** *ordIso-iff-ordLeq* **by** *blast*
moreover have $\alpha =o\ \alpha' \longrightarrow (\ \forall a \in (\mathfrak{L}\ f\ \alpha').\ r``\{a\} \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha') \vee r``\{a\}{\cap}(f\ \alpha' - w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha')){\neq}\{\}\ )$
proof
assume $e1$: $\alpha =o\ \alpha'$
have *Well-order* $\alpha0$ **using** *c1* **unfolding** *sc-ord-def ordLess-def* **by** *blast*
moreover then have $(f\ \alpha0) \in SF\ r$
using *a5* **unfolding** $\mathcal{N}5$*-def* **using** *ordLeq-reflexive* **by** *blast*
moreover have $f\ \alpha = F\ \alpha0\ (f\ \alpha0)$ **using** *c1 b3* **by** *blast*
ultimately have $e2$: $\forall a \in (f\ \alpha0).\ r``\{a\} \subseteq w\text{-}dncl\ r\ (f\ \alpha0) \vee r``\{a\}{\cap}(f\ \alpha - w\text{-}dncl\ r\ (f\ \alpha0)){\neq}\{\}$
using *a4* **by** *metis*
have $\mathfrak{L}\ f\ \alpha \subseteq f\ \alpha0$
proof
fix p
assume $p \in \mathfrak{L}\ f\ \alpha$
then obtain $\beta{::}'U\ rel$ **where** $\beta <o\ \alpha \wedge p \in f\ \beta$ **unfolding** $\mathfrak{L}$*-def* **by** *blast*
moreover then have $\beta \leq o\ \alpha0 \wedge \alpha0 \leq o\ \alpha0$ **using** *c1* **unfolding** *sc-ord-def*
using *not-ordLess-iff-ordLeq ordLess-Well-order-simp* **by** *blast*
moreover then have $f \in \mathcal{N}1\ r\ \alpha0$ **using** *a0 a1 lem-Shinf-N1*[*of f F*] *ordLeq-Well-order-simp* **by** *metis*

ultimately show $p \in f\ \alpha 0$ **unfolding** $\mathcal{N}1$*-def* **by** *blast*
qed
moreover have $f\ \alpha 0 \subseteq \mathfrak{L}\ f\ \alpha$ **using** *c1* **unfolding** *sc-ord-def* $\mathfrak{L}$*-def* **by** *blast*
ultimately have $\mathfrak{L}\ f\ \alpha = f\ \alpha 0$ **by** *blast*
then have $\mathfrak{L}\ f\ \alpha' = f\ \alpha 0$ **using** *e1 lem-shrel-L-eq* **by** *blast*
then show $\forall a \in (\mathfrak{L}\ f\ \alpha').\ r``\{a\} \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha') \vee r``\{a\} \cap (f\ \alpha' - w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha')) \neq \{\}$
using *e2 e1 b5* **by** *metis*
qed
ultimately show $\forall a \in (\mathfrak{L}\ f\ \alpha').\ r``\{a\} \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha') \vee r``\{a\} \cap (f\ \alpha' - w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha')) \neq \{\}$ **by** *blast*
qed
then show $f \in \mathcal{N}4\ r\ \alpha$ **unfolding** $\mathcal{N}4$*-def Dbk-def* $\mathcal{W}$*-def* **by** *blast*
qed
moreover have $\forall \alpha.\ lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}4\ r\ \beta) \longrightarrow f \in \mathcal{N}4\ r\ \alpha$
proof (*intro allI impI*)
fix α::$'U\ rel$
assume *c1*: $lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}4\ r\ \beta)$
then have *c2*: $f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \}$ **using** *b4* **by** *blast*
have $\forall \alpha'::'U\ rel.\ \alpha' \leq o\ \alpha \wedge (\alpha' = \{\} \vee isSuccOrd\ \alpha') \longrightarrow$
$(\ \forall a \in (\mathfrak{L}\ f\ \alpha').\ r``\{a\} \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha') \vee r``\{a\} \cap (f\ \alpha' - w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha')) \neq \{\}\ )$
proof (*intro allI impI*)
fix α'::$'U\ rel$
assume *d1*: $\alpha' \leq o\ \alpha \wedge (\alpha' = \{\} \vee isSuccOrd\ \alpha')$
then have $\alpha' <o\ \alpha \vee \alpha' =o\ \alpha$ **using** *ordLeq-iff-ordLess-or-ordIso* **by** *blast*
moreover have $\alpha' <o\ \alpha \longrightarrow (\ \forall a \in (\mathfrak{L}\ f\ \alpha').\ r``\{a\} \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha') \vee r``\{a\} \cap (f\ \alpha' - w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha')) \neq \{\}\ )$
proof
assume $\alpha' <o\ \alpha$
moreover then have $\alpha' \leq o\ \alpha'$ **using** *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
ultimately show $(\ \forall a \in (\mathfrak{L}\ f\ \alpha').\ r``\{a\} \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha') \vee r``\{a\} \cap (f\ \alpha' - w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha')) \neq \{\}\ )$
using *c1 d1* **unfolding** $\mathcal{N}4$*-def Dbk-def* $\mathcal{W}$*-def* **by** *blast*
qed
moreover have $\alpha' =o\ \alpha \longrightarrow False$
proof
assume $\alpha' =o\ \alpha$
moreover then have $\alpha' = \{\} \vee isSuccOrd\ \alpha$ **using** *d1 lem-osucc-eq* **by** *blast*
moreover have $\neg\ (\alpha = \{\} \vee isSuccOrd\ \alpha)$ **using** *c1* **unfolding** *lm-ord-def* **by** *blast*
ultimately have $\alpha' =o\ \alpha \wedge \alpha' = \{\} \wedge \alpha \neq \{\}$ **by** *blast*
then show *False* **by** (*metis iso-ozero-empty ordIso-symmetric ozero-def*)
qed
ultimately show $\forall a \in (\mathfrak{L}\ f\ \alpha').\ r``\{a\} \subseteq w\text{-}dncl\ r\ (\mathfrak{L}\ f\ \alpha') \vee r``\{a\} \cap (f\ \alpha'$

− *w-dncl r* ($\mathfrak{L}$ *f* α'))≠{} **by** *blast*
  qed
  then show $f \in \mathcal{N}4\ r\ \alpha$ **unfolding** *$\mathcal{N}$4-def Dbk-def $\mathcal{W}$-def* **by** *blast*
 qed
 ultimately show *?thesis* **using** *lem-sclm-ordind*[*of* $\lambda\ \alpha.\ f \in \mathcal{N}4\ r\ \alpha$] **by** *blast*
qed

lemma *lem-Shinf-N5*:
fixes *r*::*'U rel* **and** *F*::*'U rel* ⇒ *'U set* ⇒ *'U set* **and** *f*::*'U rel* ⇒ *'U set*
assumes *a0*: $f \in \mathcal{T}\ F$
assumes *a5*: ∀ α *A*. (*Well-order* α ∧ *A* ∈ *SF r*) ⟶ (*F* α *A*) ∈ *SF r*
shows ∀α. *Well-order* α ⟶ $f \in \mathcal{N}5\ r\ \alpha$
proof −
 have *b2*: *f* {} = {}
  and *b3*: ∀ α*0* α::*'U rel*. (*sc-ord* α*0* α ⟶ *f* α = *F* α*0* (*f* α*0*))
  and *b4*: ∀ α. (*lm-ord* α ⟶ *f* α = ⋃ { *D*. ∃ β. β <*o* α ∧ *D* = *f* β })
  and *b5*: ∀α β. α =*o* β ⟶ *f* α = *f* β **using** *a0* **unfolding** *$\mathcal{T}$-def* **by** *blast+*
 have $f \in \mathcal{N}5\ r$ {} **using** *b2 lem-ord-subemp* **unfolding** *$\mathcal{N}$5-def SF-def Field-def*
by *blast*
 moreover have ∀α*0* α. *sc-ord* α*0* α ∧ $f \in \mathcal{N}5\ r$ α*0* ⟶ $f \in \mathcal{N}5\ r\ \alpha$
 proof (*intro allI impI*)
  fix α*0* α::*'U rel*
  assume *c1*: *sc-ord* α*0* α ∧ $f \in \mathcal{N}5\ r$ α*0*
  have ∀α'::*'U rel*. α' ≤*o* α ⟶ (*f* α') ∈ *SF r*
  proof (*intro allI impI*)
   fix α'::*'U rel*
   assume *d1*: α' ≤*o* α
   then have α*0* <*o* α' ∨ α' ≤*o* α*0* **using** *c1* **unfolding** *sc-ord-def*
   using *not-ordLeq-iff-ordLess ordLeq-Well-order-simp ordLess-Well-order-simp*
by *blast*
   moreover have α' ≤*o* α*0* ⟶ *Field* (*Restr r* (*f* α')) = (*f* α') **using** *c1*
unfolding *$\mathcal{N}$5-def SF-def* **by** *blast*
   moreover have α*0* <*o* α' ⟶ α =*o* α' **using** *d1 c1* **unfolding** *sc-ord-def*
using *ordIso-iff-ordLeq* **by** *blast*
   moreover have α =*o* α' ⟶ (*f* α') ∈ *SF r*
   proof
    assume α =*o* α'
    moreover have (*f* α) ∈ *SF r*
    proof −
     have α*0* ≤*o* α*0* **using** *c1* **unfolding** *sc-ord-def*
      using *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
     then have (*f* α*0*) ∈ *SF r* **using** *c1* **unfolding** *$\mathcal{N}$5-def* **by** *blast*
     moreover have *Well-order* α*0* **using** *c1* **unfolding** *sc-ord-def* **using**
ordLess-Well-order-simp **by** *blast*
     moreover have *f* α = *F* α*0* (*f* α*0*) **using** *c1 b3* **by** *blast*
     ultimately show (*f* α) ∈ *SF r* **using** *a5* **by** *metis*
    qed
    ultimately show (*f* α') ∈ *SF r* **using** *b5* **by** *metis*
   qed

ultimately show $(f\ \alpha') \in SF\ r$ **unfolding** *SF-def* **by** *blast*
qed
then show $f \in \mathcal{N}5\ r\ \alpha$ **unfolding** *$\mathcal{N}$5-def* **by** *blast*
qed
moreover have $\forall \alpha.\ lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}5\ r\ \beta) \longrightarrow f \in \mathcal{N}5\ r\ \alpha$
proof (*intro allI impI*)
fix α::'*U rel*
assume *c1*: $lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}5\ r\ \beta)$
then have *c2*: $f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \}$ **using** *b4* **by** *blast*
have $\forall \alpha'$::'*U rel*. $\alpha' \leq o\ \alpha \longrightarrow (f\ \alpha') \in SF\ r$
proof (*intro allI impI*)
fix α'::'*U rel*
assume *d1*: $\alpha' \leq o\ \alpha$
then have $\alpha' <o\ \alpha \vee \alpha' =o\ \alpha$ **using** *ordLeq-iff-ordLess-or-ordIso* **by** *blast*
moreover have $\alpha' <o\ \alpha \longrightarrow$ *Field* (*Restr r* $(f\ \alpha')) = (f\ \alpha')$
proof
assume $\alpha' <o\ \alpha$
moreover then have $\alpha' \leq o\ \alpha'$ **using** *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
ultimately show *Field* (*Restr r* $(f\ \alpha')) = (f\ \alpha')$ **using** *c1 d1* **unfolding** *$\mathcal{N}$5-def SF-def* **by** *blast*
qed
moreover have $\alpha' =o\ \alpha \longrightarrow (f\ \alpha') \in SF\ r$
proof
assume $\alpha' =o\ \alpha$
moreover have $(f\ \alpha) \in SF\ r$
proof −
have $\forall\ \beta.\ \beta <o\ \alpha \longrightarrow (f\ \beta) \in SF\ r$ **using** *c1* **unfolding** *$\mathcal{N}$5-def*
using *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
then show *?thesis* **using** *c2 lem-Relprop-sat-un*[*of* $\{D.\ \exists \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\}\ r\ f\ \alpha$] **unfolding** *SF-def* **by** *blast*
qed
ultimately show $(f\ \alpha') \in SF\ r$ **using** *b5* **by** *metis*
qed
ultimately show $(f\ \alpha') \in SF\ r$ **unfolding** *SF-def* **by** *blast*
qed
then show $f \in \mathcal{N}5\ r\ \alpha$ **unfolding** *$\mathcal{N}$5-def* **by** *blast*
qed
ultimately show *?thesis* **using** *lem-sclm-ordind*[*of* $\lambda\ \alpha.\ f \in \mathcal{N}5\ r\ \alpha$] **by** *blast*
qed

lemma *lem-Shinf-N6*:
fixes *r*::'*U rel* **and** *F*::'*U rel* ⇒ '*U set* ⇒ '*U set* **and** *f*::'*U rel* ⇒ '*U set*
assumes *a0*: $f \in \mathcal{T}\ F$
and *a1*: $\forall\ \alpha\ A.$ *Well-order* $\alpha \longrightarrow A \subseteq F\ \alpha\ A$
and *a5*: $\forall \alpha.$ *Well-order* $\alpha \longrightarrow f \in \mathcal{N}5\ r\ \alpha$
and *a6*: $\forall\ \alpha\ A.$ *Well-order* $\alpha \longrightarrow A \in SF\ r \longrightarrow CCR$ (*Restr r* $(F\ \alpha\ A))$
shows $\forall \alpha.$ *Well-order* $\alpha \longrightarrow f \in \mathcal{N}6\ r\ \alpha$

proof –
have *b2*: f {} = {}
and *b3*: ∀ $\alpha 0$ α::'*U rel*. (*sc-ord* $\alpha 0$ $\alpha \longrightarrow f\ \alpha = F\ \alpha 0\ (f\ \alpha 0)$)
and *b4*: ∀ α. (*lm-ord* $\alpha \longrightarrow f\ \alpha = \bigcup$ { D. ∃ β. β <*o* $\alpha \wedge D = f\ \beta$ })
and *b5*: ∀ α β. α =*o* $\beta \longrightarrow f\ \alpha = f\ \beta$ **using** *a0* **unfolding** $\mathcal{T}$*-def* **by** *blast+*
have $f \in \mathcal{N}6\ r$ {} **using** *b2 lem-ord-subemp* **unfolding** $\mathcal{N}$*6-def CCR-def Field-def* **by** *blast*
moreover have ∀ $\alpha 0$ α. *sc-ord* $\alpha 0$ $\alpha \wedge f \in \mathcal{N}6\ r\ \alpha 0 \longrightarrow f \in \mathcal{N}6\ r\ \alpha$
proof (*intro allI impI*)
fix $\alpha 0$ α::'*U rel*
assume *c1*: *sc-ord* $\alpha 0$ $\alpha \wedge f \in \mathcal{N}6\ r\ \alpha 0$
then have *c2*: $f\ \alpha = F\ \alpha 0\ (f\ \alpha 0)$ **using** *b3* **by** *blast*
have ∀ α'. α' ≤*o* $\alpha \longrightarrow$ *CCR* (*Restr r* ($f\ \alpha'$))
proof (*intro allI impI*)
fix α'::'*U rel*
assume α' ≤*o* α
then have α' ≤*o* $\alpha 0$ ∨ α' =*o* α **using** *c1* **unfolding** *sc-ord-def*
by (*meson ordIso-iff-ordLeq ordLeq-Well-order-simp ordLess-Well-order-simp ordLess-or-ordLeq*)
moreover have α' ≤*o* $\alpha 0 \longrightarrow$ *CCR* (*Restr r* ($f\ \alpha'$)) **using** *c1* **unfolding** $\mathcal{N}$*6-def* **by** *blast*
moreover have α' =*o* $\alpha \longrightarrow$ *CCR* (*Restr r* ($f\ \alpha'$))
proof
assume α' =*o* α
moreover have *CCR* (*Restr r* ($f\ \alpha$))
proof –
have *Well-order* $\alpha 0$
using *c1 ordLess-Well-order-simp* **unfolding** *sc-ord-def* **by** *blast*
moreover then have ($f\ \alpha 0$) ∈ *SF r*
using *a5* **unfolding** $\mathcal{N}$*5-def* **using** *ordLeq-reflexive* **by** *blast*
ultimately show *CCR* (*Restr r* ($f\ \alpha$)) **unfolding** *c2* **using** *a6* **by** *blast*
qed
ultimately show *CCR* (*Restr r* ($f\ \alpha'$)) **using** *b5* **by** *metis*
qed
ultimately show *CCR* (*Restr r* ($f\ \alpha'$)) **by** *blast*
qed
then show $f \in \mathcal{N}6\ r\ \alpha$ **unfolding** $\mathcal{N}$*6-def* **by** *blast*
qed
moreover have ∀ α. *lm-ord* $\alpha \wedge$ (∀ β. β <*o* $\alpha \longrightarrow f \in \mathcal{N}6\ r\ \beta$) $\longrightarrow f \in \mathcal{N}6\ r\ \alpha$
proof (*intro allI impI*)
fix α::'*U rel*
assume *c1*: *lm-ord* $\alpha \wedge$ (∀ β. β <*o* $\alpha \longrightarrow f \in \mathcal{N}6\ r\ \beta$)
then have *c2*: $f\ \alpha = \bigcup$ { D. ∃ β. β <*o* $\alpha \wedge D = f\ \beta$ } **using** *b4* **by** *blast*
have *c3*: ∀ α'. α' ≤*o* $\alpha \longrightarrow$ *CCR* (*Restr r* ($f\ \alpha'$))
proof (*intro allI impI*)
fix α'::'*U rel*
assume α' ≤*o* α
then have α' <*o* α ∨ α' =*o* α **using** *ordIso-iff-ordLeq ordLeq-Well-order-simp ordLess-or-ordLeq* **by** *blast*

moreover have $\alpha' <o\ \alpha \longrightarrow$ *CCR* (*Restr r* (*f* α')) **using** *c1* **unfolding** $\mathcal{N}$*6-def*
using *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
moreover have $\alpha' =o\ \alpha \longrightarrow$ *CCR* (*Restr r* (*f* α'))
proof
assume $\alpha' =o\ \alpha$
moreover have *CCR* (*Restr r* (*f* α))
proof −
obtain *C* **where** *f1*: $C = \{ A.\ \exists\ \beta::'U\ rel.\ \beta <o\ \alpha \wedge A = f\ \beta \}$ **by** *blast*
obtain *S* **where** *f2*: $S = \{ s.\ \exists\ A \in C.\ s = Restr\ r\ A \}$ **by** *blast*
have *f3*: $\forall A1 \in C.\ \forall A2 \in C.\ A1 \subseteq A2 \vee A2 \subseteq A1$
proof (*intro ballI*)
fix *A1 A2*
assume $A1 \in C$ **and** $A2 \in C$
then obtain $\beta 1\ \beta 2::'U\ rel$ **where** $A1 = f\ \beta 1 \wedge A2 = f\ \beta 2 \wedge \beta 1 <o\ \alpha \wedge \beta 2 <o\ \alpha$ **using** *f1* **by** *blast*
moreover then have $(\beta 1 \leq o\ \beta 2 \vee \beta 2 \leq o\ \beta 1) \wedge \beta 1 \leq o\ \alpha \wedge \beta 2 \leq o\ \alpha$
using *ordLeq-total ordLess-Well-order-simp ordLess-imp-ordLeq* **by** *blast*
moreover have $f \in \mathcal{N}1\ r\ \alpha$ **using** *a0 a1 c1 lem-Shinf-N1*[*of f F r*] **unfolding** *lm-ord-def* **by** *blast*
ultimately show $A1 \subseteq A2 \vee A2 \subseteq A1$ **unfolding** $\mathcal{N}$*1-def* **by** *blast*
qed
have $\forall\ s \in S.$ *CCR s* **using** *f1 f2 c1* **unfolding** $\mathcal{N}$*6-def*
using *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
moreover have $\forall s1 \in S.\ \forall s2 \in S.\ s1 \subseteq s2 \vee s2 \subseteq s1$ **using** *f2 f3* **by** *blast*
ultimately have *CCR* ($\bigcup$ *S*) **using** *lem-Relprop-ccr-ch-un*[*of S*] **by** *blast*
moreover have *Restr r* $(\bigcup \{D.\ \exists \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\}) = \bigcup\ S$
using *f1 f2 f3 lem-Relprop-restr-ch-un*[*of C r*] **by** *blast*
ultimately show *?thesis* **unfolding** *c2* **by** *simp*
qed
ultimately show *CCR* (*Restr r* (*f* α')) **using** *b5* **by** *metis*
qed
ultimately show *CCR* (*Restr r* (*f* α')) **by** *blast*
qed
then show $f \in \mathcal{N}6\ r\ \alpha$ **unfolding** $\mathcal{N}$*6-def* **by** *blast*
qed
ultimately show *?thesis* **using** *lem-sclm-ordind*[*of* $\lambda\ \alpha.\ f \in \mathcal{N}6\ r\ \alpha$] **by** *blast*
qed

lemma *lem-Shinf-N7*:
fixes $r::'U\ rel$ **and** $F::'U\ rel \Rightarrow 'U\ set \Rightarrow 'U\ set$ **and** $f::'U\ rel \Rightarrow 'U\ set$
assumes *a0*: $f \in \mathcal{T}\ F$
and *a1*: $\forall\ \alpha\ A.$ *Well-order* $\alpha \longrightarrow A \subseteq F\ \alpha\ A$
and *a7*: $\forall\ \alpha\ A.\ (|A| <o\ \omega\text{-}ord \longrightarrow |F\ \alpha\ A| <o\ \omega\text{-}ord)$
$\wedge\ (\omega\text{-}ord \leq o\ |A| \longrightarrow |F\ \alpha\ A| \leq o\ |A|)$
shows $\forall \alpha.$ *Well-order* $\alpha \longrightarrow f \in \mathcal{N}7\ r\ \alpha$
proof −
have *b2*: $f\ \{\} = \{\}$
and *b3*: $\forall\ \alpha 0\ \alpha::'U\ rel.\ (sc\text{-}ord\ \alpha 0\ \alpha \longrightarrow f\ \alpha = F\ \alpha 0\ (f\ \alpha 0))$

and *b4*: $\forall$ *α*. (*lm-ord α* $\longrightarrow$ *f α* = $\bigcup$ { *D*. $\exists$ *β*. *β <o α* $\wedge$ *D = f β* })
and *b5*: $\forall$ *α β*. *α =o β* $\longrightarrow$ *f α = f β* **using** *a0* **unfolding** *𝒯-def* **by** *blast+*
have $\forall$ *α::'U rel*. *α ≤o* {} $\longrightarrow$ |*f α*| *≤o α* $\wedge$ |*f α*| *<o ω-ord*
proof (*intro allI impI*)
fix *α::'U rel*
assume *α ≤o* {}
moreover then have (*f α*) = {} **using** *b2 lem-ord-subemp* **by** *blast*
ultimately show |*f α*| *≤o α* $\wedge$ |*f α*| *<o ω-ord* **using** *lem-ord-subemp*
by (*metis Field-natLeq card-of-empty1 card-of-empty5 ctwo-def ctwo-ordLess-natLeq natLeq-well-order-on not-ordLeq-iff-ordLess ordLeq-Well-order-simp*)
qed
then have *f* $\in$ *𝒩7 r* {} **unfolding** *𝒩7-def* **by** *blast*
moreover have $\forall$ *α0 α*. *sc-ord α0 α* $\wedge$ *f* $\in$ *𝒩7 r α0* $\longrightarrow$ *f* $\in$ *𝒩7 r α*
proof (*intro allI impI*)
fix *α0 α::'U rel*
assume *c1*: *sc-ord α0 α* $\wedge$ *f* $\in$ *𝒩7 r α0*
then have *c2*: *f α = F α0* (*f α0*) **using** *b3* **by** *blast*
have $\forall$ *α'*. *α' ≤o α* $\wedge$ *ω-ord ≤o α'* $\longrightarrow$ |*f α'*| *≤o α'*
proof (*intro allI impI*)
fix *α'::'U rel*
assume *d1*: *α' ≤o α* $\wedge$ *ω-ord ≤o α'*
then have *α' ≤o α0* $\vee$ *α' =o α* **using** *c1* **unfolding** *sc-ord-def*
by (*meson ordIso-iff-ordLeq ordLeq-Well-order-simp ordLess-Well-order-simp ordLess-or-ordLeq*)
moreover have *α' ≤o α0* $\longrightarrow$ |*f α'*| *≤o α'* **using** *c1 d1* **unfolding** *𝒩7-def* **by** *blast*
moreover have *α' =o α* $\longrightarrow$ |*f α'*| *≤o α'*
proof
assume *e1*: *α' =o α*
then have *e2*: *ω-ord ≤o α* **using** *d1 b5 ordLeq-transitive* **by** *blast*
then have *e3*: *ω-ord ≤o α0* **using** *c1 lem-ord-suc-ge-w* **by** *blast*
then have *Well-order α0* $\wedge$ |*f α0*| *≤o α0*
using *c1* **unfolding** *sc-ord-def 𝒩7-def* **using** *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
moreover then have |*f α*| *≤o* |*f α0*| $\vee$ |*f α*| *<o ω-ord* **unfolding** *c2* **using** *a7*
using *finite-iff-ordLess-natLeq infinite-iff-natLeq-ordLeq* **by** *blast*
moreover have *α0 ≤o α* **using** *c1* **unfolding** *sc-ord-def* **using** *ordLess-imp-ordLeq* **by** *blast*
ultimately have |*f α*| *≤o α* **using** *e3 ordLeq-transitive ordLess-imp-ordLeq* **by** *metis*
then show |*f α'*| *≤o α'* **using** *b5 e1 ordIso-iff-ordLeq ordLeq-transitive* **by** *metis*
qed
ultimately show |*f α'*| *≤o α'* **by** *blast*
qed
moreover have $\forall$ *α'*. *α' ≤o α* $\wedge$ *α' <o ω-ord* $\longrightarrow$ |*f α'*| *<o ω-ord*
proof (*intro allI impI*)
fix *α'::'U rel*

assume *d1*: $\alpha' \leq o\ \alpha \wedge \alpha' <o\ \omega\text{-}ord$
then have $\alpha' \leq o\ \alpha 0 \vee \alpha' =o\ \alpha$ **using** *c1* **unfolding** *sc-ord-def*
by (*meson ordIso-iff-ordLeq ordLeq-Well-order-simp ordLess-Well-order-simp ordLess-or-ordLeq*)
moreover have $\alpha' \leq o\ \alpha 0 \longrightarrow |f\ \alpha'| <o\ \omega\text{-}ord$ **using** *c1 d1* **unfolding** *𝒩7-def* **by** *blast*
moreover have $\alpha' =o\ \alpha \longrightarrow |f\ \alpha'| <o\ \omega\text{-}ord$
proof
assume *e1*: $\alpha' =o\ \alpha$
then have *e2*: $\alpha <o\ \omega\text{-}ord$ **using** *d1 ordIso-iff-ordLeq ordIso-ordLess-trans* **by** *blast*
then have *e3*: $\alpha 0\ <o\ \omega\text{-}ord$ **using** *c1* **unfolding** *sc-ord-def* **using** *ordLeq-ordLess-trans ordLess-imp-ordLeq* **by** *blast*
then have *Well-order* $\alpha 0 \wedge |f\ \alpha 0| <o\ \omega\text{-}ord$
using *c1* **unfolding** *sc-ord-def 𝒩7-def* **using** *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
then have $|f\ \alpha| <o\ \omega\text{-}ord$ **unfolding** *c2* **using** *a7* **by** *blast*
then show $|f\ \alpha'| <o\ \omega\text{-}ord$ **using** *b5 e1* **by** *metis*
qed
ultimately show $|f\ \alpha'| <o\ \omega\text{-}ord$ **by** *blast*
qed
ultimately show $f \in \mathcal{N}7\ r\ \alpha$ **unfolding** *𝒩7-def* **by** *blast*
qed
moreover have $\forall \alpha.\ lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}7\ r\ \beta) \longrightarrow f \in \mathcal{N}7\ r\ \alpha$
proof (*intro allI impI*)
fix α::*'U rel*
assume *c1*: $lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}7\ r\ \beta)$
then have *c2*: $f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \}$ **using** *b4* **by** *blast*
have $\forall \alpha'.\ \alpha' \leq o\ \alpha \wedge \omega\text{-}ord \leq o\ \alpha' \longrightarrow |f\ \alpha'| \leq o\ \alpha'$
proof (*intro allI impI*)
fix α'::*'U rel*
assume *e1*: $\alpha' \leq o\ \alpha \wedge \omega\text{-}ord \leq o\ \alpha'$
then have $\alpha' <o\ \alpha \vee \alpha' =o\ \alpha$ **using** *ordIso-iff-ordLeq ordLeq-Well-order-simp ordLess-or-ordLeq* **by** *blast*
moreover have $\alpha' <o\ \alpha \longrightarrow |f\ \alpha'| \leq o\ \alpha'$ **using** *c1 e1* **unfolding** *𝒩7-def*
using *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
moreover have $\alpha' =o\ \alpha \longrightarrow |f\ \alpha'| \leq o\ \alpha'$
proof
assume $\alpha' =o\ \alpha$
moreover have $|f\ \alpha| \leq o\ \alpha$
proof −
obtain S **where** *f1*: $S = \{\ A.\ \exists\ \ \beta::'U\ rel.\ \beta <o\ \alpha \wedge A = f\ \beta\ \}$ **by** *blast*
have *f2*: $\omega\text{-}ord \leq o\ \alpha$ **using** *c1 lem-lmord-inf lem-inford-ge-w* **unfolding** *lm-ord-def* **by** *blast*
have *f3*: $\forall\ \ s \in S.\ |s| \leq o\ \alpha$
proof
fix s
assume $s \in S$
then obtain β **where** $\beta <o\ \alpha \wedge s = f\ \beta$ **using** *f1* **by** *blast*

then show $|s| \leq o\ \alpha$
using *c1 f2* **unfolding** $\mathcal{N}$*7-def* **apply** *clarsimp*
by (*metis card-of-Well-order natLeq-Well-order not-ordLess-ordLeq ordLeq-reflexive ordLess-Well-order-simp ordLess-or-ordLeq ordLess-transitive*)
qed
moreover have $|S| \leq o\ \alpha$
proof −
have f ' $\{\gamma.\ \gamma <o\ \alpha\} = S$ **using** *f1* **by** *force*
then show *?thesis* **using** *f1 f2 b5 lem-ord-int-card-le-inf*[*of f* α] **by** *blast*
qed
ultimately have $|\bigcup\ S| \leq o\ \alpha$ **using** *f2 lem-card-un-bnd*[*of S* α] **by** *blast*
then show *?thesis* **unfolding** *f1 c2* **by** *blast*
qed
ultimately show $|f\ \alpha'| \leq o\ \alpha'$ **using** *b5 ordIso-iff-ordLeq ordLeq-transitive* **by** *metis*
qed
ultimately show $|f\ \alpha'| \leq o\ \alpha'$ **by** *blast*
qed
moreover have $\forall \alpha'.\ \alpha' \leq o\ \alpha \wedge \alpha' <o\ \omega\text{-}ord \longrightarrow |f\ \alpha'| <o\ \omega\text{-}ord$
proof (*intro allI impI*)
fix α'::$'U$ *rel*
assume *e1*: $\alpha' \leq o\ \alpha \wedge \alpha' <o\ \omega\text{-}ord$
then have $\alpha' <o\ \alpha \vee \alpha' =o\ \alpha$ **using** *ordIso-iff-ordLeq ordLeq-Well-order-simp ordLess-or-ordLeq* **by** *blast*
moreover have $\alpha' <o\ \alpha \longrightarrow |f\ \alpha'| <o\ \omega\text{-}ord$ **using** *c1 e1* **unfolding** $\mathcal{N}$*7-def*
using *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
moreover have $\alpha' =o\ \alpha \longrightarrow |f\ \alpha'| <o\ \omega\text{-}ord$
proof
assume $\alpha' =o\ \alpha$
moreover have $|f\ \alpha| \leq o\ \alpha$
proof −
obtain S **where** *f1*: $S = \{\ A.\ \exists\ \ \beta$::$'U$ *rel*. $\beta <o\ \alpha \wedge A = f\ \beta\ \}$ **by** *blast*
have *f2*: $\omega\text{-}ord \leq o\ \alpha$ **using** *c1 lem-lmord-inf lem-inford-ge-w* **unfolding** *lm-ord-def* **by** *blast*
have *f3*: $\forall\ \ s \in S.\ |s| \leq o\ \alpha$
proof
fix s
assume $s \in S$
then obtain β **where** $\beta <o\ \alpha \wedge s = f\ \beta$ **using** *f1* **by** *blast*
then show $|s| \leq o\ \alpha$
using *c1 f2* **unfolding** $\mathcal{N}$*7-def* **apply** *clarsimp*
by (*metis card-of-Well-order natLeq-Well-order not-ordLess-ordLeq ordLeq-reflexive ordLess-Well-order-simp ordLess-or-ordLeq ordLess-transitive*)
qed
moreover have $|S| \leq o\ \alpha$
proof −
have f ' $\{\gamma.\ \gamma <o\ \alpha\} = S$ **using** *f1* **by** *force*
then show *?thesis* **using** *f1 f2 b5 lem-ord-int-card-le-inf*[*of f* α] **by** *blast*
qed

ultimately have $|\bigcup S| \leq o\ \alpha$ **using** *f2 lem-card-un-bnd*[*of S α*] **by** *blast*
then show *?thesis* **unfolding** *f1 c2* **by** *blast*
qed
ultimately show $|f\ \alpha'| <o\ \omega\text{-}ord$ **using** *e1 b5 ordIso-iff-ordLeq ordLeq-transitive*
by (*metis card-of-Well-order natLeq-Well-order not-ordLess-ordLeq ordLess-or-ordLeq*)
qed
ultimately show $|f\ \alpha'| <o\ \omega\text{-}ord$ **by** *blast*
qed
ultimately show $f \in \mathcal{N}7\ r\ \alpha$ **unfolding** *𝒩7-def* **by** *blast*
qed
ultimately show *?thesis* **using** *lem-sclm-ordind*[*of* $\lambda\ \alpha.\ f \in \mathcal{N}7\ r\ \alpha$] **by** *blast*
qed

lemma *lem-Shinf-N8*:
fixes $r::'U\ rel$ **and** $F::'U\ rel \Rightarrow 'U\ set \Rightarrow 'U\ set$ **and** $f::'U\ rel \Rightarrow 'U\ set$ **and** $Ps::'U\ set\ set$
assumes *a0*: $f \in \mathcal{T}\ F$
and *a1*: $\forall\ \alpha\ A.\ Well\text{-}order\ \alpha \longrightarrow A \subseteq F\ \alpha\ A$
and *a5*: $\forall \alpha.\ Well\text{-}order\ \alpha \longrightarrow f \in \mathcal{N}5\ r\ \alpha$
and *a7*: $\forall\ \alpha\ A.\ (\ |A| <o\ \omega\text{-}ord \longrightarrow |F\ \alpha\ A| <o\ \omega\text{-}ord\ )$
$\wedge\ (\ \omega\text{-}ord \leq o\ |A| \longrightarrow |F\ \alpha\ A| \leq o\ |A|\ )$
and *a8*: $\forall \alpha\ A.\ A \in SF\ r \longrightarrow \mathcal{E}p\ r\ Ps\ A\ (F\ \alpha\ A)$
shows $\forall \alpha.\ Well\text{-}order\ \alpha \longrightarrow f \in \mathcal{N}8\ r\ Ps\ \alpha$
proof −
have *b2*: $f\ \{\} = \{\}$
and *b3*: $\forall\ \alpha 0\ \alpha::'U\ rel.\ (sc\text{-}ord\ \alpha 0\ \alpha \longrightarrow f\ \alpha = F\ \alpha 0\ (f\ \alpha 0))$
and *b4*: $\forall\ \alpha.\ (lm\text{-}ord\ \alpha \longrightarrow f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \})$
and *b5*: $\forall \alpha\ \beta.\ \alpha =o\ \beta \longrightarrow f\ \alpha = f\ \beta$ **using** *a0* **unfolding** *𝒯-def* **by** *blast+*
have $f \in \mathcal{N}8\ r\ Ps\ \{\}$ **using** *b2 lem-ord-subemp* **unfolding** *𝒩8-def SCF-def Field-def* **by** *blast*
moreover have $\forall \alpha 0\ \alpha.\ sc\text{-}ord\ \alpha 0\ \alpha \wedge f \in \mathcal{N}8\ r\ Ps\ \alpha 0 \longrightarrow f \in \mathcal{N}8\ r\ Ps\ \alpha$
proof (*intro allI impI*)
fix $\alpha 0\ \alpha::'U\ rel$
assume *c1*: $sc\text{-}ord\ \alpha 0\ \alpha \wedge f \in \mathcal{N}8\ r\ Ps\ \alpha 0$
have $\forall \alpha'::'U\ rel.\ \alpha' \leq o\ \alpha \wedge (\alpha' = \{\} \vee isSuccOrd\ \alpha') \longrightarrow$
$((\exists P.\ Ps = \{P\}) \vee (\neg\ finite\ Ps \wedge |Ps| \leq o\ |f\ \alpha'|\ )) \longrightarrow (\forall P{\in}Ps.\ f\ \alpha' \cap P \in SCF\ (Restr\ r\ (f\ \alpha')))$
proof (*intro allI*, *rule impI*)
fix $\alpha'::'U\ rel$
assume *d1*: $\alpha' \leq o\ \alpha \wedge (\alpha' = \{\} \vee isSuccOrd\ \alpha')$
then have $\alpha 0 <o\ \alpha' \vee \alpha' \leq o\ \alpha 0$ **using** *c1* **unfolding** *sc-ord-def*
using *not-ordLeq-iff-ordLess ordLeq-Well-order-simp ordLess-Well-order-simp* **by** *blast*
moreover have $\alpha' \leq o\ \alpha 0 \longrightarrow ((\exists P.\ Ps = \{P\}) \vee (\neg\ finite\ Ps \wedge |Ps| \leq o\ |f\ \alpha'|\ )) \longrightarrow$
$(\forall P{\in}Ps.\ f\ \alpha' \cap P \in SCF\ (Restr\ r\ (f\ \alpha')))$
using *d1 c1* **unfolding** *𝒩8-def* **by** *blast*

moreover have $\alpha 0 <o\ \alpha' \longrightarrow \alpha =o\ \alpha'$ **using** *d1 c1* **unfolding** *sc-ord-def* **using** *ordIso-iff-ordLeq* **by** *blast*
moreover have $\alpha =o\ \alpha' \longrightarrow ((\exists P.\ Ps = \{P\}) \lor (\neg\ finite\ Ps \land |Ps| \leq o\ |f\ \alpha'|\ )) \longrightarrow$
$(\forall P \in Ps.\ f\ \alpha' \cap P \in SCF\ (Restr\ r\ (f\ \alpha')))$
proof (*intro ballI impI*)
fix P
assume *e1*: $\alpha =o\ \alpha'$ **and** *e2*: $(\exists P'.\ Ps = \{P'\}) \lor (\neg\ finite\ Ps \land |Ps| \leq o\ |f\ \alpha'|\ )$ **and** *e3*: $P \in Ps$
have *e4*: $f\ \alpha' = f\ \alpha$ **using** *b5 e1* **by** *blast*
have *Well-order* $\alpha 0$ **using** *c1* **unfolding** *sc-ord-def ordLess-def* **by** *blast*
then have $(f\ \alpha 0) \in SF\ r$ **using** *a5* **unfolding** $\mathcal{N}$*5-def* **using** *ordLeq-reflexive* **by** *blast*
moreover have *e5*: $f\ \alpha = F\ \alpha 0\ (f\ \alpha 0)$ **using** *c1 b3* **by** *blast*
moreover have $\neg\ (\exists P'.\ Ps = \{P'\}) \longrightarrow (\neg\ finite\ Ps \land |Ps| \leq o\ |f\ \alpha 0|\ )$
proof
assume *f1*: $\neg\ (\exists P'.\ Ps = \{P'\})$
then have *f2*: $\omega\text{-}ord \leq o\ |Ps| \land |Ps| \leq o\ |f\ \alpha|$ **using** *e2 e4 infinite-iff-natLeq-ordLeq* **by** *metis*
then have $\neg\ |F\ \alpha 0\ (f\ \alpha 0)| <o\ \omega\text{-}ord$ **using** *e5*
by (*metis finite-ordLess-infinite2 infinite-iff-natLeq-ordLeq not-ordLess-ordLeq*)
then have $\neg\ |f\ \alpha 0| <o\ \omega\text{-}ord$ **using** *a7* **by** *blast*
then have $\omega\text{-}ord \leq o\ |f\ \alpha 0|$ **by** (*metis finite-iff-ordLess-natLeq infinite-iff-natLeq-ordLeq*)
then have $|F\ \alpha 0\ (f\ \alpha 0)| \leq o\ |f\ \alpha 0|$ **using** *a7* **by** *blast*
then have $|Ps| \leq o\ |f\ \alpha 0|$ **using** *f2 e5 ordLeq-transitive* **by** *metis*
then show $\neg\ finite\ Ps \land |Ps| \leq o\ |f\ \alpha 0|$ **using** *f1 e2* **by** *blast*
qed
ultimately show $f\ \alpha' \cap P \in SCF\ (Restr\ r\ (f\ \alpha'))$ **using** *e3 e4 a8* **unfolding** $\mathcal{E}$*p-def* **by** *metis*
qed
ultimately show $((\exists P.\ Ps = \{P\}) \lor (\neg\ finite\ Ps \land |Ps| \leq o\ |f\ \alpha'|\ )) \longrightarrow (\forall P \in Ps.\ f\ \alpha' \cap P \in SCF\ (Restr\ r\ (f\ \alpha')))$ **by** *blast*
qed
then show $f \in \mathcal{N}8\ r\ Ps\ \alpha$ **unfolding** $\mathcal{N}$*8-def* **by** *blast*
qed
moreover have $\forall \alpha.\ lm\text{-}ord\ \alpha \land (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}8\ r\ Ps\ \beta) \longrightarrow f \in \mathcal{N}8\ r\ Ps\ \alpha$
proof (*intro allI impI*)
fix α::*'U rel*
assume *c1*: $lm\text{-}ord\ \alpha \land (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}8\ r\ Ps\ \beta)$
then have *c2*: $f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \land D = f\ \beta\ \}$ **using** *b4* **by** *blast*
have $\forall \alpha'$::*'U rel*. $\alpha' \leq o\ \alpha \land (\alpha' = \{\} \lor isSuccOrd\ \alpha') \longrightarrow$
$((\exists P.\ Ps = \{P\}) \lor (\neg\ finite\ Ps \land |Ps| \leq o\ |f\ \alpha'|\ )) \longrightarrow (\forall P \in Ps.\ f\ \alpha' \cap P \in SCF\ (Restr\ r\ (f\ \alpha')))$
proof (*intro allI*, *rule impI*)
fix α'::*'U rel*
assume *d1*: $\alpha' \leq o\ \alpha \land (\alpha' = \{\} \lor isSuccOrd\ \alpha')$
then have $\alpha' <o\ \alpha \lor \alpha' =o\ \alpha$ **using** *ordLeq-iff-ordLess-or-ordIso* **by** *blast*

moreover have $\alpha' <o\ \alpha \longrightarrow ((\exists P.\ Ps = \{P\}) \vee (\neg\ finite\ Ps \wedge |Ps| \leq o\ |f\ \alpha'|\ )) \longrightarrow$
$(\forall P{\in}Ps.\ f\ \alpha' \cap P \in SCF\ (Restr\ r\ (f\ \alpha')))$
proof
assume $\alpha' <o\ \alpha$
moreover then have $\alpha' \leq o\ \alpha'$ **using** *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
ultimately show $((\exists P.\ Ps = \{P\}) \vee (\neg\ finite\ Ps \wedge |Ps| \leq o\ |f\ \alpha'|\ )) \longrightarrow$
$(\forall P{\in}Ps.\ f\ \alpha' \cap P \in SCF\ (Restr\ r\ (f\ \alpha')))$
using *c1 d1* **unfolding** *$\mathcal{N}$8-def* **by** *blast*
qed
moreover have $\alpha' =o\ \alpha \longrightarrow False$
proof
assume $\alpha' =o\ \alpha$
moreover then have $\alpha' = \{\} \vee isSuccOrd\ \alpha$ **using** *d1 lem-osucc-eq* **by** *blast*
moreover have $\neg\ (\alpha = \{\} \vee isSuccOrd\ \alpha)$ **using** *c1* **unfolding** *lm-ord-def* **by** *blast*
ultimately have $\alpha' =o\ \alpha \wedge \alpha' = \{\} \wedge \alpha \neq \{\}$ **by** *blast*
then show *False* **by** (*metis iso-ozero-empty ordIso-symmetric ozero-def*)
qed
ultimately show $((\exists P.\ Ps = \{P\}) \vee (\neg\ finite\ Ps \wedge |Ps| \leq o\ |f\ \alpha'|\ )) \longrightarrow$
$(\forall P{\in}Ps.\ f\ \alpha' \cap P \in SCF\ (Restr\ r\ (f\ \alpha')))$ **by** *blast*
qed
then show $f \in \mathcal{N}8\ r\ Ps\ \alpha$ **unfolding** *$\mathcal{N}$8-def* **by** *blast*
qed
ultimately show *?thesis* **using** *lem-sclm-ordind*[*of* $\lambda\ \alpha.\ f \in \mathcal{N}8\ r\ Ps\ \alpha$] **by** *blast*
qed

lemma *lem-Shinf-N9*:
fixes r::$'U\ rel$ **and** g::$'U\ rel \Rightarrow\ 'U$
and F::$'U\ rel \Rightarrow\ 'U\ set \Rightarrow\ 'U\ set$ **and** f::$'U\ rel \Rightarrow\ 'U\ set$
assumes *a0*: $f \in \mathcal{T}\ F$
and *a1*: $\forall\ \alpha\ A.\ Well\text{-}order\ \alpha \longrightarrow A \subseteq F\ \alpha\ A$
and *a2*: $\forall\ \alpha\ A.\ Well\text{-}order\ \alpha \longrightarrow g\ \alpha \in Field\ r \longrightarrow g\ \alpha \in F\ \alpha\ A$
and *a11*: $\omega\text{-}ord \leq o\ |Field\ r| \longrightarrow Field\ r \subseteq g\ `\ \{\ \gamma::'U\ rel.\ \gamma <o\ |Field\ r|\ \}$
shows $f \in \mathcal{N}9\ r\ |Field\ r|$
proof −
have *b3*: $\forall\ \alpha 0\ \alpha::'U\ rel.\ (sc\text{-}ord\ \alpha 0\ \alpha \longrightarrow f\ \alpha = F\ \alpha 0\ (f\ \alpha 0))$ **using** *a0* **unfolding** *$\mathcal{T}$-def* **by** *blast+*
have $\forall\ a \in Field\ r.\ \omega\text{-}ord \leq o\ |Field\ r| \longrightarrow a \in f\ |Field\ r|$
proof (*intro ballI impI*)
fix *a*
assume *c1*: $a \in Field\ r$ **and** *c2*: $\omega\text{-}ord \leq o\ |Field\ r|$
then obtain $\alpha 0$::$'U\ rel$ **where** *c4*: $\alpha 0 <o\ |Field\ r| \wedge g\ \alpha 0 = a$ **using** *a11* **by** *blast*
moreover then obtain α **where** *c5*: $sc\text{-}ord\ \alpha 0\ \alpha$ **using** *lem-sucord-ex*[*of* $\alpha 0$ $|Field\ r|$] **by** *blast*

ultimately have *c6*: $\alpha \leq o$ |*Field r*| **unfolding** *sc-ord-def* **by** *blast*
have *Well-order* |*Field r*| **by** *simp*
then have $f \in \mathcal{N}1$ *r* |*Field r*| **using** *a0 a1 lem-Shinf-N1* **unfolding** *card-order-on-def* **by** *metis*
moreover have *c7*: |*Field r*| $\leq o$ |*Field r*| **by** *simp*
moreover have $f\ \alpha = F\ \alpha 0\ (f\ \alpha 0)$ **using** *c5 b3* **by** *blast*
moreover have $a \in F\ \alpha 0\ (f\ \alpha 0)$ **using** *a2 c4 c1 ordLess-Well-order-simp* **by** *blast*
ultimately show $a \in f$ |*Field r*| **using** *c6* **unfolding** $\mathcal{N}$*1-def* **by** *blast*
qed
then show *?thesis* **unfolding** $\mathcal{N}$*9-def* **by** *blast*
qed

lemma *lem-Shinf-N10*:
fixes *r*::'*U rel* **and** *F*::'*U rel* ⇒ '*U set* ⇒ '*U set* **and** *f*::'*U rel* ⇒ '*U set*
assumes *a0*: $f \in \mathcal{T}\ F$
and *a1*: ∀ α *A*. *Well-order* α ⟶ *A* ⊆ *F* α *A*
and *a5*: ∀α. *Well-order* α ⟶ $f \in \mathcal{N}5$ *r* α
and *a10*: ∀ α *A*. *Well-order* α ⟶ *A* ∈ *SF r* ⟶
((∃ *y*. (*F* α *A*) − *dncl r A* ⊆ {*y*}) ⟶ (*Field r* ⊆ *dncl r* (*F* α *A*)))
shows ∀α. *Well-order* α ⟶ $f \in \mathcal{N}10$ *r* α
proof −
have *b2*: *f* {} = {}
and *b3*: ∀ α0 α::'*U rel*. (*sc-ord* α0 α ⟶ *f* α = *F* α0 (*f* α0))
and *b4*: ∀ α. (*lm-ord* α ⟶ *f* α = ⋃ { *D*. ∃ β. β <*o* α ∧ *D* = *f* β })
and *b5*: ∀α β. α =*o* β ⟶ *f* α = *f* β **using** *a0* **unfolding** $\mathcal{T}$*-def* **by** *blast+*
have $f \in \mathcal{N}10$ *r* {} **using** *b2 lem-ord-subemp* **unfolding** $\mathcal{N}$*10-def* $\mathcal{Q}$*-def* **by** *blast*
moreover have ∀α0 α. *sc-ord* α0 α ∧ $f \in \mathcal{N}10$ *r* α0 ⟶ $f \in \mathcal{N}10$ *r* α
proof (*intro allI impI*)
fix α0 α::'*U rel*
assume *c1*: *sc-ord* α0 α ∧ $f \in \mathcal{N}10$ *r* α0
have ∀α'::'*U rel*. α' ≤*o* α ⟶
((∃ *y*. (*f* α') − *dncl r* (𝔏 *f* α') = {*y*}) ⟶ (*Field r* ⊆ *dncl r* (*f* α')))
proof (*intro allI impI*)
fix α'::'*U rel*
assume *d1*: α' ≤*o* α **and** *d2*: ∃ *y*. (*f* α') − *dncl r* (𝔏 *f* α') = {*y*}
then have α0 <*o* α' ∨ α' ≤*o* α0 **using** *c1* **unfolding** *sc-ord-def*
using *not-ordLeq-iff-ordLess ordLeq-Well-order-simp ordLess-Well-order-simp* **by** *blast*
moreover have α' ≤*o* α0 ⟶ ((∃ *y*. (*f* α') − *dncl r* (𝔏 *f* α') = {*y*}) ⟶ (*Field r* ⊆ *dncl r* (*f* α')))
using *d1 c1* **unfolding** $\mathcal{N}$*10-def* $\mathcal{Q}$*-def* **by** *blast*
moreover have α0 <*o* α' ⟶ α =*o* α' **using** *d1 c1* **unfolding** *sc-ord-def* **using** *ordIso-iff-ordLeq* **by** *blast*
moreover have α =*o* α' ⟶ (*Field r* ⊆ *dncl r* (*f* α'))
proof
assume *e1*: α =*o* α'
have *Well-order* α0 **using** *c1* **unfolding** *sc-ord-def ordLess-def* **by** *blast*

moreover then have $(f\ \alpha 0) \in SF\ r$
using *a5* **unfolding** $\mathcal{N}$*5-def* **using** *ordLeq-reflexive* **by** *blast*
moreover have $f\ \alpha = F\ \alpha 0\ (f\ \alpha 0)$ **using** *c1 b3* **by** *blast*
ultimately have *e2*: $((\exists\, y.\ \ (f\ \alpha) - dncl\ r\ (f\ \alpha 0) \subseteq \{y\}) \longrightarrow (Field\ r \subseteq dncl\ r\ (f\ \alpha)))$
using *a10* **by** *metis*
have $\mathfrak{L}\ f\ \alpha \subseteq f\ \alpha 0$
proof
fix p
assume $p \in \mathfrak{L}\ f\ \alpha$
then obtain β::$'U\ rel$ **where** $\beta <o\ \alpha \wedge p \in f\ \beta$ **unfolding** $\mathfrak{L}$*-def* **by** *blast*
moreover then have $\beta \le o\ \alpha 0 \wedge \alpha 0 \le o\ \alpha 0$ **using** *c1* **unfolding** *sc-ord-def*
using *not-ordLess-iff-ordLeq ordLess-Well-order-simp* **by** *blast*
moreover then have $f \in \mathcal{N} 1\ r\ \alpha 0$ **using** *a0 a1 lem-Shinf-N1*[*of f F*] *ordLeq-Well-order-simp* **by** *metis*
ultimately show $p \in f\ \alpha 0$ **unfolding** $\mathcal{N}$*1-def* **by** *blast*
qed
moreover have $f\ \alpha 0 \subseteq \mathfrak{L}\ f\ \alpha$ **using** *c1* **unfolding** *sc-ord-def* $\mathfrak{L}$*-def* **by** *blast*
ultimately have $\mathfrak{L}\ f\ \alpha = f\ \alpha 0$ **by** *blast*
then have $\mathfrak{L}\ f\ \alpha' = f\ \alpha 0$ **using** *e1 lem-shrel-L-eq* **by** *blast*
then show $Field\ r \subseteq dncl\ r\ (f\ \alpha')$ **using** *d2 e2 e1 b5* **by** *force*
qed
ultimately show $Field\ r \subseteq dncl\ r\ (f\ \alpha')$ **using** *d2* **by** *blast*
qed
then show $f \in \mathcal{N} 10\ r\ \alpha$ **unfolding** $\mathcal{N}$*10-def* $\mathcal{Q}$*-def* **by** *blast*
qed
moreover have $\forall \alpha.\ lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N} 10\ r\ \beta) \longrightarrow f \in \mathcal{N} 10\ r\ \alpha$
proof (*intro allI impI*)
fix α::$'U\ rel$
assume *c1*: $lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N} 10\ r\ \beta)$
then have *c2*: $f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \}$ **using** *b4* **by** *blast*
have $\forall \alpha'$::$'U\ rel.\ \alpha' \le o\ \alpha \longrightarrow$
$((\exists\, y.\ \ (f\ \alpha') - dncl\ r\ (\mathfrak{L}\ f\ \alpha') = \{y\}) \longrightarrow (Field\ r \subseteq dncl\ r\ (f\ \alpha')))$
proof (*intro allI impI*)
fix α'::$'U\ rel$
assume *d1*: $\alpha' \le o\ \alpha$ **and** *d2*: $\exists\, y.\ (f\ \alpha') - dncl\ r\ (\mathfrak{L}\ f\ \alpha') = \{y\}$
then have $\alpha' <o\ \alpha \vee \alpha' =o\ \alpha$ **using** *ordLeq-iff-ordLess-or-ordIso* **by** *blast*
moreover have $\alpha' <o\ \alpha \longrightarrow (Field\ r \subseteq dncl\ r\ (f\ \alpha'))$
proof
assume $\alpha' <o\ \alpha$
moreover then have $\alpha' \le o\ \alpha'$ **using** *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
ultimately show $Field\ r \subseteq dncl\ r\ (f\ \alpha')$ **using** *c1 d1 d2* **unfolding** $\mathcal{N}$*10-def* $\mathcal{Q}$*-def* **by** *blast*
qed
moreover have $\alpha' =o\ \alpha \longrightarrow False$
proof

assume *e1*: $\alpha' =o\ \alpha$
moreover then have *e2*: $\mathfrak{L}\ f\ \alpha' = \mathfrak{L}\ f\ \alpha$ **using** *lem-shrel-L-eq* **by** *blast*
ultimately have $\exists\, y.\ (f\ \alpha) - dncl\ r\ (\mathfrak{L}\ f\ \alpha) = \{y\}$ **using** *d2 b5* **by** *metis*
moreover have $f\ \alpha \subseteq \mathfrak{L}\ f\ \alpha$ **using** *c2* **unfolding** *𝔏-def* **by** *blast*
ultimately show *False* **unfolding** *dncl-def* **by** *blast*
qed
ultimately show $Field\ r \subseteq dncl\ r\ (f\ \alpha')$ **using** *d2* **by** *blast*
qed
then show $f \in \mathcal{N}10\ r\ \alpha$ **unfolding** *𝒩10-def 𝒬-def* **by** *blast*
qed
ultimately show *?thesis* **using** *lem-sclm-ordind*$[of\ \lambda\ \alpha.\ f \in \mathcal{N}10\ r\ \alpha]$ **by** *blast*
qed

lemma *lem-Shinf-N11*:
fixes $r::'U\ rel$ **and** $F::'U\ rel \Rightarrow 'U\ set \Rightarrow 'U\ set$ **and** $f::'U\ rel \Rightarrow 'U\ set$
assumes *a0*: $f \in \mathcal{T}\ F$
and *a1*: $\forall\ \alpha\ A.\ Well\text{-}order\ \alpha \longrightarrow A \subseteq F\ \alpha\ A$
and *a5*: $\forall\alpha.\ Well\text{-}order\ \alpha \longrightarrow f \in \mathcal{N}5\ r\ \alpha$
and *a10*: $\forall\ \alpha\ A.\ Well\text{-}order\ \alpha \longrightarrow A \in SF\ r \longrightarrow$
$((\exists\, y.\ \ (F\ \alpha\ A) - dncl\ r\ A \subseteq \{y\}) \longrightarrow (Field\ r \subseteq dncl\ r\ (F\ \alpha\ A)))$
shows $\forall\alpha.\ Well\text{-}order\ \alpha \longrightarrow f \in \mathcal{N}11\ r\ \alpha$
proof −
have *b2*: $f\ \{\} = \{\}$
and *b3*: $\forall\ \alpha0\ \alpha::'U\ rel.\ (sc\text{-}ord\ \alpha0\ \alpha \longrightarrow f\ \alpha = F\ \alpha0\ (f\ \alpha0))$
and *b4*: $\forall\ \alpha.\ (lm\text{-}ord\ \alpha \longrightarrow f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \})$
and *b5*: $\forall\alpha\ \beta.\ \alpha =o\ \beta \longrightarrow f\ \alpha = f\ \beta$ **using** *a0* **unfolding** *𝒯-def* **by** *blast+*
have $\neg$ *isSuccOrd* $(\{\}::'U\ rel)$
using *wo-rel-def wo-rel.isSuccOrd-def* **unfolding** *Field-def* **by** *force*
then have $f \in \mathcal{N}11\ r\ \{\}$ **using** *lem-ord-subemp* **unfolding** *𝒩11-def* **by** *blast*
moreover have $\forall\alpha0\ \alpha.\ sc\text{-}ord\ \alpha0\ \alpha \wedge f \in \mathcal{N}11\ r\ \alpha0 \longrightarrow f \in \mathcal{N}11\ r\ \alpha$
proof (*intro allI impI*)
fix $\alpha0\ \alpha::'U\ rel$
assume *c1*: $sc\text{-}ord\ \alpha0\ \alpha \wedge f \in \mathcal{N}11\ r\ \alpha0$
have $\forall\alpha'::'U\ rel.\ \alpha' \le o\ \alpha \wedge (isSuccOrd\ \alpha') \longrightarrow$
$((\ (f\ \alpha') - dncl\ r\ (\mathfrak{L}\ f\ \alpha') = \{\}) \longrightarrow (Field\ r \subseteq dncl\ r\ (f\ \alpha')))$
proof (*intro allI impI*)
fix $\alpha'::'U\ rel$
assume *d1*: $\alpha' \le o\ \alpha \wedge (isSuccOrd\ \alpha')$
and *d2*: $(f\ \alpha') - dncl\ r\ (\mathfrak{L}\ f\ \alpha') = \{\}$
then have $\alpha0\ <o\ \alpha' \vee \alpha' \le o\ \alpha0$ **using** *c1* **unfolding** *sc-ord-def*
using *not-ordLeq-iff-ordLess ordLeq-Well-order-simp ordLess-Well-order-simp*
by *blast*
moreover have $\alpha' \le o\ \alpha0 \longrightarrow (((f\ \alpha') - dncl\ r\ (\mathfrak{L}\ f\ \alpha') = \{\}) \longrightarrow (Field\ r$
$\subseteq dncl\ r\ (f\ \alpha')))$
using *d1 c1* **unfolding** *𝒩11-def 𝒬-def* **by** *blast*
moreover have $\alpha0\ <o\ \alpha' \longrightarrow \alpha =o\ \alpha'$ **using** *d1 c1* **unfolding** *sc-ord-def*
using *ordIso-iff-ordLeq* **by** *blast*
moreover have $\alpha =o\ \alpha' \longrightarrow (Field\ r \subseteq dncl\ r\ (f\ \alpha'))$
proof

assume *e1*: $\alpha =o\ \alpha'$
have *Well-order* $\alpha 0$ **using** *c1* **unfolding** *sc-ord-def ordLess-def* **by** *blast*
moreover then have $(f\ \alpha 0) \in SF\ r$
using *a5* **unfolding** $\mathcal{N}$*5-def* **using** *ordLeq-reflexive* **by** *blast*
moreover have $f\ \alpha = F\ \alpha 0\ (f\ \alpha 0)$ **using** *c1 b3* **by** *blast*
ultimately have *e2*: $(((f\ \alpha) - dncl\ r\ (f\ \alpha 0) = \{\}) \longrightarrow (Field\ r \subseteq dncl\ r\ (f\ \alpha)))$
using *a10* **by** *fastforce*
have $\mathfrak{L}\ f\ \alpha \subseteq f\ \alpha 0$
proof
fix *p*
assume $p \in \mathfrak{L}\ f\ \alpha$
then obtain β::*'U rel* **where** $\beta <o\ \alpha \wedge p \in f\ \beta$ **unfolding** $\mathfrak{L}$*-def* **by** *blast*
moreover then have $\beta \leq o\ \alpha 0 \wedge \alpha 0 \leq o\ \alpha 0$ **using** *c1* **unfolding** *sc-ord-def*
using *not-ordLess-iff-ordLeq ordLess-Well-order-simp* **by** *blast*
moreover then have $f \in \mathcal{N}1\ r\ \alpha 0$ **using** *a0 a1 lem-Shinf-N1*[*of f F*] *ordLeq-Well-order-simp* **by** *metis*
ultimately show $p \in f\ \alpha 0$ **unfolding** $\mathcal{N}$*1-def* **by** *blast*
qed
moreover have $f\ \alpha 0 \subseteq \mathfrak{L}\ f\ \alpha$ **using** *c1* **unfolding** *sc-ord-def* $\mathfrak{L}$*-def* **by** *blast*
ultimately have $\mathfrak{L}\ f\ \alpha = f\ \alpha 0$ **by** *blast*
then have $\mathfrak{L}\ f\ \alpha' = f\ \alpha 0$ **using** *e1 lem-shrel-L-eq* **by** *blast*
then show $Field\ r \subseteq dncl\ r\ (f\ \alpha')$ **using** *d2 e2 e1 b5* **by** *force*
qed
ultimately show $Field\ r \subseteq dncl\ r\ (f\ \alpha')$ **using** *d2* **by** *blast*
qed
then show $f \in \mathcal{N}11\ r\ \alpha$ **unfolding** $\mathcal{N}$*11-def* $\mathcal{Q}$*-def* **by** *blast*
qed
moreover have $\forall \alpha.\ lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}11\ r\ \beta) \longrightarrow f \in \mathcal{N}11\ r\ \alpha$
proof (*intro allI impI*)
fix α::*'U rel*
assume *c1*: $lm\text{-}ord\ \alpha \wedge (\forall \beta.\ \beta <o\ \alpha \longrightarrow f \in \mathcal{N}11\ r\ \beta)$
then have *c2*: $f\ \alpha = \bigcup\ \{\ D.\ \exists\ \beta.\ \beta <o\ \alpha \wedge D = f\ \beta\ \}$ **using** *b4* **by** *blast*
have $\forall \alpha'$::*'U rel*. $\alpha' \leq o\ \alpha \wedge (isSuccOrd\ \alpha') \longrightarrow$
$(((f\ \alpha') - dncl\ r\ (\mathfrak{L}\ f\ \alpha') = \{\}) \longrightarrow (Field\ r \subseteq dncl\ r\ (f\ \alpha')))$
proof (*intro allI impI*)
fix α'::*'U rel*
assume *d1*: $\alpha' \leq o\ \alpha \wedge (isSuccOrd\ \alpha')$
and *d2*: $(f\ \alpha') - dncl\ r\ (\mathfrak{L}\ f\ \alpha') = \{\}$
then have $\alpha' <o\ \alpha \vee \alpha' =o\ \alpha$ **using** *ordLeq-iff-ordLess-or-ordIso* **by** *blast*
moreover have $\alpha' <o\ \alpha \longrightarrow (Field\ r \subseteq dncl\ r\ (f\ \alpha'))$
proof
assume $\alpha' <o\ \alpha$
moreover then have $\alpha' \leq o\ \alpha'$ **using** *ordLess-Well-order-simp ordLeq-reflexive* **by** *blast*
ultimately show $Field\ r \subseteq dncl\ r\ (f\ \alpha')$ **using** *c1 d1 d2* **unfolding** $\mathcal{N}$*11-def* $\mathcal{Q}$*-def* **by** *blast* **qed**

moreover have $\alpha' =o\ \alpha \longrightarrow$ *False*
proof
assume $\alpha' =o\ \alpha$
moreover then have $\alpha' = \{\} \vee$ *isSuccOrd* α **using** *d1 lem-osucc-eq* **by** *blast*
moreover have $\neg$ ($\alpha = \{\} \vee$ *isSuccOrd* α) **using** *c1* **unfolding** *lm-ord-def* **by** *blast*
ultimately have $\alpha' =o\ \alpha \wedge \alpha' = \{\} \wedge \alpha \neq \{\}$ **by** *blast*
then show *False* **by** (*metis iso-ozero-empty ordIso-symmetric ozero-def*)
qed
ultimately show *Field r* $\subseteq$ *dncl r* (*f* α') **using** *d2* **by** *blast*
qed
then show $f \in \mathcal{N}11\ r\ \alpha$ **unfolding** $\mathcal{N}$*11-def* $\mathcal{Q}$*-def* **by** *blast*
qed
ultimately show *?thesis* **using** *lem-sclm-ordind*[*of* $\lambda\ \alpha.\ f \in \mathcal{N}11\ r\ \alpha$] **by** *blast*
qed

lemma *lem-Shinf-N12*:
fixes *r*::'*U rel* **and** *g*::'*U rel* $\Rightarrow$ '*U*
and *F*::'*U rel* $\Rightarrow$ '*U set* $\Rightarrow$ '*U set* **and** *f*::'*U rel* $\Rightarrow$ '*U set*
assumes *a0*: $f \in \mathcal{T}\ F$
and *a1*: $\forall \alpha.$ *Well-order* $\alpha \longrightarrow f \in \mathcal{N}1\ r\ \alpha$
and *a2*: $\forall\ \alpha\ A.$ *Well-order* $\alpha \longrightarrow g\ \alpha \in$ *Field r* $\longrightarrow g\ \alpha \in F\ \alpha\ A$
and *a11*: ω*-ord* $\leq o$ |*Field r*| $\longrightarrow$ *Field r* = *g* ‘ { γ::'*U rel*. $\gamma <o$ |*Field r*| }
and *a2'*: $\forall \alpha$::'*U rel*. ω*-ord* $\leq o\ \alpha \wedge \alpha \leq o$ |*Field r*| $\longrightarrow \omega$*-ord* $\leq o$ |*g* ‘ {γ. $\gamma <o\ \alpha$}|
shows $f \in \mathcal{N}12\ r$ |*Field r*|
proof –
have *b1*: $\forall \alpha.\ \omega$*-ord* $=o\ \alpha \wedge \alpha \leq o$ |*Field r*| $\longrightarrow \omega$*-ord* $\leq o$ |$\mathfrak{L}\ f\ \alpha$|
proof (*intro allI impI*)
fix α::'*U rel*
assume *c1*: ω*-ord* $=o\ \alpha \wedge \alpha \leq o$ |*Field r*|
then have *c2*: ω*-ord* $\leq o$ |*g*‘{γ. $\gamma <o\ \alpha$}| **using** *a2' ordIso-imp-ordLeq* **by** *blast*
have *g*‘{γ. $\gamma <o\ \alpha$} $\subseteq$ *g*‘{γ. $\gamma <o$ |*Field r*|} **using** *c1 ordLess-ordLeq-trans* **by** *force*
then have *g*‘{γ. $\gamma <o\ \alpha$} $\subseteq$ *Field r*
using *c1 a11 ordLeq-transitive ordIso-imp-ordLeq*[*of* ω*-ord*] **by** *metis*
have *g*‘{γ. $\gamma <o\ \alpha$} $\subseteq \mathfrak{L}\ f\ \alpha$
proof
fix *a*
assume $a \in$ *g*‘{γ. $\gamma <o\ \alpha$}
then obtain γ **where** *d1*: $a = g\ \gamma \wedge \gamma <o\ \alpha$ **by** *blast*
obtain γ' **where** *d2*: *sc-ord* $\gamma\ \gamma'$ **using** *d1 lem-sucord-ex* **by** *blast*
then have $f\ \gamma' = F\ \gamma\ (f\ \gamma)$ **using** *a0* **unfolding** $\mathcal{T}$*-def* **by** *blast*
moreover have *Well-order* γ **using** *d2* **unfolding** *sc-ord-def* **using** *ord-Less-def* **by** *blast*
moreover have $g\ \gamma \in$ *Field r* **using** *d1 c1 a11 ordIso-ordLeq-trans ord-Less-ordLeq-trans* **by** *blast*
ultimately have $a \in f\ \gamma'$ **using** *d1 a2* **by** *blast*

moreover have $\gamma' <o\ \alpha$
proof −
have *isLimOrd ω-ord* **by** (*simp add*: *Field-natLeq card-order-infinite-isLimOrd natLeq-card-order*)
then have ¬ *isSuccOrd* α
using *c1 lem-osucc-eq ordIso-symmetric*
using *natLeq-Well-order wo-rel.isLimOrd-def wo-rel-def* **by** *blast*
then obtain β::*'U rel* **where** $\gamma <o\ \beta \wedge \neg (\alpha \leq o\ \beta)$ **using** *d1 lem-ordint-sucord* **by** *blast*
then have $\gamma <o\ \beta \wedge \beta <o\ \alpha$ **using** *d1*
by (*metis ordIso-imp-ordLeq ordLess-Well-order-simp ordLess-imp-ordLeq ordLess-or-ordIso*)
then show $\gamma' <o\ \alpha$ **using** *d2* **unfolding** *sc-ord-def* **using** *ordLeq-ordLess-trans* **by** *blast*
qed
ultimately show $a \in \mathfrak{L}\ f\ \alpha$ **unfolding** $\mathfrak{L}$*-def* **by** *blast*
qed
then have $|g`\{\gamma.\ \gamma <o\ \alpha\}| \leq o\ |\mathfrak{L}\ f\ \alpha|$ **by** *simp*
then show *ω-ord* $\leq o\ |\mathfrak{L}\ f\ \alpha|$ **using** *c2 ordLeq-transitive* **by** *blast*
qed
have $\forall \alpha.$ *ω-ord* $\leq o\ \alpha \wedge \alpha \leq o\ |Field\ r| \longrightarrow$ *ω-ord* $\leq o\ |\mathfrak{L}\ f\ \alpha|$
proof (*intro allI impI*)
fix α::*'U rel*
assume *ω-ord* $\leq o\ \alpha \wedge \alpha \leq o\ |Field\ r|$
moreover then obtain $\alpha 0$::*'U rel* **where** *d1*: *ω-ord* $=o\ \alpha 0 \wedge \alpha 0 \leq o\ \alpha$
using *internalize-ordLeq*[*of ω-ord α*] **by** *blast*
ultimately have *ω-ord* $=o\ \alpha 0 \wedge \alpha 0 \leq o\ |Field\ r|$ **using** *ordLeq-transitive* **by** *blast*
then have *ω-ord* $\leq o\ |\mathfrak{L}\ f\ \alpha 0|$ **using** *b1* **by** *blast*
moreover have $\mathfrak{L}\ f\ \alpha 0 \subseteq \mathfrak{L}\ f\ \alpha$ **using** *d1* **unfolding** $\mathfrak{L}$*-def* **using** *ordLess-ordLeq-trans* **by** *blast*
moreover then have $|\mathfrak{L}\ f\ \alpha 0| \leq o\ |\mathfrak{L}\ f\ \alpha|$ **by** *simp*
ultimately show *ω-ord* $\leq o\ |\mathfrak{L}\ f\ \alpha|$ **using** *ordLeq-transitive* **by** *blast*
qed
then show *?thesis* **unfolding** $\mathcal{N}$*12-def* **by** *blast*
qed

lemma *lem-Shinf-E-ne*:
fixes *r*::*'U rel* **and** *a0*::*'U* **and** *A*::*'U set* **and** *Ps*::*'U set set*
assumes *a2*: *CCR r* **and** *a3*: $Ps \subseteq SCF\ r$
shows $\mathcal{E}\ r\ a0\ A\ Ps \neq \{\}$
proof (*cases* $A \in SF\ r$)
assume *b0*: $A \in SF\ r$
show $\mathcal{E}\ r\ a0\ A\ Ps \neq \{\}$
proof (*cases finite A*)
assume *b1*: *finite A*
then obtain A' **where** $(a0 \in Field\ r \longrightarrow a0 \in A')$ **and** *b2*: $A \subseteq A'$ **and** *b3*: *CCR* (*Restr r* A') ∧ *finite* A'
and $(\forall a{\in}A.\ r``\{a\}{\subseteq}w\text{-}dncl\ r\ A \vee r``\{a\}{\cap}(A'{-}w\text{-}dncl\ r\ A) \neq \{\})$

and $A' \in SF\ r$ **and** *b4*: $(\exists y.\ A' - dncl\ r\ A \subseteq \{y\}) \longrightarrow Field\ r \subseteq A' \cup dncl\ r\ A$
and *b5*: $(\exists\ P.\ Ps = \{P\}) \longrightarrow (\forall\ P \in Ps.\ (A' \cap P \in SCF\ (Restr\ r\ A')))$
using *b0 a2 a3*
lem-Ccext-finsubccr-pext5-scf3[*of r A Ps a0 w-dncl r A dncl r A*]
by *metis*
moreover have $|A'| <o\ \omega\text{-}ord$ **using** *b3 finite-iff-ordLess-natLeq* **by** *blast*
moreover have $\neg\ (\ \omega\text{-}ord \leq o\ |A|\ )$ **using** *b1 infinite-iff-natLeq-ordLeq* **by** *blast*
moreover have $(\exists y.\ A' - dncl\ r\ A \subseteq \{y\}) \longrightarrow Field\ r \subseteq dncl\ r\ A'$ **using** *b2 b4* **unfolding** *dncl-def* **by** *blast*
moreover have $(\exists\ P.\ Ps = \{P\}) \vee ((\neg\ finite\ Ps) \wedge |Ps| \leq o\ |A|\ ) \longrightarrow (\exists\ P.\ Ps = \{P\})$
using *b1 card-of-ordLeq-finite* **by** *blast*
ultimately have $A' \in \mathcal{E}\ r\ a0\ A\ Ps$ **unfolding** *$\mathcal{E}$-def $\mathcal{E}$p-def* **by** *fast*
then show *?thesis* **by** *blast*
next
assume *b1*: $\neg\ finite\ A$
then obtain A' **where** *b2*: $(a0 \in Field\ r \longrightarrow a0 \in A')$ **and** *b3*: $A \subseteq A'$ **and** *b4*: $CCR\ (Restr\ r\ A')$
and *b5*: $|A'| =o\ |A|$ **and** *b6*: $(\forall\, a{\in}A.\ r``\{a\}{\subseteq}w\text{-}dncl\ r\ A \vee r``\{a\}{\cap}(A'{-}w\text{-}dncl\ r\ A) \neq \{\})$
and *b7*: $A' \in SF\ r$ **and** *b8*: $(\exists y.\ A' - dncl\ r\ A \subseteq \{y\}) \longrightarrow Field\ r \subseteq A' \cup dncl\ r\ A$
and *b9*: $(\ |Ps| \leq o\ |A| \longrightarrow (\forall\ P \in Ps.\ (A' \cap P) \in SCF\ (Restr\ r\ A'))\ )$
and *b10*: $escl\ r\ A\ A' \subseteq A'$ **and** *b11*: $clterm\ (Restr\ r\ A')\ r$
using *b0 a2 a3*
lem-Ccext-infsubccr-pext5-scf3[*of r A Ps a0 w-dncl r A dncl r A*] **by** *metis*
then have $(\omega\text{-}ord \leq o\ |A| \longrightarrow |A'| \leq o\ |A|\ )$ **using** *ordIso-iff-ordLeq* **by** *blast*
moreover have $(\ |A| <o\ \omega\text{-}ord \longrightarrow |A'| <o\ \omega\text{-}ord)$ **using** *b1 finite-iff-ordLess-natLeq* **by** *blast*
moreover have $(\exists y.\ A' - dncl\ r\ A \subseteq \{y\}) \longrightarrow (Field\ r \subseteq dncl\ r\ A')$ **using** *b3 b8* **unfolding** *dncl-def* **by** *blast*
moreover have $(\exists\ P.\ Ps = \{P\}) \vee ((\neg\ finite\ Ps) \wedge |Ps| \leq o\ |A|\ ) \longrightarrow |Ps| \leq o\ |A|$
using *b1* **by** (*metis card-of-singl-ordLeq finite.simps*)
ultimately have $A' \in \mathcal{E}\ r\ a0\ A\ Ps$ **unfolding** *$\mathcal{E}$-def $\mathcal{E}$p-def*
using *b2 b3 b4 b5 b6 b7 b8 b9 b10 b11* **by** *fast*
then show *?thesis* **by** *blast*
qed
next
assume $A \notin SF\ r$
moreover obtain A' **where** *b1*: $A' = A \cup \{a0\}$ **by** *blast*
moreover then have $|A| <o\ \omega\text{-}ord \longrightarrow |A'| <o\ \omega\text{-}ord$ **using** *finite-iff-ordLess-natLeq* **by** *blast*
moreover have $\omega\text{-}ord \leq o\ |A| \longrightarrow |A'| \leq o\ |A|$
proof
assume $\omega\text{-}ord \leq o\ |A|$

then have ¬ *finite A* **using** *finite-iff-ordLess-natLeq not-ordLeq-ordLess* **by** *blast*
then have $|A'| =o |A|$ **unfolding** *b1* **using** *infinite-card-of-insert* **by** *simp*
then show $|A'| \leq o\ |A|$ **using** *ordIso-imp-ordLeq* **by** *blast*
qed
ultimately have $A' \in \mathcal{E}\ r\ a0\ A\ Ps$ **unfolding** $\mathcal{E}$*-def* **by** *blast*
then show $\mathcal{E}\ r\ a0\ A\ Ps \neq \{\}$ **by** *blast*
qed

lemma *lem-oseq-fin-inj*:
fixes $g{::}'U\ rel \Rightarrow\ 'a$ **and** $I{::}'U\ rel \Rightarrow\ 'U\ rel\ set$ **and** $A{::}'a\ set$
assumes *a1*: $I = (\lambda\ \alpha'.\ \{\ \alpha{::}'U\ rel.\ \alpha <o\ \alpha'\ \})$
and *a2*: $\omega\text{-}ord \leq o\ |A|$
and *a3*: $\forall\ \alpha\ \beta.\ \alpha =o\ \beta \longrightarrow g\ \alpha = g\ \beta$
shows $\exists\ h.\ (\forall\ \alpha'.\ g`(I\ \alpha') \subseteq h`(I\ \alpha') \wedge h`(I\ \alpha') \subseteq g`(I\ \alpha') \cup A)$
$\wedge (\forall\ \alpha'.\ \omega\text{-}ord \leq o\ \alpha' \longrightarrow \omega\text{-}ord \leq o\ |h`(I\ \alpha')|\ )$
$\wedge (\forall\ \alpha\ \beta.\ \alpha =o\ \beta \longrightarrow h\ \alpha = h\ \beta)$
proof(*cases* $\exists\ \alpha{::}'U\ rel.\ \omega\text{-}ord \leq o\ \alpha$)
assume $\exists\ \alpha{::}'U\ rel.\ \omega\text{-}ord \leq o\ \alpha$
then obtain $\alpha m{::}'U\ rel$ **where** *b1*: $\omega\text{-}ord =o\ \alpha m$ **by** (*metis internalize-ordLeq*)
obtain $f{::}nat \Rightarrow\ 'U\ rel$ **where** *b2*: $f = (\lambda\ n.\ SOME\ \alpha.\ \alpha =o\ (natLeq\text{-}on\ n))$ **by** *blast*
have $|UNIV{::}nat\ set| \leq o\ |A|$ **using** *a2* **using** *card-of-nat ordIso-imp-ordLeq ordLeq-transitive* **by** *blast*
then obtain $xi{::}nat \Rightarrow\ 'a$ **where** *b3*: $inj\ xi \wedge xi\ `\ UNIV \subseteq A$ **by** (*meson card-of-ordLeq*)
obtain *yi* **where** *b4*: $yi = (\lambda\ n.\ if\ (\exists\ i{<}n.\ g\ (f\ n) = g\ (f\ i))\ then\ (xi\ n)\ else\ (g\ (f\ n)))$ **by** *blast*
obtain *h* **where** *b5*: $h = (\lambda\ \alpha.\ if\ (\exists\ n.\ \alpha =o\ f\ n)\ then\ (yi\ (SOME\ n.\ (\alpha =o\ f\ n)))\ else\ (g\ \alpha))$ **by** *blast*
have *b6*: $\bigwedge n{::}nat.\ f\ n =o\ (natLeq\text{-}on\ n)$
proof −
fix *n*
have $natLeq\text{-}on\ n <o\ \alpha m$ **using** *b1 natLeq-on-ordLess-natLeq ordLess-ordIso-trans* **by** *blast*
then obtain $\alpha{::}'U\ rel$ **where** $\alpha =o\ (natLeq\text{-}on\ n)$
using *internalize-ordLess ordIso-symmetric* **by** *fastforce*
then show $f\ n =o\ natLeq\text{-}on\ n$ **using** *b2 someI-ex*[*of* $\lambda\alpha{::}'U\ rel.\ \alpha =o\ (natLeq\text{-}on\ n)$] **by** *blast*
qed
then have *b7*: $\bigwedge n\ m.\ n \leq m \Longrightarrow f\ n \leq o\ f\ m$
by (*metis* (*no-types*, *lifting*) *natLeq-on-ordLeq-less-eq ordIso-imp-ordLeq ordIso-symmetric ordLeq-transitive*)
have *b8*: $\bigwedge n\ m.\ f\ n =o\ f\ m \Longrightarrow n = m$
proof −
fix *n m*
assume $f\ n =o\ f\ m$
moreover then have $natLeq\text{-}on\ n =o\ f\ m$ **using** *b6 ordIso-transitive ordIso-symmetric* **by** *blast*

ultimately have *natLeq-on n =o natLeq-on m* **using** *b6 ordIso-transitive* **by** *blast*
then show $n = m$ **using** *natLeq-on-injective-ordIso* **by** *blast*
qed
have *b9*: $\bigwedge \alpha\ n.\ \alpha =o\ f\ n \Longrightarrow h\ \alpha = yi\ n$
proof −
fix α::$'U\ rel$ **and** n::nat
assume $\alpha =o\ f\ n$
moreover obtain m **where** $m = (SOME\ n.\ (\alpha =o\ f\ n))$ **by** *blast*
ultimately have $h\ \alpha = yi\ m \wedge \alpha =o\ f\ m \wedge \alpha =o\ f\ n$ **using** *b5 someI-ex*[*of* λ $n.\ \alpha =o\ f\ n$] **by** *fastforce*
moreover then have $m = n$ **using** *b8 ordIso-transitive ordIso-symmetric* **by** *blast*
ultimately show $h\ \alpha = yi\ n$ **by** *blast*
qed
have *b10*: $\bigwedge n.\ yi`\{k.\ k \leq n\} \subseteq g`(f`(\{k.\ k \leq n\})) \cup A$
proof −
fix *n0*
show $yi`\{k.\ k \leq n0\} \subseteq g`(f`(\{k.\ k \leq n0\})) \cup A$
proof (*induct n0*)
show $yi`\{k.\ k \leq 0\} \subseteq g`(f`\{k.\ k \leq 0\}) \cup A$ **using** *b4* **by** *simp*
next
fix n
assume *d1*: $yi`\{k.\ k \leq n\} \subseteq g`(f`(\{k.\ k \leq n\})) \cup A$
show $yi`\{k.\ k \leq Suc\ n\} \subseteq g`(f`(\{k.\ k \leq (Suc\ n)\})) \cup A$
proof (*cases* $\exists\ i<Suc\ n.\ g\ (f\ (Suc\ n)) = g\ (f\ i)$)
assume $\exists\ i<Suc\ n.\ g\ (f\ (Suc\ n)) = g\ (f\ i)$
then obtain i **where** $i<Suc\ n \wedge g\ (f\ (Suc\ n)) = g\ (f\ i)$ **by** *blast*
then have $i \leq n \wedge yi\ (Suc\ n) = xi\ (Suc\ n)$ **using** *b4* **by** *force*
then have $yi\ (Suc\ n) \in g`(f`(\{k.\ k \leq Suc\ n\})) \cup A$ **using** *b3* **by** *blast*
moreover have $yi`\{k.\ k \leq n\} \subseteq g`(f`(\{k.\ k \leq Suc\ n\})) \cup A$ **using** *d1* **by** *fastforce*
moreover have $\bigwedge k.\ k \leq Suc\ n \longleftrightarrow (k \leq n \vee k = Suc\ n)$ **by** *linarith*
moreover then have $yi`\{k.\ k \leq Suc\ n\} = yi`\{k.\ k \leq n\} \cup \{yi\ (Suc\ n)\}$ **by** *fastforce*
ultimately show *?thesis* **by** *blast*
next
assume $\neg\ (\exists\ i<Suc\ n.\ g\ (f\ (Suc\ n)) = g\ (f\ i))$
then have $yi\ (Suc\ n) = g\ (f\ (Suc\ n))$ **using** *b4* **by** *force*
then have $yi\ (Suc\ n) \in g`(f`(\{k.\ k \leq Suc\ n\})) \cup A$ **by** *blast*
moreover have $yi`\{k.\ k \leq n\} \subseteq g`(f`(\{k.\ k \leq Suc\ n\})) \cup A$ **using** *d1* **by** *fastforce*
moreover have $\bigwedge k.\ k \leq Suc\ n \longleftrightarrow (k \leq n \vee k = Suc\ n)$ **by** *linarith*
moreover then have $yi`\{k.\ k \leq Suc\ n\} = yi`\{k.\ k \leq n\} \cup \{yi\ (Suc\ n)\}$ **by** *fastforce*
ultimately show *?thesis* **by** *blast*
qed
qed
qed

have ∀ α'. $g`(I\ \alpha') \subseteq h`(I\ \alpha') \wedge h`(I\ \alpha') \subseteq g`(I\ \alpha') \cup A$
proof
fix α'::$'U$ *rel*
have $g`(I\ \alpha') \subseteq h`(I\ \alpha')$
proof
fix a
assume $a \in g`(I\ \alpha')$
then obtain β **where** *d1*: $\beta <o\ \alpha' \wedge a = g\ \beta$ **using** *a1* **by** *blast*
show $a \in h`(I\ \alpha')$
proof (*cases* ∃ n. $\beta =o\ f\ n$)
assume ∃ n. $\beta =o\ f\ n$
then obtain n **where** *e1*: $\beta =o\ f\ n$ **by** *blast*
then have *e2*: $a = g\ (f\ n) \wedge h\ \beta = yi\ n$ **using** *d1 b9 a3* **by** *blast*
obtain P **where** *e3*: $P = (\lambda\ i.\ i \le n \wedge g\ (f\ n) = g\ (f\ i))$ **by** *blast*
obtain k **where** $k = (LEAST\ i.\ P\ i)$ **by** *blast*
moreover have $P\ n$ **using** *e3* **by** *blast*
ultimately have $P\ k \wedge (\forall\ i.\ P\ i \longrightarrow k \le i)$ **using** *LeastI Least-le* **by** *metis*
then have $k \le n \wedge g\ (f\ n) = g\ (f\ k) \wedge \neg\ (\exists\ i<k.\ g\ (f\ k) = g\ (f\ i))$
using *e3* **by** (*metis leD less-le-trans less-or-eq-imp-le*)
then have $a = yi\ k \wedge f\ k \le o\ f\ n$ **using** *e2 b4 b7* **by** *fastforce*
moreover then have $f\ k <o\ \alpha'$
using *e1 d1* **by** (*metis ordIso-symmetric ordLeq-ordIso-trans ordLeq-ordLess-trans*)
ultimately have $f\ k \in I\ \alpha' \wedge h\ (f\ k) = a$ **using** *a1 b7 b9 ordIso-iff-ordLeq*
by *blast*
then show *?thesis* **by** *blast*
next
assume ¬ (∃ n. $\beta =o\ f\ n$)
then have $h\ \beta = g\ \beta$ **using** *b5* **by** *simp*
then show *?thesis* **using** *d1 a1* **by** *force*
qed
qed
moreover have $h`(I\ \alpha') \subseteq g`(I\ \alpha') \cup A$
proof
fix a
assume $a \in h`(I\ \alpha')$
then obtain β **where** *d1*: $\beta <o\ \alpha' \wedge a = h\ \beta$ **using** *a1* **by** *blast*
show $a \in g`(I\ \alpha') \cup A$
proof (*cases* ∃ n. $\beta =o\ f\ n$)
assume ∃ n. $\beta =o\ f\ n$
then obtain n **where** *e1*: $\beta =o\ f\ n$ **by** *blast*
then have $a = yi\ n$ **using** *d1 b9* **by** *blast*
then have $a \in g`(f`(\{k.\ k \le n\})) \cup A$ **using** *b10* **by** *blast*
moreover have ∀ k. $k \le n \longrightarrow f\ k \in I\ \alpha'$
proof (*intro allI impI*)
fix k
assume $k \le n$
then have $f\ k \le o\ f\ n$ **using** *b7* **by** *blast*
then show $f\ k \in I\ \alpha'$ **using** *e1 a1 d1*
using *ordIso-symmetric ordLeq-ordIso-trans ordLeq-ordLess-trans* **by**

fastforce
qed
ultimately show *?thesis* **by** *blast*
next
assume ¬ (∃ *n*. *β* =*o* *f n*)
then show *?thesis* **using** *d1 a1 b5* **by** *force*
qed
qed
ultimately show *g*‘(*I α*′) ⊆ *h*‘(*I α*′) ∧ *h*‘(*I α*′) ⊆ *g*‘(*I α*′) ∪ *A* **by** *blast*
qed
moreover have ∀ *α*′. *ω-ord* ≤*o* *α*′ ⟶ *ω-ord* ≤*o* |*h*‘(*I α*′)|
proof (*intro allI impI*)
fix *α*′::′*U rel*
assume *ω-ord* ≤*o* *α*′
then have *I αm* ⊆ *I α*′
using *a1 b1* **by** (*smt mem-Collect-eq not-ordLess-ordIso ordIso-symmetric ordLeq-iff-ordLess-or-ordIso ordLeq-ordLess-trans ordLeq-transitive subsetI*)
moreover have *f*‘*UNIV* ⊆ *I αm* **using** *b1 a1*
using *b6 natLeq-on-ordLess-natLeq ordIso-ordLess-trans ordLess-ordIso-trans* **by** *fastforce*
ultimately have *h*‘(*f*‘*UNIV*) ⊆ *h*‘(*I α*′) **by** *blast*
then have |*h*‘(*f*‘*UNIV*)| ≤*o* |*h*‘(*I α*′)| **by** *simp*
moreover have *ω-ord* ≤*o* |*h*‘(*f*‘*UNIV*)|
proof −
have ∀ *n*. *h* (*f n*) = *yi n* **using** *b7 b9 ordIso-iff-ordLeq* **by** *blast*
then have *yi*‘*UNIV* ⊆ *h*‘(*f*‘*UNIV*) **by** (*smt imageE image-eqI subset-eq*)
then have |*yi*‘*UNIV*| ≤*o* |*h*‘(*f*‘*UNIV*)| **by** *simp*
moreover have *ω-ord* ≤*o* |*yi*‘*UNIV*|
proof (*cases finite* (*g*‘(*f*‘*UNIV*)))
assume *e1*: *finite*(*g*‘(*f*‘*UNIV*))
obtain *J* **where** *e3*: *J* = {*n*. ∃ *i*<*n*. *g* (*f n*) = *g* (*f i*)} **by** *blast*
have (∀ *m*. ∃ *n*>*m*. *n* ∉ *J*) ⟶ *False*
proof
assume *f1*: ∀ *m*. ∃ *n*>*m*. *n* ∉ *J*
obtain *w* **where** *f2*: *w* = (*λ m*. *SOME n*. *n*>*m* ∧ *n* ∉ *J*) **by** *blast*
have *f3*: ∀ *m*. *w m* > *m* ∧ *w m* ∉ *J*
proof
fix *m*
show *w m* > *m* ∧ *w m* ∉ *J* **using** *f1 f2 someI-ex*[*of λ n*. *n*>*m* ∧ *n* ∉ *J*] **by** *metis*
qed
obtain *p* **where** *f4*: *p* = (*λ k*::*nat*. (*w*^^*k*) *0*) **by** *blast*
have *f5*: ∀ *k*. *k* ≠ *0* ⟶ *p k* ∉ *J*
proof
fix *k*
show *k* ≠ *0* ⟶ *p k* ∉ *J*
proof (*induct k*)
show *0* ≠ *0* ⟶ *p 0* ∉ *J* **by** *blast*
next

fix *k*
assume $k \neq 0 \longrightarrow p\ k \notin J$
show $Suc\ k \neq 0 \longrightarrow p\ (Suc\ k) \notin J$ **using** *f3 f4* **by** *simp*
qed
qed
have $\forall\ j.\ \forall\ i<j.\ p\ i < p\ j$
proof
fix *j*
show $\forall i<j.\ p\ i < p\ j$
proof (*induct j*)
show $\forall i<0.\ p\ i < p\ 0$ **by** *blast*
next
fix *j*
assume $\forall i<j.\ p\ i < p\ j$
moreover have $p\ j < p\ (Suc\ j)$ **using** *f3 f4* **by** *force*
ultimately show $\forall i<Suc\ j.\ p\ i < p\ (Suc\ j)$ **by** (*metis less-antisym less-trans*)
qed
qed
then have *inj p* **unfolding** *inj-on-def* **by** (*metis nat-neq-iff*)
then have ¬ *finite* (*p‘UNIV*) **using** *finite-imageD* **by** *blast*
moreover obtain *P* **where** *f6*: $P = p`\{k.\ k \neq 0\}$ **by** *blast*
moreover have $UNIV = \{0\} \cup \{k{::}nat.\ k \neq 0\}$ **by** *blast*
moreover then have $p`UNIV = p`\{0\} \cup P \wedge finite\ (p`\{0\})$ **using** *f6* **by** *fastforce*
ultimately have *f7*: ¬ *finite P* **using** *finite-UnI* **by** *metis*
have $\forall\ n \in P.\ \forall\ m \in P.\ g\ (f\ n) = g\ (f\ m) \longrightarrow n = m$
proof (*intro ballI impI*)
fix *n m*
assume *g1*: $n \in P$ **and** *g2*: $m \in P$ **and** *g3*: $g\ (f\ n) = g\ (f\ m)$
have $n < m \longrightarrow False$
proof
assume $n < m$
moreover then have $m \notin J$ **using** *g2 f5 f6* **by** *blast*
ultimately show *False* **using** *g3 e3* **by** *force*
qed
moreover have $m < n \longrightarrow False$
proof
assume $m < n$
moreover then have $n \notin J$ **using** *g1 f5 f6* **by** *blast*
ultimately show *False* **using** *g3 e3* **by** *force*
qed
ultimately show $n = m$ **by** *force*
qed
then have *inj-on* $(g \circ f)$ *P* **unfolding** *inj-on-def* **by** *simp*
then have ¬ *finite* $((g \circ f)`UNIV)$ **using** *f7*
by (*metis finite-imageD infinite-iff-countable-subset subset-UNIV subset-image-iff*)
moreover have $(g \circ f)`UNIV = g`(f`UNIV)$ **by** *force*

ultimately show *False* **using** *e1* **by** *simp*
qed
then obtain *m* **where** $\forall$ *n>m. n* $\in$ *J* **by** *blast*
then have $\forall$ *n>m. yi n = xi n* **using** *e3 b4* **by** *force*
then have *e4*: *xi'{n. n>m}* $\subseteq$ *yi'UNIV* **by** (*metis image-Collect-subsetI rangeI*)
have *e5*: |*xi'{n. n>m}*| =*o* |{*n. n>m*}| **using** *b3* **by** (*metis card-of-image image-inv-f-f ordIso-iff-ordLeq*)
have *finite* {*n. n*$\leq$*m*} $\wedge$ ($\neg$ *finite* (*UNIV::nat set*)) $\wedge$ {*n. n*$\leq$*m*} $\cup$ {*n. n>m*} = *UNIV* **by** *force*
then have $\neg$ *finite* {*n. n>m*} **using** *finite-UnI* **by** *metis*
then have |*xi'{n. n>m}*| =*o* ω*-ord* **using** *e5* **by** (*meson card-of-UNIV card-of-nat*
finite-iff-cardOf-nat ordIso-transitive ordLeq-iff-ordLess-or-ordIso)
then show *?thesis* **using** *e4*
by (*metis finite-subset infinite-iff-natLeq-ordLeq ordIso-natLeq-infinite1*)
next
assume $\neg$ *finite* (*g'(f'UNIV)*)
moreover have *g'(f'UNIV)* $\subseteq$ *yi'UNIV*
proof
fix *a*
assume *a* $\in$ *g'(f'UNIV)*
then obtain *n* **where** *e1*: *a = g (f n)* **by** *blast*
obtain *P* **where** *e3*: *P* = (λ *i. i*$\leq$*n* $\wedge$ *g (f n) = g (f i)*) **by** *blast*
obtain *k* **where** *k* = (*LEAST i. P i*) **by** *blast*
moreover have *P n* **using** *e3* **by** *blast*
ultimately have *P k* $\wedge$ ($\forall$ *i. P i* $\longrightarrow$ *k* $\leq$ *i*) **using** *LeastI Least-le* **by** *metis*
then have *g (f n) = g (f k)* $\wedge$ $\neg$ ($\exists$ *i<k. g (f k) = g (f i)*)
using *e3* **by** (*metis leD less-le-trans less-or-eq-imp-le*)
then have *yi k = a* **using** *e1 b4 b7* **by** *fastforce*
then show *a* $\in$ *yi'UNIV* **by** *blast*
qed
ultimately have $\neg$ *finite* (*yi'UNIV*) **using** *finite-subset* **by** *metis*
then show *?thesis* **using** *infinite-iff-natLeq-ordLeq* **by** *blast*
qed
ultimately show *?thesis* **using** *ordLeq-transitive* **by** *blast*
qed
ultimately show ω*-ord* $\leq$*o* |*h'(I* α'*)*| **using** *ordLeq-transitive* **by** *blast*
qed
moreover have $\forall$ α β. α =*o* β $\longrightarrow$ *h* α = *h* β
proof (*intro allI impI*)
fix α*::'U rel* **and** β*::'U rel*
assume *c1*: α =*o* β
show *h* α = *h* β
proof (*cases* $\exists$ *n.* α =*o f n*)
assume $\exists$ *n.* α =*o f n*
moreover then have $\exists$ *n.* β =*o f n* **using** *c1 ordIso-transitive ordIso-symmetric* **by** *metis*

moreover have $\forall$ *n*. (α =*o f n*) = (β =*o f n*) **using** *c1 ordIso-transitive ordIso-symmetric* **by** *metis*
ultimately show *h* α = *h* β **using** *b5* **by** *simp*
next
assume ¬ ($\exists$ *n*. α =*o f n*)
moreover then have ¬ ($\exists$ *n*. β =*o f n*) **using** *c1 ordIso-transitive* **by** *metis*
ultimately show *h* α = *h* β **using** *b5 c1 a3* **by** *simp*
qed
qed
ultimately show *?thesis* **by** *blast*
next
assume ¬ ($\exists$ α::'*U rel*. ω-*ord* $\le$*o* α)
then show *?thesis* **using** *a3* **by** *blast*
qed

lemma *lem-Shinf-N-ne*:
fixes *r*::'*U rel* **and** *Ps*::'*U set set*
assumes *CCR r* **and** *Ps* $\subseteq$ *SCF r*
shows $\mathcal{N}$ *r Ps* $\neq$ {}
proof −
obtain *E* :: '*U* $\Rightarrow$ '*U set* $\Rightarrow$ '*U set* **where** *E* = (λ *a A*. *SOME A'*. *A'* $\in$ $\mathcal{E}$ *r a A Ps*) **by** *blast*
moreover have $\forall$ *a A*. $\exists$ *A'*. *A'* $\in$ $\mathcal{E}$ *r a A Ps* **using** *assms lem-Shinf-E-ne*[*of r Ps*] **by** *blast*
ultimately have *b1*: $\forall$ *a A*. *E a A* $\in$ $\mathcal{E}$ *r a A Ps* **by** (*meson someI-ex*)
have $\exists$ *g*::'*U rel* $\Rightarrow$ '*U*. (ω-*ord* $\le$*o* |*Field r*| $\longrightarrow$ *Field r* = *g* ‘ {γ. γ <*o* |*Field r*|}) $\wedge$
($\forall$ α'::'*U rel*. ω-*ord* $\le$*o* α' $\wedge$ α' $\le$*o* |*Field r*| $\longrightarrow$ ω-*ord* $\le$*o* |*g* ‘ {γ. γ <*o* α'}|) $\wedge$
($\forall$ α β. α =*o* β $\longrightarrow$ *g* α = *g* β)
proof(*cases* ω-*ord* $\le$*o* |*Field r*|)
assume *c1*: ω-*ord* $\le$*o* |*Field r*|
moreover have *Card-order* |*Field r*| $\wedge$ |*Field r*| $\le$*o* |*Field r*| **by** *simp*
ultimately obtain *g0*::'*U rel* $\Rightarrow$ '*U* **where**
c2: *Field r* $\subseteq$ *g0* ‘ {γ. γ <*o* |*Field r*| }
and *c3*: $\forall$ α β. α =*o* β $\longrightarrow$ *g0* α = *g0* β
using *c1 lem-card-setcv-inf-stab*[*of* |*Field r*| *Field r*] **by** *blast*
have *Field r* $\neq$ {} **using** *c1* **by** (*metis finite.emptyI infinite-iff-natLeq-ordLeq*)
then obtain *a0* **where** *a0* $\in$ *Field r* **by** *blast*
moreover obtain *t* **where** *t* = (λ *a*. *if* (*a* $\in$ *Field r*) *then a else a0*) **by** *blast*
moreover obtain *g1* **where** *g1* = (λ α. *t* (*g0* α)) **by** *blast*
ultimately have *c4*: *Field r* $\subseteq$ *g1*‘{γ . γ <*o* |*Field r*| }
and *c5*: $\forall$ α β. α =*o* β $\longrightarrow$ *g1* α = *g1* β **and** *c6*: *g1*‘*UNIV* $\subseteq$ *Field r* **using** *c2 c3* **by** *force+*
obtain *I* **where** *c7*: *I* = ($\lambda\alpha'$::'*U rel*. {α::'*U rel*. α <*o* α'}) **by** *blast*
then obtain *g* **where** *c8*: ($\forall$ α'. *g1*‘(*I* α') $\subseteq$ *g*‘(*I* α') $\wedge$ *g*‘(*I* α') $\subseteq$ *g1*‘(*I* α') $\cup$ (*Field r*))
and *c9*: $\forall$ α'. ω-*ord* $\le$*o* α' $\longrightarrow$ ω-*ord* $\le$*o* |*g*‘(*I* α')|
and *c10*: ($\forall$ α β. α =*o* β $\longrightarrow$ *g* α = *g* β) **using** *c1 c5 lem-oseq-fin-inj*[*of*

I Field r g1] **by** *blast*
have *g1*‘(*I* |*Field r*|) ⊆ *Field r* **using** *c6* **by** *blast*
then have *g* ‘ { *γ*. *γ* <*o* |*Field r*| } ⊆ *Field r* **using** *c7 c8* **by** *blast*
moreover have *Field r* ⊆ *g*‘{ *γ*. *γ* <*o* |*Field r*| } **using** *c4 c7 c8* **by** *force*
ultimately have *ω-ord* ≤*o* |*Field r*| ⟶ *Field r* = *g*‘{ *γ*. *γ* <*o* |*Field r*| } **by** *blast*
then show *?thesis* **using** *c7 c9 c10* **by** *blast*
next
assume ¬ *ω-ord* ≤*o* |*Field r*|
moreover then have ∀ *α*′::′*U rel*. ¬ (*ω-ord* ≤*o* *α*′ ∧ *α*′ ≤*o* |*Field r*|) **using** *ordLeq-transitive* **by** *blast*
moreover have ∃ *g*::′*U rel* ⇒ ′*U*. (∀ *α β*. *α* =*o* *β* ⟶ *g α* = *g β*) **by** *force*
ultimately show *?thesis* **by** *blast*
qed
then obtain *g*::′*U rel* ⇒ ′*U* **where**
b4: *ω-ord* ≤*o* |*Field r*| ⟶ *Field r* = *g* ‘ { *γ*::′*U rel*. *γ* <*o* |*Field r*| }
and *b4*′: ∀ *α*′::′*U rel*. *ω-ord* ≤*o* *α*′ ∧ *α*′ ≤*o* |*Field r*| ⟶ *ω-ord* ≤*o* |*g* ‘ {*γ*. *γ* <*o* *α*′}|
and *b5*: ∀ *α β*. *α* =*o* *β* ⟶ *g α* = *g β* **by** *blast*
obtain *F*::′*U rel* ⇒ ′*U set* ⇒ ′*U set* **where** *b6*: *F* = (*λ α A*. *E* (*g α*) *A*) **by** *blast*
then have ∀ *α β*. *α* =*o* *β* ⟶ *F α* = *F β* **using** *b5* **by** *fastforce*
then obtain *f*::′*U rel* ⇒ ′*U set* **where** *b7*: *f* ∈ 𝒯 *F*
unfolding 𝒯*-def* **using** *lem-ordseq-rec-sets*[*of F* {}] **by** *clarsimp*
have *b8*: *Well-order* |*Field r*| **by** *simp*
have 𝒩 *r Ps* ≠ {}
proof −
have *c0*: ∀ *α A*. *A* ∈ *SF r* ⟶ *F α A* ∈ *SF r* **using** *b6 b1* **unfolding** ℰ*-def* **by** *simp*
have *c1*: ∀ *α A*. *A* ⊆ *F α A* **using** *b6 b1* **unfolding** ℰ*-def* **by** *simp*
have *c2*: ∀ *α A*. (*g α* ∈ *Field r* ⟶ *g α* ∈ *F α A*) **using** *b6 b1* **unfolding** ℰ*-def* **by** *blast*
have *c3*: ∀ *α A*. *A* ∈ *SF r* ⟶ *ω-ord* ≤*o* |*A*| ⟶ *escl r A* (*F α A*) ⊆ (*F α A*) ∧ *clterm* (*Restr r* (*F α A*)) *r*
using *b6 b1* **unfolding** ℰ*-def* **by** *blast*
have *c4*: ∀ *α A*. *A* ∈ *SF r* ⟶
(∀ *a*∈*A*. *r* “ {*a*} ⊆ *w-dncl r A* ∨ *r* “ {*a*} ∩ (*F α A* − *w-dncl r A*) ≠ {})
using *b6 b1* **unfolding** ℰ*-def* **by** *blast*
have *c6*: ∀ *α A*. *A* ∈ *SF r* ⟶ *CCR* (*Restr r* (*F α A*))
using *b6 b1* **unfolding** ℰ*-def* **by** *blast*
have *c7*: ∀ *α A*. (|*A*| <*o* *ω-ord* ⟶ |*F α A*| <*o* *ω-ord*) ∧ (*ω-ord* ≤*o* |*A*| ⟶ |*F α A*| ≤*o* |*A*|)
using *b6 b1* **unfolding** ℰ*-def* **by** *blast*
have *c8*: ∀ *α A*. *A* ∈ *SF r* ⟶ ℰ*p r Ps A* (*F α A*) **using** *b6 b1* **unfolding** ℰ*-def* ℰ*p-def* **by** *blast*
have *c10*: ∀ *α A*. *A* ∈ *SF r* ⟶ ((∃ *y*. (*F α A*) − *dncl r A* ⊆ {*y*}) ⟶ (*Field r* ⊆ *dncl r* (*F α A*)))
using *b6 b1* **unfolding** ℰ*-def* **by** *blast*

have *c1'*: ∀α. *Well-order* α ⟶ *f* ∈ 𝒩*1 r* α **using** *b7 b8 c1 lem-Shinf-N1*[*of f F r*] **by** *blast*
have *c5'*: ∀α. *Well-order* α ⟶ *f* ∈ 𝒩*5 r* α **using** *b7 b8 c0 lem-Shinf-N5*[*of f F r*] **by** *blast*
have *f* ∈ 𝒩*1 r* |*Field r*| **using** *b7 b8 c1 lem-Shinf-N1*[*of f F r*] **by** *blast*
moreover have *f* ∈ 𝒩*2 r* |*Field r*| **using** *b7 b8 lem-Shinf-N2*[*of f F r*] **by** *blast*
moreover have *f* ∈ 𝒩*3 r* |*Field r*| **using** *b7 b8 c1 c3 c5' lem-Shinf-N3*[*of f F r*] **by** *blast*
moreover have *f* ∈ 𝒩*4 r* |*Field r*| **using** *b7 b8 c1 c4 c5' lem-Shinf-N4*[*of f F r*] **by** *blast*
moreover have *f* ∈ 𝒩*5 r* |*Field r*| **using** *b7 b8 c0 lem-Shinf-N5*[*of f F r*] **by** *blast*
moreover have *f* ∈ 𝒩*6 r* |*Field r*| **using** *b7 b8 c1 c6 c5' lem-Shinf-N6*[*of f F r*] **by** *blast*
moreover have *f* ∈ 𝒩*7 r* |*Field r*| **using** *b7 b8 c1 c7 lem-Shinf-N7*[*of f F r*] **by** *blast*
moreover have *f* ∈ 𝒩*8 r Ps* |*Field r*| **using** *b7 b8 c1 c7 c8 c5' lem-Shinf-N8*[*of f F r Ps*] **by** *blast*
moreover have *f* ∈ 𝒩*9 r* |*Field r*| **using** *b7 b4 c1 c2 lem-Shinf-N9*[*of f F g r*] **by** *blast*
moreover have *f* ∈ 𝒩*10 r* |*Field r*| **using** *b7 b8 c1 c10 c5' lem-Shinf-N10*[*of f F r*] **by** *metis*
moreover have *f* ∈ 𝒩*11 r* |*Field r*| **using** *b7 b8 c1 c10 c5' lem-Shinf-N11*[*of f F r*] **by** *metis*
moreover have *f* ∈ 𝒩*12 r* |*Field r*| **using** *b7 c1' c2 b4 b4' lem-Shinf-N12*[*of f F r g*] **by** *blast*
moreover have ∀ α β. α =*o* β ⟶ *f* α = *f* β **using** *b7* **unfolding** 𝒯*-def* **by** *blast*
ultimately show *?thesis* **unfolding** 𝒩*-def* **by** *blast*
qed
then show *?thesis* **by** *blast*
qed

lemma *lem-wrankrel-eq*: *wrank-rel r A0* α ⟹ α =*o* β ⟹ *wrank-rel r A0* β
proof −
assume *a1*: *wrank-rel r A0* α **and** *a2*: α =*o* β
then obtain *B* **where** *B* ∈ *wbase r A0* ∧ |*B*| =*o* α ∧ (∀ *B'* ∈ *wbase r A0*. |*B*| ≤*o* |*B'*|) **unfolding** *wrank-rel-def* **by** *blast*
moreover then have |*B*| =*o* β **using** *a2* **by** (*metis ordIso-transitive*)
ultimately show *wrank-rel r A0* β **unfolding** *wrank-rel-def* **by** *blast*
qed

lemma *lem-wrank-wrankrel*:
fixes *r*::*'U rel* **and** *A0*::*'U set*
shows *wrank-rel r A0* (*wrank r A0*)
proof −
have *b1*: *wbase r A0* ≠ {} **using** *lem-wdn-range-lb*[*of A0 r*] **unfolding** *wbase-def* **by** *blast*
obtain *Q* **where** *b2*: *Q* = { α::*'U rel*. ∃ *A* ∈ *wbase r A0*. α =*o* |*A*| } **by** *blast*

have *b3*: $\forall$ $A \in$ *wbase r A0*. $\exists$ $\alpha \in Q$. $\alpha \leq o$ $|A|$
proof
fix *A*
assume $A \in$ *wbase r A0*
then have $|A| \in Q \wedge |A| =o$ $|A|$ **using** *b2 ordIso-symmetric* **by** *force*
then show $\exists$ $\alpha \in Q$. $\alpha \leq o$ $|A|$ **using** *ordIso-iff-ordLeq* **by** *blast*
qed
then have $Q \neq \{\}$ **using** *b1* **by** *blast*
then obtain α **where** *b4*: $\alpha \in Q \wedge (\forall \alpha'. \alpha' <o\ \alpha \longrightarrow \alpha' \notin Q)$ **using** *wf-ordLess wf-eq-minimal*[*of ordLess*] **by** *blast*
moreover have $\forall$ $\alpha' \in Q$. *Card-order* α' **using** *b2* **using** *ordIso-card-of-imp-Card-order* **by** *blast*
ultimately have $\forall$ $\alpha' \in Q$. $\neg$ $(\alpha' <o\ \alpha) \longrightarrow \alpha \leq o\ \alpha'$ **by** *simp*
then have *b5*: $\alpha \in Q \wedge (\forall$ $\alpha' \in Q$. $\alpha \leq o\ \alpha')$ **using** *b4* **by** *blast*
then obtain *A* **where** *b6*: $A \in$ *wbase r A0* $\wedge$ $|A| =o\ \alpha$ **using** *b2 ordIso-symmetric* **by** *blast*
moreover have $\forall$ $B \in$*wbase r A0*. $|A| \leq o$ $|B|$
proof
fix *B*
assume $B \in$ *wbase r A0*
then obtain α' **where** $\alpha' \in Q \wedge \alpha' \leq o$ $|B|$ **using** *b3* **by** *blast*
moreover then have $|A| =o\ \alpha \wedge \alpha \leq o\ \alpha'$ **using** *b5 b6* **by** *blast*
ultimately show $|A| \leq o$ $|B|$ **using** *ordIso-ordLeq-trans ordLeq-transitive* **by** *blast*
qed
ultimately have *wrank-rel r A0* α **unfolding** *wrank-rel-def* **by** *blast*
then show *?thesis* **unfolding** *wrank-def* **by** (*metis someI2*)
qed

lemma *lem-wrank-uset*:
fixes *r*::'*U rel* **and** *A0*::'*U set*
shows $\exists$ $A \in$ *wbase r A0*. $|A| =o$ *wrank r A0* $\wedge$ ($\forall$ $B \in$ *wbase r A0*. $|A| \leq o$ $|B|$)
using *lem-wrank-wrankrel* **unfolding** *wrank-rel-def* **by** *blast*

lemma *lem-wrank-uset-mem-bnd*:
fixes *r*::'*U rel* **and** *A0 B*::'*U set*
assumes $B \in$ *wbase r A0*
shows *wrank r A0* $\leq o$ $|B|$
proof −
obtain *A* **where** $A \in$ *wbase r A0* $\wedge$ $|A| =o$ *wrank r A0* $\wedge$ ($\forall$ $A' \in$ *wbase r A0*. $|A| \leq o$ $|A'|$) **using** *assms lem-wrank-uset* **by** *blast*
moreover then have $|A| \leq o$ $|B|$ **using** *assms* **by** *blast*
ultimately show *?thesis* **by** (*metis ordIso-iff-ordLeq ordLeq-transitive*)
qed

lemma *lem-wrank-cardord*: *Card-order* (*wrank r A0*)
proof −
obtain *A* **where** $A \in$ *wbase r A0* $\wedge$ $|A| =o$ *wrank r A0* **using** *lem-wrank-uset*

by *blast*
then show *Card-order* (*wrank r A0*) **using** *Card-order-ordIso2 card-of-Card-order*
by *blast*
qed

lemma *lem-wrank-ub*: *wrank r A0* $\leq o$ |*A0*|
using *lem-wdn-range-lb*[*of A0 r*] *lem-wrank-uset-mem-bnd* **unfolding** *wbase-def*
by *blast*

lemma *lem-card-un2-bnd*: ω-*ord* $\leq o$ α $\Longrightarrow$ |*A*| $\leq o$ α $\Longrightarrow$ |*B*| $\leq o$ α $\Longrightarrow$ |*A* $\cup$ *B*| $\leq o$ α
proof −
assume ω-*ord* $\leq o$ α **and** |*A*| $\leq o$ α **and** |*B*| $\leq o$ α
moreover have |{*A*, *B*}| $\leq o$ ω-*ord* **using** *finite-iff-ordLess-natLeq ordLess-imp-ordLeq*
by *blast*
ultimately have |$\bigcup${*A*, *B*}| $\leq o$ α **using** *lem-card-un-bnd*[*of* {*A*,*B*}] *ordLeq-transitive*
by *blast*
then show |*A* $\cup$ *B*| $\leq o$ α **by** *simp*
qed

lemma *lem-card-un2-lsbnd*: ω-*ord* $\leq o$ α $\Longrightarrow$ |*A*| $<o$ α $\Longrightarrow$ |*B*| $<o$ α $\Longrightarrow$ |*A* $\cup$ *B*| $<o$ α
proof −
assume *b1*: ω-*ord* $\leq o$ α **and** *b2*: |*A*| $<o$ α **and** *b3*: |*B*| $<o$ α
have ¬ *finite A* $\longrightarrow$ |*A* $\cup$ *B*| $<o$ α
proof
assume *c1*: ¬ *finite A*
show |*A* $\cup$ *B*| $<o$ α
proof (*cases* |*A*| $\leq o$ |*B*|)
assume |*A*| $\leq o$ |*B*|
then have |*A* $\cup$ *B*| $=o$ |*B*| **using** *c1* **by** (*metis card-of-Un-infinite card-of-ordLeq-finite*)
then show *?thesis* **using** *b3* **by** (*metis ordIso-ordLess-trans*)
next
assume ¬ |*A*| $\leq o$ |*B*|
then have |*B*| $\leq o$ |*A*| **by** (*metis card-of-Well-order ordLeq-total*)
then have |*A* $\cup$ *B*| $=o$ |*A*| **using** *c1* **by** (*metis card-of-Un-infinite*)
then show *?thesis* **using** *b2* **by** (*metis ordIso-ordLess-trans*)
qed
qed
moreover have ¬ *finite B* $\longrightarrow$ |*A* $\cup$ *B*| $<o$ α
proof
assume *c1*: ¬ *finite B*
show |*A* $\cup$ *B*| $<o$ α
proof (*cases* |*A*| $\leq o$ |*B*|)
assume |*A*| $\leq o$ |*B*|
then have |*A* $\cup$ *B*| $=o$ |*B*| **using** *c1* **by** (*metis card-of-Un-infinite*)
then show *?thesis* **using** *b3* **by** (*metis ordIso-ordLess-trans*)
next
assume ¬ |*A*| $\leq o$ |*B*|

then have $|B| \leq o\ |A|$ **by** (*metis card-of-Well-order ordLeq-total*)
then have $|A \cup B| =o\ |A|$ **using** *c1* **by** (*metis card-of-Un-infinite card-of-ordLeq-finite*)
then show *?thesis* **using** *b2* **by** (*metis ordIso-ordLess-trans*)
qed
qed
moreover have *finite A* $\wedge$ *finite B* $\longrightarrow |A \cup B| <o\ \alpha$
proof
assume *finite A* $\wedge$ *finite B*
then have *finite* $(A \cup B)$ **by** *blast*
then show $|A \cup B| <o\ \alpha$ **using** *b1*
by (*meson card-of-nat finite-iff-cardOf-nat ordIso-imp-ordLeq ordLess-ordLeq-trans*)

qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-wrank-un-bnd*:
fixes r::$'U$ *rel* **and** S::$'U$ *set set* **and** α::$'U$ *rel*
assumes *a1*: $\forall\ A{\in}S.$ *wrank r A* $\leq o\ \alpha$ **and** *a2*: $|S| \leq o\ \alpha$ **and** *a3*: ω*-ord* $\leq o\ \alpha$
shows *wrank r* $(\bigcup\ S) \leq o\ \alpha$
proof −
obtain *h* **where** *b1*: $h = (\lambda\ A\ B.\ B \in$ *wbase r A* $\wedge\ |B| =o$ *wrank r A*) **by** *blast*
obtain *Bi* **where** *b2*: *Bi* = (λ *A. SOME B. h A B*) **by** *blast*
have $\forall\,A{\in}S.\ \exists\ B.$ *h A B* **using** *b1 lem-wrank-uset*[*of r*] **by** *blast*
then have $\forall\,A{\in}S.$ *h A* (*Bi A*) **using** *b2* **by** (*metis someI-ex*)
then have *b3*: $\forall\,A{\in}S.$ (*Bi A*) $\in$ *wbase r A* $\wedge$ $|Bi\ A| =o$ *wrank r A* **using** *b1* **by** *blast*
then have *b4*: $\forall\ A \in S.\ |Bi\ A| \leq o\ \alpha$ **using** *assms ordIso-ordLeq-trans* **by** *blast*
obtain S' **where** *b5*: $S' = Bi$ ‘ S **by** *blast*
then have $|S'| \leq o\ |S| \wedge (\forall\ X \in S'.\ |X| \leq o\ \alpha)$ **using** *b4* **by** *simp*
moreover then have $|S'| \leq o\ \alpha$ **using** *a2* **by** (*metis ordLeq-transitive*)
ultimately have $|\bigcup S'| \leq o\ \alpha$ **using** *a3 lem-card-un-bnd*[*of* S' α] **by** *blast*
moreover obtain *B* **where** *b6*: $B = (\bigcup A{\in}S.\ Bi\ A)$ **by** *blast*
ultimately have *b7*: $|B| \leq o\ \alpha$ **using** *b5* **by** *simp*
have $\forall\,A{\in}S.\ A \subseteq$ *w-dncl r* (*Bi A*) **using** *b3* **unfolding** *wbase-def* **by** *blast*
then have $\bigcup S \subseteq$ *w-dncl r B* **using** *b6 lem-wdn-mon*[*of - B r*] **by** *blast*
then have $B \in$ *wbase r* $(\bigcup S)$ **unfolding** *wbase-def* **by** *blast*
then have *wrank r* $(\bigcup S) \leq o\ |B|$ **using** *lem-wrank-uset-mem-bnd* **by** *blast*
then show *?thesis* **using** *b7* **by** (*metis ordLeq-transitive*)
qed

lemma *lem-wrank-un-bnd-stab*:
fixes r::$'U$ *rel* **and** S::$'U$ *set set* **and** α::$'U$ *rel*
assumes *a1*: $\forall\ A{\in}S.$ *wrank r A* $<o\ \alpha$ **and** *a2*: $|S| <o\ \alpha$ **and** *a3*: *stable* α
shows *wrank r* $(\bigcup\ S) <o\ \alpha$
proof −
obtain *h* **where** *b1*: $h = (\lambda\ A\ B.\ B \in$ *wbase r A* $\wedge\ |B| =o$ *wrank r A*) **by** *blast*
obtain *Bi* **where** *b2*: *Bi* = (λ *A. SOME B. h A B*) **by** *blast*
have $\forall\,A{\in}S.\ \exists\ B.$ *h A B* **using** *b1 lem-wrank-uset*[*of r*] **by** *blast*

then have $\forall A{\in}S.\ h\ A\ (Bi\ A)$ **using** *b2* **by** (*metis someI-ex*)
then have *b3*: $\forall A{\in}S.\ (Bi\ A) \in wbase\ r\ A \wedge |Bi\ A| =o\ wrank\ r\ A$ **using** *b1* **by** *blast*
then have *b4*: $\forall\ A \in S.\ |Bi\ A| <o\ \alpha$ **using** *assms ordIso-ordLess-trans* **by** *blast*
obtain S' **where** *b5*: $S' = Bi\ `\ S$ **by** *blast*
then have $|S'| \leq o\ |S| \wedge (\forall\ X \in S'.\ |X| <o\ \alpha)$ **using** *b4* **by** *simp*
moreover then have $|S'| <o\ \alpha$ **using** *a2* **by** (*metis ordLeq-ordLess-trans*)
ultimately have $|\bigcup S'| <o\ \alpha$ **using** *a3 lem-card-un-bnd-stab*[*of* α S'] **by** *blast*
moreover obtain B **where** *b6*: $B = (\bigcup A{\in}S.\ Bi\ A)$ **by** *blast*
ultimately have *b7*: $|B| <o\ \alpha$ **using** *b5* **by** *simp*
have $\forall A{\in}S.\ A \subseteq w\text{-}dncl\ r\ (Bi\ A)$ **using** *b3* **unfolding** *wbase-def* **by** *blast*
then have $\bigcup S \subseteq w\text{-}dncl\ r\ B$ **using** *b6 lem-wdn-mon*[*of - B r*] **by** *blast*
then have $B \in wbase\ r\ (\bigcup S)$ **unfolding** *wbase-def* **by** *blast*
then have $wrank\ r\ (\bigcup S) \leq o\ |B|$ **using** *lem-wrank-uset-mem-bnd* **by** *blast*
then show *?thesis* **using** *b7* **by** (*metis ordLeq-ordLess-trans*)
qed

lemma *lem-wrank-fw*:
fixes r::$'U\ rel$ **and** K::$'U\ set$ **and** α::$'U\ rel$
assumes *a1*: $\omega\text{-}ord \leq o\ \alpha$ **and** *a2*: $wrank\ r\ K \leq o\ \alpha$ **and** *a3*: $\forall\ b{\in}K.\ wrank\ r\ (r``\{b\}) \leq o\ \alpha$
shows $wrank\ r\ (\bigcup b{\in}K.\ (r``\{b\})) \leq o\ \alpha$
proof −
obtain h **where** *b1*: $h = (\lambda\ A\ B.\ B \in wbase\ r\ A \wedge |B| =o\ wrank\ r\ A)$ **by** *blast*
obtain Bi **where** *b2*: $Bi = (\lambda\ b.\ SOME\ B.\ h\ (r``\{b\})\ B)$ **by** *blast*
have $\forall b{\in}K.\ \exists\ B.\ h\ (r``\{b\})\ B$ **using** *b1 lem-wrank-uset*[*of r*] **by** *blast*
then have $\forall b{\in}K.\ h\ (r``\{b\})\ (Bi\ b)$ **using** *b2* **by** (*metis someI-ex*)
then have *b3*: $\forall b{\in}K.\ (Bi\ b) \in wbase\ r\ (r``\{b\}) \wedge |Bi\ b| =o\ wrank\ r\ (r``\{b\})$ **using** *b1* **by** *blast*
obtain BK **where** *b4*: $BK \in wbase\ r\ K \wedge |BK| =o\ wrank\ r\ K$ **using** *lem-wrank-uset*[*of r K*] **by** *blast*
obtain BU **where** *b5*: $BU = BK \cup (\bigcup b{\in}(K{\cap}BK).\ Bi\ b)$ **by** *blast*
obtain S **where** *b6*: $S = (\bigcup b{\in}K.\ (r``\{b\}))$ **by** *blast*
have *b7*: $\forall\ b \in K{\cap}BK.\ (r``\{b\}) \subseteq w\text{-}dncl\ r\ BU$
proof
fix b
assume $b \in K \cap BK$
then have $Bi\ b \subseteq BU \wedge (Bi\ b) \in wbase\ r\ (r``\{b\})$ **using** *b3 b5* **by** *blast*
then show $r``\{b\} \subseteq w\text{-}dncl\ r\ BU$ **using** *lem-wdn-mon* **unfolding** *wbase-def* **by** *blast*
qed
have $BU \in wbase\ r\ S$
proof −
have $\forall\ b \in K.\ r``\{b\} \subseteq dncl\ r\ BU$
proof
fix b
assume *d1*: $b \in K$
show $r``\{b\} \subseteq dncl\ r\ BU$
proof (*cases* $b \in BK$)

assume $b \in BK$
then show *?thesis* **using** *d1 b7* **unfolding** *w-dncl-def* **by** *blast*
next
assume *e1*: $b \notin BK$
have $\forall\ t \in r``\{b\}.\ t \notin dncl\ r\ BU \longrightarrow False$
proof (*intro ballI impI*)
fix *t*
assume *f1*: $t \in r``\{b\}$ **and** *f2*: $t \notin dncl\ r\ BU$
then have *f3*: $t \notin dncl\ r\ BK$ **using** *b5* **unfolding** *dncl-def* **by** *blast*
moreover have $b \in w\text{-}dncl\ r\ BK$ **using** *d1 b4* **unfolding** *wbase-def* **by** *blast*
ultimately have *f4*: $\forall F \in \mathcal{F}\ r\ b\ t.\ F \cap BK \neq \{\}$ **unfolding** *w-dncl-def* **by** *blast*
obtain *f* **where** *f5*: $f = (\lambda\ n{::}nat.\ if\ (n = 0)\ then\ b\ else\ t)$ **by** *blast*
then have $f\ 0 = b \wedge f\ 1 = t$ **by** *simp*
moreover then have $\forall i{<}1.\ (f\ i,\ f\ (Suc\ i)) \in r$ **using** *f1* **by** *simp*
ultimately have $f \in rpth\ r\ b\ t\ 1 \wedge \{b,\ t\} = f\ `\ \{i.\ i \leq 1\}$
using *f5* **unfolding** *rpth-def* **by** *force*
then have $\{b,\ t\} \in \mathcal{F}\ r\ b\ t$ **unfolding** *$\mathcal{F}$-def* **by** *blast*
then have $\{b,\ t\} \cap BK \neq \{\}$ **using** *f4* **by** *blast*
then show *False* **using** *e1 f3* **unfolding** *dncl-def* **by** *blast*
qed
then show *?thesis* **by** *blast*
qed
qed
then have *c1*: $S \subseteq dncl\ r\ BU$ **using** *b6* **by** *blast*
moreover have $\forall\ x \in S.\ \forall c.\ \forall F{\in}\mathcal{F}\ r\ x\ c.\ c \notin dncl\ r\ BU \longrightarrow F \cap BU \neq \{\}$
proof (*intro ballI allI impI*)
fix *x c F*
assume *d1*: $x \in S$ **and** *d2*: $F \in \mathcal{F}\ r\ x\ c$ **and** *d3*: $c \notin dncl\ r\ BU$
then obtain *b* **where** *d4*: $b \in K \wedge (b,x) \in r$ **using** *b6* **by** *blast*
show $F \cap BU \neq \{\}$
proof (*cases* $b \in BK$)
assume $b \in BK$
then have $x \in w\text{-}dncl\ r\ BU$ **using** *b7 d4* **by** *blast*
then show *?thesis* **using** *d2 d3* **unfolding** *w-dncl-def* **by** *blast*
next
assume *e1*: $b \notin BK$
have *e2*: $b \in w\text{-}dncl\ r\ BK$ **using** *d4 b4* **unfolding** *wbase-def* **by** *blast*
obtain *f n* **where** *e3*: $f \in rpth\ r\ x\ c\ n$ **and** *e4*: $F = f\ `\ \{i.\ i{\leq}n\}$
using *d2* **unfolding** *$\mathcal{F}$-def* **by** *blast*
obtain *g* **where** *e5*: $g = (\lambda\ k{::}nat.\ if\ (k{=}0)\ then\ b\ else\ (f\ (k{-}1)))$ **by** *blast*
then have $g \in rpth\ r\ b\ c\ (Suc\ n)$
using *e3 d4* **unfolding** *rpth-def*
by (*simp, metis Suc-le-eq diff-Suc-Suc diff-zero gr0-implies-Suc less-Suc-eq-le*)
then have $g\ `\ \{i.\ i \leq (Suc\ n)\} \in \mathcal{F}\ r\ b\ c \wedge c \notin dncl\ r\ BK$
using *d3 b5* **unfolding** *$\mathcal{F}$-def dncl-def* **by** *blast*
then have $g\ `\ \{i.\ i \leq (Suc\ n)\} \cap BK \neq \{\}$ **using** *e2* **unfolding** *w-dncl-def* **by** *blast*

moreover have g ‘ $\{i.\ i \leq (Suc\ n)\} \subseteq F \cup \{b\}$
proof
fix a
assume $a \in g$ ‘ $\{i.\ i \leq (Suc\ n)\}$
then obtain i **where** $i \leq (Suc\ n) \wedge a = g\ i$ **by** *blast*
then show $a \in F \cup \{b\}$ **using** *e4 e5* **by** *force*
qed
ultimately have $(F \cup \{b\}) \cap BK \neq \{\}$ **by** *blast*
then show *?thesis* **using** *e1 b5* **by** *blast*
qed
qed
ultimately have $S \subseteq$ *w-dncl r BU* **unfolding** *w-dncl-def* **by** *blast*
then show *?thesis* **unfolding** *wbase-def* **by** *blast*
qed
moreover have $|BU| \leq o\ \alpha$
proof $-$
have *c1*: $|BK| \leq o\ \alpha$ **using** *b4 a2* **by** (*metis ordIso-ordLeq-trans*)
then have $|K \cap BK| \leq o\ \alpha$ **by** (*meson card-of-mono1 inf-le2 ordLeq-transitive*)
then have $|Bi$ ‘ $(K \cap BK)| \leq o\ \alpha$ **by** (*metis card-of-image ordLeq-transitive*)
moreover have $\forall\ b \in (K \cap BK).\ |Bi\ b| \leq o\ \alpha$ **using** *b3 a3* **by** (*meson Int-iff ordIso-ordLeq-trans*)
ultimately have $|\bigcup(Bi$ ‘ $(K \cap BK))| \leq o\ \alpha$ **using** *a1 lem-card-un-bnd*[*of* Bi‘$(K \cap BK)\ \alpha$] **by** *blast*
then show $|BU| \leq o\ \alpha$ **using** *c1 b5 a1 lem-card-un2-bnd*[*of* $\alpha\ BK \bigcup(Bi$ ‘ $(K \cap BK))$] **by** *simp*
qed
ultimately have *wrank r S* $\leq o\ \alpha$ **using** *b6 lem-wrank-uset-mem-bnd ordLeq-transitive* **by** *blast*
then show *?thesis* **using** *b6* **by** *blast*
qed

lemma *lem-wrank-fw-stab*:
fixes r::$'U$ *rel* **and** K::$'U$ *set* **and** α::$'U$ *rel*
assumes *a1*: ω*-ord* $\leq o\ \alpha \wedge$ *stable* α **and** *a2*: *wrank r K* $< o\ \alpha$ **and** *a3*: $\forall\ b \in K.$ *wrank r* $(r``\{b\}) < o\ \alpha$
shows *wrank r* $(\bigcup b \in K.\ (r``\{b\})) < o\ \alpha$
proof $-$
obtain h **where** *b1*: $h = (\lambda\ A\ B.\ B \in$ *wbase r A* $\wedge |B| =o$ *wrank r A*$)$ **by** *blast*
obtain Bi **where** *b2*: $Bi = (\lambda\ b.$ *SOME* $B.\ h\ (r``\{b\})\ B)$ **by** *blast*
have $\forall b \in K.\ \exists\ B.\ h\ (r``\{b\})\ B$ **using** *b1 lem-wrank-uset*[*of r*] **by** *blast*
then have $\forall b \in K.\ h\ (r``\{b\})\ (Bi\ b)$ **using** *b2* **by** (*metis someI-ex*)
then have *b3*: $\forall b \in K.\ (Bi\ b) \in$ *wbase r* $(r``\{b\}) \wedge |Bi\ b| =o$ *wrank r* $(r``\{b\})$ **using** *b1* **by** *blast*
obtain BK **where** *b4*: $BK \in$ *wbase r K* $\wedge |BK| =o$ *wrank r K* **using** *lem-wrank-uset*[*of r K*] **by** *blast*
obtain BU **where** *b5*: $BU = BK \cup (\bigcup b \in (K \cap BK).\ Bi\ b)$ **by** *blast*
obtain S **where** *b6*: $S = (\bigcup b \in K.\ (r``\{b\}))$ **by** *blast*
have *b7*: $\forall\ b \in K \cap BK.\ (r``\{b\}) \subseteq$ *w-dncl r BU*
proof

fix *b*
assume $b \in K \cap BK$
then have $Bi\ b \subseteq BU \wedge (Bi\ b) \in wbase\ r\ (r``\{b\})$ **using** *b3 b5* **by** *blast*
then show $r``\{b\} \subseteq w\text{-}dncl\ r\ BU$ **using** *lem-wdn-mon* **unfolding** *wbase-def* **by** *blast*
qed
have $BU \in wbase\ r\ S$
proof –
have $\forall\ b \in K.\ r``\{b\} \subseteq dncl\ r\ BU$
proof
fix *b*
assume *d1*: $b \in K$
show $r``\{b\} \subseteq dncl\ r\ BU$
proof (*cases* $b \in BK$)
assume $b \in BK$
then show *?thesis* **using** *d1 b7* **unfolding** *w-dncl-def* **by** *blast*
next
assume *e1*: $b \notin BK$
have $\forall\ t \in r``\{b\}.\ t \notin dncl\ r\ BU \longrightarrow False$
proof (*intro ballI impI*)
fix *t*
assume *f1*: $t \in r``\{b\}$ **and** *f2*: $t \notin dncl\ r\ BU$
then have *f3*: $t \notin dncl\ r\ BK$ **using** *b5* **unfolding** *dncl-def* **by** *blast*
moreover have $b \in w\text{-}dncl\ r\ BK$ **using** *d1 b4* **unfolding** *wbase-def* **by** *blast*
ultimately have *f4*: $\forall F \in \mathcal{F}\ r\ b\ t.\ F \cap BK \neq \{\}$ **unfolding** *w-dncl-def* **by** *blast*
obtain *f* **where** *f5*: $f = (\lambda\ n{::}nat.\ if\ (n = 0)\ then\ b\ else\ t)$ **by** *blast*
then have $f\ 0 = b \wedge f\ 1 = t$ **by** *simp*
moreover then have $\forall i{<}1.\ (f\ i,\ f\ (Suc\ i)) \in r$ **using** *f1* **by** *simp*
ultimately have $f \in rpth\ r\ b\ t\ 1 \wedge \{b,\ t\} = f\ `\ \{i.\ i \leq 1\}$
using *f5* **unfolding** *rpth-def* **by** *force*
then have $\{b,\ t\} \in \mathcal{F}\ r\ b\ t$ **unfolding** *$\mathcal{F}$-def* **by** *blast*
then have $\{b,\ t\} \cap BK \neq \{\}$ **using** *f4* **by** *blast*
then show *False* **using** *e1 f3* **unfolding** *dncl-def* **by** *blast*
qed
then show *?thesis* **by** *blast*
qed
qed
then have *c1*: $S \subseteq dncl\ r\ BU$ **using** *b6* **by** *blast*
moreover have $\forall\ x \in S.\ \forall c.\ \forall F{\in}\mathcal{F}\ r\ x\ c.\ c \notin dncl\ r\ BU \longrightarrow F \cap BU \neq \{\}$
proof (*intro ballI allI impI*)
fix *x c F*
assume *d1*: $x \in S$ **and** *d2*: $F \in \mathcal{F}\ r\ x\ c$ **and** *d3*: $c \notin dncl\ r\ BU$
then obtain *b* **where** *d4*: $b \in K \wedge (b,x) \in r$ **using** *b6* **by** *blast*
show $F \cap BU \neq \{\}$
proof (*cases* $b \in BK$)
assume $b \in BK$
then have $x \in w\text{-}dncl\ r\ BU$ **using** *b7 d4* **by** *blast*

then show *?thesis* using *d2 d3* unfolding *w-dncl-def* by *blast*
next
assume *e1*: $b \notin BK$
have *e2*: $b \in$ *w-dncl r BK* using *d4 b4* unfolding *wbase-def* by *blast*
obtain *f n* where *e3*: $f \in$ *rpth r x c n* and *e4*: $F = f$ ‘ $\{i.\ i \leq n\}$
using *d2* unfolding *$\mathcal{F}$-def* by *blast*
obtain *g* where *e5*: $g = (\lambda\ k::nat.\ if\ (k=0)\ then\ b\ else\ (f\ (k-1)))$ by *blast*
then have $g \in$ *rpth r b c* (*Suc n*)
using *e3 d4* unfolding *rpth-def*
by (*simp, metis Suc-le-eq diff-Suc-Suc diff-zero gr0-implies-Suc less-Suc-eq-le*)
then have g ‘ $\{i.\ i \leq (Suc\ n)\} \in \mathcal{F}\ r\ b\ c \wedge c \notin dncl\ r\ BK$
using *d3 b5* unfolding *$\mathcal{F}$-def dncl-def* by *blast*
then have g ‘ $\{i.\ i \leq (Suc\ n)\} \cap BK \neq \{\}$ using *e2* unfolding *w-dncl-def* by *blast*
moreover have g ‘ $\{i.\ i \leq (Suc\ n)\} \subseteq F \cup \{b\}$
proof
fix *a*
assume $a \in g$ ‘ $\{i.\ i \leq (Suc\ n)\}$
then obtain *i* where $i \leq (Suc\ n) \wedge a = g\ i$ by *blast*
then show $a \in F \cup \{b\}$ using *e4 e5* by *force*
qed
ultimately have $(F \cup \{b\}) \cap BK \neq \{\}$ by *blast*
then show *?thesis* using *e1 b5* by *blast*
qed
qed
ultimately have $S \subseteq$ *w-dncl r BU* unfolding *w-dncl-def* by *blast*
then show *?thesis* unfolding *wbase-def* by *blast*
qed
moreover have $|BU| <o\ \alpha$
proof −
have *c1*: $|BK| <o\ \alpha$ using *b4 a2* by (*metis ordIso-imp-ordLeq ordLeq-ordLess-trans*)
then have $|K \cap BK| <o\ \alpha$ by (*meson Int-iff card-of-mono1 ordLeq-ordLess-trans subsetI*)
then have $|Bi$ ‘ $(K \cap BK)| <o\ \alpha$ by (*metis card-of-image ordLeq-ordLess-trans*)
moreover have $\forall\ b{\in}(K{\cap}BK).\ |Bi\ b| <o\ \alpha$ using *b3 a3* by (*meson Int-iff ordIso-ordLess-trans*)
ultimately have $|\bigcup(Bi$ ‘ $(K \cap BK))| <o\ \alpha$ using *a1 lem-card-un-bnd-stab*[*of* α *Bi*‘(*K*∩*BK*)] by *blast*
then show $|BU| <o\ \alpha$ using *c1 b5 a1 lem-card-un2-lsbnd*[*of* α *BK* $\bigcup(Bi$ ‘ $(K \cap BK))$] by *simp*
qed
ultimately have *wrank r S* $<o\ \alpha$ using *b6 lem-wrank-uset-mem-bnd*[*of BU r S*] by (*metis ordLeq-ordLess-trans*)
then show *?thesis* using *b6* by *blast*
qed

lemma *lem-wnb-neib*:
fixes $r::'U\ rel$ and $\alpha::'U\ rel$
assumes *a1*: $\omega\text{-}ord \leq o\ \alpha$ and *a2*: $\alpha <o\ \|r\|$

shows ∀ *a* ∈ *Field r*. ∃ *b* ∈ *Mwn r α*. (*a*,*b*) ∈ *r*^*
proof
  fix *a*
  assume *b1*: *a* ∈ *Field r*
  have ¬ (∃ *b* ∈ *Mwn r α*. (*a*,*b*) ∈ *r*^*) ⟶ *False*
  proof
    assume *c1*: ¬ (∃ *b* ∈ *Mwn r α*. (*a*,*b*) ∈ *r*^*)
    obtain *B* **where** *c2*: *B* = (*r*^*)‘‘{*a*} **by** *blast*
    obtain *S* **where** *c3*: *S* = ((λ *n*. (*r*^^*n*)‘‘{*a*}) ‘ (*UNIV*::*nat set*)) **by** *blast*
    have *c4*: ∀ *b* ∈ *B*. *wrank r* (*r*‘‘{*b*}) ≤*o* *α*
    proof
      fix *b*
      assume *d1*: *b* ∈ *B*
      then obtain *k* **where** *b* ∈ (*r*^^*k*)‘‘{*a*} **using** *c2 rtrancl-power* **by** *blast*
      moreover have ∀ *n*. (*r*^^*n*) ‘‘ {*a*} ⊆ *Field r*
      proof
        fix *n*
        show (*r*^^*n*) ‘‘ {*a*} ⊆ *Field r* **using** *b1*
        by (*induct n*, *force*, *meson FieldI2 Image-singleton-iff relpow-Suc-E subsetI*)
      qed
      ultimately have *b* ∈ *Field r* **by** *blast*
      moreover have *b* ∉ *Mwn r α* **using** *d1 c1 c2* **by** *blast*
      ultimately have *b* ∈ *Field r* − *Mwn r α* **by** *blast*
      moreover have *Well-order α* **using** *assms* **unfolding** *ordLess-def* **by** *blast*
      moreover have *Well-order* (*wrank r* (*r*‘‘{*b*})) **using** *lem-wrank-cardord* **by**
(*metis card-order-on-well-order-on*)
      ultimately show *wrank r* (*r*‘‘{*b*}) ≤*o* *α* **unfolding** *Mwn-def* **by** *simp*
    qed
    have ∀ *n*. *wrank r* ((*r*^^*n*)‘‘{*a*}) ≤*o* *α*
    proof
      fix *n0*
      show *wrank r* ((*r*^^*n0*)‘‘{*a*}) ≤*o* *α*
      proof (*induct n0*)
        have |{*a*}| ≤*o* *ω-ord* **using** *card-of-Well-order finite.emptyI*
          infinite-iff-natLeq-ordLeq natLeq-Well-order ordLeq-total **by** *blast*
        then have |(*r*^^*0*)‘‘{*a*}| ≤*o* *ω-ord* **by** *simp*
        then show *wrank r* ((*r*^^*0*)‘‘{*a*}) ≤*o* *α*
          using *a1 lem-wrank-ub*[*of r* (*r*^^*0*)‘‘{*a*}] **by** (*metis ordLeq-transitive*)
      next
        fix *n*
        assume *e1*: *wrank r* ((*r*^^*n*)‘‘{*a*}) ≤*o* *α*
        obtain *K* **where** *e2*: *K* = (*r*^^*n*)‘‘{*a*} **by** *blast*
        obtain *S′* **where** *e3*: *S′* = ((λ *b*. *r*‘‘{*b*}) ‘ *K*) **by** *blast*
        have *wrank r K* ≤*o* *α* **using** *e1 e2* **by** *blast*
        moreover have ∀ *A*∈*S′*. *wrank r A* ≤*o* *α*
        proof
          fix *A*
          assume *A* ∈ *S′*
          then obtain *b* **where** *b* ∈ *K* ∧ *A* = *r*‘‘{*b*} **using** *e3* **by** *blast*

moreover then have $b \in B$ **using** *c2 e2 rtrancl-power* **by** *blast*
ultimately show *wrank r A* $\leq o$ α **using** *c4* **by** *blast*
qed
ultimately have *e4*: *wrank r* $(\bigcup S')$ $\leq o$ α
using *a1 e3 lem-wrank-fw*[*of* α *r K*] **by** *fastforce*
have $(r \hat{\ } \hat{\ } (Suc\ n))$ "$\{a\}$ = r"K **using** *e2* **by** *force*
moreover have r"$K = \bigcup S'$ **using** *e3* **by** *blast*
ultimately have $(r \hat{\ } \hat{\ } (Suc\ n))$ "$\{a\} = \bigcup S'$ **using** *e2* **by** *blast*
then show *wrank r* $((r \hat{\ } \hat{\ } (Suc\ n))$ "$\{a\})$ $\leq o$ α **using** *e4* **by** *simp*
qed
qed
then have $\forall A \in S.$ *wrank r A* $\leq o$ α **using** *c3* **by** *blast*
moreover have $B = \bigcup S$ **using** *c2 c3 rtrancl-power*
apply (*simp*)
by *blast*
moreover have $|S| \leq o$ α
proof −
have $|S| \leq o$ $|UNIV::nat\ set|$ **using** *c3* **by** *simp*
moreover have $|UNIV::nat\ set| =o$ ω*-ord* **using** *card-of-nat* **by** *blast*
ultimately show *?thesis* **using** *a1 ordLeq-ordIso-trans ordLeq-transitive* **by** *blast*
qed
ultimately have *wrank r B* $\leq o$ α **using** *a1 lem-wrank-un-bnd*[*of S r* α] **by** *blast*
moreover obtain *B0* **where** $B0 \in$ *wbase r B* $\wedge$ $|B0| =o$ *wrank r B* **using** *lem-wrank-uset*[*of r B*] **by** *blast*
ultimately have *c5*: $B \subseteq$ *dncl r B0* $\wedge$ $|B0| \leq o$ α
unfolding *wbase-def w-dncl-def* **using** *ordIso-ordLeq-trans* **by** *blast*
have $((\{\}::'U\ rel) <o \|r\|)$ **using** *a2* **by** (*metis ordLeq-ordLess-trans ordLess-Well-order-simp ozero-def ozero-ordLeq*)
then have *c6*: *CCR r* **using** *lem-Rcc-eq1-31* **by** *blast*
obtain *B1* **where** *c7*: $B1 = B0 \cap$ *Field r* **by** *blast*
then have *c8*: $|B1| \leq o$ α **using** *c5* **by** (*meson IntE card-of-mono1 ordLeq-transitive subsetI*)
have $B1 \subseteq$ *Field r* **using** *c7* **by** *blast*
moreover have $\forall x \in$ *Field r*. $\exists y \in B1$. $(x, y) \in r \hat{\ } *$
proof
fix x
assume *e1*: $x \in$ *Field r*
then obtain y **where** $(x,y) \in r \hat{\ } * \wedge (a,y) \in r \hat{\ } *$ **using** *c6 b1* **unfolding** *CCR-def* **by** *blast*
moreover then have $y \in B$ **unfolding** *c2* **by** *blast*
moreover then obtain y' **where** $y' \in B0 \wedge (y,y') \in r \hat{\ } *$ **using** *c5* **unfolding** *dncl-def* **by** *blast*
ultimately have $y' \in B0 \wedge (x,y') \in r \hat{\ } *$ **by** *force*
moreover then have $x = y' \vee y' \in$ *Field r* **using** *lem-rtr-field*[*of x y'*] **by** *blast*
ultimately have $y' \in B1 \wedge (x,y') \in r \hat{\ } *$ **using** *e1 c7* **by** *blast*
then show $\exists y \in B1$. $(x, y) \in r \hat{\ } *$ **by** *blast*

qed
ultimately have *B1* ∈ *SCF r* **unfolding** *SCF-def* **by** *blast*
then have *scf r* ≤*o* |*B1*| **using** *lem-scf-uset-mem-bnd* **by** *blast*
then have *scf r* ≤*o* α **using** *c8* **by** (*metis ordLeq-transitive*)
moreover have ‖*r*‖ =*o scf r* **using** *c6 lem-scf-ccr-scf-rcc-eq*[*of r*] **by** *blast*
ultimately show *False* **using** *a2* **by** (*metis not-ordLeq-ordLess ordIso-ordLeq-trans*)
qed
then show ∃ *b* ∈ *Mwn r* α. (*a*,*b*) ∈ *r*^∗ **by** *blast*
qed

lemma *lem-wnb-neib3*:
fixes *r*::′*U rel*
assumes *a1*: ω-*ord* <*o* ‖*r*‖ **and** *a2*: *stable* ‖*r*‖
shows ∀ *a* ∈ *Field r*. ∃ *b* ∈ *Mwnm r*. (*a*,*b*) ∈ *r*^∗
proof
fix *a*
assume *b1*: *a* ∈ *Field r*
have ¬ (∃ *b* ∈ *Mwnm r*. (*a*,*b*) ∈ *r*^∗) ⟶ *False*
proof
assume *c1*: ¬ (∃ *b* ∈ *Mwnm r*. (*a*,*b*) ∈ *r*^∗)
obtain *B* **where** *c2*: *B* = (*r*^∗)‘‘{*a*} **by** *blast*
obtain *S* **where** *c3*: *S* = ((λ *n*. (*r*^^*n*)‘‘{*a*}) ‘ (*UNIV*::*nat set*)) **by** *blast*
have *c4*: ∀ *b* ∈ *B*. *wrank r* (*r* ‘‘{*b*}) <*o* ‖*r*‖
proof
fix *b*
assume *d1*: *b* ∈ *B*
then obtain *k* **where** *b* ∈ (*r*^^*k*)‘‘{*a*} **using** *c2 rtrancl-power* **by** *blast*
moreover have ∀ *n*. (*r*^^*n*) ‘‘ {*a*} ⊆ *Field r*
proof
fix *n*
show (*r*^^*n*) ‘‘ {*a*} ⊆ *Field r* **using** *b1*
by (*induct n*, *force*, *meson FieldI2 Image-singleton-iff relpow-Suc-E subsetI*)
qed
ultimately have *b* ∈ *Field r* **by** *blast*
moreover have *b* ∉ *Mwnm r* **using** *d1 c1 c2* **by** *blast*
ultimately have *b* ∈ *Field r* − *Mwnm r* **by** *blast*
moreover have *Well-order* (*wrank r* (*r*‘‘{*b*})) **using** *lem-wrank-cardord* **by**
(*metis card-order-on-well-order-on*)
moreover have *Well-order* ‖*r*‖ **using** *lem-rcc-cardord* **unfolding** *card-order-on-def*
by *blast*
ultimately show *wrank r* (*r*‘‘{*b*}) <*o* ‖*r*‖ **unfolding** *Mwnm-def* **by** *simp*
qed
have ∀ *n*. *wrank r* ((*r*^^*n*)‘‘{*a*}) <*o* ‖*r*‖
proof
fix *n0*
show *wrank r* ((*r*^^*n0*)‘‘{*a*}) <*o* ‖*r*‖
proof (*induct n0*)
have |{*a*}| ≤*o* ω-*ord* **using** *card-of-Well-order finite.emptyI*
infinite-iff-natLeq-ordLeq natLeq-Well-order ordLeq-total **by** *blast*

then have $|(r^{\frown\frown}0)``\{a\}| \le o$ *ω-ord* **by** *simp*
then show *wrank r* $((r^{\frown\frown}0)``\{a\}) < o \ \|r\|$
using *a1 lem-wrank-ub*[*of r* $(r^{\frown\frown}0)``\{a\}$] **by** (*metis ordLeq-ordLess-trans*)
next
fix *n*
assume *e1*: *wrank r* $((r^{\frown\frown}n)``\{a\}) < o \ \|r\|$
obtain *K* **where** *e2*: $K = (r^{\frown\frown}n)``\{a\}$ **by** *blast*
obtain S' **where** *e3*: $S' = ((\lambda\ b.\ r``\{b\})\ `\ K)$ **by** *blast*
have *wrank r K* $< o \ \|r\|$ **using** *e1 e2* **by** *blast*
moreover have $\forall A \in S'$. *wrank r A* $< o \ \|r\|$
proof
fix *A*
assume $A \in S'$
then obtain *b* **where** $b \in K \wedge A = r``\{b\}$ **using** *e3* **by** *blast*
moreover then have $b \in B$ **using** *c2 e2 rtrancl-power* **by** *blast*
ultimately show *wrank r A* $< o \ \|r\|$ **using** *c4* **by** *blast*
qed
moreover have *ω-ord* $\le o \ \|r\|$ **using** *a1* **by** (*metis ordLess-imp-ordLeq*)
ultimately have *e4*: *wrank r* $(\bigcup\ S') < o \ \|r\|$
using *e3 a2 lem-wrank-fw-stab*[*of* $\|r\|$ *r K*] **by** *fastforce*
have $(r^{\frown\frown}(Suc\ n))``\{a\} = r``K$ **using** *e2* **by** *force*
moreover have $r``K = \bigcup\ S'$ **using** *e3* **by** *blast*
ultimately have $(r^{\frown\frown}(Suc\ n))``\{a\} = \bigcup\ S'$ **using** *e2* **by** *blast*
then show *wrank r* $((r^{\frown\frown}(Suc\ n))\ ``\ \{a\}) < o \ \|r\|$ **using** *e4* **by** *simp*
qed
qed
then have $\forall A \in S$. *wrank r A* $< o \ \|r\|$ **using** *c3* **by** *blast*
moreover have $B = \bigcup\ S$ **using** *c2 c3 rtrancl-power*
apply (*simp*)
by *blast*
moreover have $|S| < o \ \|r\|$
proof −
have $|S| \le o \ |UNIV::nat\ set|$ **using** *c3* **by** *simp*
moreover have $|UNIV::nat\ set| = o$ *ω-ord* **using** *card-of-nat* **by** *blast*
ultimately show *?thesis* **using** *a1 ordLeq-ordIso-trans ordLeq-ordLess-trans* **by** *blast*
qed
ultimately have *wrank r B* $< o \ \|r\|$ **using** *a2 lem-wrank-un-bnd-stab*[*of S r* $\|r\|$] **by** *blast*
moreover obtain *B0* **where** $B0 \in$ *wbase r B* $\wedge |B0| = o$ *wrank r B* **using** *lem-wrank-uset*[*of r B*] **by** *blast*
ultimately have *c5*: $B \subseteq$ *dncl r B0* $\wedge |B0| < o \ \|r\|$
unfolding *wbase-def w-dncl-def*
by (*metis* (*no-types*, *lifting*) *mem-Collect-eq ordIso-ordLess-trans subsetI subset-trans*)
have $((\{\}::'U\ rel) < o \ \|r\|)$ **using** *a1* **by** (*metis ordLeq-ordLess-trans ordLess-Well-order-simp ozero-def ozero-ordLeq*)
then have *c6*: *CCR r* **using** *lem-Rcc-eq1-31* **by** *blast*
obtain *B1* **where** *c7*: $B1 = B0 \cap$ *Field r* **by** *blast*

then have *c8*: |*B1*| <*o* ‖*r*‖ **using** *c5* **by** (*meson IntE card-of-mono1 ordLeq-ordLess-trans subsetI*)
have *B1* ⊆ *Field r* **using** *c7* **by** *blast*
moreover have ∀ *x* ∈ *Field r*. ∃ *y* ∈ *B1*. (*x*, *y*) ∈ *r*^*
proof
fix *x*
assume *e1*: *x* ∈ *Field r*
then obtain *y* **where** (*x*,*y*) ∈ *r*^* ∧ (*a*,*y*) ∈ *r*^* **using** *c6 b1* **unfolding** *CCR-def* **by** *blast*
moreover then have *y* ∈ *B* **unfolding** *c2* **by** *blast*
moreover then obtain *y'* **where** *y'* ∈ *B0* ∧ (*y*,*y'*) ∈ *r*^* **using** *c5* **unfolding** *dncl-def* **by** *blast*
ultimately have *y'* ∈ *B0* ∧ (*x*,*y'*) ∈ *r*^* **by** *force*
moreover then have *x* = *y'* ∨ *y'* ∈ *Field r* **using** *lem-rtr-field*[*of x y'*] **by** *blast*
ultimately have *y'* ∈ *B1* ∧ (*x*,*y'*) ∈ *r*^* **using** *e1 c7* **by** *blast*
then show ∃ *y*∈*B1*. (*x*, *y*) ∈ *r*^* **by** *blast*
qed
ultimately have *B1* ∈ *SCF r* **unfolding** *SCF-def* **by** *blast*
then have *scf r* ≤*o* |*B1*| **using** *lem-scf-uset-mem-bnd* **by** *blast*
then have *scf r* <*o* ‖*r*‖ **using** *c8* **by** (*metis ordLeq-ordLess-trans*)
moreover have ‖*r*‖ =*o scf r* **using** *c6 lem-scf-ccr-scf-rcc-eq*[*of r*] **by** *blast*
ultimately show *False* **by** (*metis not-ordLess-ordIso ordIso-symmetric*)
qed
then show ∃ *b* ∈ *Mwnm r*. (*a*,*b*) ∈ *r*^* **by** *blast*
qed

lemma *lem-scfgew-ncl*: *ω-ord* ≤*o scf r* ⟹ ¬ *Conelike r*
proof (*cases CCR r*)
assume *ω-ord* ≤*o scf r* **and** *CCR r*
then have *ω-ord* ≤*o* ‖*r*‖ **using** *lem-scf-ccr-scf-rcc-eq*[*of r*]
by (*metis ordIso-iff-ordLeq ordLeq-transitive*)
then have ∀ *a*. ¬ (‖*r*‖ ≤*o* |{*a*}|) **using** *finite-iff-ordLess-natLeq ordLess-ordLeq-trans*[*of - ω-ord* ‖*r*‖] *not-ordLess-ordLeq*[*of -* ‖*r*‖] **by** *blast*
then show ¬ *Conelike r* **using** *lem-Rcc-eq2-12*[*of r*] **by** *metis*
next
assume *ω-ord* ≤*o scf r* **and** ¬ *CCR r*
then show ¬ *Conelike r* **unfolding** *CCR-def Conelike-def* **by** *fastforce*
qed

lemma *lem-wnb-P-ncl-reg-grw*:
fixes *r*::'*U rel*
assumes *a1*: *CCR r* **and** *a2*: *ω-ord* <*o scf r* **and** *a3*: *regularCard* (*scf r*)
shows ∃ *P* ∈ *SCF r*. (∀ *α*::'*U rel*. *α* <*o scf r* ⟶ (∀ *a* ∈ *P*. *α* <*o wrank r* (*r*``{*a*})))
proof −
have ¬ *Conelike r* **using** *a2 lem-scfgew-ncl ordLess-imp-ordLeq* **by** *blast*
moreover obtain *P* **where** *b1*: *P* = { *a* ∈ *Field r*. *scf r* ≤*o wrank r* (*r* ``{*a*}) } **by** *blast*

ultimately have *stable* (*scf r*)
using *a1 a3 lem-scf-ccr-finscf-cl lem-scf-cardord regularCard-stable* **by** *blast*
then have *stable* $\|r\|$ **using** *a1 lem-scf-ccr-scf-rcc-eq stable-ordIso1* **by** *blast*
moreover have *ω-ord <o* $\|r\|$ **using** *a1 a2 lem-scf-ccr-scf-rcc-eq*[*of r*]
by (*metis ordIso-iff-ordLeq ordLess-ordLeq-trans*)
ultimately have ∀ *a*∈*Field r*. ∃ *b* ∈ *Mwnm r*. (*a*, *b*) ∈ *r*^∗ **using** *lem-wnb-neib3* **by** *blast*
moreover have *Mwnm r* ⊆ *P* **unfolding** *b1 Mwnm-def* **using** *a1 lem-scf-ccr-scf-rcc-eq*[*of r*]
by (*clarsimp*, *metis ordIso-ordLeq-trans ordIso-symmetric*)
moreover have *P* ⊆ *Field r* **using** *b1* **by** *blast*
ultimately have *P* ∈ *SCF r* **unfolding** *SCF-def* **by** *blast*
moreover have ∀ *α*::′*U rel*. *α <o scf r* ⟶ (∀ *a* ∈ *P*. *α <o wrank r* (*r*‘‘{*a*}))
using *b1 ordLess-ordLeq-trans* **by** *blast*
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-wnb-P-ncl-nreg*:
fixes *r*::′*U rel*
assumes *a1*: *CCR r* **and** *a2*: *ω-ord* ≤*o scf r* **and** *a3*: ¬ *regularCard* (*scf r*)
shows ∃ *Ps*::′*U set set*. *Ps* ⊆ *SCF r* ∧ |*Ps*| *<o scf r*
∧ (∀ *α*::′*U rel*. *α <o scf r* ⟶ (∃ *P* ∈ *Ps*. ∀ *a* ∈ *P*. *α <o wrank r* (*r*‘‘{*a*})))
proof −
have ¬ *Conelike r* **using** *a2 lem-scfgew-ncl* **by** *blast*
then have *b1*: ¬ *finite* (*Field* (*scf r*)) **using** *a1 lem-scf-ccr-finscf-cl* **by** *blast*
have *b2*: ⋀ *α*::′*U rel*. *ω-ord* ≤*o α* ⟹ *α <o scf r* ⟹ { *a* ∈ *Field r*. *α <o wrank r* (*r* ‘‘{*a*}) } ∈ *SCF r*
proof −
fix *α*::′*U rel*
assume *c1*: *ω-ord* ≤*o α* **and** *c2*: *α <o scf r*
have *α <o* $\|r\|$ **using** *a1 c2 lem-scf-ccr-scf-rcc-eq ordIso-iff-ordLeq ordLess-ordLeq-trans* **by** *blast*
then have ∀ *a* ∈ *Field r*. ∃ *b* ∈ *Mwn r α*. (*a*,*b*) ∈ *r*^∗ **using** *c1 lem-wnb-neib* **by** *blast*
then show { *a* ∈ *Field r*. *α <o wrank r* (*r* ‘‘{*a*}) } ∈ *SCF r* **unfolding** *SCF-def Mwn-def* **by** *blast*
qed
have *b3*: *ω-ord <o scf r*
proof −
have *c1*: ¬ *stable* (*scf r*) **using** *b1 a3 lem-scf-cardord stable-regularCard* **by** *blast*
have *ω-ord* ≤*o scf r* **using** *b1 lem-inford-ge-w lem-scf-cardord* **unfolding** *card-order-on-def* **by** *blast*
moreover have *ω-ord* =*o scf r* ⟶ *False* **using** *c1 stable-ordIso stable-natLeq* **by** *blast*
ultimately show *?thesis* **using** *ordLeq-iff-ordLess-or-ordIso* **by** *blast*
qed
obtain *S*::′*U rel set* **where** *b4*: |*S*| *<o scf r* **and** *b5*: ∀ *α*∈*S*. *α <o scf r*

and *b6*: ∀ *α*::(*'U rel*). *α* <*o scf r* ⟶ (∃ *β*∈*S*. *α* ≤*o* *β*)
using *b1 a3 lem-scf-cardord*[*of r*] *lem-card-nreg-inf-osetlm*[*of scf r*] **by** *blast*
obtain *S1*::*'U rel set* **where** *b7*: *S1* = { *α* ∈ *S*. *ω-ord* ≤*o* *α* } **by** *blast*
obtain *f*::*'U rel* ⇒ *'U set* **where** *b8*: *f* = (*λ α*. { *a* ∈ *Field r*. *α* <*o wrank r* (*r* ‘‘{*a*}) }) **by** *blast*
obtain *Ps*::*'U set set* **where** *b9*: *Ps* = *f* ‘ *S1* **by** *blast*
have *Ps* ⊆ *SCF r* **using** *b2 b5 b7 b8 b9* **by** *blast*
moreover have |*Ps*| <*o scf r*
proof −
have |*Ps*| ≤*o* |*S1*| **using** *b9* **by** *simp*
moreover have |*S1*| ≤*o* |*S*| **using** *b7 card-of-mono1*[*of S1 S*] **by** *blast*
ultimately show *?thesis* **using** *b4 ordLeq-ordLess-trans ordLeq-transitive* **by** *blast*
qed
moreover have ∀ *α*::*'U rel*. *α* <*o scf r* ⟶ (∃ *P* ∈ *Ps*. ∀ *a* ∈ *P*. *α* <*o wrank r* (*r*‘‘{*a*}))
proof (*intro allI impI*)
fix *α*::*'U rel*
assume *c1*: *α* <*o scf r*
have ∃ *αm*::(*'U rel*). *ω-ord* ≤*o* *αm* ∧ *α* ≤*o* *αm* ∧ *αm* <*o scf r*
proof (*cases ω-ord* ≤*o* *α*)
assume *ω-ord* ≤*o* *α*
then show *?thesis* **using** *c1 ordLeq-reflexive* **unfolding** *ordLeq-def* **by** *blast*
next
assume ¬ (*ω-ord* ≤*o* *α*)
then have *d1*: *α* ≤*o* *ω-ord* **using** *c1 natLeq-Well-order ordLess-Well-order-simp*

ordLess-imp-ordLeq ordLess-or-ordLeq **by** *blast*
have *isLimOrd* (*scf r*)
using *b1 lem-scf-cardord*[*of r*] *card-order-infinite-isLimOrd*[*of scf r*] **by** *blast*
then obtain *αm*::*'U rel* **where** *ω-ord* ≤*o* *αm* ∧ *αm* <*o scf r*
using *b3 lem-lmord-prec*[*of ω-ord scf r*] *ordLess-imp-ordLeq* **by** *blast*
then show *?thesis* **using** *d1 ordLeq-transitive* **by** *blast*
qed
then obtain *αm*::*'U rel* **where** *ω-ord* ≤*o* *αm* ∧ *α* ≤*o* *αm* ∧ *αm* <*o scf r* **by** *blast*
moreover then obtain *β*::*'U rel* **where** *β* ∈ *S* ∧ *αm* ≤*o* *β* **using** *b6* **by** *blast*
ultimately have *c2*: *α* ≤*o* *β* **and** *c3*: *β* ∈ *S1* **using** *b7 ordLeq-transitive* **by** *blast+*
obtain *P* **where** *c4*: *P* = *f β* **by** *blast*
then have *P* ∈ *Ps* **using** *c3 b9* **by** *blast*
moreover have ∀ *a* ∈ *P*. *α* <*o wrank r* (*r*‘‘{*a*}) **using** *c2 c4 b8 ordLeq-ordLess-trans* **by** *blast*
ultimately show ∃ *P* ∈ *Ps*. ∀ *a* ∈ *P*. *α* <*o wrank r* (*r*‘‘{*a*}) **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Wf-ext-arc*:

fixes r::$'U$ *rel* **and** Ps::$'U$ *set set* **and** f::$'U$ *rel* ⇒ $'U$ *set* **and** α::$'U$ *rel* **and** a::$'U$
assumes *a1*: *scf r =o |Field r|* **and** *a2*: $f \in \mathcal{N}$ *r Ps*
  and *a3*: $\forall\gamma$::$'U$ *rel*. γ *<o scf r* ⟶ ($\forall a \in P$. γ *<o wrank r* (*r*‘‘{*a*}))
  and *a4*: ω*-ord* $\leq o$ α **and** *a5*: $a \in f\ \alpha \cap P$
shows ⋀ β. α *<o* β ∧ β *<o |Field r|* ∧ (β = {} ∨ *isSuccOrd* β) ⟹ (*r*‘‘{*a*} ∩ ($\mathcal{W}$ *r f* β) ≠ {})
proof (*elim conjE*)
 fix β::$'U$ *rel*
 assume *b1*: α *<o* β **and** *b2*: β *<o |Field r|* **and** *b3*: β = {} ∨ *isSuccOrd* β
 have *b4*: ω*-ord* $\leq o$ β **using** *b1 a4* **by** (*metis ordLeq-ordLess-trans ordLess-imp-ordLeq*)
 have *b5*: $a \in$ ($\mathfrak{L}$ *f* β) ∩ *P* **using** *b1 a5* **unfolding** $\mathfrak{L}$*-def* **by** *blast*
 show *r*‘‘{*a*} ∩ ($\mathcal{W}$ *r f* β) ≠ {}
 proof −
  have *r*‘‘{*a*} ⊆ *w-dncl r* ($\mathfrak{L}$ *f* β) ∨ (*r*‘‘{*a*} ∩ ($\mathcal{W}$ *r f* β)≠{})
   using *b2 b3 b5 a2* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*4-def* **using** *ordLess-imp-ordLeq* **by** *blast*
  moreover have *r*‘‘{*a*} ⊆ *w-dncl r* ($\mathfrak{L}$ *f* β) ⟶ *False*
  proof
   assume *r*‘‘{*a*} ⊆ *w-dncl r* ($\mathfrak{L}$ *f* β)
   then have $\mathfrak{L}$ *f* β ∈ *wbase r* (*r*‘‘{*a*}) **unfolding** *wbase-def* **by** *blast*
   then have *d1*: *wrank r* (*r*‘‘{*a*}) $\leq o$ |$\mathfrak{L}$ *f* β| **using** *lem-wrank-uset-mem-bnd* **by** *blast*
    have $\mathfrak{L}$ *f* β ⊆ *f* β **using** *b2 a2* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*1-def* $\mathfrak{L}$*-def* **using** *ordLess-imp-ordLeq* **by** *blast*
   then have |$\mathfrak{L}$ *f* β| $\leq o$ |*f* β| **by** *simp*
   moreover have |*f* β| $\leq o$ β **using** *a2 b2 b4* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*7-def* **using** *ordLess-imp-ordLeq* **by** *blast*
   ultimately have *wrank r* (*r*‘‘{*a*}) $\leq o$ β **using** *d1 ordLeq-transitive* **by** *blast*
   moreover have β *<o wrank r* (*r* ‘‘ {*a*}) **using** *b2 b5 a1 a3* **by** (*meson IntE ordIso-symmetric ordLess-ordIso-trans*)
   ultimately show *False* **by** (*metis not-ordLeq-ordLess*)
  qed
  ultimately show *?thesis* **by** *blast*
 qed
qed

lemma *lem-Wf-esc-pth*:
fixes r::$'U$ *rel* **and** Ps::$'U$ *set set* **and** f::$'U$ *rel* ⇒ $'U$ *set* **and** α::$'U$ *rel*
assumes *a1*: *Refl r* ∧ ¬ *finite r* **and** *a2*: $f \in \mathcal{N}$ *r Ps*
  and *a3*: ω*-ord* $\leq o$ |$\mathfrak{L}$ *f* α| **and** *a4*: α *<o |Field r|*
shows ⋀ *F*. *F* ∈ *SCF* (*Restr r* (*f* α)) ⟹
   ∀ *a* ∈ $\mathcal{W}$ *r f* α. ∃ *b* ∈ (*F* ∩ ($\mathcal{W}$ *r f* α)). (*a*,*b*) ∈ (*Restr r* ($\mathcal{W}$ *r f* α))^*
proof −
 fix *F*
 assume *a5*: *F* ∈ *SCF* (*Restr r* (*f* α))
 show ∀ *a* ∈ ($\mathcal{W}$ *r f* α). ∃ *b* ∈ (*F* ∩ ($\mathcal{W}$ *r f* α)). (*a*,*b*) ∈ (*Restr r* ($\mathcal{W}$ *r f* α))^*
 proof
  fix *a*
  assume *b1*: *a* ∈ $\mathcal{W}$ *r f* α

have *b2*: *SF r* = {*A*. *A* ⊆ *Field r*} **using** *a1* **unfolding** *SF-def refl-on-def Field-def* **by** *fast*
moreover have *f α* ⊆ *Field r*
using *a2 a4* **unfolding** *𝒩-def 𝒩5-def SF-def Field-def* **using** *ordLess-imp-ordLeq* **by** *blast*
ultimately have ∀ *x* ∈ *f α*. ∃ *y* ∈ *f α* ∩ *F*. (*x*, *y*) ∈ (*Restr r* (*f α*))^*
using *a5* **unfolding** *SF-def SCF-def* **by** *blast*
then have *b3*: ∀ *x* ∈ *𝒬 r f α*. ∃ *y* ∈ (*f α* ∩ *F* ∩ *𝒬 r f α*). (*x*, *y*) ∈ (*Restr r* (*𝒬 r f α*))^*
using *lem-der-qinv3*[*of* (*f α*) ∩ *F f α r*] **by** *blast*
have *b4*: *Restr r* (*𝒬 r f α*) ∈ 𝔘 (*Restr r* (*𝒲 r f α*))
using *a1 a2 a3 a4 lem-der-inf-qw-restr-uset*[*of r f Ps α*] **by** *blast*
moreover have *a* ∈ *Field* (*Restr r* (*𝒲 r f α*))
proof −
have *𝒲 r f α* ⊆ *Field r* **using** *a2 a4 lem-qw-range ordLess-imp-ordLeq* **by** *blast*
then have *𝒲 r f α* ∈ *SF r* **using** *b2* **by** *blast*
then show *?thesis* **using** *b1* **unfolding** *SF-def* **by** *blast*
qed
ultimately obtain *a′* **where** *b5*: *a′* ∈ *𝒬 r f α* ∧ (*a*, *a′*) ∈ (*Restr r* (*𝒲 r f α*))^*
unfolding *𝔘-def Field-def* **by** *blast*
then obtain *b* **where** *b6*: *b* ∈ (*f α* ∩ *F* ∩ *𝒬 r f α*) ∧ (*a′*, *b*) ∈ (*Restr r* (*𝒬 r f α*))^* **using** *b3* **by** *blast*
then have *b* ∈ (*F* ∩ (*𝒲 r f α*)) ∧ (*a*, *b*) ∈ (*Restr r* (*𝒲 r f α*))^*
using *b5 lem-QS-subs-WS*[*of r f α*] *rtrancl-mono*[*of Restr r* (*𝒬 r f α*) *Restr r* (*𝒲 r f α*)] **by** *force*
then show ∃ *b* ∈ (*F* ∩ (*𝒲 r f α*)). (*a*,*b*) ∈ (*Restr r* (*𝒲 r f α*))^* **by** *blast*
qed
qed

lemma *lem-Nf-lewfbnd*:
assumes *a1*: *f* ∈ *𝒩 r Ps* **and** *a2*: *α* ≤*o* |*Field r*| **and** *a3*: *ω-ord* ≤*o* |𝔏 *f α*|
shows *ω-ord* ≤*o* *α*
proof −
have 𝔏 *f α* ⊆ *f α* **using** *a1 a2* **unfolding** *𝒩-def 𝒩1-def 𝔏-def* **using** *ordLess-imp-ordLeq* **by** *blast*
then have *ω-ord* ≤*o* |*f α*| **using** *a3* **by** (*metis card-of-mono1 ordLeq-transitive*)
moreover have *α* <*o* *ω-ord* ⟶ |*f α*| <*o* *ω-ord* **using** *a1 a2* **unfolding** *𝒩-def 𝒩7-def* **by** *blast*
ultimately show *?thesis* **using** *a2 not-ordLess-ordLeq* **by** *force*
qed

lemma *lem-regcard-iso*: *κ* =*o* *κ′* ⟹ *regularCard κ′* ⟹ *regularCard κ*
proof −
assume *a1*: *κ* =*o* *κ′* **and** *a2*: *regularCard κ′*
then obtain *f* **where** *b1*: *iso κ κ′ f* **unfolding** *ordIso-def* **by** *blast*
have ∀ *K*. *K* ⊆ *Field κ* ∧ *cofinal K κ* ⟶ |*K*| =*o* *κ*
proof (*intro allI impI*)

fix *K*
assume *c1*: $K \subseteq$ *Field* $\kappa \wedge$ *cofinal* K κ
moreover then obtain K' **where** *c2*: $K' = f$ ‘ K **by** *blast*
ultimately have $K' \subseteq$ *Field* κ' **using** *b1* **unfolding** *iso-def bij-betw-def* **by** *blast*
moreover have *cofinal* K' κ'
proof −
have $\forall\, a' \in$*Field* κ'. $\exists\, b' \in K'$. $a' \neq b' \wedge (a', b') \in \kappa'$
proof
fix a'
assume $a' \in$ *Field* κ'
then obtain a **where** *e1*: $a' = f\ a \wedge a \in$ *Field* κ **using** *b1* **unfolding** *iso-def bij-betw-def* **by** *blast*
then obtain b **where** *e2*: $b \in K \wedge a \neq b \wedge (a, b) \in \kappa$ **using** *c1* **unfolding** *cofinal-def* **by** *blast*
then have $f\ b \in K'$ **using** *c2* **by** *blast*
moreover have $a' \neq f\ b$ **using** *e1 e2 c1 b1* **unfolding** *iso-def bij-betw-def inj-on-def* **by** *blast*
moreover have $(a', f\ b) \in \kappa'$
proof −
have $(a,b) \in \kappa$ **using** *e2* **by** *blast*
moreover have *embed* κ κ' f **using** *b1* **unfolding** *iso-def* **by** *blast*
ultimately have $(f\ a, f\ b) \in \kappa'$ **using** *compat-def embed-compat* **by** *metis*
then show *?thesis* **using** *e1* **by** *blast*
qed
ultimately show $\exists\, b' \in K'$. $a' \neq b' \wedge (a', b') \in \kappa'$ **by** *blast*
qed
then show *?thesis* **unfolding** *cofinal-def* **by** *blast*
qed
ultimately have *c3*: $|K'|$ $=o$ κ' **using** *a2* **unfolding** *regularCard-def* **by** *blast*
have *inj-on f K* **using** *c1 b1* **unfolding** *iso-def bij-betw-def inj-on-def* **by** *blast*
then have *bij-betw f K* K' **using** *c2* **unfolding** *bij-betw-def* **by** *blast*
then have $|K|$ $=o$ $|K'|$ **using** *card-of-ordIsoI* **by** *blast*
then have $|K|$ $=o$ κ' **using** *c3 ordIso-transitive* **by** *blast*
then show $|K|$ $=o$ κ **using** *a1 ordIso-symmetric ordIso-transitive* **by** *blast*
qed
then show *regularCard* κ **unfolding** *regularCard-def* **by** *blast*
qed

lemma *lem-cardsuc-inf-gwreg*: $\neg$ *finite* $A \Longrightarrow \kappa$ $=o$ *cardSuc* $|A| \Longrightarrow \omega$*-ord* $<o$ κ $\wedge$ *regularCard* κ
proof −
assume *a1*: $\neg$ *finite* A **and** *a2*: κ $=o$ *cardSuc* $|A|$
moreover then have *regularCard* (*cardSuc* $|A|$) **using** *infinite-cardSuc-regularCard* **by** *force*
ultimately have *a3*: *regularCard* κ **using** *lem-regcard-iso ordIso-transitive* **by** *blast*
have $|A|$ $<o$ *cardSuc* $|A|$ **by** *simp*
then have $|A|$ $<o$ κ **using** *a2 ordIso-symmetric ordLess-ordIso-trans* **by** *blast*

moreover have *ω-ord* ≤*o* |*A*| **using** *a1 infinite-iff-natLeq-ordLeq* **by** *blast*
ultimately have *ω-ord* <*o* *κ* **using** *ordLeq-ordLess-trans* **by** *blast*
then show *?thesis* **using** *a3* **by** *blast*
qed

lemma *lem-ccr-rcscf-struct*:
fixes *r*::′*U rel*
assumes *a1*: *Refl r* **and** *a2*: *CCR r* **and** *a3*: *ω-ord* <*o scf r* **and** *a4*: *regularCard* (*scf r*)
and *a5*: *scf r* =*o* |*Field r*|
shows ∃ *Ps*. ∃ *f* ∈ 𝒩 *r Ps*.
∀ *α*. *ω-ord* ≤*o* |𝔏 *f α*| ∧ *α* <*o* |*Field r*| ∧ *isSuccOrd α* ⟶
CCR (*Restr r* (𝒲 *r f α*)) ∧ |*Restr r* (𝒲 *r f α*)| <*o* |*Field r*|
∧ (∀ *a* ∈ 𝒲 *r f α*. *wesc-rel r f α a* (*wesc r f α a*))
proof −
obtain *P* **where** *b1*: *P* ∈ *SCF r*
and *b2*: ∀ *α*::′*U rel*. *α* <*o scf r* ⟶ (∀ *a* ∈ *P*. *α* <*o wrank r* (*r*‘‘{*a*}))
using *a2 a3 a4 lem-wnb-P-ncl-reg-grw*[*of r*] **by** *blast*
then obtain *f* **where** *b3*: *f* ∈ 𝒩 *r* {*P*} **using** *a1 a2 lem-Shinf-N-ne*[*of r* {*P*}] **by** *blast*
moreover have ∀ *α*. *ω-ord* ≤*o* |𝔏 *f α*| ∧ *α* <*o* |*Field r*| ∧ (*α* = {} ∨ *isSuccOrd α*) ⟶
CCR (*Restr r* (𝒲 *r f α*)) ∧ |*Restr r* (𝒲 *r f α*)| <*o* |*Field r*|
∧ (∀ *a* ∈ 𝒲 *r f α*. *wesc-rel r f α a* (*wesc r f α a*))
proof (*intro allI impI*)
fix *α*
assume *c1*: *ω-ord* ≤*o* |𝔏 *f α*| ∧ *α* <*o* |*Field r*| ∧ (*α* = {} ∨ *isSuccOrd α*)
then have *c2*: (*f α* ∩ *P*) ∈ *SCF* (*Restr r* (*f α*))
using *b3* **unfolding** 𝒩*-def* 𝒩*8-def* **using** *ordLess-imp-ordLeq* **by** *blast*
have *c3*: ¬ *finite r* **using** *a2 a3 lem-scfgew-ncl lem-scf-ccr-scf-uset*[*of r*]
unfolding 𝔘*-def* **using** *ordLess-imp-ordLeq finite-subset*[*of - r*] **by** *blast*
have *CCR* (*Restr r* (𝒲 *r f α*)) **using** *c1 c3 b3 a1 lem-der-inf-qw-restr-ccr*[*of r f* {*P*} *α*] **by** *blast*
moreover have |*Restr r* (𝒲 *r f α*)| <*o* |*Field r*| **using** *c1 c3 b3 lem-der-inf-qw-restr-card*[*of r f* {*P*} *α*] **by** *blast*
moreover have ∀ *a* ∈ 𝒲 *r f α*. *wesc-rel r f α a* (*wesc r f α a*)
proof
fix *a*
assume *a* ∈ 𝒲 *r f α*
then obtain *b* **where** *d1*: *b* ∈ (*P* ∩ (𝒲 *r f α*)) **and** *d2*: (*a*,*b*) ∈ (*Restr r* (𝒲 *r f α*))^∗
using *c1 c2 c3 b3 a1 lem-Wf-esc-pth*[*of r f* {*P*} *α f α* ∩ *P*] **by** *blast*
moreover then have *b* ∈ (*f α*) ∩ *P* **unfolding** 𝒲*-def* **by** *blast*
moreover have *ω-ord* ≤*o α* **using** *c1 b3 lem-Nf-lewfbnd*[*of f r* {*P*} *α*] *ordLess-imp-ordLeq* **by** *blast*
ultimately have ∀ *β*. *α* <*o β* ∧ *β* <*o* |*Field r*| ∧ (*β* = {} ∨ *isSuccOrd β*) ⟶ *r* ‘‘ {*b*} ∩ 𝒲 *r f β* ≠ {}
using *b2 b3 a5 lem-Wf-ext-arc*[*of r f* {*P*} *P α b*] **by** *blast*
then have *wesc-rel r f α a b* **using** *d1 d2* **unfolding** *wesc-rel-def* **by** *blast*

then have ∃ b. *wesc-rel* $r\ f\ \alpha\ a\ b$ **by** *blast*
then show *wesc-rel* $r\ f\ \alpha\ a$ (*wesc* $r\ f\ \alpha\ a$)
using *someI-ex*[*of* λ b. *wesc-rel* $r\ f\ \alpha\ a\ b$] **unfolding** *wesc-def* **by** *blast*
qed
ultimately show *CCR* (*Restr* r ($\mathcal{W}$ $r\ f\ \alpha$))
∧ |*Restr* r ($\mathcal{W}$ $r\ f\ \alpha$)| <*o* |*Field* r|
∧ (∀ $a \in \mathcal{W}$ $r\ f\ \alpha$. *wesc-rel* $r\ f\ \alpha\ a$ (*wesc* $r\ f\ \alpha\ a$)) **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-oint-infcard-sc-cf*:
fixes $\alpha 0$::*'a rel* **and** κ::*'U rel* **and** S::*'U rel set*
assumes *a1*: *Card-order* κ **and** *a2*: ω*-ord* ≤*o* κ
and *a3*: $S = \{\alpha \in \mathcal{O}$::*'U rel set*. $\alpha 0$ ≤*o* α ∧ *isSuccOrd* α ∧ α <*o* $\kappa\}$
shows ∀ $\alpha \in S$. ∃ $\beta \in S$. α <*o* β
proof
fix α
assume *b1*: $\alpha \in S$
then have α <*o* κ **using** *a3* **by** *blast*
then obtain β **where** *b2*: *sc-ord* α β **using** *lem-sucord-ex* **by** *blast*
obtain β' **where** *b3*: β' = *nord* β **by** *blast*
have *b4*: *isSuccOrd* β **using** *b2* **unfolding** *sc-ord-def* **using** *lem-ordint-sucord* **by** *blast*
moreover have β =*o* β' **using** *b2 b3 lem-nord-l* **unfolding** *sc-ord-def ordLess-def* **by** *blast*
ultimately have *isSuccOrd* β' **using** *lem-osucc-eq* **by** *blast*
moreover have $\beta' \in \mathcal{O}$ **using** *b2 b3 lem-nordO-ls-r* **unfolding** *sc-ord-def* **by** *blast*
moreover have $\alpha 0$ ≤*o* β' **using** *b1 b2 b3 a3* **unfolding** *sc-ord-def*
using *lem-nord-le-r ordLeq-ordLess-trans ordLess-imp-ordLeq* **by** *blast*
moreover have β' <*o* κ
proof −
have β ≤*o* κ **using** *b1 b2 a3* **unfolding** *sc-ord-def* **by** *blast*
moreover have β =*o* κ ⟶ *False*
proof
assume β =*o* κ
then have *isSuccOrd* κ **using** *b4 lem-osucc-eq* **by** *blast*
moreover have *isLimOrd* κ **using** *a1 a2 lem-ge-w-inford* **by** (*metis card-order-infinite-isLimOrd*)
moreover have *Well-order* κ **using** *a1* **unfolding** *card-order-on-def* **by** *blast*
ultimately show *False* **using** *wo-rel.isLimOrd-def* **unfolding** *wo-rel-def* **by** *blast*
qed
ultimately have β <*o* κ **using** *ordLeq-iff-ordLess-or-ordIso* **by** *blast*
then show *?thesis* **using** *b3 lem-nord-ls-l* **by** *blast*
qed
moreover have α <*o* β' **using** *b2 b3 lem-nord-ls-r* **unfolding** *sc-ord-def* **by** *blast*
ultimately have $\beta' \in S$ ∧ α <*o* β' **using** *a3* **by** *blast*

then show $\exists\ \beta \in S.\ \alpha <o\ \beta$ **by** *blast*
qed

lemma *lem-oint-infcard-gew-sc-cfbnd*:
fixes $\alpha 0::'a\ rel$ **and** $\kappa::'U\ rel$ **and** $S::'U\ rel\ set$
assumes *a1*: *Card-order* κ **and** *a2*: ω-*ord* $\leq o\ \kappa$ **and** *a3*: $\alpha 0 <o\ \kappa$ **and** *a4*: $\alpha 0 =o\ \omega$-*ord*
and *a5*: $S = \{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha 0 \leq o\ \alpha \wedge isSuccOrd\ \alpha \wedge \alpha <o\ \kappa\}$
shows $|\{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}| \leq o\ |S|$
$\wedge (\exists\ f.\ (\forall\ \alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha 0 \leq o\ \alpha \wedge \alpha <o\ \kappa \longrightarrow \alpha \leq o\ f\ \alpha \wedge f\ \alpha \in S))$
proof $-$
have $|UNIV::nat\ set| <o\ \kappa$ **using** *a3 a4* **by** (*meson card-of-nat ordIso-ordLess-trans ordIso-symmetric*)
then obtain N **where** $N \subseteq Field\ \kappa \wedge |UNIV::nat\ set| =o\ |N|$
using *internalize-card-of-ordLess*[*of UNIV*::*nat set* κ] **by** *force*
moreover obtain $\alpha 0'::'U\ rel$ **where** $\alpha 0' = |N|$ **by** *blast*
ultimately have *b0*: $\alpha 0' =o\ \omega$-*ord* **using** *card-of-nat ordIso-symmetric ordIso-transitive* **by** *blast*
then have *b0'*: $\alpha 0' <o\ \kappa$ **using** *a3 a4 ordIso-symmetric ordIso-ordLess-trans* **by** *metis*
have *b0''*: $\alpha 0 =o\ \alpha 0'$ **using** *b0 a4 ordIso-symmetric ordIso-transitive* **by** *blast*
obtain *S1* **where** *b1*: $S1 = \{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha 0 \leq o\ \alpha \wedge \alpha <o\ \kappa\}$ **by** *blast*
obtain f **where** $f = (\lambda\alpha::'U\ rel.\ SOME\ \beta.\ sc\text{-}ord\ \alpha\ \beta)$ **by** *blast*
moreover have $\forall\ \alpha \in S1.\ \exists\ \beta.\ sc\text{-}ord\ \alpha\ \beta$ **using** *b1 lem-sucord-ex* **by** *blast*
ultimately have *b2*: $\bigwedge \alpha.\ \alpha \in S1 \Longrightarrow sc\text{-}ord\ \alpha\ (f\ \alpha)$ **using** *someI-ex* **by** *metis*
have *b3*: $(nord \circ f)$ ' $S1 \subseteq S$
proof
fix α
assume $\alpha \in (nord \circ f)$ ' $S1$
then obtain α' **where** *c1*: $\alpha' \in S1 \wedge \alpha = nord\ (f\ \alpha')$ **by** *force*
then have *c2*: *sc-ord* α' $(f\ \alpha')$ **using** *b2* **by** *blast*
then have *c3*: *isSuccOrd* $(f\ \alpha')$ **unfolding** *sc-ord-def* **using** *lem-ordint-sucord* **by** *blast*
moreover have $f\ \alpha' =o\ \alpha$ **using** *c1 c2 lem-nord-l* **unfolding** *sc-ord-def ordLess-def* **by** *blast*
ultimately have *c4*: *isSuccOrd* α **using** *lem-osucc-eq* **by** *blast*
have $\alpha 0 \leq o\ \alpha' \wedge \alpha' <o\ \kappa$ **using** *c1 b1* **by** *blast*
then have *c5*: $\alpha 0 \leq o\ (f\ \alpha') \wedge (f\ \alpha') \leq o\ \kappa$
using *c1 b2* **unfolding** *sc-ord-def* **using** *ordLeq-ordLess-trans ordLess-imp-ordLeq* **by** *blast*
then have *c6*: $\alpha 0 \leq o\ \alpha$ **using** *c1 lem-nord-le-r* **by** *blast*
have *c7*: $\alpha \in \mathcal{O}$ **using** *c1 c2 lem-nordO-ls-r* **unfolding** *sc-ord-def* **by** *blast*
have $(f\ \alpha') =o\ \kappa \longrightarrow False$
proof
assume $(f\ \alpha') =o\ \kappa$
then have *isSuccOrd* κ **using** *c3 lem-osucc-eq* **by** *blast*
moreover have *isLimOrd* κ **using** *a1 a2 lem-ge-w-inford* **by** (*metis card-order-infinite-isLimOrd*)
moreover have *Well-order* κ **using** *a1* **unfolding** *card-order-on-def* **by** *blast*
ultimately show *False* **using** *wo-rel.isLimOrd-def* **unfolding** *wo-rel-def* **by**

```
blast
    qed
    then have f α′ <o κ using c5 using ordLeq-iff-ordLess-or-ordIso by blast
    then have α <o κ using c1 lem-nord-ls-l by blast
    then show α ∈ S using c4 c6 c7 a5 by blast
  qed
  moreover have inj-on (nord ∘ f) S1
  proof −
    have ∀α∈S1. ∀β∈S1. (nord ∘ f) α = (nord ∘ f) β ⟶ α = β
    proof (intro ballI impI)
      fix α β
      assume d1: α ∈ S1 and d2: β ∈ S1 and (nord ∘ f) α = (nord ∘ f) β
      then have nord (f α) = nord (f β) by simp
      moreover have Well-order (f α) ∧ Well-order (f β)
        using d1 d2 b2 unfolding sc-ord-def ordLess-def by blast
      ultimately have d3: f α =o f β using lem-nord-req by blast
      have d4: sc-ord α (f α) ∧ sc-ord β (f β) using d1 d2 b2 by blast
       have Well-order α ∧ Well-order β using d1 d2 b1 unfolding ordLess-def
by blast
      moreover have α <o β ⟶ False
      proof
        assume α <o β
        then have f α ≤o β ∧ β <o f β using d4 unfolding sc-ord-def by blast
         then show False using d3 using not-ordLess-ordIso ordLeq-ordLess-trans
by blast
      qed
      moreover have β <o α ⟶ False
      proof
        assume β <o α
        then have f β ≤o α ∧ α <o f α using d4 unfolding sc-ord-def by blast
         then show False using d3 using not-ordLess-ordIso ordLeq-ordLess-trans
ordIso-symmetric by blast
      qed
      ultimately have α =o β using ordIso-or-ordLess by blast
      then show α = β using d1 d2 b1 lem-Oeq by blast
    qed
    then show ?thesis unfolding inj-on-def by blast
  qed
  ultimately have b4: |S1| ≤o |S| using card-of-ordLeq by blast
  obtain S2 where b5: S2 = { α ∈ 𝒪::′U rel set. α <o α0 } by blast
  have b6: |UNIV::nat set| ≤o |S1|
  proof −
    obtain xi where c1: xi = (λ i::nat. ((nord ∘ f)^^i) (nord α0′)) by blast
    have c2: ∀ i. xi i ∈ S1
    proof
      fix i0
      show xi i0 ∈ S1
      proof (induct i0)
        have α0′ ≤o nord α0′
```

using *b0′ lem-nord-l* **unfolding** *ordLess-def* **using** *ordIso-iff-ordLeq* **by** *blast*
then have $\alpha 0 \leq o$ *nord* $\alpha 0'$ **using** *b0″ ordIso-ordLeq-trans* **by** *blast*
moreover then have *nord* $\alpha 0' <o\ \kappa \wedge$ *nord* $\alpha 0' \in \mathcal{O}$
using *b0′ lem-nordO-ls-l lem-nord-ls-l ordLeq-ordLess-trans* **by** *blast*
ultimately show *xi 0* ∈ *S1* **using** *c1 b1* **by** *simp*
next
fix *i*
assume *xi i* ∈ *S1*
then have (*nord* ∘ *f*) (*xi i*) ∈ *S* **using** *b3* **by** *blast*
then show *xi* (*Suc i*) ∈ *S1* **using** *c1 b1 a5* **by** *simp*
qed
qed
have *c3*: ∀ *j*. ∀ *i*<*j*. *xi i* <*o xi j*
proof
fix *j0*
show ∀ *i*<*j0*. *xi i* <*o xi j0*
proof (*induct j0*)
show ∀ *i*<*0*. *xi i* <*o xi 0* **by** *blast*
next
fix *j*
assume *e1*: ∀ *i*<*j*. *xi i* <*o xi j*
show ∀ *i*<*Suc j*. *xi i* <*o xi* (*Suc j*)
proof(*intro allI impI*)
fix *i*
assume *f1*: *i* < *Suc j*
have *xi j* <*o nord* (*f* (*xi j*)) **using** *c2 b2* **unfolding** *sc-ord-def* **using** *lem-nord-ls-r* **by** *blast*
then have *xi j* <*o xi* (*Suc j*) **using** *c1* **by** *simp*
moreover then have *i* < *j* ⟶ *xi i* <*o xi* (*Suc j*) **and** *i* = *j* ⟶ *xi i* <*o xi* (*Suc j*)
using *e1 ordLess-transitive* **by** *blast+*
moreover have *i* < *j* ∨ *i* = *j* **using** *f1* **by** *force*
ultimately show *xi i* <*o xi* (*Suc j*) **by** *blast*
qed
qed
qed
then have ∀ *i j*. *xi i* = *xi j* ⟶ *i* = *j* **by** (*metis linorder-neqE-nat ordLess-irreflexive*)
then have *inj xi* **unfolding** *inj-on-def* **by** *blast*
moreover have *xi* ‘ *UNIV* ⊆ *S1* **using** *c2* **by** *blast*
ultimately show |*UNIV*::*nat set*| ≤*o* |*S1*| **using** *card-of-ordLeq* **by** *blast*
qed
then have ¬ *finite S1* **using** *infinite-iff-card-of-nat* **by** *blast*
moreover have |*S1*| ≤*o* |*S2*| ∨ |*S2*| ≤*o* |*S1*|
using *card-of-Well-order ordLess-imp-ordLeq ordLess-or-ordLeq* **by** *blast*
ultimately have |*S1* ∪ *S2*| ≤*o* |*S1*| ∨ |*S1* ∪ *S2*| ≤*o* |*S2*|
by (*metis card-of-Un1 card-of-Un-ordLeq-infinite card-of-ordLeq-finite sup.idem*)
moreover have |*S2*| ≤*o* |*S*|

proof −
have $|UNIV::nat\ set| \leq o\ |S|$ **using** *b4 b6 ordLeq-transitive* **by** *blast*
moreover have $|S2| \leq o\ |UNIV::nat\ set|$
proof −
have $\forall\ \alpha \in S2.\ \alpha <o\ \omega\text{-}ord \wedge \alpha \in \mathcal{O}$ **using** *b5 a4 ordLess-ordIso-trans* **by** *blast*
then have *d1*: $\forall\ \alpha \in S2.\ \alpha =o\ natLeq\text{-}on\ (card\ (Field\ \alpha)) \wedge \alpha \in \mathcal{O}$ **using** *lem-wolew-nat* **by** *blast*
obtain *A* **where** *d2*: $A = natLeq\text{-}on$ ‘ $UNIV$ **by** *blast*
moreover obtain *f* **where** *d3*: $f = (\lambda\ \alpha::'U\ rel.\ natLeq\text{-}on\ (card\ (Field\ \alpha)))$ **by** *blast*
ultimately have f ‘ $UNIV \subseteq A$ **by** *force*
moreover have *inj-on f S2*
proof −
have $\forall\ \alpha \in S2.\ \forall\ \beta \in S2.\ f\ \alpha = f\ \beta \longrightarrow \alpha = \beta$
proof (*intro ballI impI*)
fix $\alpha\ \beta$
assume $\alpha \in S2$ **and** $\beta \in S2$ **and** $f\ \alpha = f\ \beta$
then have $\alpha =o\ natLeq\text{-}on\ (card\ (Field\ \alpha))$ **and** $\beta =o\ natLeq\text{-}on\ (card\ (Field\ \beta))$
and $natLeq\text{-}on\ (card\ (Field\ \alpha)) = natLeq\text{-}on\ (card\ (Field\ \beta))$
and $\alpha \in \mathcal{O} \wedge \beta \in \mathcal{O}$ **using** *d1 d3* **by** *blast+*
moreover then have $\alpha =o\ \beta$
by (*metis* (*no-types*, *lifting*) *ordIso-symmetric ordIso-transitive*)
ultimately show $\alpha = \beta$ **using** *lem-Oeq* **by** *blast*
qed
then show *?thesis* **unfolding** *inj-on-def* **by** *blast*
qed
ultimately have $|S2| \leq o\ |A|$ **using** *card-of-ordLeq*[*of S2 A*] **by** *blast*
moreover have $|A| \leq o\ |UNIV::nat\ set|$ **using** *d2* **by** *simp*
ultimately show *?thesis* **using** *ordLeq-transitive* **by** *blast*
qed
ultimately show *?thesis* **using** *ordLeq-transitive* **by** *blast*
qed
ultimately have *b7*: $|S1 \cup S2| \leq o\ |S|$ **using** *b4 ordLeq-transitive* **by** *blast*
have $\{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\} \subseteq S1 \cup S2$ **using** *b1 b5 a1 a3* **by** *fastforce*
then have $|\{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}| \leq o\ |S1 \cup S2|$ **by** *simp*
moreover have $\forall\ \alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha 0 \leq o\ \alpha \wedge \alpha <o\ \kappa \longrightarrow \alpha \leq o\ (nord \circ f)\ \alpha \wedge (nord \circ f)\ \alpha \in S$
proof (*intro ballI impI*)
fix $\alpha::'U\ rel$
assume *c1*: $\alpha \in \mathcal{O}$ **and** *c2*: $\alpha 0 \leq o\ \alpha \wedge \alpha <o\ \kappa$
then have *c3*: $(nord \circ f)\ \alpha \in S$ **using** *b1 b3* **by** *blast*
moreover have $\alpha <o\ f\ \alpha$ **using** *c1 c2 b1 b2*[*of* α] **unfolding** *sc-ord-def* **by** *blast*
then have $\alpha \leq o\ f\ \alpha$ **using** *ordLess-imp-ordLeq* **by** *blast*
then have $\alpha \leq o\ (nord \circ f)\ \alpha$ **using** *lem-nord-le-r* **by** *simp*
then show $\alpha \leq o\ (nord \circ f)\ \alpha \wedge (nord \circ f)\ \alpha \in S$ **using** *c3* **by** *blast*
qed

ultimately show *?thesis* **using** *b7 ordLeq-transitive* **by** *blast*
qed

lemma *lem-rcc-uset-rcc-bnd*:
assumes $s \in \mathfrak{U}\ r$
shows $\|r\| \leq o\ \|s\|$
proof −
obtain *s0* **where** *b1*: $s0 \in \mathfrak{U}\ r \wedge |s0| =o\ \|r\| \wedge |s0| \leq o\ |s| \wedge (\ \forall\ s' \in \mathfrak{U}\ r.\ |s0| \leq o\ |s'|\ )$
using *assms lem-rcc-uset-ne* **by** *blast*
have *CCR s* **using** *assms* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
then obtain *t* **where** *b2*: $t \in \mathfrak{U}\ s \wedge |t| =o\ \|s\| \wedge (\ \forall\ s' \in \mathfrak{U}\ s.\ |t| \leq o\ |s'|\ )$
using *lem-Rcc-eq1-12 lem-rcc-uset-ne* **by** *blast*
have $t \in \mathfrak{U}\ r$ **using** *b2 assms lem-rcc-uset-tr* **by** *blast*
then have $\|r\| \leq o\ |t|$ **using** *lem-rcc-uset-mem-bnd* **by** *blast*
then show $\|r\| \leq o\ \|s\|$ **using** *b2 ordLeq-ordIso-trans* **by** *blast*
qed

lemma *lem-dc2-ccr-scf-lew*:
fixes *r*::*'U rel*
assumes *a1*: *CCR r* **and** *a2*: *scf r* $\leq o$ ω*-ord*
shows *DCR 2 r*
proof −
have $\exists\ s.\ s \in \mathfrak{U}\ r \wedge$ *single-valued s*
proof (*cases scf r* $<o$ ω*-ord*)
assume *scf r* $<o$ ω*-ord*
then have *b1*: *Conelike r* **using** *a1 lem-scf-ccr-finscf-cl lem-fin-fl-rel lem-wolew-fin* **by** *blast*
show *?thesis*
proof (*cases* $r = \{\}$)
assume $r = \{\}$
then have $r \in \mathfrak{U}\ r \wedge$ *single-valued r*
unfolding $\mathfrak{U}$*-def CCR-def single-valued-def Field-def* **by** *blast*
then show *?thesis* **by** *blast*
next
assume $r \neq \{\}$
then obtain *m* **where** *c2*: $m \in Field\ r \wedge (\forall\ a \in Field\ r.\ (a,m) \in r\hat{}*)$
using *b1* **unfolding** *Conelike-def* **by** *blast*
then obtain *a b* **where** $(a,b) \in r \wedge (m = a \vee m = b)$ **unfolding** *Field-def* **by** *blast*
moreover obtain *s* **where** $s = \{(a,b)\}$ **by** *blast*
ultimately have $s \in \mathfrak{U}\ r$ **and** *single-valued s*
using *c2* **unfolding** $\mathfrak{U}$*-def CCR-def Field-def single-valued-def* **by** *blast+*
then show *?thesis* **by** *blast*
qed
next
assume $\neg$ (*scf r* $<o$ ω*-ord*)
then have *scf r* $=o$ ω*-ord* **using** *a2 ordLeq-iff-ordLess-or-ordIso* **by** *blast*
then obtain *s* **where** *b1*: $s \in Span\ r$ **and** *b2*: *CCR s* **and** *b3*: *single-valued s*

using *a1 lem-sv-span-scfeqw* **by** *blast*
then have $s \in \mathfrak{U}\ r \wedge$ *single-valued s* **unfolding** *Span-def* $\mathfrak{U}$*-def* **by** *blast*
then show *?thesis* **by** *blast*
qed
then obtain *s* **where** *b1*: $s \in \mathfrak{U}\ r \wedge$ *single-valued s* **by** *blast*
moreover have *DCR 1 s*
proof −
obtain *g* **where** $g = (\lambda\ \alpha{::}nat.\ s)$ **by** *blast*
moreover then have *DCR-generating g*
using *b1* **unfolding** $\mathfrak{D}$*-def single-valued-def DCR-generating-def* **by** *blast*
ultimately show *?thesis* **unfolding** *DCR-def* **by** *blast*
qed
ultimately have *DCR* (*Suc 1*) *r* **using** *lem-Ldo-uset-reduc*[*of s r 1*] **by** *fastforce*
moreover have (*Suc 1*) = (*2*::*nat*) **by** *simp*
ultimately show *?thesis* **by** *metis*
qed

lemma *lem-dc3-ccr-refl-scf-wsuc*:
fixes *r*::′*U rel*
assumes *a1*: *Refl r* **and** *a2*: *CCR r*
and *a3*: |*Field r*| =*o cardSuc* |*UNIV*::*nat set*| **and** *a4*: *scf r* =*o* |*Field r*|
shows *DCR 3 r*
proof −
obtain κ::′*U rel* **where** *b0*: κ = |*Field r*| **by** *blast*
have *b1*: ω*-ord* <*o* (*scf r*) ∧ *regularCard* (*scf r*)
and *b2*: ω*-ord* <*o* |*Field r*|
using *a3 a4 lem-cardsuc-inf-gwreg ordIso-transitive* **by** *blast+*
then obtain *Ps f*
where *b3*: $f \in \mathcal{N}\ r\ Ps$
and *b4*: $\bigwedge\alpha.\ \omega\text{-}ord \le o\ |\mathfrak{L}\ f\ \alpha| \wedge \alpha < o\ \kappa \wedge isSuccOrd\ \alpha \Longrightarrow$
CCR (*Restr r* ($\mathcal{W}$ *r f* α)) ∧ |*Restr r* ($\mathcal{W}$ *r f* α)| <*o* κ
∧ (∀ *a* ∈ $\mathcal{W}$ *r f* α. *wesc-rel r f* α *a* (*wesc r f* α *a*))
using *b0 a1 a2 a4 lem-ccr-rcscf-struct* **by** *blast*
have *q0*: $\bigwedge$ α. ω*-ord* ≤*o* α ∧ α <*o* κ ∧ *isSuccOrd* α ⟹ ¬ *Conelike* (*Restr r* (*f* α))
proof −
fix α::′*U rel*
assume ω*-ord* ≤*o* α ∧ α <*o* κ ∧ *isSuccOrd* α
then have *Conelike* (*Restr r* (*f* α)) ⟶ *Conelike r*
using *b3 b0* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*3-def* $\mathcal{N}$*12-def clterm-def* **using** *ordLess-imp-ordLeq* **by** *blast*
moreover have *Conelike r* ⟶ *False*
proof
assume *Conelike r*
then have *finite* (*Field* (*scf r*)) **using** *a2 lem-scf-ccr-finscf-cl* **by** *blast*
then show *False* **using** *b2 a4*
by (*metis Field-card-of infinite-iff-natLeq-ordLeq ordIso-finite-Field ordLess-imp-ordLeq*)
qed

ultimately show ¬ *Conelike* (*Restr r* (*f* α)) **by** *blast*
qed
have *q1*: ⋀ α. *ω-ord* ≤*o* α ∧ α <*o* κ ∧ *isSuccOrd* α ⟹
ω-ord ≤*o* |𝔏 *f* α| ∧ *scf* (*Restr r* (*f* α)) =*o* *ω-ord*
proof −
fix α::′*U rel*
assume *c1*: *ω-ord* ≤*o* α ∧ α <*o* κ ∧ *isSuccOrd* α
have *Card-order ω-ord* ∧ ¬*finite* (*Field ω-ord*) ∧ *Well-order ω-ord*
using *natLeq-Card-order Field-natLeq* **by** *force*
then have ¬ *isSuccOrd ω-ord*
using *card-order-infinite-isLimOrd wo-rel.isLimOrd-def wo-rel-def* **by** *blast*
then have *ω-ord* <*o* α **using** *c1* **using** *lem-osucc-eq ordIso-symmetric ordLeq-iff-ordLess-or-ordIso* **by** *blast*
then obtain α*0*::′*U rel* **where** *c2*: *ω-ord* =*o* α*0* ∧ α*0* <*o* α **using** *internalize-ordLess*[*of ω-ord* α] **by** *blast*
then have *c3*: *f* α*0* ⊆ 𝔏 *f* α **unfolding** 𝔏*-def* **by** *blast*
obtain γ **where** *c4*: γ = *scf* (*Restr r* (*f* α)) **by** *blast*
have ¬ *Conelike* (*Restr r* (*f* α)) **using** *c1 q0* **by** *blast*
moreover have *CCR* (*Restr r* (*f* α)) **using** *c1 b0 b3* **unfolding** 𝒩*-def* 𝒩*6-def*

using *ordLess-imp-ordLeq* **by** *blast*
ultimately have *Card-order* γ ∧ ¬ *finite* (*Field* γ) **and** *c5*: ¬ *finite* (*Restr r* (*f* α))
using *c4 lem-scf-ccr-finscf-cl lem-scf-cardord lem-Relprop-fin-ccr* **by** *blast+*
then have *c6*: *ω-ord* ≤*o* γ
by (*meson card-of-Field-ordIso infinite-iff-natLeq-ordLeq ordIso-iff-ordLeq ordLeq-transitive*)
have *ω-ord* ≤*o* |𝔏 *f* α| **using** *c1 b0 b3* **unfolding** 𝒩*-def* 𝒩*12-def* **using** *ordLess-imp-ordLeq* **by** *blast*
moreover have *scf* (*Restr r* (*f* α)) =*o* *ω-ord*
proof −
have |*f* α| ≤*o* α **using** *c1 b0 b3* **unfolding** 𝒩*-def* 𝒩*7-def* **using** *ordLess-imp-ordLeq* **by** *blast*
then have |*Restr r* (*f* α)| ≤*o* α **using** *c1 lem-restr-ordbnd* **by** *blast*
then have γ ≤*o* α **using** *c4 c5 lem-rel-inf-fld-card*[*of Restr r* (*f* α)]
lem-scf-relfldcard-bnd ordLeq-ordIso-trans ordLeq-transitive **by** *blast*
then have γ <*o* *cardSuc* |*UNIV*::*nat set*| **using** *c1 b0 a3*
using *ordIso-iff-ordLeq ordLeq-ordLess-trans ordLess-ordLeq-trans* **by** *blast*
moreover have *Card-order* γ **using** *c4 lem-scf-cardord* **by** *blast*
ultimately have γ ≤*o* |*UNIV*::*nat set*| **by** *simp*
then show *?thesis* **using** *c4 c6* **using** *card-of-nat ordIso-iff-ordLeq ordLeq-ordIso-trans* **by** *blast*
qed
ultimately show *ω-ord* ≤*o* |𝔏 *f* α| ∧ *scf* (*Restr r* (*f* α)) =*o* *ω-ord* **by** *blast*
qed
obtain *is-st*::′*U rel* ⇒ ′*U rel* ⇒ *bool*
where *q3*: *is-st* = (λ *s t*. *t* ∈ *Span s* ∧ *t* ≠ {} ∧ *CCR t* ∧
single-valued t ∧ *acyclic t* ∧ (∀ *x*∈*Field t*. *t*‘‘{*x*} ≠ {})) **by** *blast*
obtain *st* **where** *q4*: *st* = (λ *s*::′*U rel*. *SOME t*. *is-st s t*) **by** *blast*

have *q5*: $\bigwedge s.\ CCR\ s \wedge scf\ s =o\ \omega\text{-}ord \Longrightarrow is\text{-}st\ s\ (st\ s)$
proof −
fix $s::'U\ rel$
assume $CCR\ s \wedge scf\ s =o\ \omega\text{-}ord$
then obtain t **where** $is\text{-}st\ s\ t$ **using** *q3 lem-sv-span-scfeqw[of s]* **by** *blast*
then show $is\text{-}st\ s\ (st\ s)$ **using** *q4 someI-ex* **by** *metis*
qed
obtain $\kappa 0$ **where** *b5*: $\kappa 0 = \omega\text{-}ord$ **by** *blast*
obtain S **where** *b6*: $S = \{\alpha \in \mathcal{O}::'U\ rel\ set.\ \kappa 0 \le o\ \alpha \wedge isSuccOrd\ \alpha \wedge \alpha <o\ \kappa\}$ **by** *blast*
obtain R **where** *b8*: $R = (\lambda\ \alpha.\ st\ (Restr\ r\ (\mathcal{W}\ r\ f\ \alpha)))$ **by** *blast*
obtain $T::'U\ rel\ set$ **where** *b11*: $T = \{\ t.\ t \neq \{\} \wedge CCR\ t \wedge single\text{-}valued\ t \wedge$
$acyclic\ t \wedge (\forall x{\in}Field\ t.\ t``\{x\} \neq \{\})\ \}$ **by** *blast*
obtain $W::'U\ rel \Rightarrow 'U\ set$ **where** *b12*: $W = (\lambda\ \alpha.\ \mathcal{W}\ r\ f\ \alpha)$ **by** *blast*
obtain Wa **where** *b13*: $Wa = (\bigcup \alpha{\in}S.\ W\ \alpha)$ **by** *blast*
obtain $r1$ **where** *b14*: $r1 = Restr\ r\ Wa$ **by** *blast*
have *b15*: $\bigwedge\ \alpha.\ \alpha \in S \Longrightarrow Restr\ r\ (\mathcal{W}\ r\ f\ \alpha) = Restr\ r1\ (W\ \alpha)$ **using** *b12 b13 b14* **by** *blast*
have *b16*: $\bigwedge\ \alpha.\ \alpha \in S \Longrightarrow Restr\ r\ (\mathcal{W}\ r\ f\ \alpha) \in \mathfrak{U}\ (Restr\ r\ (f\ \alpha))$
proof −
fix α
assume *c1*: $\alpha \in S$
have *d1*: $\neg\ finite\ r$ **using** *b2 lem-fin-fl-rel* **by** (*metis infinite-iff-natLeq-ordLeq ordLess-imp-ordLeq*)
moreover have $\alpha <o\ scf\ r$ **using** *c1 b0 b6 a4* **using** *ordIso-symmetric ordLess-ordIso-trans* **by** *blast*
moreover have $\omega\text{-}ord \le o\ |\mathfrak{L}\ f\ \alpha|$ **using** *c1 b5 b6 q1* **by** *blast*
moreover have $isSuccOrd\ \alpha$ **using** *c1 b6* **by** *blast*
ultimately show $Restr\ r\ (\mathcal{W}\ r\ f\ \alpha) \in \mathfrak{U}\ (Restr\ r\ (f\ \alpha))$
using *b3 a1 a2 lem-der-qw-uset[of r f Ps α]* **by** *blast*
qed
have $\kappa =o\ cardSuc\ |UNIV::nat\ set|$ **using** *b0 a3* **by** *blast*
moreover have $Refl\ r1$ **using** *a1 b14* **unfolding** *refl-on-def Field-def* **by** *blast*
moreover have $S \subseteq \{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}$ **using** *b6* **by** *blast*
moreover have *b17*: $|\{\alpha \in \mathcal{O}::'U\ rel\ set.\ \alpha <o\ \kappa\}| \le o\ |S|$
$\wedge\ (\exists h.\ \forall \alpha{\in}\mathcal{O}::'U\ rel\ set.\ \kappa 0 \le o\ \alpha \wedge \alpha <o\ \kappa \longrightarrow \alpha \le o\ h\ \alpha \wedge h\ \alpha \in S)$
proof −
have $Card\text{-}order\ \kappa$ **using** *b0* **by** *simp*
moreover have $\omega\text{-}ord \le o\ \kappa$ **using** *b0 b2 ordLess-imp-ordLeq* **by** *blast*
moreover have $\kappa 0 <o\ \kappa$ **using** *b0 b2 b5* **by** *blast*
moreover have $\kappa 0 =o\ \omega\text{-}ord$ **using** *b5 ordIso-refl natLeq-Card-order* **by** *blast*
ultimately show *?thesis* **using** *b6 lem-oint-infcard-gew-sc-cfbnd[of κ κ0 S]* **by** *blast*
qed
moreover have $\forall\ \alpha \in S.\ \exists\ \beta \in S.\ \alpha <o\ \beta$
proof −
have $Card\text{-}order\ \kappa$ **using** *b0* **by** *simp*
moreover have $\omega\text{-}ord \le o\ \kappa$ **using** *b0 b2 ordLess-imp-ordLeq* **by** *blast*
ultimately show *?thesis* **using** *b6 lem-oint-infcard-sc-cf[of κ S κ0]* **by** *blast*

qed
moreover have *b18*: *Field r1* = (⋃α∈*S*. *W* α)
proof −
have *SF r* = {*A*. *A* ⊆ *Field r*} **using** *a1* **unfolding** *SF-def Field-def refl-on-def*
by *fast*
moreover have *Wa* ⊆ *Field r*
using *b0 b3 b6 b12 b13 lem-qw-range*[*of f r Ps* -] *ordLess-imp-ordLeq*[*of* - κ]
by *blast*
ultimately have *Field r1* = *Wa* **using** *b14* **unfolding** *SF-def* **by** *blast*
then show *?thesis* **using** *b13* **by** *blast*
qed
moreover have ∀α∈*S*. ∀ β∈*S*. α ≠ β ⟶ *W* α ∩ *W* β = {}
proof (*intro ballI impI*)
fix α β
assume α ∈ *S* **and** β ∈ *S* **and** α ≠ β
then have *Well-order* α ∧ *Well-order* β ∧ ¬ (α =*o* β) **using** *b6 lem-Owo*
lem-Oeq **by** *blast*
then show *W* α ∩ *W* β = {} **using** *b12 lem-Der-inf-qw-disj* **by** *blast*
qed
moreover have ⋀ α. α ∈ *S* ⟹ *R* α ∈ *T* ∧ *R* α ⊆ *Restr r1* (*W* α) ∧ |*W* α|
≤*o* |*UNIV*::*nat set*|
∧ *Field* (*R* α) = *W* α ∧ ¬ *Conelike* (*Restr r1* (*W* α))
proof −
fix α
assume *c1*: α ∈ *S*
then have *c2*: *CCR* (*Restr r* (𝒲 *r f* α)) ∧ *scf* (*Restr r* (*f* α)) =*o* ω-*ord* **using**
b4 q1 b5 b6 **by** *blast*
moreover have *c3*: *scf* (*Restr r* (𝒲 *r f* α)) =*o* ω-*ord* ∧ |𝒲 *r f* α| ≤*o*
|*UNIV*::*nat set*|
proof −
have *d1*: ¬ *finite r* **using** *b2 lem-fin-fl-rel* **by** (*metis infinite-iff-natLeq-ordLeq*
ordLess-imp-ordLeq)
have *Restr r* (𝒲 *r f* α) ∈ 𝔘 (*Restr r* (*f* α)) **using** *c1 b16* **by** *blast*
then have *d2*: ‖*Restr r* (*f* α)‖ ≤*o* ‖*Restr r* (𝒲 *r f* α)‖ **using** *lem-rcc-uset-rcc-bnd*
by *blast*
have *scf* (*Restr r* (*f* α)) =*o* ω-*ord* **using** *c1 b5 b6 q1* **by** *blast*
moreover have *CCR* (*Restr r* (*f* α))
using *c1 b0 b3 b6* **unfolding** 𝒩-*def* 𝒩*6-def* **using** *ordLess-imp-ordLeq* **by**
blast
ultimately have ω-*ord* =*o* ‖*Restr r* (*f* α)‖
using *lem-scf-ccr-scf-rcc-eq ordIso-symmetric ordIso-transitive* **by** *blast*
then have *d3*: ω-*ord* ≤*o* ‖*Restr r* (𝒲 *r f* α)‖ **using** *d2 ordIso-ordLeq-trans*
by *blast*
have |*Restr r* (𝒲 *r f* α)| <*o* |*Field r*| **using** *d1 c1 b0 b3 b6 lem-der-inf-qw-restr-card*
by *blast*
then have |*Restr r* (𝒲 *r f* α)| <*o* *cardSuc* |*UNIV*::*nat set*| **using** *a3 ord-*
Less-ordIso-trans **by** *blast*
then have *d4*: |*Restr r* (𝒲 *r f* α)| ≤*o* |*UNIV*::*nat set*| **by** *simp*
then have ‖*Restr r* (𝒲 *r f* α)‖ ≤*o* ω-*ord* **using** *lem-Rcc-relcard-bnd*

by (*metis ordLeq-transitive card-of-nat ordLeq-ordIso-trans*)
then have ‖*Restr r* (𝒲 *r f* α)‖ =*o* ω-*ord* **using** *d3* **using** *ordIso-iff-ordLeq* **by** *blast*
moreover have |𝒲 *r f* α| ≤*o* |*UNIV*::*nat set*|
proof −
have 𝒲 *r f* α ⊆ *f* α **unfolding** 𝒲-*def* **by** *blast*
then have |𝒲 *r f* α| ≤*o* |*f* α| **by** *simp*
moreover have |*f* α| <*o* |*Field r*| **using** *c1 b3 b5 b6 b0* **unfolding** 𝒩-*def* 𝒩*7-def*
using *ordLess-imp-ordLeq ordLeq-ordLess-trans* **by** *blast*
ultimately have |𝒲 *r f* α| <*o cardSuc* |*UNIV*::*nat set*|
using *a3 ordLeq-ordLess-trans ordLess-ordIso-trans* **by** *blast*
then show *?thesis* **by** *simp*
qed
ultimately show *?thesis* **using** *c2 lem-scf-ccr-scf-rcc-eq*[*of Restr r* (𝒲 *r f* α)]
by (*metis ordIso-symmetric ordIso-transitive*)
qed
ultimately have *c4*: *is-st* (*Restr r* (𝒲 *r f* α)) (*R* α) **using** *q5 b8* **by** *blast*
then have *c5*: *R* α ∈ *Span* (*Restr r* (𝒲 *r f* α)) **using** *q3* **by** *blast*
then have *Field* (*R* α) = *Field* (*Restr r* (𝒲 *r f* α)) **unfolding** *Span-def* **by** *blast*
moreover have *SF r* = {*A*. *A* ⊆ *Field r*} **using** *a1* **unfolding** *SF-def refl-on-def Field-def* **by** *fast*
moreover have 𝒲 *r f* α ⊆ *Field r* **using** *c1 b0 b3 b6 lem-qw-range ordLess-imp-ordLeq* **by** *blast*
ultimately have *Field* (*R* α) = 𝒲 *r f* α **unfolding** *SF-def* **by** *blast*
then have *R* α ⊆ *Restr r1* (*W* α) ∧ *Field* (*R* α) = *W* α
using *c1 c5 b12 b13 b14* **unfolding** *Span-def* **by** *blast*
moreover have *R* α ∈ *T* **using** *c4 q3 b11* **by** *blast*
moreover have ¬ *Conelike* (*Restr r1* (*W* α))
proof −
obtain *s1* **where** *d1*: *s1* = *Restr r* (𝒲 *r f* α) **by** *blast*
then have *scf s1* =*o* ω-*ord* ∧ *CCR s1* **using** *c2 c3* **by** *blast*
moreover then have ¬ *finite* (*Field* (*scf s1*))
by (*metis Field-natLeq infinite-UNIV-nat ordIso-finite-Field*)
ultimately have ¬ *Conelike s1* **using** *lem-scf-ccr-finscf-cl* **by** *blast*
then show *?thesis* **using** *d1 c1 b15*[*of* α] **by** *metis*
qed
ultimately show *R* α ∈ *T* ∧ *R* α ⊆ *Restr r1* (*W* α) ∧ |*W* α| ≤*o* |*UNIV*::*nat set*|
∧ *Field* (*R* α) = *W* α ∧ ¬ *Conelike* (*Restr r1* (*W* α)) **using** *c3 b12* **by** *blast*
qed
moreover have ⋀ α *x*. α ∈ *S* ⟹ *x* ∈ *W* α ⟹
∃ *a*. ((*x*,*a*) ∈ (*Restr r1* (*W* α))^* ∧ (∀ β ∈ *S*. α <*o* β ⟶ (*r1*‘‘{*a*} ∩ *W* β) ≠ {}))
proof −
fix α *x*

assume *c1*: $\alpha \in S$ **and** *c2*: $x \in W\ \alpha$
moreover obtain *a* **where** $a = wesc\ r\ f\ \alpha\ x$ **by** *blast*
ultimately have *wesc-rel r f* α *x a* **using** *b4 b0 b5 b6 b12 q1* **by** *blast*
then have *c3*: $a \in \mathcal{W}\ r\ f\ \alpha \wedge (x,a) \in (Restr\ r\ (\mathcal{W}\ r\ f\ \alpha))\hat{}*$ **and**
c4: $\forall \beta.\ \alpha <o\ \beta \wedge \beta <o\ |Field\ r| \wedge (\beta = \{\} \vee isSuccOrd\ \beta) \longrightarrow r``\{a\} \cap \mathcal{W}$
$r\ f\ \beta \neq \{\}$
unfolding *wesc-rel-def* **by** *blast+*
have $(x,a) \in (Restr\ r1\ (W\ \alpha))\hat{}*$ **using** *c1 c3 b15* **by** *metis*
moreover have $\forall\ \beta \in S.\ \alpha <o\ \beta \longrightarrow (r1``\{a\} \cap W\ \beta) \neq \{\}$
proof (*intro ballI impI*)
fix β
assume *d1*: $\beta \in S$ **and** $\alpha <o\ \beta$
then obtain *b* **where** $(a,b) \in r \wedge b \in W\ \beta$ **using** *c4 b6 b0 b12* **by** *blast*
moreover then have $b \in Wa$ **using** *d1 b13* **by** *blast*
moreover have $a \in Wa$ **using** *c1 c3 b12 b13* **by** *blast*
ultimately have $(a,b) \in r1 \wedge b \in W\ \beta$ **using** *b14* **by** *blast*
then show $(r1``\{a\} \cap W\ \beta) \neq \{\}$ **by** *blast*
qed
ultimately show $\exists\ a.\ ((x,a) \in (Restr\ r1\ (W\ \alpha))\hat{}*$
$\wedge (\forall\ \beta \in S.\ \alpha <o\ \beta \longrightarrow (r1``\{a\} \cap W\ \beta) \neq \{\}))$ **by** *blast*
qed
ultimately obtain r' **where** *b19*: $CCR\ r' \wedge DCR\ 2\ r' \wedge r' \subseteq r1$
and $\forall\ a \in Field\ r1.\ \exists\ b \in Field\ r'.\ (a,b) \in r1\hat{}*$
using *b11 lem-cfcomp-d2uset*[*of* κ *T r1 S W R*] **by** *blast*
then have *b20*: $r' \in \mathfrak{U}\ r1$ **unfolding** $\mathfrak{U}$*-def Span-def* **by** *blast*
moreover have $r1 \in \mathfrak{U}\ r$
proof −
have $\forall\ a \in Field\ r.\ \exists\ \alpha \in S.\ a \in f\ \alpha$
proof
fix *a*
assume *d1*: $a \in Field\ r$
obtain *A* **where** *d2*: $A = \{\alpha \in \mathcal{O}::'U\ rel\ set.\ \kappa 0 \leq o\ \alpha \wedge \alpha <o\ \kappa\}$ **by** *blast*
have *d3*: $a \in f\ |Field\ r| \wedge \omega\text{-}ord \leq o\ |Field\ r|$ **using** *d1 b3 b2*
unfolding $\mathcal{N}$*-def* $\mathcal{N}$*9-def* **using** *ordLess-imp-ordLeq* **by** *blast*
moreover have *Card-order* $|Field\ r|$ **by** *simp*
ultimately have $\neg\ (\ |Field\ r| = \{\} \vee isSuccOrd\ |Field\ r|\ )$ **using** *lem-card-inf-lim*
by *blast*
moreover have $|Field\ r| \leq o\ |Field\ r|$ **by** *simp*
ultimately have $(\nabla\ f\ |Field\ r|\ ) = \{\}$ **using** *b3* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*2-def*
by *blast*
then have $f\ |Field\ r| \subseteq \mathfrak{L}\ f\ |Field\ r|$ **unfolding** *Dbk-def* **by** *blast*
then obtain γ **where** *d4*: $\gamma <o\ \kappa \wedge a \in f\ \gamma$ **using** *d3 b0* **unfolding** $\mathfrak{L}$*-def*
by *blast*
have $\exists\ \alpha \in A.\ a \in f\ \alpha$
proof (*cases* $\kappa 0 \leq o\ \gamma$)
assume $\kappa 0 \leq o\ \gamma$
then have $nord\ \gamma \in A \wedge nord\ \gamma =o\ \gamma$ **using** *d4 d2 lem-nord-le-r lem-nord-ls-l*
lem-nord-r lem-nordO-le-r ordLess-Well-order-simp **by** *blast*

moreover then have f $(nord\ \gamma) = f\ \gamma$ **using** *b3* **unfolding** $\mathcal{N}$*-def* **by** *blast*
ultimately have $nord\ \gamma \in A \wedge a \in f\ (nord\ \gamma)$ **using** *d4* **by** *blast*
then show *?thesis* **by** *blast*
next
assume $\neg\ \kappa 0 \leq o\ \gamma$
moreover have *Well-order* $\kappa 0$ $\wedge$ *Well-order* γ
using *d4 b5 natLeq-Well-order ordLess-Well-order-simp* **by** *blast*
ultimately have $\gamma \leq o\ \kappa 0$ **using** *ordLeq-total* **by** *blast*
moreover have $\kappa 0 < o\ \kappa$ **using** *b0 b2 b5* **by** *blast*
moreover then obtain $\alpha 0$::$'U\ rel$ **where** $\kappa 0 = o\ \alpha 0 \wedge \alpha 0 < o\ \kappa$
using *internalize-ordLess*[*of* $\kappa 0\ \kappa$] **by** *blast*
ultimately have $\gamma \leq o\ \alpha 0 \wedge \kappa 0 \leq o\ \alpha 0 \wedge \alpha 0 < o\ \kappa$
using *ordLeq-ordIso-trans ordIso-iff-ordLeq* **by** *blast*
then have $\gamma \leq o\ nord\ \alpha 0 \wedge \kappa 0 \leq o\ nord\ \alpha 0 \wedge nord\ \alpha 0 < o\ \kappa \wedge nord\ \alpha 0 \in \mathcal{O}$
using *lem-nord-le-r lem-nord-le-r lem-nord-ls-l lem-nordO-le-r ordLess-Well-order-simp* **by** *blast*
moreover then have $f\ \gamma \subseteq f\ (nord\ \alpha 0)$
using *b3 b0 ordLess-imp-ordLeq* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*1-def* **by** *blast*
ultimately have $a \in f\ (nord\ \alpha 0) \wedge nord\ \alpha 0 \in A$ **using** *d4 d2* **by** *blast*
then show *?thesis* **by** *blast*
qed
then obtain $\alpha\ \alpha'$ **where** $\alpha' \in S \wedge \alpha \leq o\ \alpha' \wedge \alpha \in A \wedge a \in f\ \alpha$ **using** *d2 b17* **by** *blast*
moreover then have $\alpha' \leq o\ |Field\ r|$ **using** *b6 b0* **using** *ordLess-imp-ordLeq* **by** *blast*
ultimately have $\alpha' \in S \wedge a \in f\ \alpha'$ **using** *b3 b0 b0* **unfolding** $\mathcal{N}$*-def* $\mathcal{N}$*1-def* **by** *blast*
then show $\exists\ \alpha \in S.\ a \in f\ \alpha$ **by** *blast*
qed
moreover have $\forall\ \alpha \in S.\ f\ \alpha \subseteq dncl\ r\ (Field\ r1)$
proof
fix α
assume *d1*: $\alpha \in S$
show $f\ \alpha \subseteq dncl\ r\ (Field\ r1)$
proof
fix a
assume $a \in f\ \alpha$
moreover have $f\ \alpha \in SF\ r$ **using** *d1 b0 b3 b6*
unfolding $\mathcal{N}$*-def* $\mathcal{N}$*5-def* **using** *ordLess-imp-ordLeq* **by** *blast*
ultimately have $a \in Field\ (Restr\ r\ (f\ \alpha))$ **unfolding** *SF-def* **by** *blast*
moreover have $Restr\ r\ (\mathcal{W}\ r\ f\ \alpha) \in \mathfrak{U}\ (Restr\ r\ (f\ \alpha))$ **using** *d1 b16* **by** *blast*
ultimately obtain b **where** $b \in Field\ (Restr\ r\ (\mathcal{W}\ r\ f\ \alpha)) \wedge (a,\ b) \in (Restr\ r\ (f\ \alpha))\hat{}*$
unfolding $\mathfrak{U}$*-def* **by** *blast*
then have $b \in \mathcal{W}\ r\ f\ \alpha \wedge (a,b) \in r\hat{}*$
unfolding *Field-def* **using** *rtrancl-mono*[*of Restr r* $(f\ \alpha)$ *r*] **by** *blast*

moreover then have $b \in$ *Field r1* **using** *d1 b12 b18* **by** *blast*
ultimately show $a \in$ *dncl r* (*Field r1*) **unfolding** *dncl-def* **by** *blast*
qed
qed
ultimately have $\forall\ a \in$ *Field r*. $\exists\ b \in$ *Field r1*. $(a, b) \in r\hat{\ }*$ **unfolding** *dncl-def* **by** *blast*
moreover have *CCR r1* **using** *b20 lem-rcc-uset-ne-ccr* **by** *blast*
moreover have $r1 \subseteq r$ **using** *b14* **by** *blast*
ultimately show $r1 \in \mathfrak{U}\ r$ **unfolding** $\mathfrak{U}$*-def* **by** *blast*
qed
ultimately have $r' \in \mathfrak{U}\ r$ **using** *lem-rcc-uset-tr* **by** *blast*
then show *DCR 3 r* **using** *b19 lem-Ldo-uset-reduc*[*of r′ r 2*] **by** *simp*
qed

lemma *lem-dc3-ccr-scf-lewsuc*:
fixes *r*::*′U rel*
assumes *a1*: *CCR r* **and** *a2*: |*Field r*| $\leq o$ *cardSuc* |*UNIV*::*nat set*|
shows *DCR 3 r*
proof (*cases scf r* $\leq o$ ω*-ord*)
assume *scf r* $\leq o$ ω*-ord*
then have *DCR 2 r* **using** *a1 lem-dc2-ccr-scf-lew* **by** *blast*
moreover have $r \in \mathfrak{U}\ r$ **using** *a1* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
ultimately show *DCR 3 r* **using** *lem-Ldo-uset-reduc*[*of r r 2*] **by** *simp*
next
assume ¬ (*scf r* $\leq o$ ω*-ord*)
then have ω*-ord* $<o$ |*Field r*| **using** *lem-scf-relfldcard-bnd lem-scf-inf*
by (*metis ordIso-iff-ordLeq ordLeq-iff-ordLess-or-ordIso ordLeq-transitive*)
then have |*UNIV*::*nat set*| $<o$ |*Field r*| **using** *card-of-nat ordIso-ordLess-trans* **by** *blast*
then have *cardSuc* |*UNIV*::*nat set*| $\leq o$ |*Field r*| **by** (*meson cardSuc-ordLess-ordLeq card-of-Card-order*)
then have *b0*: |*Field r*| $=o$ *cardSuc* |*UNIV*::*nat set*| **using** *a2*
using *not-ordLeq-ordLess ordLeq-iff-ordLess-or-ordIso* **by** *blast*
obtain *r1* **where** *b1*: $r1 = r \cup \{(x,y).\ x = y \wedge x \in$ *Field r*$\}$ **by** *blast*
have *b2*: *Field r1* = *Field r* **using** *b1* **unfolding** *Field-def* **by** *blast*
have $r \in \mathfrak{U}\ r1$ **using** *b1 b2 a1* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
then have *b3*: *CCR r1* **using** *lem-rcc-uset-ne-ccr*[*of r1*] **by** *blast*
have (¬ (*scf r1* $\leq o$ ω*-ord*)) $\longrightarrow$ *scf r1* $=o$ |*Field r1*|
proof
assume ¬ (*scf r1* $\leq o$ ω*-ord*)
then have ω*-ord* $<o$ *scf r1*
using *lem-scf-inf* **by** (*metis ordIso-iff-ordLeq ordLeq-iff-ordLess-or-ordIso*)
then have |*UNIV*::*nat set*| $<o$ *scf r1* $\wedge$ *Card-order* (*scf r1*)
using *lem-scf-cardord* **by** (*metis card-of-nat ordIso-ordLess-trans*)
then have *cardSuc* |*UNIV*::*nat set*| $\leq o$ *scf r1* **by** (*meson cardSuc-ordLess-ordLeq card-of-Card-order*)
then have |*Field r1*| $\leq o$ *scf r1* **using** *b0 b2* **by** (*metis ordIso-ordLeq-trans*)
then show *scf r1* $=o$ |*Field r1*| **using** *lem-scf-relfldcard-bnd*[*of r1*]
by (*metis not-ordLeq-ordLess ordLeq-iff-ordLess-or-ordIso*)

qed
moreover have *scf r1* $\leq o$ ω*-ord* $\longrightarrow$ *DCR 3 r1*
proof
assume *scf r1* $\leq o$ ω*-ord*
then have *DCR 2 r1* **using** *b3 lem-dc2-ccr-scf-lew* **by** *blast*
moreover have *r1* $\in$ $\mathfrak{U}$ *r1* **using** *b3* **unfolding** $\mathfrak{U}$*-def* **by** *blast*
ultimately show *DCR 3 r1* **using** *lem-Ldo-uset-reduc*[*of r1 r1 2*] **by** *simp*
qed
moreover have *scf r1* $=o$ |*Field r1*| $\longrightarrow$ *DCR 3 r1*
proof
assume *scf r1* $=o$ |*Field r1*|
moreover have *Refl r1* **using** *b1* **unfolding** *refl-on-def Field-def* **by** *force*
ultimately show *DCR 3 r1* **using** *b0 b2 b3 lem-dc3-ccr-refl-scf-wsuc*[*of r1*] **by** *simp*
qed
ultimately have *DCR 3 r1* **by** *blast*
moreover have $\bigwedge$ *n. n* $\neq$ *0* $\Longrightarrow$ *DCR n r1* $\Longrightarrow$ *DCR n r* **using** *b1 lem-Ldo-eqid* **by** *blast*
ultimately show *DCR 3 r* **by** *force*
qed

lemma *lem-Cprf-conf-ccr-decomp*:
fixes *r*::*'U rel*
assumes *confl-rel r*
shows $\exists$ *S*::(*'U rel set*). ($\forall$ *s*$\in$*S*. *CCR s*) $\wedge$ (*r* = $\bigcup$ *S*) $\wedge$ ($\forall$ *s1*$\in$*S*. $\forall$ *s2*$\in$*S*. *s1* $\neq$ *s2* $\longrightarrow$ *Field s1* $\cap$ *Field s2* = {})
proof −
obtain $\mathcal{D}$ **where** *b1*: $\mathcal{D}$ = { *D*. $\exists$ *x* $\in$ *Field r*. *D* = (*r*^<−>∗) " {*x*} } **by** *blast*
obtain *S* **where** *b2*: *S* = { *s*. $\exists$ *D* $\in$ $\mathcal{D}$. *s* = *Restr r D* } **by** *blast*
have *r* = $\bigcup$ *S*
proof
show *r* $\subseteq$ $\bigcup$ *S*
proof
fix *a b*
assume *d1*: (*a*,*b*) $\in$ *r*
then have *a* $\in$ *Field r* **unfolding** *Field-def* **by** *blast*
moreover obtain *D* **where** *d2*: *D* = (*r*^<−>∗) " {*a*} **by** *blast*
ultimately have *D* $\in$ $\mathcal{D}$ **using** *b1* **by** *blast*
moreover then have (*a*,*b*) $\in$ *Restr r D* **using** *d1 d2* **by** *blast*
ultimately show (*a*,*b*) $\in$ $\bigcup$ *S* **using** *b2* **by** *blast*
qed
next
show $\bigcup$ *S* $\subseteq$ *r* **using** *b2* **by** *blast*
qed
moreover have $\forall$ *s1*$\in$*S*. $\forall$ *s2*$\in$*S*. *Field s1* $\cap$ *Field s2* $\neq$ {} $\longrightarrow$ *s1* = *s2*
proof (*intro ballI impI*)
fix *s1 s2*
assume *s1* $\in$ *S* **and** *s2* $\in$ *S* **and** *Field s1* $\cap$ *Field s2* $\neq$ {}
moreover then obtain *D1 D2* **where** *c1*: *D1* $\in$ $\mathcal{D}$ $\wedge$ *D2* $\in$ $\mathcal{D}$ $\wedge$ *s1* = *Restr*

r D1 ∧ s2 = Restr r D2 **using** b2 **by** blast
  ultimately have c2: D1 ∩ D2 ≠ {} **unfolding** Field-def **by** blast
  obtain a b c **where** c3: c ∈ D1 ∩ D2 ∧ D1 = (r^<->*) `` {a} ∧ D2 = (r^<->\*) `` {b} **using** b1 c1 c2 **by** blast
  then have (a,c) ∈ r^<->* ∧ (b,c) ∈ r^<->* **by** blast
  then have (a,b) ∈ r^<->* **by** (metis conversion-inv conversion-rtrancl rtrancl.intros(2))
  moreover have equiv UNIV (r^<->*) **unfolding** equiv-def
   by (simp add: conversion-sym conversion-trans refl-on-def)
  ultimately have D1 = D2 **using** c3 equiv-class-eq **by** simp
  then show s1 = s2 **using** c1 **by** blast
 qed
 moreover have ∀ s∈S. CCR s
 proof
  fix s
  assume s ∈ S
  then obtain D **where** c1: D ∈ 𝒟 ∧ s = Restr r D **using** b2 **by** blast
  then obtain x **where** c2: x ∈ Field r ∧ D = (r^<->*) `` {x} **using** b1 **by** blast
  have c3: r `` D ⊆ D
  proof
   fix b
   assume b ∈ r `` D
   then obtain a **where** d1: a ∈ D ∧ (a,b) ∈ r **by** blast
   then have (x,a) ∈ r^<->* **using** c2 **by** blast
   then have (x,b) ∈ r^<->* **using** d1
   by (metis conversionI′ conversion-rtrancl rtrancl.rtrancl-into-rtrancl rtrancl.rtrancl-refl)
   then show b ∈ D **using** c2 **by** blast
  qed
  have c4: r^* ∩ (D × (UNIV::′U set)) ⊆ s^*
  proof −
   have ∀ n. ∀ a b. (a,b) ∈ r^^n ∧ a ∈ D ⟶ (a,b) ∈ s^*
   proof
    fix n0
    show ∀ a b. (a,b) ∈ r^^n0 ∧ a ∈ D ⟶ (a,b) ∈ s^*
    proof (induct n0)
     show ∀ a b. (a,b) ∈ r^^0 ∧ a ∈ D ⟶ (a,b) ∈ s^* **by** simp
    next
     fix n
     assume f1: ∀ a b. (a,b) ∈ r^^n ∧ a ∈ D ⟶ (a,b) ∈ s^*
     show ∀ a b. (a,b) ∈ r^^(Suc n) ∧ a ∈ D ⟶ (a,b) ∈ s^*
     proof (intro allI impI)
      fix a b
      assume g1: (a,b) ∈ r^^(Suc n) ∧ a ∈ D
      moreover then obtain c **where** g2: (a,c) ∈ r^^n ∧ (c,b) ∈ r **by** force
      ultimately have g3: (a,c) ∈ s^* **using** f1 **by** blast
      have c ∈ D **using** c2 g1 g2
     by (metis Image-singleton-iff conversionI′ conversion-rtrancl relpow-imp-rtrancl rtrancl.rtrancl-into-rtrancl)
      then have (c,b) ∈ s **using** c1 c3 g2 **by** blast

then show $(a,b) \in s\hat{}*$ **using** *g3* **by** (*meson rtrancl.rtrancl-into-rtrancl*)
qed
qed
qed
then show *?thesis* **using** *rtrancl-power* **by** *blast*
qed
have $\forall\ a \in Field\ s.\ \forall\ b \in Field\ s.\ \exists\ c \in Field\ s.\ (a,c) \in s\hat{}* \wedge (b,c) \in s\hat{}*$
proof (*intro ballI*)
fix *a b*
assume *d1*: $a \in Field\ s$ **and** *d2*: $b \in Field\ s$
then have *d3*: $a \in D \wedge b \in D$ **using** *c1* **unfolding** *Field-def* **by** *blast*
then have $(x,a) \in r\hat{}{<}{-}{>}* \wedge (x,b) \in r\hat{}{<}{-}{>}*$ **using** *c2* **by** *blast*
then have $(a,b) \in r\hat{}{<}{-}{>}*$ **by** (*metis conversion-inv conversion-rtrancl rtrancl.rtrancl-into-rtrancl*)
moreover have *CR r* **using** *assms* **unfolding** *confl-rel-def Abstract-Rewriting.CR-on-def* **by** *blast*
ultimately obtain *c* **where** $(a,c) \in r\hat{}* \wedge (b,c) \in r\hat{}*$
by (*metis Abstract-Rewriting.CR-imp-conversionIff-join Abstract-Rewriting.joinD*)
then have $(a,c) \in s\hat{}* \wedge (b,c) \in s\hat{}*$ **using** *c4 d3* **by** *blast*
moreover then have $c \in Field\ s$ **using** *d1* **unfolding** *Field-def* **by** (*metis Range.intros Un-iff rtrancl.cases*)
ultimately show $\exists\ c \in Field\ s.\ (a,c) \in s\hat{}* \wedge (b,c) \in s\hat{}*$ **by** *blast*
qed
then show *CCR s* **unfolding** *CCR-def* **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-Cprf-dc-disj-fld-un*:
fixes *S*::*'U rel set* **and** *n*::*nat*
assumes *a1*: $\forall\ s1 \in S.\ \forall s2 \in S.\ s1 \neq s2 \longrightarrow Field\ s1 \cap Field\ s2 = \{\}$
and *a2*: $\forall\ s \in S.\ DCR\ n\ s$
shows $DCR\ n\ (\bigcup\ S)$
proof –
obtain *gi*::*'U rel* $\Rightarrow$ *nat* $\Rightarrow$ *'U rel*
where *b1*: $gi = (\lambda\ s.\ (SOME\ g.\ DCR\text{-}generating\ g \wedge s = \bigcup\{r'.\ \exists \alpha'{<}n.\ r' = g\ \alpha'\}))$ **by** *blast*
obtain *ga* **where** *b2*: $ga = (\lambda\ \alpha.\ if\ (\alpha < n)\ then\ \bigcup s{\in}S.\ gi\ s\ \alpha\ else\ \{\})$ **by** *blast*
have *b3*: $\bigwedge\ s.\ s \in S \Longrightarrow DCR\text{-}generating\ (gi\ s) \wedge s = \bigcup\{r'.\ \exists \alpha'{<}n.\ r' = gi\ s\ \alpha'\}$
proof –
fix *s*
assume $s \in S$
then obtain *g* **where** $DCR\text{-}generating\ g \wedge s = \bigcup\{r'.\ \exists \alpha'{<}n.\ r' = g\ \alpha'\}$
using *a2* **unfolding** *DCR-def* **by** *force*
then show $DCR\text{-}generating\ (gi\ s) \wedge s = \bigcup\{r'.\ \exists \alpha'{<}n.\ r' = gi\ s\ \alpha'\}$
using *b1 someI-ex*[$of\ \lambda\ g.\ DCR\text{-}generating\ g \wedge s = \bigcup\{r'.\ \exists \alpha'{<}n.\ r' = g\ \alpha'\}$] **by** *blast*
qed

have $\forall\alpha\ \beta\ a\ b\ c.\ (a,\ b) \in ga\ \alpha \wedge (a,\ c) \in ga\ \beta \longrightarrow$
$(\exists b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ ga\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ ga\ \beta\ \alpha)$
proof (*intro allI impI*)
fix $\alpha\ \beta\ a\ b\ c$
assume *c1*: $(a,\ b) \in ga\ \alpha \wedge (a,\ c) \in ga\ \beta$
moreover have $\alpha < n$ **using** *c1 b2* **by** (*cases* $\alpha{<}n$, *simp+*)
moreover have $\beta < n$ **using** *c1 b2* **by** (*cases* $\beta{<}n$, *simp+*)
ultimately obtain *s1 s2* **where** *c2*: $\alpha < n \wedge s1 \in S \wedge (a,b) \in gi\ s1\ \alpha$
and *c3*: $\beta < n \wedge s2 \in S \wedge (a,c) \in gi\ s2\ \beta$ **using** *c1 b2*
by *fastforce*
then have $(a,b) \in s1 \wedge (a,c) \in s2$ **using** *b3* **by** *blast*
then have $s1 = s2$ **using** *c2 c3 a1* **unfolding** *Field-def* **by** *blast*
then obtain $b'\ b''\ c'\ c''\ d$
where *c4*: $(b,\ b',\ b'',\ d) \in \mathfrak{D}\ (gi\ s1)\ \alpha\ \beta$ **and** *c5*: $(c,\ c',\ c'',\ d) \in \mathfrak{D}\ (gi\ s1)$
$\beta\ \alpha$
using *c2 c3 b3*[*of s1*] **unfolding** *DCR-generating-def* **by** *blast*
have $(b,\ b',\ b'',\ d) \in \mathfrak{D}\ ga\ \alpha\ \beta$
proof −
have *d1*: $(b,\ b') \in (\mathfrak{L}1\ (gi\ s1)\ \alpha)\hat{}{*} \wedge (b',\ b'') \in (gi\ s1\ \beta)\hat{}{=} \wedge (b'',\ d) \in (\mathfrak{L}v$
$(gi\ s1)\ \alpha\ \beta)\hat{}{*}$
using *c4* **unfolding** $\mathfrak{D}$-*def* **by** *blast*
have $\mathfrak{L}1\ (gi\ s1)\ \alpha \subseteq \mathfrak{L}1\ ga\ \alpha$
proof
fix p
assume $p \in \mathfrak{L}1\ (gi\ s1)\ \alpha$
then obtain γ **where** $\gamma < \alpha \wedge p \in gi\ s1\ \gamma$ **unfolding** $\mathfrak{L}1$-*def* **by** *blast*
moreover then have $p \in ga\ \gamma$ **using** *c2 b2* **by** *fastforce*
ultimately show $p \in \mathfrak{L}1\ ga\ \alpha$ **unfolding** $\mathfrak{L}1$-*def* **by** *blast*
qed
then have *d2*: $(b,\ b') \in (\mathfrak{L}1\ ga\ \alpha)\hat{}{*}$ **using** *d1 rtrancl-mono* **by** *blast*
have $gi\ s1\ \beta \subseteq ga\ \beta$ **using** *c2 c3 b2* **by** *fastforce*
then have *d3*: $(b',\ b'') \in (ga\ \beta)\hat{}{=}$ **using** *d1* **by** *blast*
have $\mathfrak{L}v\ (gi\ s1)\ \alpha\ \beta \subseteq \mathfrak{L}v\ ga\ \alpha\ \beta$
proof
fix p
assume $p \in \mathfrak{L}v\ (gi\ s1)\ \alpha\ \beta$
then obtain γ **where** $(\gamma < \alpha \vee \gamma < \beta) \wedge p \in gi\ s1\ \gamma$ **unfolding** $\mathfrak{L}v$-*def*
by *blast*
moreover then have $p \in ga\ \gamma$ **using** *c2 c3 b2* **by** *fastforce*
ultimately show $p \in \mathfrak{L}v\ ga\ \alpha\ \beta$ **unfolding** $\mathfrak{L}v$-*def* **by** *blast*
qed
then have $(b'',\ d) \in (\mathfrak{L}v\ ga\ \alpha\ \beta)\hat{}{*}$ **using** *d1 rtrancl-mono* **by** *blast*
then show *?thesis* **using** *d2 d3* **unfolding** $\mathfrak{D}$-*def* **by** *blast*
qed
moreover have $(c,\ c',\ c'',\ d) \in \mathfrak{D}\ ga\ \beta\ \alpha$
proof −
have *d1*: $(c,\ c') \in (\mathfrak{L}1\ (gi\ s1)\ \beta)\hat{}{*} \wedge (c',\ c'') \in (gi\ s1\ \alpha)\hat{}{=} \wedge (c'',\ d) \in (\mathfrak{L}v$
$(gi\ s1)\ \beta\ \alpha)\hat{}{*}$
using *c5* **unfolding** $\mathfrak{D}$-*def* **by** *blast*

have $\mathfrak{L}1$ $(gi\ s1)$ $\beta \subseteq \mathfrak{L}1\ ga\ \beta$
proof
fix p
assume $p \in \mathfrak{L}1\ (gi\ s1)\ \beta$
then obtain γ **where** $\gamma < \beta \wedge p \in gi\ s1\ \gamma$ **unfolding** *𝔏1-def* **by** *blast*
moreover then have $p \in ga\ \gamma$ **using** *c2 c3 b2* **by** *fastforce*
ultimately show $p \in \mathfrak{L}1\ ga\ \beta$ **unfolding** *𝔏1-def* **by** *blast*
qed
then have *d2*: $(c, c') \in (\mathfrak{L}1\ ga\ \beta)\widehat{\ }*$ **using** *d1 rtrancl-mono* **by** *blast*
have $gi\ s1\ \alpha \subseteq ga\ \alpha$ **using** *c2 b2* **by** *fastforce*
then have *d3*: $(c', c'') \in (ga\ \alpha)\widehat{\ }=$ **using** *d1* **by** *blast*
have $\mathfrak{L}v\ (gi\ s1)\ \beta\ \alpha \subseteq \mathfrak{L}v\ ga\ \beta\ \alpha$
proof
fix p
assume $p \in \mathfrak{L}v\ (gi\ s1)\ \beta\ \alpha$
then obtain γ **where** $(\gamma < \beta \vee \gamma < \alpha) \wedge p \in gi\ s1\ \gamma$ **unfolding** *𝔏v-def* **by** *blast*
moreover then have $p \in ga\ \gamma$ **using** *c2 c3 b2* **by** *fastforce*
ultimately show $p \in \mathfrak{L}v\ ga\ \beta\ \alpha$ **unfolding** *𝔏v-def* **by** *blast*
qed
then have $(c'', d) \in (\mathfrak{L}v\ ga\ \beta\ \alpha)\widehat{\ }*$ **using** *d1 rtrancl-mono* **by** *blast*
then show *?thesis* **using** *d2 d3* **unfolding** *𝔇-def* **by** *blast*
qed
ultimately show $\exists\, b'\ b''\ c'\ c''\ d.\ (b, b', b'', d) \in \mathfrak{D}\ ga\ \alpha\ \beta \wedge (c, c', c'', d) \in \mathfrak{D}\ ga\ \beta\ \alpha$ **by** *blast*
qed
then have *DCR-generating ga* **unfolding** *DCR-generating-def* **by** *blast*
moreover have $\bigcup\ S = \bigcup\{r'.\ \exists\alpha'<n.\ r' = ga\ \alpha'\}$
proof
show $\bigcup\ S \subseteq \bigcup\{r'.\ \exists\alpha'<n.\ r' = ga\ \alpha'\}$
proof
fix p
assume $p \in \bigcup\ S$
then obtain s **where** $s \in S \wedge p \in s$ **by** *blast*
moreover then obtain α **where** $\alpha<n \wedge p \in gi\ s\ \alpha$ **using** *b3* **by** *blast*
ultimately have $\alpha<n \wedge p \in ga\ \alpha$ **using** *b2* **by** *force*
then show $p \in \bigcup\{r'.\ \exists\alpha'<n.\ r' = ga\ \alpha'\}$ **by** *blast*
qed
next
show $\bigcup\{r'.\ \exists\alpha'<n.\ r' = ga\ \alpha'\} \subseteq \bigcup\ S$
proof
fix p
assume $p \in \bigcup\{r'.\ \exists\alpha'<n.\ r' = ga\ \alpha'\}$
then obtain α **where** $\alpha<n \wedge p \in ga\ \alpha$ **by** *blast*
moreover then obtain s **where** $s \in S \wedge p \in gi\ s\ \alpha$ **using** *b2* **by** *force*
ultimately have $s \in S \wedge p \in s$ **using** *b3* **by** *blast*
then show $p \in \bigcup\ S$ **by** *blast*
qed
qed

ultimately show *?thesis* **unfolding** *DCR-def* **by** *blast*
qed

lemma *lem-dc3-to-d3*:
fixes *r*::*'U rel*
assumes *DCR 3 r*
shows *DCR3 r*
proof −
obtain *g* **where** *b1*: *DCR-generating g* **and** *b2*: $r = \bigcup\{r'.\ \exists\alpha'<3.\ r' = g\ \alpha'\}$
using *assms* **unfolding** *DCR-def* **by** *blast*
have $\forall\ \alpha$::*nat*. $\alpha<2 \longleftrightarrow \alpha = 0 \lor \alpha = 1$ **by** *force*
then have *b3*: $\mathfrak{L}1\ g\ 0 = \{\} \land \mathfrak{L}1\ g\ 1 = g\ 0 \land \mathfrak{L}1\ g\ 2 = g\ 0 \cup g\ 1$
$\land\ \mathfrak{L}v\ g\ 0\ 0 = \{\} \land \mathfrak{L}v\ g\ 1\ 0 = g\ 0 \land \mathfrak{L}v\ g\ 0\ 1 = g\ 0 \land \mathfrak{L}v\ g\ 1\ 1 = g\ 0$
$\land\ \mathfrak{L}v\ g\ 2\ 0 = g\ 0 \cup g\ 1 \land \mathfrak{L}v\ g\ 2\ 1 = g\ 0 \cup g\ 1$
$\land\ \mathfrak{L}v\ g\ 2\ 2 = g\ 0 \cup g\ 1 \land \mathfrak{L}v\ g\ 0\ 2 = g\ 0 \cup g\ 1 \land \mathfrak{L}v\ g\ 1\ 2 = g\ 0 \cup g\ 1$
unfolding $\mathfrak{L}$*1-def* $\mathfrak{L}$*v-def* **by** (*simp-all*, *blast*+)
have $r = (g\ 0) \cup (g\ 1) \cup (g\ 2)$
proof
show $r \subseteq (g\ 0) \cup (g\ 1) \cup (g\ 2)$
proof
fix *p*
assume $p \in r$
then obtain α **where** $p \in g\ \alpha \land \alpha < 3$ **using** *b2* **by** *blast*
moreover have $\forall\ \alpha$::*nat*. $\alpha<3 \longleftrightarrow \alpha = 0 \lor \alpha = 1 \lor \alpha = 2$ **by** *force*
ultimately show $p \in (g\ 0) \cup (g\ 1) \cup (g\ 2)$ **by** *force*
qed
next
have (*0*::*nat*) < (*3*::*nat*) ∧ (*1*::*nat*) < (*3*::*nat*) ∧ (*2*::*nat*) < (*3*::*nat*) **by** *simp*
then show $(g\ 0) \cup (g\ 1) \cup (g\ 2) \subseteq r$ **using** *b2* **by** *blast*
qed
moreover have $\forall\ a\ b\ c.\ (a,b) \in (g\ 0) \land (a,c) \in (g\ 0) \longrightarrow$ *jn00* (*g 0*) *b c*
proof (*intro allI impI*)
fix *a b c*
assume $(a,b) \in (g\ 0) \land (a,c) \in (g\ 0)$
then obtain $b'\ b''\ c'\ c''\ d$ **where** $(b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ 0\ 0 \land (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ 0\ 0$
using *b1* **unfolding** *DCR-generating-def* **by** *blast*
then show *jn00* (*g 0*) *b c* **unfolding** *jn00-def* $\mathfrak{D}$*-def* $\mathfrak{L}$*1-def* $\mathfrak{L}$*v-def* **by** *force*
qed
moreover have $\forall\ a\ b\ c.\ (a,b) \in (g\ 0) \land (a,c) \in (g\ 1) \longrightarrow$ *jn01* (*g 0*) (*g 1*) *b c*
proof (*intro allI impI*)
fix *a b c*
assume $(a,b) \in (g\ 0) \land (a,c) \in (g\ 1)$
then obtain $b'\ b''\ c'\ c''\ d$ **where**
$(b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ 0\ 1 \land (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ 1\ 0$
using *b1* **unfolding** *DCR-generating-def* **by** *blast*
then show *jn01* (*g 0*) (*g 1*) *b c* **unfolding** *jn01-def* $\mathfrak{D}$*-def* $\mathfrak{L}$*1-def* $\mathfrak{L}$*v-def* **by** *force*
qed

moreover have ∀ *a b c*. (*a*,*b*) ∈ (*g 1*) ∧ (*a*,*c*) ∈ (*g 1*) ⟶ *jn11* (*g 0*) (*g 1*) *b c*
proof (*intro allI impI*)
fix *a b c*
assume (*a*,*b*) ∈ (*g 1*) ∧ (*a*,*c*) ∈ (*g 1*)
then obtain *b′ b′′ c′ c′′ d* **where** (*b*, *b′*, *b′′*, *d*) ∈ 𝔇 *g 1 1* ∧ (*c*, *c′*, *c′′*, *d*) ∈ 𝔇 *g 1 1*
using *b1* **unfolding** *DCR-generating-def* **by** *blast*
then show *jn11* (*g 0*) (*g 1*) *b c* **unfolding** *jn11-def* 𝔇*-def*
apply (*simp only*: *b3*)
by *blast*
qed
moreover have ∀ *a b c*. (*a*,*b*) ∈ (*g 0*) ∧ (*a*,*c*) ∈ (*g 2*) ⟶ *jn02* (*g 0*) (*g 1*) (*g 2*) *b c*
proof (*intro allI impI*)
fix *a b c*
assume (*a*,*b*) ∈ (*g 0*) ∧ (*a*,*c*) ∈ (*g 2*)
then obtain *b′ b′′ c′ c′′ d* **where** *c1*: (*b*, *b′*, *b′′*, *d*) ∈ 𝔇 *g 0 2* ∧ (*c*, *c′*, *c′′*, *d*) ∈ 𝔇 *g 2 0*
using *b1* **unfolding** *DCR-generating-def* **by** *blast*
then have (*c*, *c′*) ∈ (*g 0* ∪ *g 1*)^* ∧ (*c′*,*c′′*) ∈ (*g 0*)^= ∧ (*c′′*,*d*) ∈ (*g 0* ∪ *g 1*)^*
unfolding 𝔇*-def* **by** (*simp add*: *b3*)
moreover then have (*c′*,*c′′*) ∈ (*g 0* ∪ *g 1*)^* **by** *blast*
ultimately have (*c*, *d*) ∈ (*g 0* ∪ *g 1*)^* **by** *force*
then show *jn02* (*g 0*) (*g 1*) (*g 2*) *b c*
using *c1* **unfolding** *jn02-def* 𝔇*-def*
apply (*simp add*: *b3*)
by *blast*
qed
moreover have ∀ *a b c*. (*a*,*b*) ∈ (*g 1*) ∧ (*a*,*c*) ∈ (*g 2*) ⟶ *jn12* (*g 0*) (*g 1*) (*g 2*) *b c*
proof (*intro allI impI*)
fix *a b c*
assume (*a*,*b*) ∈ (*g 1*) ∧ (*a*,*c*) ∈ (*g 2*)
then obtain *b′ b′′ c′ c′′ d* **where** *c1*: (*b*, *b′*, *b′′*, *d*) ∈ 𝔇 *g 1 2* ∧ (*c*, *c′*, *c′′*, *d*) ∈ 𝔇 *g 2 1*
using *b1* **unfolding** *DCR-generating-def* **by** *blast*
then have (*c*, *c′*) ∈ (*g 0* ∪ *g 1*)^* ∧ (*c′*,*c′′*) ∈ (*g 1*)^= ∧ (*c′′*,*d*) ∈ (*g 0* ∪ *g 1*)^*
unfolding 𝔇*-def* **apply** (*simp only*: *b3*)
by *blast*
moreover then have (*c′*,*c′′*) ∈ (*g 0* ∪ *g 1*)^* **by** *blast*
ultimately have (*c*, *d*) ∈ (*g 0* ∪ *g 1*)^* **by** *force*
then show *jn12* (*g 0*) (*g 1*) (*g 2*) *b c*
using *c1* **unfolding** *jn12-def* 𝔇*-def* **apply** (*simp only*: *b3*)
by *blast*
qed
moreover have ∀ *a b c*. (*a*,*b*) ∈ (*g 2*) ∧ (*a*,*c*) ∈ (*g 2*) ⟶ *jn22* (*g 0*) (*g 1*) (*g 2*) *b c*

proof (*intro allI impI*)
fix *a b c*
assume $(a,b) \in (g\ 2) \wedge (a,c) \in (g\ 2)$
then obtain b' b'' c' c'' *d* **where** *c1*: $(b, b', b'', d) \in \mathfrak{D}\ g\ 2\ 2 \wedge (c, c', c'', d) \in \mathfrak{D}\ g\ 2\ 2$
using *b1* **unfolding** *DCR-generating-def* **by** *blast*
then show *jn22* (*g 0*) (*g 1*) (*g 2*) *b c*
unfolding *jn22-def* $\mathfrak{D}$*-def* **apply** (*simp only*: *b3*)
by *blast*
qed
ultimately have *LD3 r* (*g 0*) (*g 1*) (*g 2*) **unfolding** *LD3-def* **by** *blast*
then show *?thesis* **unfolding** *DCR3-def* **by** *blast*
qed

lemma *lem-dc3-confl-lewsuc*:
fixes *r*::*'U rel*
assumes *a1*: *confl-rel r* **and** *a2*: |*Field r*| $\leq$*o cardSuc* |*UNIV*::*nat set*|
shows *DCR 3 r*
proof −
obtain *S* **where** *b1*: $r = \bigcup S$
and *b2*: $\forall\ s1 \in S.\ \forall\ s2 \in S.\ s1 \neq s2 \longrightarrow$ *Field s1* $\cap$ *Field s2* = {}
and *b3*: $\forall\ s \in S.$ *CCR s* **using** *a1 lem-Cprf-conf-ccr-decomp*[*of r*] **by** *blast*
have $\forall\ s \in S.$ *DCR 3 s*
proof
fix *s*
assume $s \in S$
then have *CCR s* $\wedge$ *Field s* $\subseteq$ *Field r* **using** *b1 b3* **unfolding** *Field-def* **by** *blast*
moreover then have |*Field s*| $\leq$*o* |*Field r*| **by** *simp*
ultimately have *CCR s* $\wedge$ |*Field s*| $\leq$*o cardSuc* |*UNIV*::*nat set*| **using** *a2 ordLeq-transitive* **by** *blast*
then show *DCR 3 s* **using** *lem-dc3-ccr-scf-lewsuc* **by** *blast*
qed
then show *DCR 3 r* **using** *b1 b2 lem-Cprf-dc-disj-fld-un*[*of S*] **by** *blast*
qed

lemma *lem-cle-eqdef*: |*A*| $\leq$*o* |*B*| = ($\exists$ *g* . $A \subseteq g$‘*B*)
by (*metis surj-imp-ordLeq card-of-ordLeq2 empty-subsetI order-refl*)

lemma *lem-cardLeN1-eqdef*:
fixes *A*::*'a set*
shows *cardLeN1 A* = (|*A*| $\leq$*o cardSuc* |{*n*::*nat* . *True*}|)
proof
assume *b1*: *cardLeN1 A*
obtain κ **where** *b2*: κ = *cardSuc* |*UNIV*::*nat set*| **by** *blast*
have *cardSuc* |*UNIV*::*nat set*| <*o* |*A*| $\longrightarrow$ *False*
proof
assume *cardSuc* |*UNIV*::*nat set*| <*o* |*A*|

then have *c1*: $\kappa <o\ |A| \wedge |Field\ \kappa| =o\ \kappa$ **using** *b2* **by** *simp*
then have $|Field\ \kappa| \leq o\ |A|$ **using** *ordIso-ordLess-trans ordLess-imp-ordLeq* **by** *blast*
then obtain B **where** *c2*: $B \subseteq A \wedge |Field\ \kappa| =o\ |B|$
using *internalize-card-of-ordLeq2*[*of Field* κ *A*] **by** *blast*
moreover have $|UNIV::nat\ set| <o\ \kappa$ **using** *b2* **by** *simp*
ultimately have *c3*: $B \subseteq A \wedge |UNIV::nat\ set| <o\ |B|$
using *c1* **by** (*meson ordIso-imp-ordLeq ordIso-symmetric ordLess-ordLeq-trans*)
then obtain C **where** *c4*: $C \subseteq B \wedge |UNIV::nat\ set| =o\ |C|$
using *internalize-card-of-ordLeq2*[*of UNIV::nat set B*] *ordLess-imp-ordLeq* **by** *blast*
obtain c **where** $c \in C$ **using** *c4* **using** *card-of-empty2* **by** *fastforce*
moreover obtain D **where** *c5*: $D = C - \{c\}$ **by** *blast*
ultimately have *c6*: $C = D \cup \{c\}$ **by** *blast*
have $\neg$ *finite D* **using** *c4 c5* **using** *card-of-ordIso-finite* **by** *force*
moreover then have $|\{c\}| \leq o\ |D|$ **by** (*metis card-of-singl-ordLeq finite.emptyI*)
ultimately have $|C| \leq o\ |D|$ **using** *c6* **using** *card-of-Un-infinite ordIso-imp-ordLeq* **by** *blast*
then obtain f **where** $C \subseteq f\ `\ D$ **by** (*metis card-of-ordLeq2 empty-subsetI order-refl*)
moreover have $D \subset C \wedge C \subseteq B \wedge B \subseteq A$ **using** *c3 c4 c5 c6* **by** *blast*
ultimately have $(\exists f.\ B \subseteq f\ `\ C) \vee (\exists g.\ A \subseteq g`B)$ **using** *b1* **unfolding** *cardLeN1-def* **by** *metis*
moreover have $(\exists f.\ B \subseteq f\ `\ C) \longrightarrow False$
proof
assume $\exists f.\ B \subseteq f\ `\ C$
then obtain f **where** $B \subseteq f\ `\ C$ **by** *blast*
then have $|B| \leq o\ |f`C|$ **by** *simp*
moreover have $|f`C| \leq o\ |C|$ **by** *simp*
ultimately have $|B| \leq o\ |C|$ **using** *ordLeq-transitive* **by** *blast*
then show *False* **using** *c3 c4 not-ordLess-ordIso ordLess-ordLeq-trans* **by** *blast*
qed
moreover have $(\exists g.\ A \subseteq g`B) \longrightarrow False$
proof
assume $\exists g.\ A \subseteq g`B$
then obtain g **where** $A \subseteq g`B$ **by** *blast*
then have $|A| \leq o\ |g`B|$ **by** *simp*
moreover have $|g`B| \leq o\ |B|$ **by** *simp*
ultimately have $|A| \leq o\ |B|$ **using** *ordLeq-transitive* **by** *blast*
then show *False* **using** *c1 c2*
by (*metis BNF-Cardinal-Order-Relation.ordLess-Field not-ordLess-ordIso ordLess-ordLeq-trans*)
qed
ultimately show *False* **by** *blast*
qed
then show $|A| \leq o$ *cardSuc* $|\{n::nat\ .\ True\}|$ **by** *simp*
next
assume $|A| \leq o$ *cardSuc* $|\{n::nat\ .\ True\}|$

then have *b1*: $|A| \leq o$ *cardSuc* $|UNIV::nat\ set|$ **by** *simp*
have $\forall\ B \subseteq A.$ ($\forall\ C \subseteq B$. (($\exists\ D\ f.\ D \subset C \wedge C \subseteq f`D$ ) $\longrightarrow$ ( $\exists\ f.\ B \subseteq f`C$)))
$\vee$ ($\exists\ g$. $A \subseteq g`B$)
proof (*intro allI impI*)
fix *B*
assume $B \subseteq A$
show ($\forall\ C \subseteq B$. (($\exists\ D\ f.\ D \subset C \wedge C \subseteq f`D$ ) $\longrightarrow$ ( $\exists\ f.\ B \subseteq f`C$))) $\vee$ ($\exists\ g$. $A \subseteq g`B$)
proof (*cases* $|B| \leq o\ |UNIV::nat\ set|$)
assume *d1*: $|B| \leq o\ |UNIV::nat\ set|$
have $\forall\ C \subseteq B$. (($\exists\ D\ f.\ D \subset C \wedge C \subseteq f`D$ ) $\longrightarrow$ ( $\exists\ f.\ B \subseteq f`C$))
proof (*intro allI impI*)
fix *C*
assume $C \subseteq B$ **and** $\exists\ D\ f.\ D \subset C \wedge C \subseteq f`D$
then obtain *D f* **where** *e1*: $D \subset C \wedge C \subseteq f`D$ **by** *blast*
have *finite C* $\longrightarrow$ *False*
proof
assume *finite C*
moreover then have *finite D* **using** *e1 finite-subset* **by** *blast*
ultimately have $|D| <o\ |C|$
using *e1* **by** (*metis finite-card-of-iff-card3 psubset-card-mono*)
moreover have $|C| \leq o\ |D|$ **using** *e1* **using** *surj-imp-ordLeq* **by** *blast*
ultimately show *False* **using** *not-ordLeq-ordLess* **by** *blast*
qed
then have $|B| \leq o\ |C|$ **using** *d1* **by** (*metis infinite-iff-card-of-nat ordLeq-transitive*)
then show $\exists\ f.\ B \subseteq f`C$ **by** (*metis card-of-ordLeq2 empty-subsetI order-refl*)
qed
then show *?thesis* **by** *blast*
next
assume $\neg\ |B| \leq o\ |UNIV::nat\ set|$
then have $|A| \leq o\ |B|$ **using** *b1 lem-cord-lin*
by (*metis cardSuc-ordLeq-ordLess card-of-Card-order ordLess-ordLeq-trans*)
then have $\exists\ g$. $A \subseteq g`B$ **by** (*metis card-of-ordLeq2 empty-subsetI order-refl*)
then show *?thesis* **by** *blast*
qed
qed
then show *cardLeN1 A* **unfolding** *cardLeN1-def* **by** *blast*
qed

lemma *lem-cleN1-eqdef*:
fixes *r*::($'U \times 'U$) *set*
shows ($|r| \leq o$ *cardSuc* $|\{n::nat\ .\ True\}|$)
$\longleftrightarrow$ ($\forall\ s \subseteq r.$ (($\forall\ t \subseteq s$. (($\exists\ t'\ f.\ t' \subset t \wedge t \subseteq f`t'$) $\longrightarrow$ ($\exists\ f.\ s \subseteq f`t$)))
$\vee$ ($\exists\ g$. $r \subseteq g`s$)
))
using *lem-cardLeN1-eqdef*[*of r*] *cardLeN1-def* **by** *blast*

1.2.3 Result

The next theorem has the following meaning: if the cardinality of a confluent binary relation r does not exceed the first uncountable cardinal, then confluence of r can be proved with the help of the decreasing diagrams method using no more than 3 labels (e.g. 0, 1, 2 ordered in the usual way).

theorem *thm-main*:
fixes r::('U×'U) *set*
assumes ∀ *a b c* . (*a*,*b*) ∈ *r*^∗ ∧ (*a*,*c*) ∈ *r*^∗ ⟶ (∃ *d*. (*b*,*d*) ∈ *r*^∗ ∧ (*c*,*d*) ∈ *r*^∗)
  and |*r*| ≤*o* *cardSuc* |{*n*::*nat* . *True*}|
shows ∃ *r0 r1 r2* . (
  (*r* = (*r0* ∪ *r1* ∪ *r2*))
  ∧ (∀ *a b c*. (*a*,*b*) ∈ *r0* ∧ (*a*,*c*) ∈ *r0*
    ⟶ (∃ *d*.
      (*b*,*d*) ∈ *r0*^=
      ∧ (*c*,*d*) ∈ *r0*^=))
  ∧ (∀ *a b c*. (*a*,*b*) ∈ *r0* ∧ (*a*,*c*) ∈ *r1*
    ⟶ (∃ *b′ d*.
      (*b*,*b′*) ∈ *r1*^= ∧ (*b′*,*d*) ∈ *r0*^∗
      ∧ (*c*,*d*) ∈ *r0*^∗))
  ∧ (∀ *a b c*. (*a*,*b*) ∈ *r1* ∧ (*a*,*c*) ∈ *r1*
    ⟶ (∃ *b′ b″ c′ c″ d*.
      (*b*,*b′*) ∈ *r0*^∗ ∧ (*b′*,*b″*) ∈ *r1*^= ∧ (*b″*,*d*) ∈ *r0*^∗
      ∧ (*c*,*c′*) ∈ *r0*^∗ ∧ (*c′*,*c″*) ∈ *r1*^= ∧ (*c″*,*d*) ∈ *r0*^∗))
  ∧ (∀ *a b c*. (*a*,*b*) ∈ *r0* ∧ (*a*,*c*) ∈ *r2*
    ⟶ (∃ *b′ d*.
      (*b*,*b′*) ∈ *r2*^= ∧ (*b′*,*d*) ∈ (*r0* ∪ *r1*)^∗
      ∧ (*c*,*d*) ∈ (*r0* ∪ *r1*)^∗))
  ∧ (∀ *a b c*. (*a*,*b*) ∈ *r1* ∧ (*a*,*c*) ∈ *r2*
    ⟶ (∃ *b′ b″ d*.
      (*b*,*b′*) ∈ *r0*^∗ ∧ (*b′*,*b″*) ∈ *r2*^= ∧ (*b″*,*d*) ∈ (*r0* ∪ *r1*)^∗
      ∧ (*c*,*d*) ∈ (*r0* ∪ *r1*)^∗))
  ∧ (∀ *a b c*. (*a*,*b*) ∈ *r2* ∧ (*a*,*c*) ∈ *r2*
    ⟶ (∃ *b′ b″ c′ c″ d*.
      (*b*,*b′*) ∈ (*r0* ∪ *r1*)^∗ ∧ (*b′*,*b″*) ∈ *r2*^= ∧ (*b″*,*d*) ∈ (*r0* ∪ *r1*)^∗
      ∧ (*c*,*c′*) ∈ (*r0* ∪ *r1*)^∗ ∧ (*c′*,*c″*) ∈ *r2*^= ∧ (*c″*,*d*) ∈ (*r0* ∪ *r1*)^∗
))
)
proof −
 have *b0*: |*r*| ≤*o* *cardSuc* |*UNIV*::*nat set*| **using** *assms*(*2*) **by** *simp*
 obtain κ **where** *b1*: κ = *cardSuc* |*UNIV*::*nat set*| **by** *blast*
 have |*Field r*| ≤*o* κ
 proof (*cases finite r*)
  assume *finite r*
  then show *?thesis* **using** *b1 lem-fin-fl-rel* **by** (*metis Field-card-of Field-natLeq cardSuc-ordLeq-ordLess*
   card-of-card-order-on card-of-mono2 finite-iff-ordLess-natLeq ordLess-imp-ordLeq)
 next
  assume ¬ *finite r*

then show *?thesis* **using** *b0 b1 lem-rel-inf-fld-card* **using** *ordIso-ordLeq-trans* **by** *blast*
qed
moreover have *confl-rel r* **using** *assms(1)* **unfolding** *confl-rel-def* **by** *blast*
ultimately have *DCR3 r* **using** *b1 lem-dc3-confl-lewsuc[of r] lem-dc3-to-d3* **by** *blast*
then show *?thesis* **unfolding** *DCR3-def LD3-def*
jn00-def jn01-def jn02-def jn11-def jn12-def jn22-def **by** *fast*
qed

end

1.3 Optimality of the DCR3 method for proving confluence of relations of the least uncountable cardinality

theory *DCR3-Optimality*
imports
HOL−Cardinals.Cardinals
Finite-DCR-Hierarchy
begin

1.3.1 Auxiliary definitions

datatype $Lev = \mathrm{l}0 \mid \mathrm{l}1 \mid \mathrm{l}2 \mid \mathrm{l}3 \mid \mathrm{l}4 \mid \mathrm{l}5 \mid \mathrm{l}6 \mid \mathrm{l}7 \mid \mathrm{l}8$

type-synonym $'U\ rD = Lev \times 'U\ set \times 'U\ set \times 'U\ set$

fun $rP :: Lev \Rightarrow 'U\ set \Rightarrow 'U\ set \Rightarrow 'U\ set \Rightarrow Lev \Rightarrow 'U\ set \Rightarrow 'U\ set \Rightarrow 'U\ set \Rightarrow bool$
where
$rP\ \mathrm{l}0\ A\ B\ C\ n'\ A'\ B'\ C' = (A = \{\} \wedge B = \{\} \wedge C = \{\} \wedge n' = \mathrm{l}1 \wedge finite\ A' \wedge B' = \{\} \wedge C' = \{\})$
$\mid rP\ \mathrm{l}1\ A\ B\ C\ n'\ A'\ B'\ C' = (finite\ A \wedge B = \{\} \wedge C = \{\} \wedge n' = \mathrm{l}2 \wedge A' = A \wedge B' = \{\} \wedge C' = \{\})$
$\mid rP\ \mathrm{l}2\ A\ B\ C\ n'\ A'\ B'\ C' = (finite\ A \wedge B = \{\} \wedge C = \{\} \wedge n' = \mathrm{l}3 \wedge A' = A \wedge finite\ B' \wedge C' = \{\})$
$\mid rP\ \mathrm{l}3\ A\ B\ C\ n'\ A'\ B'\ C' = (finite\ A \wedge finite\ B \wedge C = \{\} \wedge n' = \mathrm{l}4 \wedge A' = A \wedge B' = B \wedge C' = \{\})$
$\mid rP\ \mathrm{l}4\ A\ B\ C\ n'\ A'\ B'\ C' = (finite\ A \wedge finite\ B \wedge C = \{\} \wedge n' = \mathrm{l}5 \wedge A' = A \wedge B' = B \wedge finite\ C')$
$\mid rP\ \mathrm{l}5\ A\ B\ C\ n'\ A'\ B'\ C' = (finite\ A \wedge finite\ B \wedge finite\ C \wedge n' = \mathrm{l}6 \wedge A' = A \wedge B' = B \wedge C' = C)$
$\mid rP\ \mathrm{l}6\ A\ B\ C\ n'\ A'\ B'\ C' = (finite\ A \wedge finite\ B \wedge finite\ C \wedge n' = \mathrm{l}7 \wedge A' = A \cup B \cup C \wedge B' = A' \wedge C' = A')$
$\mid rP\ \mathrm{l}7\ A\ B\ C\ n'\ A'\ B'\ C' = (finite\ A \wedge B = A \wedge C = A \wedge n' = \mathrm{l}8 \wedge A' = A \wedge B' = A' \wedge C' = A')$
$\mid rP\ \mathrm{l}8\ A\ B\ C\ n'\ A'\ B'\ C' = (finite\ A \wedge B = A \wedge C = A \wedge n' = \mathrm{l}7 \wedge A \subset A' \wedge finite\ A' \wedge B' = A' \wedge C' = A')$

definition $rC :: 'U\ set \Rightarrow 'U\ set \Rightarrow 'U\ set \Rightarrow 'U\ set \Rightarrow bool$

where
  $rC\ S\ A\ B\ C = (A \subseteq S \wedge B \subseteq S \wedge C \subseteq S)$

definition $rE :: 'U\ set \Rightarrow ('U\ rD)\ rel$
where
  $rE\ S = \{\ ((n1,\ A1,\ B1,\ C1),\ (n2,\ A2,\ B2,\ C2)).\ rP\ n1\ A1\ B1\ C1\ n2\ A2\ B2\ C2 \wedge rC\ S\ A1\ B1\ C1 \wedge rC\ S\ A2\ B2\ C2\ \}$

fun *lev-next* :: *Lev* ⇒ *Lev*
where
  lev-next l*0* = l*1*
| *lev-next* l*1* = l*2*
| *lev-next* l*2* = l*3*
| *lev-next* l*3* = l*4*
| *lev-next* l*4* = l*5*
| *lev-next* l*5* = l*6*
| *lev-next* l*6* = l*7*
| *lev-next* l*7* = l*8*
| *lev-next* l*8* = l*7*

fun *levrd* :: $'U\ rD \Rightarrow Lev$
where
  $levrd\ (n,\ A,\ B,\ C) = n$

fun *wrd* :: $'U\ rD \Rightarrow 'U\ set$
where
  $wrd\ (n,\ A,\ B,\ C) = A \cup B \cup C$

definition *Wrd* :: $'U\ rD\ set \Rightarrow 'U\ set$
where
  $Wrd\ S = (\bigcup\ (wrd\ `\ S))$

definition *bkset* :: $'a\ rel \Rightarrow 'a\ set \Rightarrow 'a\ set$
where
  $bkset\ r\ A = ((r\hat{}*)\hat{}-1)``A$

1.3.2 Auxiliary lemmas

lemma *lem-rtr-field*: $(x,y) \in r\hat{}* \Longrightarrow (x = y) \vee (x \in Field\ r \wedge y \in Field\ r)$
  by (*metis Field-def Not-Domain-rtrancl Range.RangeI UnCI rtranclE*)

lemma *lem-fin-fl-rel*: *finite* (*Field r*) = *finite r*
  using *finite-Field finite-subset trancl-subset-Field2* **by** *fastforce*

lemma *lem-rel-inf-fld-card*:
fixes $r::'U\ rel$
assumes ¬ *finite r*
shows $|Field\ r| =o\ |r|$
proof −

obtain *f1*::′*U* × ′*U* ⇒ ′*U* **where** *b1*: *f1* = (λ (*x*,*y*). *x*) **by** *blast*
obtain *f2*::′*U* × ′*U* ⇒ ′*U* **where** *b2*: *f2* = (λ (*x*,*y*). *y*) **by** *blast*
then have *f1* ‘ *r* = *Domain r* ∧ *f2* ‘ *r* = *Range r* **using** *b1 b2* **by** *force*
then have *b3*: |*Domain r*| ≤*o* |*r*| ∧ |*Range r*| ≤*o* |*r*|
using *card-of-image*[*of f1 r*] *card-of-image*[*of f2 r*] **by** *simp*
have |*Domain r*| ≤*o* |*Range r*| ∨ |*Range r*| ≤*o* |*Domain r*| **by** (*simp add*: *ordLeq-total*)
moreover have |*Domain r*| ≤*o* |*Range r*| ⟶ |*Field r*| ≤*o* |*r*|
proof
assume *c1*: |*Domain r*| ≤*o* |*Range r*|
moreover have *finite* (*Domain r*) ∧ *finite* (*Range r*) ⟶ *finite* (*Field r*) **unfolding** *Field-def* **by** *blast*
ultimately have ¬ *finite* (*Range r*)
using *assms lem-fin-fl-rel card-of-ordLeq-finite* **by** *blast*
then have |*Field r*| =*o* |*Range r*| **using** *c1 card-of-Un-infinite* **unfolding** *Field-def* **by** *blast*
then show |*Field r*| ≤*o* |*r*| **using** *b3 ordIso-ordLeq-trans* **by** *blast*
qed
moreover have |*Range r*| ≤*o* |*Domain r*| ⟶ |*Field r*| ≤*o* |*r*|
proof
assume *c1*: |*Range r*| ≤*o* |*Domain r*|
moreover have *finite* (*Domain r*) ∧ *finite* (*Range r*) ⟶ *finite* (*Field r*) **unfolding** *Field-def* **by** *blast*
ultimately have ¬ *finite* (*Domain r*)
using *assms lem-fin-fl-rel card-of-ordLeq-finite* **by** *blast*
then have |*Field r*| =*o* |*Domain r*| **using** *c1 card-of-Un-infinite* **unfolding** *Field-def* **by** *blast*
then show |*Field r*| ≤*o* |*r*| **using** *b3 ordIso-ordLeq-trans* **by** *blast*
qed
ultimately have |*Field r*| ≤*o* |*r*| **by** *blast*
moreover have |*r*| ≤*o* |*Field r*|
proof −
have *r* ⊆ (*Field r*) × (*Field r*) **unfolding** *Field-def* **by** *force*
then have *c1*: |*r*| ≤*o* |*Field r* × *Field r*| **by** *simp*
have ¬ *finite* (*Field r*) **using** *assms lem-fin-fl-rel* **by** *blast*
then have *c2*: |*Field r* × *Field r*| =*o* |*Field r*| **by** *simp*
show *?thesis* **using** *c1 c2* **using** *ordLeq-ordIso-trans* **by** *blast*
qed
ultimately show *?thesis* **using** *ordIso-iff-ordLeq* **by** *blast*
qed

lemma *lem-confl-field*: *confl-rel r* = (∀ *a* ∈ *Field r*. ∀ *b* ∈ *Field r*. ∀ *c* ∈ *Field r*. (*a*,*b*) ∈ *r*^∗ ∧ (*a*,*c*) ∈ *r*^∗ ⟶
(∃ *d* ∈ *Field r*. (*b*,*d*) ∈ *r*^∗ ∧ (*c*,*d*) ∈ *r*^∗))
proof
assume *b1*: *confl-rel r*
show ∀ *a* ∈ *Field r*. ∀ *b* ∈ *Field r*. ∀ *c* ∈ *Field r*. (*a*,*b*) ∈ *r*^∗ ∧ (*a*,*c*) ∈ *r*^∗ ⟶

(∃ *d* ∈ *Field r*. (*b*,*d*) ∈ *r*^∗ ∧ (*c*,*d*) ∈ *r*^∗)

proof (*intro ballI impI*)
fix *a b c*
assume *c1*: $a \in$ *Field r* **and** *c2*: $b \in$ *Field r* **and** *c3*: $c \in$ *Field r* **and** *c4*: $(a,b) \in r\hat{}* \wedge (a,c) \in r\hat{}*$
obtain *d* **where** $(b,d) \in r\hat{}* \wedge (c,d) \in r\hat{}*$ **using** *b1 c4* **unfolding** *confl-rel-def* **by** *blast*
moreover then have $d \in$ *Field r* **using** *c2* **using** *lem-rtr-field* **by** *fastforce*
ultimately show $\exists\ d \in$ *Field r*. $(b,d) \in r\hat{}* \wedge (c,d) \in r\hat{}*$ **by** *blast*
qed
next
assume *b1*: $\forall\ a \in$ *Field r*. $\forall\ b \in$ *Field r*. $\forall\ c \in$ *Field r*. $(a,b) \in r\hat{}* \wedge (a,c) \in r\hat{}* \longrightarrow$
$(\exists\ d \in$ *Field r*. $(b,d) \in r\hat{}* \wedge (c,d) \in r\hat{}*)$
have $\forall a\ b\ c.\ (a,\ b) \in r\hat{}* \wedge (a,\ c) \in r\hat{}* \longrightarrow (\exists d.\ (b,\ d) \in r\hat{}* \wedge (c,\ d) \in r\hat{}*)$
proof (*intro allI impI*)
fix *a b c*
assume $(a,\ b) \in r\hat{}* \wedge (a,\ c) \in r\hat{}*$
moreover then have $a \notin$ *Field r* $\vee\ b \notin$ *Field r* $\vee\ c \notin$ *Field r* $\longrightarrow a = b \vee a = c$ **by** (*meson lem-rtr-field*)
ultimately show $\exists d.\ (b,\ d) \in r\hat{}* \wedge (c,\ d) \in r\hat{}*$ **using** *b1* **by** *blast*
qed
then show *confl-rel r* **unfolding** *confl-rel-def* **by** *blast*
qed

lemma *lem-d2-to-dc2*:
fixes *r*::*'U rel*
assumes *DCR2 r*
shows *DCR 2 r*
proof −
obtain *r0 r1* **where** *b1*: $r = r0 \cup r1$
and *b2*: $\forall\ a\ b\ c.\ (a,b) \in r0 \wedge (a,c) \in r0 \longrightarrow$ *jn00 r0 b c*
and *b3*: $\forall\ a\ b\ c.\ (a,b) \in r0 \wedge (a,c) \in r1 \longrightarrow$ *jn01 r0 r1 b c*
and *b4*: $\forall\ a\ b\ c.\ (a,b) \in r1 \wedge (a,c) \in r1 \longrightarrow$ *jn11 r0 r1 b c*
using *assms* **unfolding** *DCR2-def LD2-def* **by** *blast*
obtain *g*::*nat* $\Rightarrow$ *'U rel*
where *b5*: $g = (\lambda\ \alpha$::*nat*. *if* $\alpha = 0$ *then r0 else* (*if* $\alpha = 1$ *then r1 else* {})) **by** *blast*
have *b6*: $g\ 0 = r0 \wedge g\ 1 = r1$ **using** *b5* **by** *simp*
have *b7*: $\forall\ n.\ (\neg\ (n = 0 \vee n = 1)) \longrightarrow g\ n = \{\}$ **using** *b5* **by** *simp*
have $\forall \alpha\ \beta\ a\ b\ c.\ (a,\ b) \in g\ \alpha \wedge (a,\ c) \in g\ \beta \longrightarrow$
$(\exists b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ \beta\ \alpha)$
proof (*intro allI impI*)
fix $\alpha\ \beta$ *a b c*
assume *c1*: $(a,\ b) \in g\ \alpha \wedge (a,\ c) \in g\ \beta$
then have *c2*: $(\alpha = 0 \vee \alpha = 1) \wedge (\beta = 0 \vee \beta = 1)$ **using** *b7* **by** *blast*
show $\exists b'\ b''\ c'\ c''\ d.\ (b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ \beta\ \alpha$
proof −
have $\alpha = 0 \wedge \beta = 0 \longrightarrow$ *?thesis*
proof

assume *e1*: $\alpha = 0 \wedge \beta = 0$
then have *jn00 r0 b c* **using** *c1 b2 b6* **by** *blast*
then obtain d **where** $(b, d) \in r0\hat{}{=} \wedge (c, d) \in r0\hat{}{=}$ **unfolding** *jn00-def* **by** *blast*
then have $(b, b, d, d) \in \mathfrak{D}\ g\ 0\ 0 \wedge (c, c, d, d) \in \mathfrak{D}\ g\ 0\ 0$ **using** *b6* **unfolding** $\mathfrak{D}$*-def* **by** *blast*
then show $\exists\, b'\ b''\ c'\ c''\ d.\ (b, b', b'', d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c, c', c'', d) \in \mathfrak{D}\ g\ \beta\ \alpha$ **using** *e1* **by** *blast*
qed
moreover have $\alpha = 0 \wedge \beta = 1 \longrightarrow ?thesis$
proof
assume *e1*: $\alpha = 0 \wedge \beta = 1$
then have *jn01 r0 r1 b c* **using** *c1 b3 b6* **by** *blast*
then obtain b'' d **where** $(b,b'') \in r1\hat{}{=} \wedge (b'',d) \in r0\hat{}{*} \wedge (c,d) \in r0\hat{}{*}$ **unfolding** *jn01-def* **by** *blast*
moreover have $\mathfrak{L}v\ g\ 0\ 1 = g\ 0 \wedge \mathfrak{L}v\ g\ 1\ 0 = g\ 0$ **using** *b6 b7* **unfolding** $\mathfrak{L}v$*-def* **by** *blast*
ultimately have $(b, b, b'', d) \in \mathfrak{D}\ g\ 0\ 1 \wedge (c, c, c, d) \in \mathfrak{D}\ g\ 1\ 0$ **using** *b6* **unfolding** $\mathfrak{D}$*-def* **by** *simp*
then show $\exists\, b'\ b''\ c'\ c''\ d.\ (b, b', b'', d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c, c', c'', d) \in \mathfrak{D}\ g\ \beta\ \alpha$ **using** *e1* **by** *blast*
qed
moreover have $\alpha = 1 \wedge \beta = 0 \longrightarrow ?thesis$
proof
assume *e1*: $\alpha = 1 \wedge \beta = 0$
then have *jn01 r0 r1 c b* **using** *c1 b3 b6* **by** *blast*
then obtain c'' d **where** $(c,c'') \in r1\hat{}{=} \wedge (c'',d) \in r0\hat{}{*} \wedge (b,d) \in r0\hat{}{*}$ **unfolding** *jn01-def* **by** *blast*
moreover have $\mathfrak{L}v\ g\ 0\ 1 = g\ 0 \wedge \mathfrak{L}v\ g\ 1\ 0 = g\ 0$ **using** *b6 b7* **unfolding** $\mathfrak{L}v$*-def* **by** *blast*
ultimately have $(b, b, b, d) \in \mathfrak{D}\ g\ 1\ 0 \wedge (c, c, c'', d) \in \mathfrak{D}\ g\ 0\ 1$ **using** *b6* **unfolding** $\mathfrak{D}$*-def* **by** *simp*
then show $\exists\, b'\ b''\ c'\ c''\ d.\ (b, b', b'', d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c, c', c'', d) \in \mathfrak{D}\ g\ \beta\ \alpha$ **using** *e1* **by** *blast*
qed
moreover have $\alpha = 1 \wedge \beta = 1 \longrightarrow ?thesis$
proof
assume *e1*: $\alpha = 1 \wedge \beta = 1$
then have *jn11 r0 r1 b c* **using** *c1 b4 b6* **by** *blast*
then obtain $b'\ b''\ c'\ c''\ d$ **where**
e2: $(b,b') \in r0\hat{}{*} \wedge (b',b'') \in r1\hat{}{=} \wedge (b'',d) \in r0\hat{}{*}$
and *e3*: $(c,c') \in r0\hat{}{*} \wedge (c',c'') \in r1\hat{}{=} \wedge (c'',d) \in r0\hat{}{*}$ **unfolding** *jn11-def* **by** *blast*
moreover have $\mathfrak{L}v\ g\ 1\ 1 = g\ 0 \wedge \mathfrak{L}1\ g\ 1 = g\ 0$ **using** *b6 b7* **unfolding** $\mathfrak{L}1$*-def* $\mathfrak{L}v$*-def* **by** *blast*
ultimately have $(b, b', b'', d) \in \mathfrak{D}\ g\ 1\ 1 \wedge (c, c', c'', d) \in \mathfrak{D}\ g\ 1\ 1$ **using** *b6* **unfolding** $\mathfrak{D}$*-def* **by** *simp*
then show $\exists\, b'\ b''\ c'\ c''\ d.\ (b, b', b'', d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c, c', c'', d) \in \mathfrak{D}\ g\ \beta\ \alpha$ **using** *e1* **by** *blast*

```
      qed
      ultimately show ?thesis using c2 by blast
    qed
  qed
  then have DCR-generating g unfolding DCR-generating-def by blast
  moreover have r = ⋃{r'. ∃α'<2. r' = g α'}
  proof
    show r ⊆ ⋃{r'. ∃α'<2. r' = g α'}
    proof
      fix p
      assume p ∈ r
      then have p ∈ r0 ∨ p ∈ r1 using b1 by blast
      moreover have (0::nat) < (2::nat) ∧ (1::nat) < (2::nat) by simp
      ultimately show p ∈ ⋃{r'. ∃α'<2. r' = g α'} using b6 by blast
    qed
  next
    show ⋃{r'. ∃α'<2. r' = g α'} ⊆ r
    proof
      fix p
      assume p ∈ ⋃{r'. ∃α'<2. r' = g α'}
      then obtain α' where α'<2 ∧ p ∈ g α' by blast
      moreover then have α' = 0 ∨ α' = 1 by force
      ultimately show p ∈ r using b1 b6 by blast
    qed
  qed
  ultimately show ?thesis unfolding DCR-def by blast
qed

lemma lem-dc2-to-d2:
fixes r::'U rel
assumes DCR 2 r
shows DCR2 r
proof –
  obtain g where b1: DCR-generating g and b2: r = ⋃{r'. ∃α'<2. r' = g α'}
      using assms unfolding DCR-def by blast
  have ∀ α::nat. α<2 ⟷ α = 0 ∨ α = 1 by force
  then have b3: 𝔏1 g 0 = {} ∧ 𝔏1 g 1 = g 0 ∧ 𝔏1 g 2 = g 0 ∪ g 1
      ∧ 𝔏v g 0 0 = {} ∧ 𝔏v g 1 0 = g 0 ∧ 𝔏v g 0 1 = g 0 ∧ 𝔏v g 1 1 = g 0
    unfolding 𝔏1-def 𝔏v-def by (simp-all, blast+)
  have r = (g 0) ∪ (g 1)
  proof
    show r ⊆ (g 0) ∪ (g 1)
    proof
      fix p
      assume p ∈ r
      then obtain α where p ∈ g α ∧ α < 2 using b2 by blast
      moreover have ∀ α::nat. α<2 ⟷ α = 0 ∨ α = 1 by force
      ultimately show p ∈ (g 0) ∪ (g 1) by force
    qed
```

next
  have $(0::nat) < (2::nat) \wedge (1::nat) < (2::nat)$ **by** *simp*
  then show $(g\ 0) \cup (g\ 1) \subseteq r$ **using** *b2* **by** *blast*
qed
moreover have $\forall\ a\ b\ c.\ (a,b) \in (g\ 0) \wedge (a,c) \in (g\ 0) \longrightarrow jn00\ (g\ 0)\ b\ c$
proof (*intro allI impI*)
  fix *a b c*
  assume $(a,b) \in (g\ 0) \wedge (a,c) \in (g\ 0)$
  then obtain $b'\ b''\ c'\ c''\ d$ **where** $(b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ 0\ 0 \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ 0\ 0$
    using *b1* **unfolding** *DCR-generating-def* **by** *blast*
  then show $jn00\ (g\ 0)\ b\ c$ **unfolding** *jn00-def* $\mathfrak{D}$-*def* $\mathfrak{L}1$-*def* $\mathfrak{L}v$-*def* **by** *force*
qed
moreover have $\forall\ a\ b\ c.\ (a,b) \in (g\ 0) \wedge (a,c) \in (g\ 1) \longrightarrow jn01\ (g\ 0)\ (g\ 1)\ b\ c$
proof (*intro allI impI*)
  fix *a b c*
  assume $(a,b) \in (g\ 0) \wedge (a,c) \in (g\ 1)$
  then obtain $b'\ b''\ c'\ c''\ d$ **where**
    $(b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ 0\ 1 \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ 1\ 0$
      using *b1* **unfolding** *DCR-generating-def* **by** *blast*
  then show $jn01\ (g\ 0)\ (g\ 1)\ b\ c$ **unfolding** *jn01-def* $\mathfrak{D}$-*def* $\mathfrak{L}1$-*def* $\mathfrak{L}v$-*def* **by** *force*
qed
moreover have $\forall\ a\ b\ c.\ (a,b) \in (g\ 1) \wedge (a,c) \in (g\ 1) \longrightarrow jn11\ (g\ 0)\ (g\ 1)\ b\ c$
proof (*intro allI impI*)
  fix *a b c*
  assume $(a,b) \in (g\ 1) \wedge (a,c) \in (g\ 1)$
  then obtain $b'\ b''\ c'\ c''\ d$ **where** $(b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ 1\ 1 \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ 1\ 1$
      using *b1* **unfolding** *DCR-generating-def* **by** *blast*
  then show $jn11\ (g\ 0)\ (g\ 1)\ b\ c$
      unfolding *jn11-def* $\mathfrak{D}$-*def* **apply** (*simp only*: *b3*)
      by *blast*
qed
ultimately have $LD2\ r\ (g\ 0)\ (g\ 1)$ **unfolding** *LD2-def* **by** *blast*
then show *?thesis* **unfolding** *DCR2-def* **by** *blast*
qed

lemma *lem-rP-inv*: $rP\ n\ A\ B\ C\ n'\ A'\ B'\ C' \Longrightarrow (\ A \subseteq A' \wedge B \subseteq B' \wedge C \subseteq C'$
    $\wedge$ *finite* $A \wedge$ *finite* $B \wedge$ *finite* $C \wedge$ *finite* $A' \wedge$ *finite* $B' \wedge$ *finite* $C'\ )$
  by (*cases n*, *cases n'*, *force+*)

lemma *lem-infset-finext*:
fixes $S::'U\ set$ **and** $A::'U\ set$
assumes $\neg$ *finite* S **and** *finite* A **and** $A \subseteq S$
shows $\exists\ B.\ B \subseteq S \wedge A \subset B \wedge$ *finite* B
proof –
  have *b1*: *finite* A **using** *assms lem-rP-inv* **by** *blast*
  then have $A \neq S$ **using** *assms* **by** *blast*

then obtain *A2 x* **where** $x \in S \wedge A2 = A \cup \{x\} \wedge x \notin A \wedge A2 \subseteq S$ **using** *assms* **by** *force*
moreover then have *finite A2* **using** *b1* **by** *blast*
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-rE-df*:
fixes $S::'U$ *set*
shows $(u,v) \in rE\ S \Longrightarrow (u,w) \in rE\ S \Longrightarrow (v,t) \in (rE\ S)\hat{}{=} \Longrightarrow (w,t) \in (rE\ S)\hat{}{=} \Longrightarrow v = w$
proof −
assume $(u,v) \in rE\ S$ **and** $(u,w) \in rE\ S$ **and** $(v,t) \in (rE\ S)\hat{}{=}$ **and** $(w,t) \in (rE\ S)\hat{}{=}$
moreover have $\bigwedge u\ v\ w\ t.\ (u,v) \in rE\ S \Longrightarrow (u,\ w) \in rE\ S \Longrightarrow (v,\ t) \in rE\ S \vee v = t \Longrightarrow (w,\ t) \in rE\ S \Longrightarrow v = w$
proof −
fix *u v w t*
assume $(u,v) \in (rE\ S)$ **and** $(u,\ w) \in (rE\ S)$ **and** $(v,\ t) \in (rE\ S) \vee v = t$ **and** $(w,\ t) \in (rE\ S)$
moreover obtain *n::Lev* **and** *a b c* **where** $u = (n,a,b,c)$ **using** *prod-cases4* **by** *blast*
moreover obtain *n′::Lev* **and** *a′ b′ c′* **where** $v = (n',a',b',c')$ **using** *prod-cases4* **by** *blast*
moreover obtain *n′′::Lev* **and** *a′′ b′′ c′′* **where** $w = (n'',a'',b'',c'')$ **using** *prod-cases4* **by** *blast*
moreover obtain *n′′′::Lev* **and** *a′′′ b′′′ c′′′* **where** $t = (n''',a''',b''',c''')$ **using** *prod-cases4* **by** *blast*
ultimately show $v = w$
apply (*simp add*: *rE-def*)
apply (*cases n*)
apply (*cases n′*)
apply (*cases n′′*)
apply (*cases n′′′*)
by *simp+*
qed
ultimately show *?thesis* **by** *blast*
qed

lemma *lem-rE-succ-lev*:
fixes $S::'U$ *set*
assumes $(u,v) \in rE\ S$
shows *levrd* $v =$ (*lev-next* (*levrd* u))
proof −
obtain *n A B C* **where** *b1*: $u = (n,A,B,C)$ **using** *prod-cases4* **by** *blast*
moreover obtain *n′ A′ B′ C′* **where** *b2*: $v = (n',A',B',C')$ **using** *prod-cases4* **by** *blast*
ultimately have *rP n A B C n′ A′ B′ C′* **using** *assms* **unfolding** *rE-def* **by** *blast*
then have $n' =$ (*lev-next n*) **by** (*cases n*, *auto+*)

then show *?thesis* **using** *b1 b2* **by** *simp*
qed

lemma *lem-rE-levset-inv*:
fixes *S*::*'U set* **and** *L u v*
assumes *a1*: (*u*,*v*) ∈ (*rE S*)^* **and** *a2*: *levrd u* ∈ *L* **and** *a3*: *lev-next* ' *L* ⊆ *L*
shows *levrd v* ∈ *L*
proof –
have ⋀ *k*. ∀ *u v*::*'U rD*. (*u*,*v*) ∈ (*rE S*)^^*k* ∧ *levrd u* ∈ *L* ⟶ *levrd v* ∈ *L*
proof –
fix *k*
show ∀ *u v*::*'U rD*. (*u*,*v*) ∈ (*rE S*)^^*k* ∧ *levrd u* ∈ *L* ⟶ *levrd v* ∈ *L*
proof (*induct k*)
show ∀ *u v*::*'U rD*. (*u*,*v*) ∈ (*rE S*)^^*0* ∧ *levrd u* ∈ *L* ⟶ *levrd v* ∈ *L* **by** *simp*
next
fix *k*
assume *d1*: ∀ *u v*::*'U rD*. (*u*,*v*) ∈ (*rE S*)^^*k* ∧ *levrd u* ∈ *L* ⟶ *levrd v* ∈ *L*
show ∀ *u v*::*'U rD*. (*u*,*v*) ∈ (*rE S*)^^(*Suc k*) ∧ *levrd u* ∈ *L* ⟶ *levrd v* ∈ *L*
proof (*intro allI impI*)
fix *u v*::*'U rD*
assume (*u*,*v*) ∈ (*rE S*)^^(*Suc k*) ∧ *levrd u* ∈ *L*
moreover then obtain *v'* **where** *e1*: (*u*,*v'*) ∈ (*rE S*)^^*k* ∧ (*v'*,*v*) ∈ (*rE S*) **by** *force*
ultimately have *levrd v'* ∈ *L* **using** *d1* **by** *blast*
then have *levrd v* ∈ *lev-next* ' *L* **using** *e1 lem-rE-succ-lev*[*of v' v*] **by** *force*
then show *levrd v* ∈ *L* **using** *a3* **by** *force*
qed
qed
qed
then show *?thesis* **using** *a1 a2 rtrancl-imp-relpow* **by** *blast*
qed

lemma *lem-rE-levun*:
fixes *S*::*'U set*
shows *u* ∈ *Domain* (*rE S*) ⟹ *levrd u* ∈ {l*1*, l*3*, l*5*} ⟹ ∃ *v*. (*rE S*)``{*u*} ⊆ {*v*}
proof –
assume *a1*: *u* ∈ *Domain* (*rE S*) **and** *a2*: *levrd u* ∈ {l*1*, l*3*, l*5*}
then obtain *v* **where** *b1*: (*u*,*v*) ∈ (*rE S*) **by** *blast*
obtain *n a b c* **where** *b2*: *u* = (*n*,*a*,*b*,*c*) **using** *prod-cases4* **by** *blast*
obtain *n' a' b' c'* **where** *b3*: *v* = (*n'*,*a'*,*b'*,*c'*) **using** *prod-cases4* **by** *blast*
have *b4*: *rP n a b c n' a' b' c'* **using** *b1 b2 b3* **unfolding** *rE-def* **by** *blast*
have *n* = l*1* ∨ *n* = l*3* ∨ *n* = l*5* **using** *b2 a2* **by** *simp*
moreover have *n* = l*1* ⟶ (*rE S*) `` {*u*} ⊆ {*v*} **using** *b2 b3 b4* **unfolding** *rE-def* **by** *force*
moreover have *n* = l*3* ⟶ (*rE S*) `` {*u*} ⊆ {*v*} **using** *b2 b3 b4* **unfolding** *rE-def* **by** *force*
moreover have *n* = l*5* ⟶ (*rE S*) `` {*u*} ⊆ {*v*} **using** *b2 b3 b4* **unfolding** *rE-def* **by** *force*

ultimately show ∃ *v*. (*rE S*)‘‘{*u*} ⊆ {*v*} **by** *blast*
qed

lemma *lem-rE-domfield*:
fixes *S*::'*U set*
assumes ¬ *finite S*
shows *Domain* (*rE S*) = *Field* (*rE S*)
proof −
 have ⋀ *u2 u1*::'*U rD*. (*u2*,*u1*) ∈ *rE S* ⟹ ∃ *u3*. (*u1*,*u3*) ∈ *rE S*
 proof −
  fix *u2 u1*::'*U rD*
  assume *c1*: (*u2*,*u1*) ∈ *rE S*
  obtain *n1 A1 B1 C1* **where** *c2*: *u1* = (*n1*,*A1*,*B1*,*C1*) **using** *prod-cases4* **by** *blast*
  obtain *n2 A2 B2 C2* **where** *c3*: *u2* = (*n2*,*A2*,*B2*,*C2*) **using** *prod-cases4* **by** *blast*
  have *c4*: *rP n2 A2 B2 C2 n1 A1 B1 C1* ∧ *rC S A2 B2 C2* ∧ *rC S A1 B1 C1* **using** *c1 c2 c3* **unfolding** *rE-def* **by** *blast*
  then have *finite* (*A1* ∪ *A2*) **using** *lem-rP-inv* **by** *blast*
  moreover have *A1* ∪ *A2* ⊆ *S* **using** *c4* **unfolding** *rC-def* **by** *blast*
  ultimately obtain *A3* **where** *c5*: *A3* ⊆ *S* ∧ *A1* ⊂ *A3* ∧ *A2* ⊂ *A3* ∧ *finite A3*
   using *assms lem-infset-finext*[*of S A1* ∪ *A2*] **by** *blast*
  have ∃ *n3 A3 B3 C3*. (*rP n1 A1 B1 C1 n3 A3 B3 C3* ∧ *rC S A3 B3 C3*)
  using *c4* **unfolding** *rC-def*
   apply (*cases n1*)
   apply (*cases n2*, *simp*+)
   apply (*cases n2*, *simp*+)
   apply (*cases n2*, *simp*+)
   apply (*force*, *simp*+)
   apply (*cases n2*, *simp*+)
   apply (*cases n2*, *simp*+)
   apply (*force*, *simp*+)
   apply (*cases n2*, *simp*+)
   apply (*cases n2*, *simp*+)
   apply (*cases n2*, *simp*+)
   using *c5* **apply** (*cases n2*)
   apply *simp*+
   apply *blast*
   apply *simp*
   done
  then obtain *n3 A3 B3 C3* **where** *rP n1 A1 B1 C1 n3 A3 B3 C3* ∧ *rC S A3 B3 C3* **by** *blast*
  moreover obtain *u3* **where** *u3* = (*n3*, *A3*, *B3*, *C3*) **by** *blast*
  moreover have *rC S A1 B1 C1* **using** *c1 c2* **unfolding** *rE-def* **by** *blast*
  ultimately have (*u1*,*u3*) ∈ *rE S* **using** *c2* **unfolding** *rE-def* **by** *blast*
  then show ∃ *u3*. (*u1*,*u3*) ∈ *rE S* **by** *blast*
 qed
 then show *?thesis* **unfolding** *Field-def* **by** *blast*

qed

lemma *lem-wrd-fin-field-rE*:
fixes *S*::*'U set*
assumes ¬ *finite S*
shows *u* ∈ *Field* (*rE S*) ⟹ *finite* (*wrd u*)
proof −
  assume *u* ∈ *Field* (*rE S*)
  then have *u* ∈ *Domain* (*rE S*) **using** *assms lem-rE-domfield* **by** *blast*
  then show *finite* (*wrd u*) **using** *lem-rP-inv* **unfolding** *rE-def* **by** *force*
qed

lemma *lem-rE-rtr-wrd-mon*:
fixes *S*::*'U set* **and** *u v*::*'U rD*
shows (*u*,*v*) ∈ (*rE S*)^* ⟹ *wrd u* ⊆ *wrd v*
proof −
  assume *a1*: (*u*,*v*) ∈ (*rE S*)^*
  have *b1*: ⋀ *u v*::*'U rD*. (*u*,*v*) ∈ (*rE S*) ⟹ *wrd u* ⊆ *wrd v*
  proof −
    fix *u v*::*'U rD*
    assume *a1*: (*u*,*v*) ∈ (*rE S*)
    obtain *n A B C* **where** *b1*: *u* = (*n*,*A*,*B*,*C*) **using** *prod-cases4* **by** *blast*
    obtain *n' A' B' C'* **where** *b2*: *v* = (*n'*,*A'*,*B'*,*C'*) **using** *prod-cases4* **by** *blast*
    have *wrd u* = *A* ∪ *B* ∪ *C* ∧ *wrd v* = *A'*∪ *B'*∪ *C'* **using** *a1 b1 b2* **by** *simp*
    then show *wrd u* ⊆ *wrd v* **using** *a1 b1 b2 lem-rP-inv* **unfolding** *rE-def* **by** *fast*
  qed
  have ⋀ *n*. ∀ *u v*::*'U rD*. (*u*,*v*) ∈ (*rE S*)^^*n* ⟶ *wrd u* ⊆ *wrd v*
  proof −
    fix *n*
    show ∀ *u v*::*'U rD*. (*u*,*v*) ∈ (*rE S*)^^*n* ⟶ *wrd u* ⊆ *wrd v*
    proof (*induct n*)
      show ∀ *u v*. (*u*,*v*) ∈ (*rE S*)^^*0* ⟶ *wrd u* ⊆ *wrd v* **by** *simp*
    next
      fix *m*
      assume *d1*: ∀ *u v*::*'U rD*. (*u*,*v*) ∈ (*rE S*)^^*m* ⟶ *wrd u* ⊆ *wrd v*
      show ∀ *u v*::*'U rD*. (*u*,*v*) ∈ (*rE S*)^^(*Suc m*) ⟶ *wrd u* ⊆ *wrd v*
      proof (*intro allI impI*)
        fix *u v*::*'U rD*
        assume (*u*,*v*) ∈ (*rE S*)^^(*Suc m*)
        then obtain *v'* **where** (*u*,*v'*) ∈ (*rE S*)^^*m* ∧ (*v'*,*v*) ∈ (*rE S*) **by** *force*
        then show *wrd u* ⊆ *wrd v* **using** *d1 b1* **by** *blast*
      qed
    qed
  qed
  then show *wrd u* ⊆ *wrd v* **using** *a1 rtrancl-imp-relpow* **by** *blast*
qed

lemma *lem-Wrd-bkset-rE*: *Wrd* (*bkset* (*rE S*) *U*) = *Wrd U*

proof
  show *Wrd* (*bkset* (*rE S*) *U*) ⊆ *Wrd U*
  proof
    fix *y*
    assume *y* ∈ *Wrd* (*bkset* (*rE S*) *U*)
    then obtain *u v* **where** *u* ∈ *U* ∧ (*v*,*u*) ∈ (*rE S*)^* ∧ *y* ∈ *wrd v* **unfolding** *Wrd-def bkset-def* **by** *force*
    moreover then have *wrd v* ⊆ *wrd u* **using** *lem-rE-rtr-wrd-mon* **by** *blast*
    ultimately show *y* ∈ *Wrd U* **unfolding** *Wrd-def* **by** *blast*
  qed
next
  show *Wrd U* ⊆ *Wrd* (*bkset* (*rE S*) *U*) **unfolding** *Wrd-def bkset-def* **by** *blast*
qed

lemma *lem-Wrd-rE-field-subs-cnt*:
fixes *S*::'*U set* **and** *U*::('*U rD*) *set*
assumes ¬ *finite S*
shows *U* ⊆ *Field* (*rE S*) ⟹ |*U*| ≤*o* |*UNIV*::*nat set*| ⟹ |*Wrd U*| ≤*o* |*UNIV*::*nat set*|
proof −
  assume *b1*: *U* ⊆ *Field* (*rE S*) **and** *a2*: |*U*| ≤*o* |*UNIV*::*nat set*|
  moreover have ∀ *u*∈*U*. |*wrd u*| ≤*o* |*UNIV*::*nat set*|
  proof
    fix *u*::'*U rD*
    assume *u* ∈ *U*
    then have *finite* (*wrd u*) **using** *b1 assms lem-wrd-fin-field-rE* **by** *blast*
    then show |*wrd u*| ≤*o* |*UNIV*::*nat set*| **using** *ordLess-imp-ordLeq* **by** *force*
  qed
  ultimately have |⋃ *u*∈*U*. *wrd u*| ≤*o* |*UNIV*::*nat set*|
    using *card-of-UNION-ordLeq-infinite infinite-UNIV-nat* **by** *blast*
  then show |*Wrd U*| ≤*o* |*UNIV*::*nat set*| **unfolding** *Wrd-def* **by** *simp*
qed

lemma *lem-rE-dn-cnt*:
fixes *S*::'*U set* **and** *U*::('*U rD*) *set*
assumes ¬ *finite S*
shows *U* ⊆ *Field* (*rE S*) ⟹ |*U*| ≤*o* |*UNIV*::*nat set*| ⟹ *V* ⊆ *bkset* (*rE S*) *U* ⟹ |*Wrd V*| ≤*o* |*UNIV*::*nat set*|
proof −
  assume *a1*: *U* ⊆ *Field* (*rE S*) **and** *a2*: |*U*| ≤*o* |*UNIV*::*nat set*| **and** *a3*: *V* ⊆ *bkset* (*rE S*) *U*
  have *Wrd V* ⊆ *Wrd* (*bkset* (*rE S*) *U*) **using** *a3* **unfolding** *Wrd-def* **by** *blast*
  then have |*Wrd V*| ≤*o* |*Wrd* (*bkset* (*rE S*) *U*)| **by** *simp*
  moreover have |*Wrd* (*bkset* (*rE S*) *U*)| ≤*o* |*UNIV*::*nat set*|
    using *a1 a2 assms lem-Wrd-bkset-rE*[*of S U*] *lem-Wrd-rE-field-subs-cnt*[*of S U*] **by** *force*
  ultimately show |*Wrd V*| ≤*o* |*UNIV*::*nat set*| **using** *ordLeq-transitive* **by** *blast*
qed

lemma *lem-rE-succ-Wrd-univ*: $(u,w) \in (rE\ S) \Longrightarrow levrd\ u \in \{l0, l2, l4\} \Longrightarrow S - wrd\ w \subseteq Wrd\ (((rE\ S)``\{u\}) - \{w\})$
proof –
 assume *a1*: $(u,w) \in (rE\ S)$ **and** *a2*: $levrd\ u \in \{l0, l2, l4\}$
 moreover obtain $n\ a\ b\ c$ **where** *b2*: $u = (n,a,b,c)$ **using** *prod-cases4* **by** *blast*
 moreover obtain $n'\ a'\ b'\ c'$ **where** *b3*: $w = (n',a',b',c')$ **using** *prod-cases4* **by** *blast*
 ultimately have *b4*: $rP\ n\ a\ b\ c\ n'\ a'\ b'\ c' \wedge rC\ S\ a\ b\ c \wedge rC\ S\ a'\ b'\ c'$ **unfolding** *rE-def* **by** *blast*
 have $\forall\ y \in S.\ y \notin wrd\ w \longrightarrow (\exists\ v \in (rE\ S)``\{u\} - \{w\}.\ y \in wrd\ v)$
 proof (*intro ballI impI*)
  fix y
  assume *c0*: $y \in S$ **and** *c1*: $y \notin wrd\ w$
  have $n = l0 \longrightarrow (\exists\ v \in (rE\ S)``\{u\} - \{w\}.\ y \in wrd\ v)$
  proof
   assume $n = l0$
   then have $(u, (l1, \{y\}, \{\}, \{\})) \in (rE\ S)$ **using** *c0 b2 b4* **unfolding** *rE-def rC-def* **by** *force*
   then show $\exists\ v \in (rE\ S)``\{u\} - \{w\}.\ y \in wrd\ v$ **using** *c1* **by** *force*
  qed
  moreover have $n = l2 \longrightarrow (\exists\ v \in (rE\ S)``\{u\} - \{w\}.\ y \in wrd\ v)$
  proof
   assume $n = l2$
   then have $(u, (l3,\ a,\ \{y\},\ \{\})) \in (rE\ S)$ **using** *c0 b2 b4* **unfolding** *rE-def rC-def* **by** *force*
   then show $\exists\ v \in (rE\ S)``\{u\} - \{w\}.\ y \in wrd\ v$ **using** *c1* **by** *force*
  qed
  moreover have $n = l4 \longrightarrow (\exists\ v \in (rE\ S)``\{u\} - \{w\}.\ y \in wrd\ v)$
  proof
   assume $n = l4$
   then have $(u, (l5,\ a,\ b,\ \{y\})) \in (rE\ S)$ **using** *c0 b2 b4* **unfolding** *rE-def rC-def* **by** *force*
   then show $\exists\ v \in (rE\ S)``\{u\} - \{w\}.\ y \in wrd\ v$ **using** *c1* **by** *force*
  qed
  ultimately show $\exists\ v \in (rE\ S)``\{u\} - \{w\}.\ y \in wrd\ v$ **using** *a2 b2* **by** *force*
 qed
 then show $S - wrd\ w \subseteq Wrd\ (((rE\ S)``\{u\}) - \{w\})$ **unfolding** *Wrd-def* **by** *blast*
qed

lemma *lem-rE-succ-nocntbnd*:
fixes $S::'U\ set$ **and** $u0::'U\ rD$ **and** $v0::'U\ rD$ **and** $U::('U\ rD)\ set$
assumes *a0*: $\neg\ |S| \leq o\ |UNIV::nat\ set|$ **and** *a1*: $(u0,\ v0) \in (rE\ S)$ **and** *a2*: $levrd\ u0 \in \{l0, l2, l4\}$
 and *a3*: $U \subseteq Field\ (rE\ S)$ **and** *a4*: $((rE\ S)\ ``\ \{u0\}) - \{v0\} \subseteq bkset\ (rE\ S)\ U$
shows $\neg\ |U| \leq o\ |UNIV::nat\ set|$
proof
 assume $|U| \leq o\ |UNIV::nat\ set|$
 moreover have *c0*: $\neg$ *finite* S **using** *a0* **by** (*meson card-of-Well-order infi-*

nite-iff-card-of-nat ordLeq-total)
ultimately have *c1*: $|Wrd\ (((rE\ S)``\{u0\}) - \{v0\})| \leq o\ |UNIV::nat\ set|$ **using** *a3 a4 lem-rE-dn-cnt* **by** *blast*
have $v0 \in Field\ (rE\ S)$ **using** *a1* **unfolding** *Field-def* **by** *blast*
then have *finite* (*wrd v0*) **using** *c0 a0 lem-wrd-fin-field-rE* **by** *blast*
then have $\neg\ |S - wrd\ v0| \leq o\ |UNIV::nat\ set|$ **using** *a0*
by (*metis card-of-infinite-diff-finite finite-iff-cardOf-nat ordIso-symmetric ordLeq-iff-ordLess-or-ordIso ordLeq-transitive*)
moreover have $S - wrd\ v0 \subseteq Wrd\ (((rE\ S)``\{u0\}) - \{v0\})$ **using** *lem-rE-succ-Wrd-univ a1 a2* **by** *blast*
ultimately have $\neg\ |Wrd\ (((rE\ S)``\{u0\}) - \{v0\})| \leq o\ |UNIV::nat\ set|$ **by** (*metis card-of-mono1 ordLeq-transitive*)
then show *False* **using** *c1* **by** *blast*
qed

lemma *lem-rE-succ-nocntbnd2*:
fixes $S::'U\ set$ **and** $u0::'U\ rD$ **and** $v0::'U\ rD$
assumes *a0*: $\neg\ |S| \leq o\ |UNIV::nat\ set|$
and *a1*: $(u0, v0) \in (rE\ S)$ **and** *a2*: $levrd\ u0 \in \{l0, l2, l4\}$
and *a3*: $r \subseteq (rE\ S)$ **and** *a4*: $\forall\ u.\ |r``\{u\}| \leq o\ |UNIV::nat\ set|$
and *a5*: $((rE\ S)\ ``\ \{u0\}) - \{v0\} \subseteq bkset\ (rE\ S)\ ((r\hat{}*)``\{u0\})$
shows *False*
proof −
have *b1*: $\bigwedge n::nat.\ \bigwedge u::('U\ rD).\ u \in Field\ (rE\ S) \longrightarrow (r\hat{}\hat{}n)``\{u\} \subseteq Field\ (rE\ S) \wedge |(r\hat{}\hat{}n)``\{u\}| \leq o\ |UNIV::nat\ set|$
proof (*intro impI*)
fix $n::nat$ **and** $u::'U\ rD$
assume *c1*: $u \in Field\ (rE\ S)$
show $(r\hat{}\hat{}n)``\{u\} \subseteq Field\ (rE\ S) \wedge |(r\hat{}\hat{}n)\ ``\ \{u\}| \leq o\ |UNIV::nat\ set|$
proof (*induct n*)
show $(r\hat{}\hat{}0)``\{u\} \subseteq Field\ (rE\ S) \wedge |(r\ \hat{}\hat{}\ 0)\ ``\ \{u\}| \leq o\ |UNIV::nat\ set|$ **using** *c1* **by** *simp*
next
fix *m*
assume *d1*: $(r\hat{}\hat{}m)``\{u\} \subseteq Field\ (rE\ S) \wedge |(r\hat{}\hat{}m)``\{u\}| \leq o\ |UNIV::nat\ set|$
moreover have $\forall\ v \in (r\hat{}\hat{}m)``\{u\}.\ |r``\{v\}| \leq o\ |UNIV::nat\ set|$ **using** *a4* **by** *blast*
moreover have $(r\ \hat{}\hat{}\ Suc\ m)\ ``\ \{u\} = (\bigcup v\in((r\hat{}\hat{}m)``\{u\}).\ r\ ``\{v\})$ **by** *force*
ultimately have $|(r\ \hat{}\hat{}\ Suc\ m)\ ``\ \{u\}| \leq o\ |UNIV::nat\ set|$
using *card-of-UNION-ordLeq-infinite*[*of UNIV*::*nat set* $(r\hat{}\hat{}m)``\{u\}$] *infinite-UNIV-nat* **by** *simp*
moreover have $(r\ \hat{}\hat{}\ Suc\ m)``\{u\} \subseteq Field\ (rE\ S)$ **using** *d1 a3* **unfolding** *Field-def* **by** *fastforce*
ultimately show $(r\ \hat{}\hat{}\ Suc\ m)``\{u\} \subseteq Field\ (rE\ S) \wedge |(r\ \hat{}\hat{}\ Suc\ m)\ ``\ \{u\}| \leq o\ |UNIV::nat\ set|$ **by** *blast*
qed
qed
have *b2*: $\bigwedge u::'U\ rD.\ u \in Field\ (rE\ S) \longrightarrow |(r\hat{}*)\ ``\ \{u\}| \leq o\ |UNIV::nat\ set|$
proof (*intro impI*)

fix *u*::*'U rD*
assume *c1*: $u \in$ *Field* (*rE S*)
have |*UNIV*::*nat set*| $\leq$ |*UNIV*::*nat set*| **by** *simp*
moreover have $\forall$ *n*. |(*r*^^*n*) `` {*u*}| $\leq o$ |*UNIV*::*nat set*| **using** *c1 b1* **by** *blast*
ultimately have *c1*: |$\bigcup$ *n*. (*r*^^*n*) `` {*u*}| $\leq o$ |*UNIV*::*nat set*|
using *card-of-UNION-ordLeq-infinite*[*of UNIV*::*nat set UNIV*::*nat set*] *infinite-UNIV-nat* **by** *simp*
have (*r*^*) `` {*u*} $\subseteq$ ($\bigcup$ *n*. (*r*^^*n*) `` {*u*}) **by** (*simp add*: *rtrancl-is-UN-relpow subset-eq*)
then have |(*r*^*) `` {*u*}| $\leq o$ |$\bigcup$ *n*. (*r*^^*n*) `` {*u*}| **by** *simp*
then show |(*r*^*) `` {*u*}| $\leq o$ |*UNIV*::*nat set*| **using** *c1 ordLeq-transitive* **by** *blast*
qed
obtain *U* **where** *b3*: *U* = ((*r*^*) `` {*u0*}) **by** *blast*
have *U* $\subseteq$ ($\bigcup$ *n*. (*r*^^*n*) `` {*u0*}) **using** *b3* **by** (*simp add*: *rtrancl-is-UN-relpow subset-eq*)
moreover have *u0* $\in$ *Field* (*rE S*) **using** *a1* **unfolding** *Field-def* **by** *blast*
ultimately have *U* $\subseteq$ *Field* (*rE S*) $\wedge$ |*U*| $\leq o$ |*UNIV*::*nat set*| **using** *b1 b2 b3* **by** *blast*
moreover have ((*rE S*) `` {*u0*}) − {*v0*} $\subseteq$ *bkset* (*rE S*) *U* **using** *b3 a5* **by** *blast*
ultimately show *False* **using** *a0 a1 a2 lem-rE-succ-nocntbnd*[*of S u0 v0 U*] **by** *blast*
qed

lemma *lem-rE-diamsubr-un*:
fixes *S*::*'U set*
assumes *a1*: *r0* $\subseteq$ (*rE S*) **and** *a2*: $\forall$ *a b c*. (*a*,*b*) $\in$ *r0* $\wedge$ (*a*,*c*) $\in$ *r0* $\longrightarrow$ ($\exists$ *d*. (*b*,*d*) $\in$ *r0*^= $\wedge$ (*c*,*d*) $\in$ *r0*^=)
shows $\forall$ *u*. $\exists$ *v*. *r0*``{*u*} $\subseteq$ {*v*}
proof
fix *u*
have $\forall$ *v w*. (*u*,*v*) $\in$ *r0* $\wedge$ (*u*,*w*) $\in$ *r0* $\longrightarrow$ *v* = *w*
proof (*intro allI impI*)
fix *v w*
assume (*u*,*v*) $\in$ *r0* $\wedge$ (*u*,*w*) $\in$ *r0*
moreover then obtain *t* **where** (*v*,*t*) $\in$ *r0*^= $\wedge$ (*w*,*t*) $\in$ *r0*^= **using** *a2* **by** *blast*
ultimately have (*u*,*v*) $\in$ (*rE S*) $\wedge$ (*u*,*w*) $\in$ (*rE S*) $\wedge$ (*v*,*t*) $\in$ (*rE S*)^= $\wedge$ (*w*,*t*) $\in$ (*rE S*)^= **using** *a1* **by** *blast*
then show *v* = *w* **using** *lem-rE-df* **by** *blast*
qed
then show $\exists$ *v*. *r0*``{*u*} $\subseteq$ {*v*} **by** *blast*
qed

lemma *lem-rE-succ-nocntbnd3*:
fixes *S*::*'U set* **and** *u0*::*'U rD* **and** *v0*::*'U rD*
assumes *a0*: ¬ |*S*| $\leq o$ |*UNIV*::*nat set*|
and *a1*: *LD2* (*rE S*) *r0 r1*

and *a2*: (*u0*, *v0*) ∈ (*rE S*) **and** *a3*: *levrd u0* ∈ {l*0*, l*2*, l*4*}
and *a4*: *r* = {(*u*,*v*) ∈ *rE S*. *u* = *v0*} ∪ *r0*
and *a5*: ((*rE S*) ‘‘ {*u0*}) − {*v0*} ⊆ *bkset* (*rE S*) ((*r*^*∗*)‘‘{*u0*})
shows *False*
proof −
have *b1*: *r0* ⊆ (*rE S*) **using** *a1* **unfolding** *LD2-def* **by** *blast*
then have *r* ⊆ (*rE S*) **using** *a4* **by** *blast*
moreover have ∀ *u*. |*r*‘‘{*u*}| ≤*o* |*UNIV*::*nat set*|
proof
fix *u*
have ∀ *a b c*. (*a*,*b*) ∈ *r0* ∧ (*a*,*c*) ∈ *r0* ⟶ (∃ *d*. (*b*,*d*) ∈ *r0*^= ∧ (*c*,*d*) ∈ *r0*^=)
using *a1* **unfolding** *LD2-def jn00-def* **by** *blast*
then obtain *v* **where** *r0*‘‘{*u*} ⊆ {*v*} **using** *b1 lem-rE-diamsubr-un*[*of r0*] **by** *blast*
moreover have *r*‘‘{*u*} ⊆ *r0*‘‘{*u*} ∪ (*rE S*)‘‘{*v0*} **using** *a4* **by** *blast*
ultimately have *r*‘‘{*u*} ⊆ {*v*} ∪ (*rE S*)‘‘{*v0*} **by** *blast*
moreover have |{*v*} ∪ (*rE S*)‘‘{*v0*}| ≤*o* |*UNIV*::*nat set*|
proof −
have *levrd v0* ∈ {l*1*, l*3*, l*5*} **using** *a2 a3* **unfolding** *rE-def* **by** *force*
moreover have ¬ *finite S* **using** *a0* **by** (*meson card-of-Well-order infinite-iff-card-of-nat ordLeq-total*)
moreover then have *v0* ∈ *Domain* (*rE S*) **using** *a2 a0 lem-rE-domfield* **unfolding** *Field-def* **by** *blast*
ultimately obtain *v0′* **where** (*rE S*)‘‘{*v0*} ⊆ {*v0′*} **using** *lem-rE-levun* **by** *blast*
then have {*v*} ∪ (*rE S*)‘‘{*v0*} ⊆ {*v*,*v0′*} **by** *blast*
then have *finite* ({*v*} ∪ (*rE S*)‘‘{*v0*}) **by** (*meson finite.emptyI finite.insertI rev-finite-subset*)
then show *?thesis* **by** (*simp add*: *ordLess-imp-ordLeq*)
qed
ultimately show |*r*‘‘{*u*}| ≤*o* |*UNIV*::*nat set*| **using** *card-of-mono1 ordLeq-transitive* **by** *blast*
qed
ultimately show *?thesis* **using** *a0 a2 a3 a5 lem-rE-succ-nocntbnd2*[*of S u0 v0 r*] **by** *blast*
qed

lemma *lem-rE-one*:
fixes *S*::*′U set* **and** *u0*::*′U rD* **and** *v0*::*′U rD*
assumes *a0*: ¬ |*S*| ≤*o* |*UNIV*::*nat set*| **and** *a1*: *LD2* (*rE S*) *r0 r1*
and *a2*: (*u0*, *v0*) ∈ *r0* **and** *a3*: *levrd u0* ∈ {l*0*, l*2*, l*4*}
shows *False*
proof −
obtain *r* **where** *b1*: *r* = {(*u*,*v*) ∈ *rE S*. *u* = *v0*} ∪ *r0* **by** *blast*
moreover have (*u0*, *v0*) ∈ (*rE S*) **using** *a1 a2* **unfolding** *LD2-def* **by** *blast*
moreover have ((*rE S*) ‘‘ {*u0*}) − {*v0*} ⊆ *bkset* (*rE S*) ((*r*^*∗*)‘‘{*u0*})
proof
fix *v*
assume *c1*: *v* ∈ ((*rE S*) ‘‘ {*u0*}) − {*v0*}

have $\exists$ *v. r0‘‘*{*u0*} $\subseteq$ {*v*} **using** *a1 lem-rE-diamsubr-un*[*of r0 S*] **unfolding** *LD2-def jn00-def* **by** *blast*
then have *r0* ‘‘ {*u0*} $\subseteq$ {*v0*} **using** *a2* **by** *blast*
moreover have *c2*: (*rE S*) = *r0* $\cup$ *r1* **using** *a1* **unfolding** *LD2-def* **by** *blast*
ultimately have (*u0*, *v*) $\in$ *r1* **using** *c1* **by** *blast*
then have *jn01 r0 r1 v0 v* **using** *a1 a2* **unfolding** *LD2-def* **by** *blast*
then obtain *v0′ d* **where** *c3*: (*v0*, *v0′*) $\in$ *r1*^= $\wedge$ (*v0′*, *d*) $\in$ *r0*^* $\wedge$ (*v*, *d*) $\in$ *r0*^* **unfolding** *jn01-def* **by** *blast*
obtain *U* **where** *c4*: *U* = (*r*^*)‘‘{*u0*} **by** *blast*
have (*u0*, *d*) $\in$ *r*^*
proof −
have *v0* = *v0′* $\vee$ (*v0*,*v0′*) $\in$ (*rE S*) **using** *c2 c3* **by** *blast*
then have (*v0*, *v0′*) $\in$ *r*^= **using** *b1* **by** *blast*
moreover have (*u0*, *v0*) $\in$ *r* **using** *b1 a2* **by** *blast*
ultimately have (*u0*, *v0′*) $\in$ *r*^* **by** *force*
moreover have (*v0′*,*d*) $\in$ *r*^* **using** *c3 b1 rtrancl-mono*[*of r0 r*] **by** *blast*
ultimately show *?thesis* **by** *force*
qed
then have *d* $\in$ *U* **using** *c4* **by** *blast*
then have *c3*: *v* $\in$ *bkset r0 U* **using** *c3* **unfolding** *bkset-def* **by** *blast*
have *r0* $\subseteq$ (*rE S*) **using** *a1* **unfolding** *LD2-def* **by** *blast*
then have *bkset r0 U* $\subseteq$ *bkset* (*rE S*) *U* **unfolding** *bkset-def* **by** (*simp add*: *Image-mono rtrancl-mono*)
then show *v* $\in$ *bkset* (*rE S*) ((*r*^*)‘‘{*u0*}) **using** *c3 c4* **by** *blast*
qed
ultimately show *False* **using** *a0 a1 a3 lem-rE-succ-nocntbnd3*[*of S r0 r1 u0 v0 r*] **by** *blast*
qed

lemma *lem-rE-jn0*:
fixes *S*::′*U set* **and** *u1*::′*U rD* **and** *u2*::′*U rD* **and** *v*::′*U rD*
assumes *a1*: (*u1*,*v*) $\in$ (*rE S*) **and** *a2*: (*u2*,*v*) $\in$ (*rE S*) **and** *a3*: *u1* $\neq$ *u2*
shows *levrd v* $\in$ {17, 18}
proof −
obtain *n1 a1 b1 c1* **where** *b1*: *u1* = (*n1*,*a1*,*b1*,*c1*) **using** *prod-cases4* **by** *blast*
obtain *n2 a2 b2 c2* **where** *b2*: *u2* = (*n2*,*a2*,*b2*,*c2*) **using** *prod-cases4* **by** *blast*
obtain *n a b c* **where** *b3*: *v* = (*n*,*a*,*b*,*c*) **using** *prod-cases4* **by** *blast*
have *rP n1 a1 b1 c1 n a b c* **using** *b1 b3 a1* **unfolding** *rE-def* **by** *blast*
moreover have *rP n2 a2 b2 c2 n a b c* **using** *b2 b3 a2* **unfolding** *rE-def* **by** *blast*
moreover have (*n1*,*a1*,*b1*,*c1*) $\neq$ (*n2*,*a2*,*b2*,*c2*) **using** *a3 b1 b2* **by** *blast*
ultimately have *n* $\in$ { 17, 18}
apply (*cases n1*, *cases n2*)
apply (*simp+*, *cases n2*)
apply (*simp+*, *cases n2*)
apply (*simp+*, *cases n2*)
apply (*simp+*, *cases n2*)
apply (*simp+*, *cases n2*)
apply *simp+*

  done
 then show *?thesis* **using** *b3* **by** *simp*
qed

lemma *lem-rE-jn1*:
fixes *S*::*'U set* **and** *u1*::*'U rD* **and** *u2*::*'U rD* **and** *v*::*'U rD*
assumes *a1*: $(u1,v) \in (rE\ S)$ **and** *a2*: $(u2,v) \in (rE\ S)\hat{}*$ **and** *a3*: $(u1,u2) \notin (rE\ S) \wedge (u2,u1) \notin (rE\ S)\hat{}*$
shows $levrd\ v \in \{17, 18\}$
proof −
 have $\bigwedge k2.\ \forall\ u1\ u2\ v::'U\ rD.\ \forall\ i.\ i \leq k2 \wedge (u1,u2) \notin (rE\ S) \wedge (u2,u1) \notin (rE\ S)\hat{}* \longrightarrow (u1,v) \in (rE\ S) \longrightarrow (u2,v) \in (rE\ S)\hat{}\hat{}i \longrightarrow levrd\ v \in \{17, 18\}$
 proof −
  fix *k2*
  show $\forall\ u1\ u2\ v::'U\ rD.\ \forall\ i.\ i \leq k2 \wedge (u1,u2) \notin (rE\ S) \wedge (u2,u1) \notin (rE\ S)\hat{}* \longrightarrow (u1,v) \in (rE\ S) \longrightarrow (u2,v) \in (rE\ S)\hat{}\hat{}i \longrightarrow levrd\ v \in \{17, 18\}$
  proof (*induct k2*)
   show $\forall u1\ u2\ v::'U\ rD.\ \forall\ i.\ i \leq 0 \wedge (u1,u2) \notin (rE\ S) \wedge (u2,u1) \notin (rE\ S)\hat{}* \longrightarrow (u1,\ v) \in (rE\ S) \longrightarrow (u2,\ v) \in (rE\ S)\hat{}\hat{}i \longrightarrow levrd\ v \in \{17, 18\}$ **by** *force*
  next
   fix *k2*
   assume *d1*: $\forall u1\ u2\ v::'U\ rD.\ \forall\ i.\ \ i \leq k2 \wedge (u1,u2) \notin (rE\ S) \wedge (u2,\ u1) \notin (rE\ S)\hat{}* \longrightarrow$
     $(u1,\ v) \in (rE\ S) \longrightarrow (u2,\ v) \in (rE\ S)\hat{}\hat{}i \longrightarrow levrd\ v \in \{17, 18\}$
   show $\forall u1\ u2\ v::'U\ rD.\ \forall\ i.\ i \leq Suc\ k2 \wedge (u1,u2) \notin (rE\ S) \wedge (u2,\ u1) \notin (rE\ S)\hat{}* \longrightarrow$
    $(u1,\ v) \in (rE\ S) \longrightarrow (u2,\ v) \in (rE\ S)\hat{}\hat{}i \longrightarrow levrd\ v \in \{17, 18\}$
   proof (*intro allI impI*)
    fix *u1 u2 v*::*'U rD* **and** *i*
    assume *e1*: $i \leq Suc\ k2 \wedge (u1,\ u2) \notin (rE\ S) \wedge (u2,\ u1) \notin (rE\ S)\hat{}*$
     and *e2*: $(u1,\ v) \in (rE\ S)$ **and** *e3*: $(u2,\ v) \in (rE\ S)\hat{}\hat{}i$
    show $levrd\ v \in \{17, 18\}$
    proof (*cases* $i = Suc\ k2$)
     assume *f1*: $i = Suc\ k2$
     then obtain v' **where** *f2*: $(u2,\ v') \in (rE\ S)$ **and** *f3*: $(v',\ v) \in (rE\ S)\hat{}\hat{}k2$ **using** *e3* **by** (*meson relpow-Suc-E2*)
     moreover have $k2 \leq k2$ **using** *e1* **by** *force*
     ultimately have $(v',u1) \notin (rE\ S)\hat{}* \wedge (u1,v') \notin (rE\ S) \longrightarrow levrd\ v \in \{17, 18\}$ **using** *e2 d1* **by** *blast*
     moreover have $(v',u1) \in (rE\ S)\hat{}* \longrightarrow False$
     proof
      assume $(v',u1) \in (rE\ S)\hat{}*$
      then have $(u2,u1) \in (rE\ S)\hat{}*$ **using** *f2* **by** *force*
      then show *False* **using** *e1* **by** *blast*
     qed
     moreover have $(u1,v') \in (rE\ S) \longrightarrow levrd\ v \in \{17, 18\}$
     proof
      assume $(u1,v') \in (rE\ S)$
      moreover have $u1 \neq u2$ **using** *e1* **by** *force*

ultimately have *levrd v'* ∈ {*17*, *18*} **using** *f2 lem-rE-jn0*[*of u1 v' S u2*] **by** *blast*
moreover have (*v'*, *v*) ∈ (*rE S*)^∗ **using** *f3 rtrancl-power* **by** *blast*
moreover have *lev-next* ‘ {*17*, *18*} ⊆ {*17*, *18*} **by** *simp*
ultimately show *levrd v* ∈ {*17*, *18*} **using** *lem-rE-levset-inv*[*of v' v S* {*17*, *18*}] **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
next
assume *i* ≠ *Suc k2*
then have *i* ≤ *k2* **using** *e1* **by** *force*
then show *?thesis* **using** *d1 e1 e2 e3* **by** *blast*
qed
qed
qed
qed
moreover obtain *k2* **where** (*u2*,*v*) ∈ (*rE S*)^^*k2* **using** *a2 rtrancl-imp-relpow* **by** *blast*
moreover have *k2* ≤ *k2* **by** *force*
ultimately show *?thesis* **using** *a1 a3* **by** *blast*
qed

lemma *lem-rE-jn2*:
fixes *S*::*'U set* **and** *u1*::*'U rD* **and** *u2*::*'U rD* **and** *v*::*'U rD*
assumes *a1*: (*u1*,*v*) ∈ (*rE S*)^∗ **and** *a2*: (*u2*,*v*) ∈ (*rE S*)^∗ **and** *a3*: (*u1*,*u2*) ∉ (*rE S*)^∗ ∧ (*u2*,*u1*) ∉ (*rE S*)^∗
shows *levrd v* ∈ {*17*, *18*}
proof −
have ⋀ *k1*. ∀ *u1 u2 v*::*'U rD*. ∀ *i*. *i* ≤ *k1* ∧ (*u1*,*u2*) ∉ (*rE S*)^∗ ∧ (*u2*,*u1*) ∉ (*rE S*)^∗ ⟶ (*u1*,*v*) ∈ (*rE S*)^^*i* ⟶ (*u2*,*v*) ∈ (*rE S*)^∗ ⟶ *levrd v* ∈ {*17*, *18*}
proof −
fix *k1*
show ∀ *u1 u2 v*::*'U rD*. ∀ *i*. *i* ≤ *k1* ∧ (*u1*,*u2*) ∉ (*rE S*)^∗ ∧ (*u2*,*u1*) ∉ (*rE S*)^∗ ⟶ (*u1*,*v*) ∈ (*rE S*)^^*i* ⟶ (*u2*,*v*) ∈ (*rE S*)^∗ ⟶ *levrd v* ∈ {*17*, *18*}
proof (*induct k1*)
show ∀ *u1 u2 v*::*'U rD*. ∀ *i*. *i* ≤ *0* ∧ (*u1*,*u2*) ∉ (*rE S*)^∗ ∧ (*u2*,*u1*) ∉ (*rE S*)^∗ ⟶ (*u1*, *v*) ∈ (*rE S*)^^*i* ⟶ (*u2*, *v*) ∈ (*rE S*)^∗ ⟶ *levrd v* ∈ {*17*, *18*}
proof (*intro allI impI*)
fix *u1 u2 v*::*'U rD* **and** *i*
assume *i* ≤ *0* ∧ (*u1*,*u2*) ∉ (*rE S*)^∗ ∧ (*u2*,*u1*) ∉ (*rE S*)^∗ **and** (*u1*, *v*) ∈ (*rE S*)^^*i* **and** (*u2*, *v*) ∈ (*rE S*)^∗
moreover then have (*u2*,*u1*) ∈ (*rE S*)^∗ **using** *rtrancl-power* **by** *fastforce*
ultimately have *False* **by** *blast*
then show *levrd v* ∈ {*17*, *18*} **by** *blast*
qed
next
fix *k1*
assume *d1*: ∀ *u1 u2 v*::*'U rD*. ∀ *i*. *i* ≤ *k1* ∧ (*u1*, *u2*) ∉ (*rE S*)^∗ ∧ (*u2*, *u1*) ∉ (*rE S*)^∗ ⟶

```
              (u1, v) ∈ (rE S) ^^ i ⟶ (u2, v) ∈ (rE S)^* ⟶ levrd v ∈ {17, 18}
      show ∀ u1 u2 v::'U rD. ∀ i. i ≤ Suc k1 ∧ (u1, u2) ∉ (rE S)^* ∧ (u2, u1)
∉ (rE S)^* ⟶
            (u1, v) ∈ (rE S) ^^ i ⟶ (u2, v) ∈ (rE S)^* ⟶ levrd v ∈ {17, 18}
      proof (intro allI impI)
        fix u1 u2 v::'U rD and i
        assume e1: i ≤ Suc k1 ∧ (u1, u2) ∉ (rE S)^* ∧ (u2, u1) ∉ (rE S)^*
          and e2: (u1, v) ∈ (rE S)^^i and e3: (u2, v) ∈ (rE S)^*
        show levrd v ∈ {17, 18}
        proof (cases i = Suc k1)
          assume f1: i = Suc k1
          then obtain v' where f2: (u1, v') ∈ (rE S) and f3: (v', v) ∈ (rE S)^^k1
using e2 by (meson relpow-Suc-E2)
          moreover have k1 ≤ k1 using e1 by force
          ultimately have (v',u2) ∉ (rE S)^* ∧ (u2,v') ∉ (rE S)^* ⟶ levrd v ∈
{17, 18} using e3 d1 by blast
          moreover have (v',u2) ∈ (rE S)^* ⟶ False
          proof
            assume (v',u2) ∈ (rE S)^*
            then have (u1,u2) ∈ (rE S)^* using f2 by force
            then show False using e1 by blast
          qed
          moreover have (u2,v') ∈ (rE S)^* ⟶ levrd v ∈ {17, 18}
          proof
            assume (u2,v') ∈ (rE S)^*
            then have levrd v' ∈ {17, 18} using e1 f2 lem-rE-jn1[of u1 v' S u2] by
blast
            moreover have (v', v) ∈ (rE S)^* using f3 rtrancl-power by blast
            moreover have lev-next ` {17, 18} ⊆ {17, 18} by simp
            ultimately show levrd v ∈ {17, 18} using lem-rE-levset-inv[of v' v S
{17, 18}] by blast
          qed
          ultimately show ?thesis by blast
        next
          assume i ≠ Suc k1
          then have i ≤ k1 using e1 by force
          then show ?thesis using d1 e1 e2 e3 by blast
        qed
      qed
    qed
  qed
  moreover obtain k1 where (u1,v) ∈ (rE S)^^k1 using a1 rtrancl-imp-relpow
by blast
  moreover have k1 ≤ k1 by force
  ultimately show ?thesis using a2 a3 by blast
qed

lemma lem-rel-pow2fw: (u,u1) ∈ r ∧ (u1,v) ∈ r⟶ (u,v) ∈ r^^2
  by (metis Suc-1 relpow-1 relpow-Suc-I)
```

lemma *lem-rel-pow3fw*: $(u,u1) \in r \wedge (u1,u2) \in r \wedge (u2,v) \in r \longrightarrow (u,v) \in r \hat{}\hat{} 3$
by (*metis One-nat-def numeral-3-eq-3 relpow-1 relpow-Suc-I*)

lemma *lem-rel-pow3*: $(u,v) \in r \hat{}\hat{} 3 \Longrightarrow \exists\ u1\ u2.\ (u,u1) \in r \wedge (u1,u2) \in r \wedge (u2,v) \in r$
by (*metis One-nat-def numeral-3-eq-3 relpow-1 relpow-Suc-E*)

lemma *lem-rel-pow4*: $(u,v) \in r \hat{}\hat{} 4 \Longrightarrow \exists\ u1\ u2\ u3.\ (u,u1) \in r \wedge (u1,u2) \in r \wedge (u2,u3) \in r \wedge (u3,v) \in r$
proof −
assume $(u,v) \in r \hat{}\hat{} 4$
then obtain *u3* **where** $(u,u3) \in r \hat{}\hat{} 3 \wedge (u3,v) \in r$ **using** *relpow-E* **by** *force*
moreover then obtain *u1 u2* **where** $(u,u1) \in r \wedge (u1,u2) \in r \wedge (u2,u3) \in r$
by (*metis One-nat-def numeral-3-eq-3 relpow-1 relpow-Suc-E*)
ultimately show $\exists\ u1\ u2\ u3.\ (u,u1) \in r \wedge (u1,u2) \in r \wedge (u2,u3) \in r \wedge (u3,v) \in r$ **by** *blast*
qed

lemma *lem-rel-pow5*: $(u,v) \in r \hat{}\hat{} 5 \Longrightarrow \exists\ u1\ u2\ u3\ u4.\ (u,u1) \in r \wedge (u1,u2) \in r \wedge (u2,u3) \in r \wedge (u3,u4) \in r \wedge (u4,v) \in r$
proof −
assume $(u,v) \in r \hat{}\hat{} 5$
then obtain *u4* **where** $(u,u4) \in r \hat{}\hat{} 4 \wedge (u4,v) \in r$ **using** *relpow-E* **by** *force*
moreover then obtain *u1 u2 u3* **where** $(u,u1) \in r \wedge (u1,u2) \in r \wedge (u2,u3) \in r \wedge (u3, u4) \in r$
using *lem-rel-pow4*[*of u u4 r*] **by** *blast*
ultimately show $\exists\ u1\ u2\ u3\ u4.\ (u,u1) \in r \wedge (u1,u2) \in r \wedge (u2,u3) \in r \wedge (u3,u4) \in r \wedge (u4,v) \in r$ **by** *blast*
qed

lemma *lem-rE-l1-l78-dist*:
fixes $S::'U\ set$
assumes *a1*: *levrd u* = l1 **and** *a2*: *levrd v* ∈ {l7, l8} **and** *a3*: $n \leq 5$
shows $(u,v) \notin (rE\ S) \hat{}\hat{} n$
proof −
have *b0*: $(u,v) \notin (rE\ S) \hat{}\hat{} 0$ **using** *a1 a2* **by** *force*
have *b1*: $(u,v) \notin (rE\ S) \hat{}\hat{} 1$ **using** *a1 a2 lem-rE-succ-lev*[*of u v*] **by** *force*
have $\bigwedge u1.\ (u,u1) \in (rE\ S) \wedge (u1,v) \in (rE\ S) \Longrightarrow False$
using *a1 a2 lem-rE-succ-lev*
by (*metis Lev.distinct*(*49*) *Lev.distinct*(*51*) *insertE lev-next.simps*(*2*) *lev-next.simps*(*3*) *singletonD*)
then have *b2*: $(u,v) \notin (rE\ S) \hat{}\hat{} 2$ **by** (*metis Suc-1 relpow-1 relpow-Suc-D2*)
have $\bigwedge u1\ u2.\ (u,u1) \in (rE\ S) \wedge (u1,u2) \in (rE\ S) \wedge (u2,v) \in (rE\ S) \Longrightarrow False$
using *a1 a2 lem-rE-succ-lev*
by (*metis Lev.distinct*(*57*) *Lev.distinct*(*59*) *insertE lev-next.simps*(*2*) *lev-next.simps*(*3*) *lev-next.simps*(*4*) *singletonD*)
then have *b3*: $(u,v) \notin (rE\ S) \hat{}\hat{} 3$ **using** *lem-rel-pow3*[*of u v rE S*] **by** *blast*
have $\bigwedge u1\ u2\ u3.\ (u,u1) \in (rE\ S) \wedge (u1,u2) \in (rE\ S) \wedge (u2,u3) \in (rE\ S) \wedge$

$(u3,v) \in (rE\ S) \Longrightarrow False$
using *a1 a2 lem-rE-succ-lev*
by (*metis Lev.distinct*(*63*) *Lev.distinct*(*65*) *insertE lev-next.simps*(*2*) *lev-next.simps*(*3*) *lev-next.simps*(*4*) *lev-next.simps*(*5*) *singletonD*)
then have *b4*: $(u,v) \notin (rE\ S)^{\frown\frown}4$ **using** *lem-rel-pow4*[*of u v rE S*] **by** *blast*
have $\bigwedge$ *u1 u2 u3 u4*. $(u,u1) \in (rE\ S) \wedge (u1,u2) \in (rE\ S) \wedge (u2,u3) \in (rE\ S) \wedge (u3,u4) \in (rE\ S) \wedge (u4,v) \in (rE\ S) \Longrightarrow False$
using *a1 a2 lem-rE-succ-lev*
by (*metis Lev.distinct*(*67*) *Lev.distinct*(*69*) *insertE lev-next.simps*(*2*) *lev-next.simps*(*3*) *lev-next.simps*(*4*) *lev-next.simps*(*5*) *lev-next.simps*(*6*) *singletonD*)
then have *b5*: $(u,v) \notin (rE\ S)^{\frown\frown}5$ **using** *lem-rel-pow5*[*of u v rE S*] **by** *blast*
have $n = 0 \vee n = 1 \vee n = 2 \vee n = 3 \vee n = 4 \vee n = 5$ **using** *a3* **by** *force*
then show *?thesis* **using** *b0 b1 b2 b3 b4 b5* **by** *blast*
qed

lemma *lem-rE-notLD2*:
fixes *S*::*'U set* **and** *r0 r1*::(*'U rD*) *rel*
assumes *a0*: $\neg\ |S| \leq o\ |UNIV::nat\ set|$ **and** *a1*: *LD2* (*rE S*) *r0 r1*
shows *False*
proof −
obtain *x0*::*'U* **where** *b0*: $x0 \in S$ **using** *a0*
by (*metis all-not-in-conv card-of-mono1 card-of-singl-ordLeq empty-subsetI finite.emptyI infinite-UNIV-char-0 ordLeq-transitive*)
obtain *u*::*'U rD* **where** *b1*: $u = (\mathrm{l}0, \{\}, \{\}, \{\})$ **by** *blast*
obtain *v1*::*'U rD* **where** *b2*: $v1 = (\mathrm{l}1, \{\}, \{\}, \{\})$ **by** *blast*
obtain *v2*::*'U rD* **where** *b3*: $v2 = (\mathrm{l}1, \{x0\}, \{\}, \{\})$ **by** *blast*
have *levrd u* = l*0* **using** *b1* **by** *simp*
then have $(u,v1) \notin r0 \wedge (u,v2) \notin r0$ **using** *a0 a1 lem-rE-one*[*of S r0 r1 u*] **by** *blast*
moreover have $(u,v1) \in (rE\ S) \wedge (u,v2) \in (rE\ S)$ **using** *b0 b1 b2 b3* **unfolding** *rE-def rC-def* **by** *simp*
ultimately have $(u,v1) \in r1 \wedge (u,v2) \in r1$ **using** *a1* **unfolding** *LD2-def* **by** *blast*
then have *jn11 r0 r1 v1 v2* **using** *a1* **unfolding** *LD2-def* **by** *blast*
then obtain $b'\ b''\ c'\ c''\ d$ **where**
b4: $(v1, b') \in r0^{\frown}* \wedge (b', b'') \in r1^{\frown}= \wedge (b'', d) \in r0^{\frown}*$
and *b5*: $(v2, c') \in r0^{\frown}* \wedge (c', c'') \in r1^{\frown}= \wedge (c'', d) \in r0^{\frown}*$ **unfolding** *jn11-def* **by** *blast*
have *b6*: $\bigwedge$ *v v'*::*'U rD*. *levrd v* $\in \{\mathrm{l}1, \mathrm{l}3\} \wedge (v, v') \in r0^{\frown}* \Longrightarrow (v,v') \in r0^{\frown}=$
proof −
fix *v v'*::*'U rD*
assume *c1*: *levrd v* $\in \{\mathrm{l}1, \mathrm{l}3\} \wedge (v, v') \in r0^{\frown}*$
then obtain *k1* **where** *c2*: $(v, v') \in r0^{\frown\frown}k1$ **using** *rtrancl-imp-relpow* **by** *blast*
have $k1 \geq 2 \longrightarrow False$
proof
assume $k1 \geq 2$
then obtain *k* **where** $k1 = 2 + k$ **using** *le-Suc-ex* **by** *blast*
then obtain *w'* **where** $(v, w') \in r0^{\frown\frown}2$ **using** *c2 relpow-add*[*of 2 k r0*] **by** *fastforce*

then obtain *w w′* **where** $(v, w) \in r0 \wedge (w, w') \in r0$ **by** (*metis One-nat-def numeral-2-eq-2 relpow-1 relpow-Suc-E*)
moreover then have $(v, w) \in (rE\ S)$ **using** *a1* **unfolding** *LD2-def* **by** *blast*
moreover then have *levrd w* $\in$ {l*2*, l*4*} **using** *c1* **unfolding** *rE-def* **by** *force*
ultimately show *False* **using** *a0 a1 lem-rE-one* **by** *blast*
qed
then have $k1 = 0 \vee k1 = 1$ **by** (*simp add*: *less-2-cases*)
then show $(v, v') \in r0\hat{}{=}$ **using** *c2* **by** *force*
qed
then have *b7*: $(v1, b') \in r0\hat{}{=} \wedge (v2, c') \in r0\hat{}{=}$ **using** *b2 b3 b4 b5* **by** *simp*
have *b8*: *levrd d* $\in$ {l*7*, l*8*}
proof −
have $r0 \subseteq (rE\ S) \wedge r1 \subseteq (rE\ S)$ **using** *a1* **unfolding** *LD2-def* **by** *blast*
then have $r0\hat{}{*} \subseteq (rE\ S)\hat{}{*} \wedge r1\hat{}{=} \subseteq (rE\ S)\hat{}{*}$ **using** *rtrancl-mono* **by** *blast*
then have $(v1, b') \in (rE\ S)\hat{}{*} \wedge (b', b'') \in (rE\ S)\hat{}{*} \wedge (b'', d) \in (rE\ S)\hat{}{*}$
and $(v2, c') \in (rE\ S)\hat{}{*} \wedge (c', c'') \in (rE\ S)\hat{}{*} \wedge (c'', d) \in (rE\ S)\hat{}{*}$ **using** *b4 b5* **by** *blast+*
then have *e1*: $(v1,d) \in (rE\ S)\hat{}{*} \wedge (v2,d) \in (rE\ S)\hat{}{*}$ **by** *force*
have $\bigwedge$ *v v′*::*′U rD*. *levrd v* = l*1* $\longrightarrow (v,v') \in (rE\ S)\hat{}{*} \longrightarrow v \neq v' \longrightarrow$ *levrd v′* $\neq$ l*1*
proof (*intro impI*)
fix *v v′*::*′U rD*
assume *d1*: *levrd v* = l*1* **and** *d2*: $(v,v') \in (rE\ S)\hat{}{*}$ **and** *d3*: $v \neq v'$
moreover then obtain *k* **where** $(v,v') \in (rE\ S)\hat{}\hat{}k$ **using** *rtrancl-imp-relpow* **by** *blast*
ultimately obtain *k′* **where** $(v,v') \in (rE\ S)\hat{}\hat{}(Suc\ k')$ **by** (*cases k*, *force+*)
then obtain *v″* **where** $(v,v'') \in (rE\ S) \wedge (v'',v') \in (rE\ S)\hat{}\hat{}k'$ **by** (*meson relpow-Suc-D2*)
then have *levrd v″* = l*2* $\wedge (v'',v') \in (rE\ S)\hat{}{*}$ **using** *d1 lem-rE-succ-lev*[*of v v″*] *relpow-imp-rtrancl* **by** *force*
moreover have *lev-next* ‘ {l*2*, l*3*, l*4*, l*5*, l*6*, l*7*, l*8*} $\subseteq$ {l*2*, l*3*, l*4*, l*5*, l*6*, l*7*, l*8*} **by** *simp*
ultimately have *levrd v′* $\in$ {l*2*, l*3*, l*4*, l*5*, l*6*, l*7*, l*8*} **using** *lem-rE-levset-inv*[*of v″ v′ S* {l*2*, l*3*, l*4*, l*5*, l*6*, l*7*, l*8*}] **by** *simp*
then show *levrd v′* $\neq$ l*1* **by** *force*
qed
then have $(v1,v2) \notin (rE\ S)\hat{}{*}$ **and** $(v2,v1) \notin (rE\ S)\hat{}{*}$ **using** *b2 b3* **by** *fastforce+*
then show *levrd d* $\in$ {l*7*, l*8*} **using** *e1 lem-rE-jn2* **by** *blast*
qed
then have *b9*: $\forall\ n \leq 5.\ (v1,d) \notin (rE\ S)\hat{}\hat{}n \wedge (v2,d) \notin (rE\ S)\hat{}\hat{}n$ **using** *b2 b3 lem-rE-l1-l78-dist*[*of - d*] **by** *simp*
have *b10*: *levrd b″* = l*2*
proof −
have *c1*: $v1 = b' \vee (v1,b') \in (rE\ S)$ **using** *b7 a1* **unfolding** *LD2-def* **by** *blast*
then have *levrd b′* $\in$ {l*1*, l*2*} **using** *b2 lem-rE-succ-lev*[*of v1 b′*] **by** *force*
moreover have *c2*: $b' = b'' \vee (b',b'') \in (rE\ S)$ **using** *b4 a1* **unfolding** *LD2-def* **by** *blast*

ultimately have *levrd* $b'' \in \{l1, l2, l3\}$ **using** *lem-rE-succ-lev*[*of* b' b''] **by** *force*
moreover have *levrd* $b'' \in \{l1, l3\} \longrightarrow False$
proof
assume *levrd* $b'' \in \{l1, l3\}$
then have $(b'',d) \in r0\hat{}=$ **using** *b4 b6* **by** *blast*
then have *d1*: $b'' = d \vee (b'', d) \in (rE\ S)$ **using** *a1* **unfolding** *LD2-def* **by** *blast*
have $(v1,d) \in (rE\ S)\hat{}\hat{}0 \vee (v1,d) \in (rE\ S)\hat{}\hat{}1 \vee (v1,d) \in (rE\ S)\hat{}\hat{}2 \vee (v1,d) \in (rE\ S)\hat{}\hat{}3$
using *c1 c2 d1 lem-rel-pow2fw*[*of - - rE S*] *lem-rel-pow3fw*[*of - - rE S*] **by** (*metis relpow-0-I relpow-1*)
then show *False* **using** *b9*
by (*meson le0 numeral-le-iff one-le-numeral semiring-norm*(*68*) *semiring-norm*(*72*) *semiring-norm*(*73*))
qed
ultimately show *levrd* $b'' = l2$ **by** *blast*
qed
then have $b'' \neq d$ **using** *b8* **by** *force*
then obtain *t* **where** *b11*: $(b'',t) \in r0 \wedge (t, d) \in r0\hat{}*$ **using** *b4* **by** (*meson converse-rtranclE*)
then have *b12*: $(b'',t) \in (rE\ S)$ **using** *a1* **unfolding** *LD2-def* **by** *blast*
then have *levrd* $t = l3$ **using** *b10 a1 lem-rE-succ-lev*[*of* b'' *t S*] **unfolding** *LD2-def* **by** *simp*
then have $(t,d) \in r0\hat{}=$ **using** *b11 b6* **by** *blast*
then have *b13*: $t = d \vee (t,d) \in (rE\ S)$ **using** *a1* **unfolding** *LD2-def* **by** *blast*
have *b14*: $v1 = b' \vee (v1,b') \in (rE\ S)$ **using** *b7 a1* **unfolding** *LD2-def* **by** *blast*
moreover have *b15*: $b' = b'' \vee (b',b'') \in (rE\ S)$ **using** *b4 a1* **unfolding** *LD2-def* **by** *blast*
ultimately have $(v1,b'') \in (rE\ S)\hat{}\hat{}0 \vee (v1,b'') \in (rE\ S)\hat{}\hat{}1 \vee (v1,b'') \in (rE\ S)\hat{}\hat{}2$
using *lem-rel-pow2fw*[*of - - rE S*] **by** (*metis relpow-0-I relpow-1*)
then have $(v1,t) \in (rE\ S)\hat{}\hat{}1 \vee (v1,t) \in (rE\ S)\hat{}\hat{}2 \vee (v1,t) \in (rE\ S)\hat{}\hat{}3$ **using** *b12 b14 b15*
lem-rel-pow2fw[*of - - rE S*] *lem-rel-pow3fw*[*of - - rE S*] **by** (*metis relpow-1*)
moreover have $(v1,t) \in (rE\ S)\hat{}\hat{}1 \longrightarrow (v1,d) \in (rE\ S)\hat{}\hat{}1 \vee (v1,d) \in (rE\ S)\hat{}\hat{}2$ **using** *b13 lem-rel-pow2fw* **by** *fastforce*
moreover have $(v1,t) \in (rE\ S)\hat{}\hat{}2 \longrightarrow (v1,d) \in (rE\ S)\hat{}\hat{}2 \vee (v1,d) \in (rE\ S)\hat{}\hat{}3$ **using** *b13 relpow-Suc-I* **by** *fastforce*
moreover have $(v1,t) \in (rE\ S)\hat{}\hat{}3 \longrightarrow (v1,d) \in (rE\ S)\hat{}\hat{}3 \vee (v1,d) \in (rE\ S)\hat{}\hat{}4$ **using** *b13 relpow-Suc-I* **by** *fastforce*
ultimately have $\exists\ n \in \{1,2,3,4\}.\ (v1,d) \in (rE\ S)\hat{}\hat{}n$ **by** *blast*
moreover have $\forall\ n \in \{1,2,3,4\}$::*nat set*. $n \leq 5$ **by** *simp*
ultimately show *False* **using** *b9* **by** *blast*
qed

lemma *lem-rE-dominv*:
fixes *S*::'*U set*
assumes ¬ *finite S*

shows $u \in Domain\ (rE\ S) \Longrightarrow (u,v) \in (rE\ S)\hat{}* \Longrightarrow v \in Domain\ (rE\ S)$
using *assms lem-rE-domfield* **unfolding** *Field-def* **by** (*metis Range.RangeI UnCI rtranclE*)

lemma *lem-rE-next*:
fixes $S::'U\ set$
assumes $\neg$ *finite S* **and** $u \in Domain\ (rE\ S)$
shows $\exists\ v.\ (u,v) \in (rE\ S) \wedge v \in Domain\ (rE\ S) \wedge levrd\ v = (lev\text{-}next\ (levrd\ u))$
proof −
obtain u' **where** *b1*: $(u,u') \in (rE\ S)$ **using** *assms* **by** *blast*
obtain *n A B C* **where** *b2*: $u = (n,A,B,C)$ **using** *prod-cases4* **by** *blast*
obtain $n'\ A'\ B'\ C'$ **where** *b3*: $u' = (n',A',B',C')$ **using** *prod-cases4* **by** *blast*
have *b4*: $rP\ n\ A\ B\ C\ n'\ A'\ B'\ C' \wedge rC\ S\ A\ B\ C \wedge rC\ S\ A'\ B'\ C'$ **using** *b1 b2 b3* **unfolding** *rE-def* **by** *blast*
moreover then have $A \subseteq S$ **unfolding** *rC-def* **by** *blast*
moreover then have *b4'*: $\exists A2 \subseteq S.\ A \subset A2 \wedge finite\ A2$
using *b4 assms lem-rP-inv lem-infset-finext*[*of S A*] **by** *metis*
ultimately have $(\exists\ A1\ B1\ C1\ n2\ A2\ B2\ C2.\ rP\ n\ A\ B\ C\ (lev\text{-}next\ n)\ A1\ B1\ C1 \wedge rC\ S\ A1\ B1\ C1$
$\wedge\ rP\ (lev\text{-}next\ n)\ A1\ B1\ C1\ n2\ A2\ B2\ C2 \wedge rC\ S\ A2\ B2\ C2)$
apply (*cases n*)
unfolding *rC-def* **by** *auto+*
then obtain *A1 B1 C1 n2 A2 B2 C2* **where**
$rP\ n\ A\ B\ C\ (lev\text{-}next\ n)\ A1\ B1\ C1 \wedge rC\ S\ A1\ B1\ C1 \wedge rP\ (lev\text{-}next\ n)\ A1\ B1\ C1\ n2\ A2\ B2\ C2 \wedge rC\ S\ A2\ B2\ C2$ **by** *blast*
moreover obtain *v* **where** $v = ((lev\text{-}next\ n),\ A1,\ B1,\ C1)$ **by** *blast*
ultimately have $(u,v) \in (rE\ S) \wedge v \in Domain\ (rE\ S) \wedge levrd\ v = (lev\text{-}next\ (levrd\ u))$
using *b2 b4* **unfolding** *rE-def* **by** *force*
then show *?thesis* **by** *blast*
qed

lemma *lem-rE-reachl8*:
fixes $S::'U\ set$
assumes $\neg$ *finite S* **and** $u \in Domain\ (rE\ S)$
shows $\exists\ v.\ (u,v) \in (rE\ S)\hat{}* \wedge v \in Domain\ (rE\ S) \wedge levrd\ v = l8$
proof −
have $levrd\ u = l8 \longrightarrow\ ?thesis$ **using** *assms* **by** *blast*
moreover have *b0*: $\bigwedge u::'U\ rD.\ u \in Domain\ (rE\ S) \Longrightarrow levrd\ u = l7 \Longrightarrow (\exists\ v.\ (u,v) \in (rE\ S)\hat{}* \wedge v \in Domain\ (rE\ S) \wedge levrd\ v = l8)$
proof −
fix $u::'U\ rD$
assume $u \in Domain\ (rE\ S)$ **and** $levrd\ u = l7$
moreover then have $(lev\text{-}next\ (levrd\ u)) = l8$ **by** *force*
ultimately obtain *v* **where** $(u,v) \in (rE\ S) \wedge v \in Domain\ (rE\ S) \wedge levrd\ v = l8$ **using** *assms lem-rE-next* **by** *metis*
then show $\exists\ v.\ (u,v) \in (rE\ S)\hat{}* \wedge v \in Domain\ (rE\ S) \wedge levrd\ v = l8$ **by** *blast*

qed
moreover have *b1*: $\bigwedge$ *u*::'*U rD*. *u* ∈ *Domain* (*rE S*) ⟹ *levrd u* = l*6* ⟹ (∃ *v*. (*u*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8*)
proof −
fix *u*::'*U rD*
assume *u* ∈ *Domain* (*rE S*) **and** *levrd u* = l*6*
moreover then have (*lev-next* (*levrd u*)) = l*7* **by** *force*
ultimately obtain *v'* **where** (*u*,*v'*) ∈ (*rE S*) ∧ *v'* ∈ *Domain* (*rE S*) ∧ *levrd v'* = l*7* **using** *assms lem-rE-next* **by** *metis*
moreover then obtain *v* **where** (*v'*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8* **using** *b0* **by** *blast*
ultimately have (*u*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8* **by** *force*
then show ∃ *v*. (*u*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8* **by** *blast*
qed
moreover have *b2*: $\bigwedge$ *u*::'*U rD*. *u* ∈ *Domain* (*rE S*) ⟹ *levrd u* = l*5* ⟹ (∃ *v*. (*u*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8*)
proof −
fix *u*::'*U rD*
assume *u* ∈ *Domain* (*rE S*) **and** *levrd u* = l*5*
moreover then have (*lev-next* (*levrd u*)) = l*6* **by** *simp*
ultimately obtain *v'* **where** (*u*,*v'*) ∈ (*rE S*) ∧ *v'* ∈ *Domain* (*rE S*) ∧ *levrd v'* = l*6* **using** *assms lem-rE-next* **by** *metis*
moreover then obtain *v* **where** (*v'*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8* **using** *b1* **by** *blast*
ultimately have (*u*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8* **by** *force*
then show ∃ *v*. (*u*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8* **by** *blast*
qed
moreover have *b3*: $\bigwedge$ *u*::'*U rD*. *u* ∈ *Domain* (*rE S*) ⟹ *levrd u* = l*4* ⟹ (∃ *v*. (*u*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8*)
proof −
fix *u*::'*U rD*
assume *u* ∈ *Domain* (*rE S*) **and** *levrd u* = l*4*
moreover then have (*lev-next* (*levrd u*)) = l*5* **by** *simp*
ultimately obtain *v'* **where** (*u*,*v'*) ∈ (*rE S*) ∧ *v'* ∈ *Domain* (*rE S*) ∧ *levrd v'* = l*5* **using** *assms lem-rE-next* **by** *metis*
moreover then obtain *v* **where** (*v'*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8* **using** *b2* **by** *blast*
ultimately have (*u*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8* **by** *force*
then show ∃ *v*. (*u*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8* **by** *blast*
qed
moreover have *b4*: $\bigwedge$ *u*::'*U rD*. *u* ∈ *Domain* (*rE S*) ⟹ *levrd u* = l*3* ⟹ (∃ *v*. (*u*,*v*) ∈ (*rE S*)^* ∧ *v* ∈ *Domain* (*rE S*) ∧ *levrd v* = l*8*)
proof −

fix *u*::*'U rD*
assume $u \in$ *Domain* (*rE S*) **and** *levrd* $u = l3$
moreover then have (*lev-next* (*levrd u*)) = l4 **by** *simp*
ultimately obtain v' **where** $(u,v') \in$ (*rE S*) $\wedge$ $v' \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v' = l4$ **using** *assms lem-rE-next* **by** *metis*
moreover then obtain *v* **where** $(v',v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$ **using** *b3* **by** *blast*
ultimately have $(u,v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$ **by** *force*
then show $\exists$ *v*. $(u,v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$ **by** *blast*
qed
moreover have *b5*: $\bigwedge$ *u*::*'U rD*. $u \in$ *Domain* (*rE S*) $\Longrightarrow$ *levrd* $u = l2$ $\Longrightarrow$ ($\exists$ *v*. $(u,v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$)
proof −
fix *u*::*'U rD*
assume $u \in$ *Domain* (*rE S*) **and** *levrd* $u = l2$
moreover then have (*lev-next* (*levrd u*)) = l3 **by** *simp*
ultimately obtain v' **where** $(u,v') \in$ (*rE S*) $\wedge$ $v' \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v' = l3$ **using** *assms lem-rE-next* **by** *metis*
moreover then obtain *v* **where** $(v',v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$ **using** *b4* **by** *blast*
ultimately have $(u,v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$ **by** *force*
then show $\exists$ *v*. $(u,v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$ **by** *blast*
qed
moreover have *b6*: $\bigwedge$ *u*::*'U rD*. $u \in$ *Domain* (*rE S*) $\Longrightarrow$ *levrd* $u = l1$ $\Longrightarrow$ ($\exists$ *v*. $(u,v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$)
proof −
fix *u*::*'U rD*
assume $u \in$ *Domain* (*rE S*) **and** *levrd* $u = l1$
moreover then have (*lev-next* (*levrd u*)) = l2 **by** *simp*
ultimately obtain v' **where** $(u,v') \in$ (*rE S*) $\wedge$ $v' \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v' = l2$ **using** *assms lem-rE-next* **by** *metis*
moreover then obtain *v* **where** $(v',v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$ **using** *b5* **by** *blast*
ultimately have $(u,v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$ **by** *force*
then show $\exists$ *v*. $(u,v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$ **by** *blast*
qed
moreover have *b7*: $\bigwedge$ *u*::*'U rD*. $u \in$ *Domain* (*rE S*) $\Longrightarrow$ *levrd* $u = l0$ $\Longrightarrow$ ($\exists$ *v*. $(u,v) \in$ (*rE S*)^* $\wedge$ $v \in$ *Domain* (*rE S*) $\wedge$ *levrd* $v = l8$)
proof −
fix *u*::*'U rD*
assume $u \in$ *Domain* (*rE S*) **and** *levrd* $u = l0$
moreover then have (*lev-next* (*levrd u*)) = l1 **by** *simp*
ultimately obtain v' **where** $(u,v') \in$ (*rE S*) $\wedge$ $v' \in$ *Domain* (*rE S*) $\wedge$ *levrd*

$v' = l1$ **using** *assms lem-rE-next* **by** *metis*
moreover then obtain *v* **where** $(v',v) \in (rE\ S)\hat{}*\ \wedge\ v \in Domain\ (rE\ S) \wedge levrd\ v = l8$ **using** *b6* **by** *blast*
ultimately have $(u,v) \in (rE\ S)\hat{}* \wedge v \in Domain\ (rE\ S) \wedge levrd\ v = l8$ **by** *force*
then show $\exists\ v.\ (u,v) \in (rE\ S)\hat{}* \wedge v \in Domain\ (rE\ S) \wedge levrd\ v = l8$ **by** *blast*
qed
ultimately show *?thesis* **using** *assms* **by** (*meson lev-next.cases*)
qed

lemma *lem-rE-jn*:
fixes $S::'U\ set$
assumes *a0*: $\neg$ *finite S* **and** *a1*: $u1 \in Domain\ (rE\ S)$ **and** *a2*: $u2 \in Domain\ (rE\ S)$
shows $\exists\ t.\ (u1,t) \in (rE\ S)\hat{}* \wedge (u2,t) \in (rE\ S)\hat{}*$
proof −
obtain *v1* **where** *b1*: $(u1,v1) \in (rE\ S)\hat{}*$ **and** *b2*: $v1 \in Domain\ (rE\ S) \wedge levrd\ v1 = l8$ **using** *a0 a1 lem-rE-reachl8* **by** *blast*
obtain *v2* **where** *b3*: $(u2,v2) \in (rE\ S)\hat{}*$ **and** *b4*: $v2 \in Domain\ (rE\ S) \wedge levrd\ v2 = l8$ **using** *a0 a2 lem-rE-reachl8* **by** *blast*
obtain *n1 A1 B1 C1* **where** *b5*: $v1 = (n1,A1,B1,C1)$ **using** *prod-cases4* **by** *blast*
obtain *n2 A2 B2 C2* **where** *b6*: $v2 = (n2,A2,B2,C2)$ **using** *prod-cases4* **by** *blast*
have *b7*: $n1 = l8 \wedge A1 = B1 \wedge A1 = C1 \wedge$ *finite* $A1 \wedge A1 \subseteq S$ **using** *b5 b2* **unfolding** *rE-def rC-def* **by** *force*
have *b8*: $n2 = l8 \wedge A2 = B2 \wedge A2 = C2 \wedge$ *finite* $A2 \wedge A2 \subseteq S$ **using** *b6 b4* **unfolding** *rE-def rC-def* **by** *force*
have *finite* $(A1 \cup A2) \wedge A1 \cup A2 \subseteq S$ **using** *b7 b8* **by** *blast*
then obtain *A3* **where** $A3 \subseteq S \wedge A1 \cup A2 \subset A3 \wedge$ *finite A3* **using** *a0 lem-infset-finext*[*of S A1* $\cup$ *A2*] **by** *blast*
moreover obtain *t* **where** $t = (l7,\ A3,\ A3,\ A3)$ **by** *blast*
ultimately have $(v1,\ t) \in (rE\ S) \wedge (v2,\ t) \in (rE\ S)$ **using** *b5 b6 b7 b8* **unfolding** *rE-def rC-def* **by** *force*
then have $(u1,t) \in (rE\ S)\hat{}* \wedge (u2,t) \in (rE\ S)\hat{}*$ **using** *b1 b3* **by** *force*
then show *?thesis* **by** *blast*
qed

lemma *lem-rE-confl*:
fixes $S::'U\ set$
assumes $\neg$ *finite S*
shows *confl-rel* (*rE S*)
proof −
have $\forall\ a\ b\ c::'U\ rD.\ (a,b) \in (rE\ S)\hat{}* \longrightarrow (a,c) \in (rE\ S)\hat{}* \longrightarrow (\exists\ d.\ (b,d) \in (rE\ S)\hat{}* \wedge (c,d) \in (rE\ S)\hat{}*)$
proof (*intro allI impI*)
fix $a\ b\ c::'U\ rD$
assume *c1*: $(a,b) \in (rE\ S)\hat{}*$ **and** *c2*: $(a,c) \in (rE\ S)\hat{}*$

show ∃ *d*. (*b*,*d*) ∈ (*rE S*)^* ∧ (*c*,*d*) ∈ (*rE S*)^*
proof (*cases a* ∈ *Domain* (*rE S*))
assume *a* ∈ *Domain* (*rE S*)
then have *b* ∈ *Domain* (*rE S*) ∧ *c* ∈ *Domain* (*rE S*) **using** *c1 c2 assms lem-rE-dominv* **by** *blast*
then obtain *d* **where** (*b*,*d*) ∈ (*rE S*)^* ∧ (*c*,*d*) ∈ (*rE S*)^* **using** *assms lem-rE-jn* **by** *blast*
then show *?thesis* **by** *blast*
next
assume *a* ∉ *Domain* (*rE S*)
then have *a* = *b* ∧ *a* = *c* **using** *c1 c2* **by** (*meson Not-Domain-rtrancl*)
then show *?thesis* **by** *blast*
qed
qed
then show *?thesis* **unfolding** *confl-rel-def* **by** *blast*
qed

lemma *lem-rE-dc3dc2*:
fixes *S*::*'U set*
assumes ¬ |*S*| ≤*o* |*UNIV*::*nat set*|
shows *confl-rel* (*rE S*) ∧ (¬ *DCR2* (*rE S*))
proof (*intro conjI*)
have ¬ *finite S* **using** *assms* **by** (*meson card-of-Well-order infinite-iff-card-of-nat ordLeq-total*)
then show *confl-rel* (*rE S*) **using** *lem-rE-confl* **by** *blast*
next
show ¬ *DCR2* (*rE S*) **using** *assms lem-rE-notLD2* **unfolding** *DCR2-def* **by** *blast*
qed

lemma *lem-rE-cardbnd*:
fixes *S*::*'U set*
assumes ¬ *finite S*
shows |*rE S*| ≤*o* |*S*|
proof −
obtain *L* **where** *b1*: *L* = (*UNIV*::*Lev set*) **by** *blast*
obtain *F* **where** *b2*: *F* = { *A*. *A* ⊆ *S* ∧ *finite A* } **by** *blast*
obtain *D* **where** *b3*: *D* = (*L* × (*F* × (*F* × *F*))) **by** *blast*
have ∀ *u v*. (*u*,*v*) ∈ *rE S* ⟶ *u* ∈ *D* ∧ *v* ∈ *D*
proof (*intro allI impI*)
fix *u v*
assume (*u*,*v*) ∈ *rE S*
then obtain *n A B C n' A' B' C'*
where *u* = (*n*,*A*,*B*,*C*) ∧ *v* = (*n'*,*A'*,*B'*,*C'*) ∧ *rC S A B C* ∧ *rC S A' B' C'* ∧ *rP n A B C n' A' B' C'* **unfolding** *rE-def* **by** *blast*
moreover then have *n* ∈ *L* ∧ *A* ∈ *F* ∧ *B* ∈ *F* ∧ *C* ∈ *F* ∧ *n'* ∈ *L* ∧ *A'* ∈ *F* ∧ *B'* ∈ *F* ∧ *C'* ∈ *F*
using *b1 b2 lem-rP-inv* **unfolding** *rC-def* **by** *fast*
ultimately show *u* ∈ *D* ∧ *v* ∈ *D* **using** *b3* **by** *blast*

qed
then have *rE S* ⊆ *D* × *D* **by** *force*
then have |*rE S*| ≤*o* |*D* × *D*| **by** *simp*
moreover have |*D* × *D*| ≤*o* |*S*|
proof −
have *F* = *Fpow S* **using** *b2* **unfolding** *Fpow-def* **by** *simp*
then have *c1*: |*F*| =*o* |*S*| **using** *assms* **by** *simp*
then have |*F* × *F*| =*o* |*F*| ∧ ¬ *finite F* **using** *assms* **by** *simp*
then have |*F*| ≤*o* |*F*| ∧ |*F* × *F*| ≤*o* |*F*| ∧ ¬ *finite F* **using** *ordIso-iff-ordLeq* **by** *force*
then have *c2*: |*F* × (*F* × *F*)| ≤*o* |*S*| **using** *c1 card-of-Times-ordLeq-infinite ordLeq-ordIso-trans* **by** *blast*
have *L* ⊆ {l*0*,l*1*,l*2*,l*3*,l*4*,l*5*,l*6*,l*7*,l*8*}
proof
fix *l*
assume *l* ∈ *L*
show *l* ∈ {l*0*,l*1*,l*2*,l*3*,l*4*,l*5*,l*6*,l*7*,l*8*} **by** (*cases l*, *simp*+)
qed
moreover have *finite* {l*0*,l*1*,l*2*,l*3*,l*4*,l*5*,l*6*,l*7*,l*8*} **by** *simp*
ultimately have *finite L* **using** *finite-subset* **by** *blast*
then have |*L*| ≤*o* |*S*| **using** *assms ordLess-imp-ordLeq* **by** *force*
then have |*D*| ≤*o* |*S*| **using** *b3 c2 assms card-of-Times-ordLeq-infinite* **by** *blast*
then show *?thesis* **using** *assms card-of-Times-ordLeq-infinite* **by** *blast*
qed
ultimately show |*rE S*| ≤*o* |*S*| **using** *ordLeq-transitive* **by** *blast*
qed

lemma *lem-fmap-rel*:
fixes *f r s a0 b0*
assumes *a1*: (*a0*, *b0*) ∈ *r*^∗ **and** *a2*: ∀ *a b*. (*a*,*b*) ∈ *r* ⟶ (*f a*, *f b*) ∈ *s*
shows (*f a0*, *f b0*) ∈ *s*^∗
proof −
have ⋀ *n*. ∀ *a b*. (*a*,*b*) ∈ *r*^^*n* ⟶ (*f a*, *f b*) ∈ *s*^∗
proof −
fix *n0*
show ∀ *a b*. (*a*,*b*) ∈ *r*^^*n0* ⟶ (*f a*, *f b*) ∈ *s*^∗
proof (*induct n0*)
show ∀ *a b*. (*a*,*b*) ∈ *r*^^*0* ⟶ (*f a*, *f b*) ∈ *s*^∗ **by** *simp*
next
fix *n*
assume ∀ *a b*. (*a*,*b*) ∈ *r*^^*n* ⟶ (*f a*, *f b*) ∈ *s*^∗
then show ∀ *a b*. (*a*,*b*) ∈ *r*^^(*Suc n*) ⟶ (*f a*, *f b*) ∈ *s*^∗ **using** *a2* **by** *force*
qed
qed
then show *?thesis* **using** *a1 rtrancl-power* **by** *blast*
qed

lemma *lem-fmap-confl*:
fixes *r*::′*a rel* **and** *f*::′*a* ⇒ ′*b*

assumes *a1*: *inj-on f* (*Field r*) **and** *a2*: *confl-rel r*
shows *confl-rel* {(*u*,*v*). ∃ *a b*. *u* = *f a* ∧ *v* = *f b* ∧ (*a*,*b*) ∈ *r*}
proof −
  obtain *rA* **where** *q1*: *rA* = {(*u*,*v*). ∃ *a b*. *u* = *f a* ∧ *v* = *f b* ∧ (*a*,*b*) ∈ *r*} **by** *blast*
  then have *q2*: ∀ *a b*. (*a*, *b*) ∈ *r* ⟶ (*f a*, *f b*) ∈ *rA* **by** *blast*
  have *q3*: *Field rA* ⊆ *f*‘(*Field r*) **using** *q1* **unfolding** *Field-def* **by** *blast*
  obtain *g* **where** *q4*: *g* = *inv-into* (*Field r*) *f* **by** *blast*
  then have *q5*: ∀ *x* ∈ *Field r*. *g* (*f x*) = *x* **using** *a1* **by** *simp*
  have *q6*: ∀ *u v*. (*u*,*v*) ∈ *rA* ⟶ (*g u*, *g v*) ∈ *r*
  proof (*intro allI impI*)
    fix *u v*
    assume (*u*,*v*) ∈ *rA*
    then obtain *a b* **where** *u* = *f a* ∧ *v* = *f b* ∧ (*a*,*b*) ∈ *r* **using** *q1* **by** *blast*
    moreover then have *a* ∈ *Field r* ∧ *b* ∈ *Field r* **unfolding** *Field-def* **by** *blast*
    ultimately show (*g u*, *g v*) ∈ *r* **using** *q5* **by** *force*
  qed
  have ∀ *u* ∈ *Field rA*. ∀ *v* ∈ *Field rA*. ∀ *w* ∈ *Field rA*.
    (*u*,*v*) ∈ *rA*^* ∧ (*u*,*w*) ∈ *rA*^* ⟶ (∃ *t* ∈ *Field rA*. (*v*,*t*) ∈ *rA*^* ∧ (*w*,*t*) ∈ *rA*^*)
  proof (*intro ballI impI*)
    fix *u v w*
    assume *c1*: *u* ∈ *Field rA* **and** *c2*: *v* ∈ *Field rA* **and** *c3*: *w* ∈ *Field rA*
      and *c4*: (*u*,*v*) ∈ *rA*^* ∧ (*u*,*w*) ∈ *rA*^*
    then have (*g u*, *g v*) ∈ *r*^* ∧ (*g u*, *g w*) ∈ *r*^* **using** *q6 lem-fmap-rel*[*of u - rA g r*] **by** *blast*
    then obtain *d* **where** *c5*: (*g v*, *d*) ∈ *r*^* ∧ (*g w*, *d*) ∈ *r*^* **using** *a2* **unfolding** *confl-rel-def* **by** *blast*
    moreover have *c6*: *g v* ∈ *Field r* ∧ *g w* ∈ *Field r* **using** *c2 c3 q3 q5* **by** *force*
    ultimately have *d* ∈ *Field r* **using** *lem-rtr-field* **by** *fastforce*
    have *v* = *f* (*g v*) ∧ *w* = *f* (*g w*) **using** *c2 c3 q3 q4 a1* **by** *force*
    moreover have (*f* (*g v*), *f d*) ∈ *rA*^* ∧ (*f* (*g w*), *f d*) ∈ *rA*^*
      using *c5 q2 lem-fmap-rel*[*of - d r f rA*] **by** *blast*
    ultimately have (*v*, *f d*) ∈ *rA*^* ∧ (*w*, *f d*) ∈ *rA*^* **by** *simp*
    moreover then have *f d* ∈ *Field rA* **using** *c2 lem-rtr-field* **by** *fastforce*
    ultimately show ∃ *t* ∈ *Field rA*. (*v*,*t*) ∈ *rA*^* ∧ (*w*,*t*) ∈ *rA*^* **by** *blast*
  qed
  then show *?thesis* **using** *q1 lem-confl-field* **by** *blast*
qed

lemma *lem-fmap-dcn*:
fixes *r*::′*a rel* **and** *f*::′*a* ⇒ ′*b*
assumes *a1*: *inj-on f* (*Field r*) **and** *a2*: *DCR n r*
shows *DCR n* {(*u*,*v*). ∃ *a b*. *u* = *f a* ∧ *v* = *f b* ∧ (*a*,*b*) ∈ *r*}
proof −
  obtain *rA* **where** *q1*: *rA* = {(*u*,*v*). ∃ *a b*. *u* = *f a* ∧ *v* = *f b* ∧ (*a*,*b*) ∈ *r*} **by** *blast*
  have *q2*: ∀ *a* ∈ *Field r*. ∀ *b* ∈ *Field r*. (*a*,*b*) ∈ *r* ⟷ (*f a*, *f b*) ∈ *rA*
    using *a1 q1* **unfolding** *Field-def inj-on-def* **by** *blast*

have *q3*: *Field rA* $\subseteq$ *f*'(*Field r*) **using** *q1* **unfolding** *Field-def* **by** *blast*
obtain *g*::*nat* $\Rightarrow$ *'a rel* **where** *b1*: *DCR-generating g*
and *b2*: $r = \bigcup \{ r'. \exists \alpha'. \alpha' < n \wedge r' = g\ \alpha' \}$ **using** *a2* **unfolding** *DCR-def* **by** *blast*
obtain *gA*::*nat* $\Rightarrow$ *'b rel*
where *b3*: $gA = (\lambda\ \alpha.\ if\ \alpha < n\ then\ \{(x,y).\ \exists\ a\ b.\ x = f\ a \wedge y = f\ b \wedge (a,b) \in g\ \alpha\ \}\ else\ \{\})$ **by** *blast*
have $\forall \alpha\ \beta\ u\ v\ w.\ (u,\ v) \in gA\ \alpha \wedge (u,\ w) \in gA\ \beta \longrightarrow$
$(\exists v'\ v''\ w'\ w''\ e.\ (v,\ v',\ v'',\ e) \in \mathfrak{D}\ gA\ \alpha\ \beta \wedge (w,\ w',\ w'',\ e) \in \mathfrak{D}\ gA\ \beta\ \alpha)$
proof (*intro allI impI*)
fix $\alpha\ \beta\ u\ v\ w$
assume *c1*: $(u,\ v) \in gA\ \alpha \wedge (u,\ w) \in gA\ \beta$
obtain *a b* **where** *c2*: $\alpha < n \wedge u = f\ a \wedge v = f\ b \wedge (a,b) \in g\ \alpha$ **using** *c1 b3* **by** (*cases* $\alpha < n$, *force+*)
obtain *a' c* **where** *c3*: $\beta < n \wedge u = f\ a' \wedge w = f\ c \wedge (a',c) \in g\ \beta$ **using** *c1 b3* **by** (*cases* $\beta < n$, *force+*)
have $(a,b) \in r \wedge (a',c) \in r$ **using** *c2 c3 b2* **by** *blast*
then have $a' = a$ **using** *c2 c3 a1* **unfolding** *inj-on-def Field-def* **by** *blast*
then have $(a,b) \in g\ \alpha \wedge (a,c) \in g\ \beta$ **using** *c2 c3* **by** *blast*
then obtain *b' b'' c' c'' d* **where** *c4*: $(b,\ b',\ b'',\ d) \in \mathfrak{D}\ g\ \alpha\ \beta \wedge (c,\ c',\ c'',\ d) \in \mathfrak{D}\ g\ \beta\ \alpha$
using *b1* **unfolding** *DCR-generating-def* **by** *blast*
have *c5*: $\bigwedge \alpha'.\ \alpha' < n \Longrightarrow \forall\ a0\ b0.\ (a0,b0) \in \mathfrak{L}1\ g\ \alpha' \longrightarrow (f\ a0,\ f\ b0) \in \mathfrak{L}1\ gA\ \alpha'$
proof (*intro allI impI*)
fix α' *a0 b0*
assume *d1*: $\alpha' < n$ **and** $(a0,b0) \in \mathfrak{L}1\ g\ \alpha'$
then obtain α'' **where** $(a0,b0) \in g\ \alpha'' \wedge \alpha'' < \alpha'$ **unfolding** $\mathfrak{L}$*1-def* **by** *blast*
moreover then have $(f\ a0,\ f\ b0) \in gA\ \alpha''$ **using** *d1 c2 b3* **by** *force*
ultimately show $(f\ a0,\ f\ b0) \in \mathfrak{L}1\ gA\ \alpha'$ **using** *c2 b3* **unfolding** $\mathfrak{L}$*1-def* **by** *blast*
qed
have *c6*: $\bigwedge \alpha'\ a0\ b0.\ \alpha' < n \Longrightarrow (a0,b0) \in (g\ \alpha')\hat{}{=} \longrightarrow (f\ a0,\ f\ b0) \in (gA\ \alpha')\hat{}{=}$ **using** *b3* **by** *force*
have *c7*: $\bigwedge \alpha'\ \beta'.\ \alpha' < n \Longrightarrow \beta' < n \Longrightarrow \forall\ a0\ b0.\ (a0,b0) \in \mathfrak{L}v\ g\ \alpha'\ \beta' \longrightarrow (f\ a0,\ f\ b0) \in \mathfrak{L}v\ gA\ \alpha'\ \beta'$
proof (*intro allI impI*)
fix $\alpha'\ \beta'$ *a0 b0*
assume *d1*: $\alpha' < n$ **and** *d2*: $\beta' < n$ **and** $(a0,b0) \in \mathfrak{L}v\ g\ \alpha'\ \beta'$
then obtain α'' **where** $(a0,b0) \in g\ \alpha'' \wedge (\alpha'' < \alpha' \vee \alpha'' < \beta')$ **unfolding** $\mathfrak{L}$*v-def* **by** *blast*
moreover then have $(f\ a0,\ f\ b0) \in gA\ \alpha''$ **using** *d1 d2 c2 b3* **by** *force*
ultimately show $(f\ a0,\ f\ b0) \in \mathfrak{L}v\ gA\ \alpha'\ \beta'$ **using** *c2 b3* **unfolding** $\mathfrak{L}$*v-def* **by** *blast*
qed
have $(v,\ f\ b') \in (\mathfrak{L}1\ gA\ \alpha)\hat{}*$ **using** *c2 c4 c5*[*of* α] *lem-fmap-rel*[*of b b'*] **unfolding** $\mathfrak{D}$*-def* **by** *blast*
moreover have $(f\ b',\ f\ b'') \in (gA\ \beta)\hat{}{=}$ **using** *c3 c4 c6* **unfolding** $\mathfrak{D}$*-def* **by**

blast
moreover have $(f\ b'',\ f\ d) \in (\mathfrak{L}v\ gA\ \alpha\ \beta)\hat{}*$ **using** *c2 c3 c4 c7*[*of* α β] *lem-fmap-rel*[*of* b'' *d*] **unfolding** $\mathfrak{D}$-*def* **by** *blast*
moreover have $(w, f\ c') \in (\mathfrak{L}1\ gA\ \beta)\hat{}*$ **using** *c3 c4 c5*[*of* β] *lem-fmap-rel*[*of* *c* c'] **unfolding** $\mathfrak{D}$-*def* **by** *blast*
moreover have $(f\ c', f\ c'') \in (gA\ \alpha)\hat{}=$ **using** *c2 c4 c6* **unfolding** $\mathfrak{D}$-*def* **by** *blast*
moreover have $(f\ c'',\ f\ d) \in (\mathfrak{L}v\ gA\ \beta\ \alpha)\hat{}*$ **using** *c2 c3 c4 c7*[*of* β α] *lem-fmap-rel*[*of* c'' *d*] **unfolding** $\mathfrak{D}$-*def* **by** *blast*
ultimately show $\exists\, v'\ v''\ w'\ w''\ e.\ (v,\ v',\ v'',\ e) \in \mathfrak{D}\ gA\ \alpha\ \beta \wedge (w,\ w',\ w'',\ e) \in \mathfrak{D}\ gA\ \beta\ \alpha$
unfolding $\mathfrak{D}$-*def* **by** *blast*
qed
then have *DCR-generating gA* **unfolding** *DCR-generating-def* **by** *blast*
moreover have $rA = \bigcup\ \{\ r'.\ \exists\ \alpha'.\ \alpha' < n \wedge r' = gA\ \alpha'\}$
proof
show $rA \subseteq \bigcup\ \{\ r'.\ \exists\ \alpha'.\ \alpha' < n \wedge r' = gA\ \alpha'\}$
proof
fix *p*
assume $p \in rA$
then obtain *x y* **where** *d1*: $p = (x,y) \wedge p \in rA$ **by** *force*
moreover then obtain *a b* **where** *d2*: $x = f\ a \wedge y = f\ b \wedge a \in Field\ r \wedge b \in Field\ r$
using *q3* **unfolding** *Field-def* **by** *blast*
ultimately have $(a,b) \in r$ **using** *q2* **by** *blast*
then obtain α' **where** $\alpha' < n \wedge (a,b) \in g\ \alpha'$ **using** *b2* **by** *blast*
then have $\alpha' < n \wedge (x,y) \in gA\ \alpha'$ **using** *d2 b3* **by** *force*
then show $p \in \bigcup\{r'.\ \exists\alpha'{<}n.\ r' = gA\ \alpha'\}$ **using** *d1* **by** *blast*
qed
next
show $\bigcup\ \{\ r'.\ \exists\ \alpha'.\ \alpha' < n \wedge r' = gA\ \alpha'\} \subseteq rA$
proof
fix *p*
assume $p \in \bigcup\ \{\ r'.\ \exists\ \alpha'.\ \alpha' < n \wedge r' = gA\ \alpha'\}$
then obtain α' **where** *d1*: $\alpha' < n \wedge p \in gA\ \alpha'$ **by** *blast*
then obtain *x y* **where** *d2*: $p = (x,y) \wedge p \in gA\ \alpha'$ **by** *force*
then obtain *a b* **where** $x = f\ a \wedge y = f\ b \wedge (a,b) \in g\ \alpha'$ **using** *d1 b3* **by** *force*
moreover then have $(a,b) \in r$ **using** *d1 b2* **by** *blast*
ultimately show $p \in rA$ **using** *d2 q2* **unfolding** *Field-def* **by** *blast*
qed
qed
ultimately have *DCR n rA* **unfolding** *DCR-def* **by** *blast*
then show *?thesis* **using** *q1* **by** *blast*
qed

lemma *lem-not-dcr2*:
assumes *cardSuc* $|UNIV{::}nat\ set| \le o\ |UNIV{::}'U\ set|$
shows $\exists\ r{::}'U\ rel.\ confl\text{-}rel\ r \wedge |r| \le o\ cardSuc\ |UNIV{::}nat\ set| \wedge (\neg\ DCR2\ r)$

proof −
  obtain *A* **where** *b1*: $A = (UNIV::'U\ set)$ **by** *blast*
  obtain *S* **where** *b2*: $S \subseteq A \wedge |S| =o\ cardSuc\ |UNIV::nat\ set|$
    using *b1 assms*
    by (*smt Card-order-ordIso2 Field-card-of cardSuc-Card-order card-of-Field-ordIso*

        card-of-card-order-on internalize-ordLeq ordIso-symmetric ordIso-transitive)
  then have $\neg$ ($|S| \leq o\ |UNIV::nat\ set|$) **by** (*simp add*: *cardSuc-ordLess-ordLeq ordIso-iff-ordLeq*)
  moreover then have $\neg$ *finite S* **by** (*meson card-of-Well-order infinite-iff-card-of-nat ordLeq-total*)
  moreover obtain *s* **where** *b3*: $s = (rE\ S)$ **by** *blast*
  ultimately have *b4*: *confl-rel s* $\wedge \neg$ *DCR2 s* $\wedge |s| \leq o\ |S|$ **using** *lem-rE-dc3dc2 lem-rE-cardbnd* **by** *blast*
  obtain *B* **where** *b5*: $B = Field\ s$ **by** *blast*
  obtain $C::'U\ set$ **where** *b6*: $C = UNIV$ **by** *blast*
  then have $cardSuc\ |UNIV::nat\ set| \leq o\ |C|$ **using** *assms* **by** *blast*
  moreover have *b6'*: $|s| \leq o\ cardSuc\ |UNIV::nat\ set|$ **using** *b2 b4 ordLeq-ordIso-trans* **by** *blast*
  ultimately have $|s| \leq o\ |C|$ **using** *ordLeq-transitive* **by** *blast*
  moreover have *b6''*: $\neg$ *finite* (*Field s*) $\longrightarrow |Field\ s| =o\ |s|$ **using** *lem-fin-fl-rel lem-rel-inf-fld-card* **by** *blast*
  ultimately have $\neg$ *finite* (*Field s*) $\longrightarrow |Field\ s| \leq o\ |C|$ **using** *ordIso-ordLeq-trans* **by** *blast*
  moreover have $\neg$ *finite C* **using** *b6 assms ordLeq-finite-Field* **by** *fastforce*
  moreover then have *finite* (*Field s*) $\longrightarrow |Field\ s| \leq o\ |C|$ **using** *ordLess-imp-ordLeq* **by** *force*
  ultimately have $|B| \leq o\ |C|$ **using** *b5* **by** *blast*
  then obtain *f* **where** *b7*: $f`B \subseteq C \wedge$ *inj-on f B* **by** (*meson card-of-ordLeq*)
  moreover obtain *g* **where** *b8*: $g =$ *inv-into B f* **by** *blast*
  ultimately have *b9*: $\forall\ x \in B.\ g\ (f\ x) = x$ **by** *simp*
  obtain *r* **where** *b10*: $r = \{(a,b).\ \exists\ x\ y.\ a = f\ x \wedge b = f\ y \wedge (x,y) \in s\}$ **by** *blast*
  have $s \subseteq \{(x,y).\ \exists\ a\ b.\ x = g\ a \wedge y = g\ b \wedge (a,b) \in r\}$
  proof
    fix *p*
    assume $p \in s$
    then obtain *x y* **where** $p = (x,y) \wedge (x,y) \in s$ **by** (*cases p*, *blast*)
    moreover then have $(f\ x,\ f\ y) \in r \wedge x \in B \wedge y \in B$ **using** *b5 b10* **unfolding** *Field-def* **by** *blast*
    moreover then have $x = g\ (f\ x) \wedge y = g\ (f\ y)$ **using** *b9* **by** *simp*
    ultimately show $p \in \{(x,y).\ \exists\ a\ b.\ x = g\ a \wedge y = g\ b \wedge (a,b) \in r\}$ **using** *b9* **by** *blast*
  qed
  moreover have $\{(x,y).\ \exists\ a\ b.\ x = g\ a \wedge y = g\ b \wedge (a,b) \in r\} \subseteq s$
  proof
    fix *p*
    assume $p \in \{(x,y).\ \exists\ a\ b.\ x = g\ a \wedge y = g\ b \wedge (a,b) \in r\}$
    then obtain *a b* **where** $p = (g\ a,\ g\ b) \wedge (a,b) \in r$ **by** *blast*
    moreover then obtain *x y* **where** $a = f\ x \wedge b = f\ y \wedge (x,y) \in s$ **using** *b10*

by *blast*
moreover then have $x \in B \wedge y \in B$ **using** *b5* **unfolding** *Field-def* **by** *blast*
ultimately show $p \in s$ **using** *b9* **by** *force*
qed
ultimately have *b11*: $s = \{(x,y).\ \exists\ a\ b.\ x = g\ a \wedge y = g\ b \wedge (a,b) \in r\}$ **by** *blast*
have *inj-on g* (*f'B*) **using** *b8 inj-on-inv-into*[*of f'B f B*] **by** *blast*
moreover have *b12*: *Field r* $\subseteq$ *f'B*
proof
fix *c*
assume $c \in$ *Field r*
then obtain *a b* **where** $(a,b) \in r \wedge (c = a \vee c = b)$ **unfolding** *Field-def* **by** *blast*
moreover then obtain *x y* **where** $a = f\ x \wedge b = f\ y \wedge (x,y) \in s$ **using** *b10* **by** *blast*
moreover then have $x \in B \wedge y \in B$ **using** *b5* **unfolding** *Field-def* **by** *blast*
ultimately show $c \in f$ ' B **by** *blast*
qed
ultimately have *inj-on g* (*Field r*) **using** *Fun.inj-on-subset* **by** *blast*
moreover have $\neg$ *DCR 2 s* **using** *b4 lem-dc2-to-d2* **by** *blast*
ultimately have $\neg$ *DCR 2 r* **using** *b11 lem-fmap-dcn*[*of g r 2*] **by** *blast*
then have $\neg$ *DCR2 r* **using** *lem-d2-to-dc2* **by** *blast*
moreover have *confl-rel r* **using** *b4 b5 b7 b10 lem-fmap-confl*[*of f s*] **by** *blast*
moreover have $|r| \leq o$ *cardSuc* $|UNIV::nat\ set|$
proof –
have *finite* (*Field s*) $\longrightarrow |B| \leq o$ *cardSuc* $|UNIV::nat\ set|$ **using** *b2 b5*
by (*metis Field-card-of cardSuc-greater card-of-card-order-on finite-ordLess-infinite2*

infinite-UNIV-nat ordLeq-transitive ordLess-imp-ordLeq)
moreover have $\neg$ *finite* (*Field s*) $\longrightarrow |B| \leq o$ *cardSuc* $|UNIV::nat\ set|$
using *b5 b6' b6'' ordIso-ordLeq-trans* **by** *blast*
ultimately have $|B| \leq o$ *cardSuc* $|UNIV::nat\ set|$ **by** *blast*
moreover have $|f'B| \leq o\ |B|$ **by** *simp*
moreover have $|Field\ r| \leq o\ |f'B|$ **using** *b12* **by** *simp*
ultimately have $|Field\ r| \leq o$ *cardSuc* $|UNIV::nat\ set|$ **using** *ordLeq-transitive* **by** *metis*
then have $\neg$ *finite r* $\longrightarrow |r| \leq o$ *cardSuc* $|UNIV::nat\ set|$
using *lem-rel-inf-fld-card*[*of r*] *ordIso-ordLeq-trans ordIso-symmetric* **by** *blast*
moreover have *finite r* $\longrightarrow |r| \leq o$ *cardSuc* $|UNIV::nat\ set|$ **by** (*simp add*: *ordLess-imp-ordLeq*)
ultimately show *?thesis* **by** *blast*
qed
ultimately show *?thesis* **by** *blast*
qed

1.3.3 Result

The next theorem has the following meaning: if the set of elements of type $'U$ is uncountable, then there exists a confluent binary relation r on $'U$ such that the cardinality of r does not exceed the first uncountable cardinal and confluence of r cannot be proved using the decreasing diagrams method with 2 labels.

theorem *thm-example-not-dcr2*:
assumes *cardSuc* $|\{n::nat.\ True\}| \leq o\ |\{x::'U.\ True\}|$
shows $\exists\ r::'U\ rel.$ (
$(\ \forall a\ b\ c.\ (a,b) \in r\hat{}* \wedge (a,c) \in r\hat{}* \longrightarrow (\exists\ d.\ (b,d) \in r\hat{}* \wedge (c,d) \in r\hat{}*)$
)
$\wedge\ |r| \leq o\ cardSuc\ |\{n::nat.\ True\}|$
$\wedge\ (\neg\ (\ \exists r0\ r1.\ ($
$(\ r = (r0 \cup r1)\ )$
$\wedge\ (\forall\ a\ b\ c.\ (a,b) \in r0 \wedge (a,c) \in r0$
$\longrightarrow (\exists\ d.$
$(b,d) \in r0\hat{}=$
$\wedge\ (c,d) \in r0\hat{}=)\ )$
$\wedge\ (\forall\ a\ b\ c.\ (a,b) \in r0 \wedge (a,c) \in r1$
$\longrightarrow (\exists b'\ d.$
$(b,b') \in r1\hat{}= \wedge (b',d) \in r0\hat{}*$
$\wedge\ (c,d) \in r0\hat{}*)\ )$
$\wedge\ (\forall\ a\ b\ c.\ (a,b) \in r1 \wedge (a,c) \in r1$
$\longrightarrow (\exists b'\ b''\ c'\ c''\ d.$
$(b,b') \in r0\hat{}* \wedge (b',b'') \in r1\hat{}= \wedge (b'',d) \in r0\hat{}*$
$\wedge\ (c,c') \in r0\hat{}* \wedge (c',c'') \in r1\hat{}= \wedge (c'',d) \in r0\hat{}*)\ )\ )\ )$
$)\ )$
proof −
have *cardSuc* $|UNIV::nat\ set| \leq o\ |UNIV::'U\ set|$ **using** *assms* **by** (*simp only*: *UNIV-def*)
then have $\exists\ r::'U\ rel.$ *confl-rel* $r \wedge |r| \leq o\ cardSuc\ |UNIV::nat\ set| \wedge (\neg\ DCR2\ r)$
using *assms lem-not-dcr2* **by** *blast*
then show *?thesis* **unfolding** *confl-rel-def DCR2-def LD2-def jn00-def jn01-def jn11-def*
by (*simp only*: *UNIV-def*)
qed

corollary *cor-example-not-dcr2*:
shows $\exists\ r::(nat\ set)\ rel.$ (
$(\ \forall a\ b\ c.\ (a,b) \in r\hat{}* \wedge (a,c) \in r\hat{}* \longrightarrow (\exists\ d.\ (b,d) \in r\hat{}* \wedge (c,d) \in r\hat{}*)$
)
$\wedge\ |r| \leq o\ cardSuc\ |\{n::nat.\ True\}|$
$\wedge\ (\neg\ (\ \exists r0\ r1.\ ($
$(\ r = (r0 \cup r1)\ )$
$\wedge\ (\forall\ a\ b\ c.\ (a,b) \in r0 \wedge (a,c) \in r0$
$\longrightarrow (\exists\ d.$
$(b,d) \in r0\hat{}=$
$\wedge\ (c,d) \in r0\hat{}=)\ )$
$\wedge\ (\forall\ a\ b\ c.\ (a,b) \in r0 \wedge (a,c) \in r1$
$\longrightarrow (\exists b'\ d.$
$(b,b') \in r1\hat{}= \wedge (b',d) \in r0\hat{}*$

∧ (c,d) ∈ r0^*))
∧ (∀ a b c. (a,b) ∈ r1 ∧ (a,c) ∈ r1
⟶ (∃ b′ b′′ c′ c′′ d.
(b,b′) ∈ r0^* ∧ (b′,b′′) ∈ r1^= ∧ (b′′,d) ∈ r0^*
∧ (c,c′) ∈ r0^* ∧ (c′,c′′) ∈ r1^= ∧ (c′′,d) ∈ r0^*))))
))
proof −
have *cardSuc* |{*x*::*nat*. *True*}| ≤*o* |{*x*::*nat set*. *True*}| **by** *force*
then show *?thesis* **using** *thm-example-not-dcr2* **by** *blast*
qed

end

1.4 DCR implies LD Property

theory *Main-Result-DCR-N1*
imports
DCR3-Method
Decreasing−Diagrams.Decreasing-Diagrams
begin

1.4.1 Auxiliary definitions

definition *map-seq-labels* :: (′*b* ⇒ ′*c*) ⇒ (′*a*,′*b*) *seq* ⇒ (′*a*,′*c*) *seq*
where
map-seq-labels f σ = (*fst σ*, *map* (λ(α,*a*). (*f* α, *a*)) (*snd σ*))

fun *map-diag-labels* :: (′*b* ⇒ ′*c*) ⇒
(′*a*,′*b*) *seq* × (′*a*,′*b*) *seq* × (′*a*,′*b*) *seq* × (′*a*,′*b*) *seq* ⇒
(′*a*,′*c*) *seq* × (′*a*,′*c*) *seq* × (′*a*,′*c*) *seq* × (′*a*,′*c*) *seq*
where
map-diag-labels f (τ,σ,σ′,τ′) = ((*map-seq-labels f* τ), (*map-seq-labels f* σ), (*map-seq-labels f* σ′), (*map-seq-labels f* τ′))

fun *f-to-ls* :: (*nat* ⇒ ′*a*) ⇒ *nat* ⇒ ′*a list*
where
f-to-ls f 0 = []
| *f-to-ls f* (*Suc n*) = (*f-to-ls f n*) @ [(*f n*)]

1.4.2 Auxiliary lemmas

lemma *lem-ftofs-len*: *length* (*f-to-ls f n*) = *n* **by** (*induct n*, *simp+*)

lemma *lem-irr-inj-im-irr*:
fixes *r*::′*a rel* **and** *r*′::′*b rel* **and** *f*::′*a* ⇒ ′*b*
assumes *irrefl r* **and** *inj-on f* (*Field r*)
and *r*′ = {(*a*′,*b*′). ∃ *a b*. *a*′ = *f a* ∧ *b*′ = *f b* ∧ (*a*,*b*) ∈ *r*}
shows *irrefl r*′
using *assms* **unfolding** *inj-on-def Field-def irrefl-def* **by** *blast*

lemma *lem-tr-inj-im-tr*:
fixes r::$'a$ *rel* **and** r'::$'b$ *rel* **and** f::$'a \Rightarrow 'b$
assumes *trans r* **and** *inj-on f* (*Field r*)
  and $r' = \{(a',b').\ \exists\ a\ b.\ a' = f\ a \wedge b' = f\ b \wedge (a,b) \in r\}$
shows *trans* r'
  using *assms* **unfolding** *inj-on-def Field-def trans-def* **by** *blast*

lemma *lem-lpeak-expr*: *local-peak lrs* $(\tau, \sigma) = (\exists\ a\ b\ c\ \alpha\ \beta.\ (a,\alpha,b) \in lrs \wedge (a,\beta,c) \in lrs \wedge \tau = (a,[(\alpha,b)]) \wedge \sigma = (a,[(\beta,c)]))$
proof
  assume *local-peak lrs* (τ, σ)
  then show $\exists\ a\ b\ c\ \alpha\ \beta.\ (a,\alpha,b) \in lrs \wedge (a,\beta,c) \in lrs \wedge \tau = (a,[(\alpha,b)]) \wedge \sigma = (a,[(\beta,c)])$
    unfolding *Decreasing-Diagrams.local-peak-def Decreasing-Diagrams.peak-def*
    apply(*cases* τ, *cases* σ, *simp*)
    using *Decreasing-Diagrams.seq-tail1*(*2*)
    by (*metis* (*no-types*, *lifting*) *Suc-length-conv length-0-conv prod.collapse*)
next
  assume $\exists\ a\ b\ c\ \alpha\ \beta.\ (a,\alpha,b) \in lrs \wedge (a,\beta,c) \in lrs \wedge \tau = (a,[(\alpha,b)]) \wedge \sigma = (a,[(\beta,c)])$
  then obtain $a\ b\ c\ \alpha\ \beta$ **where** $(a,\alpha,b) \in lrs \wedge (a,\beta,c) \in lrs \wedge \tau = (a,[(\alpha,b)]) \wedge \sigma = (a,[(\beta,c)])$ **by** *blast*
  then show *local-peak lrs* (τ, σ)
    unfolding *Decreasing-Diagrams.local-peak-def Decreasing-Diagrams.peak-def*
    by (*simp add*: *Decreasing-Diagrams.seq.intros*)
qed

lemma *lem-map-seq*:
fixes *lrs*::$('a,'b)$ *lars* **and** f::$'b \Rightarrow 'c$ **and** *lrs'*::$('a,'c)$ *lars* **and** σ::$('a,'b)$ *seq*
assumes *a1*: $lrs' = \{(a,l',b).\ \exists l.\ l' = f\ l \wedge (a,l,b) \in lrs\ \}$
  and *a2*: $\sigma \in$ *Decreasing-Diagrams.seq lrs*
shows (*map-seq-labels f* σ) $\in$ *Decreasing-Diagrams.seq lrs'*
proof −
  have $\forall\ s\ a.\ (a,s) \in$ *Decreasing-Diagrams.seq lrs* $\longrightarrow$ (*map-seq-labels f* (a,s)) $\in$ *Decreasing-Diagrams.seq lrs'*
  proof
    fix *s*
    show $\forall\ a.\ (a,s) \in$ *Decreasing-Diagrams.seq lrs* $\longrightarrow$ (*map-seq-labels f* (a,s)) $\in$ *Decreasing-Diagrams.seq lrs'*
    proof (*induct s*)
      show $\forall a.\ (a, []) \in$ *Decreasing-Diagrams.seq lrs* $\longrightarrow$ *map-seq-labels f* $(a, []) \in$ *Decreasing-Diagrams.seq lrs'*
        unfolding *map-seq-labels-def* **by** (*simp add*: *seq.intros*(*1*))
    next
      fix *p s1*
      assume *d1*: $\forall b.\ (b,\ s1) \in$ *Decreasing-Diagrams.seq lrs* $\longrightarrow$ *map-seq-labels f* $(b,\ s1) \in$ *Decreasing-Diagrams.seq lrs'*
      show $\forall b.\ (b,\ p\ \#\ s1) \in$ *Decreasing-Diagrams.seq lrs* $\longrightarrow$ *map-seq-labels f* $(b,\ p\ \#\ s1) \in$ *Decreasing-Diagrams.seq lrs'*

proof (*intro allI impI*)
fix *b*
assume *e1*: (*b*, *p* # *s1*) ∈ *Decreasing-Diagrams.seq lrs*
moreover obtain *l b′* **where** *e2*: *p* = (*l*, *b′*) **by** *force*
ultimately have *e3*: (*b*,*l*,*b′*) ∈ *lrs* ∧ (*b′*,*s1*) ∈ *Decreasing-Diagrams.seq lrs*
by (*metis Decreasing-Diagrams.seq-tail1*(*1*) *Decreasing-Diagrams.seq-tail1*(*2*) *prod.collapse snd-conv*)
then have (*b*,*f l*,*b′*) ∈ *lrs′* **using** *a1* **by** *blast*
moreover have *map-seq-labels f* (*b′*, *s1*) ∈ *Decreasing-Diagrams.seq lrs′* **using** *d1 e3* **by** *blast*
ultimately show *map-seq-labels f* (*b*, *p* # *s1*) ∈ *Decreasing-Diagrams.seq lrs′*
using *e2* **unfolding** *map-seq-labels-def* **by** (*simp add*: *seq.intros*(*2*))
qed
qed
qed
moreover obtain *a s* **where** *σ* = (*a*,*s*) **by** *force*
ultimately show (*map-seq-labels f σ*) ∈ *Decreasing-Diagrams.seq lrs′* **using** *a2* **by** *blast*
qed

lemma *lem-map-diag*:
fixes *lrs*::(*′a*,*′b*) *lars* **and** *f*::*′b* ⇒ *′c* **and** *lrs′*::(*′a*,*′c*) *lars*
and *d*::(*′a*,*′b*) *seq* × (*′a*,*′b*) *seq* × (*′a*,*′b*) *seq* × (*′a*,*′b*) *seq*
assumes *a1*: *lrs′* = {(*a*,*l′*,*b*). ∃ *l*. *l′* = *f l* ∧ (*a*,*l*,*b*) ∈ *lrs* }
and *a2*: *diagram lrs d*
shows *diagram lrs′* (*map-diag-labels f d*)
proof −
obtain *τ σ σ′ τ′* **where** *b1*: *d* = (*τ*, *σ*, *σ′*, *τ′*) **using** *prod-cases4* **by** *blast*
moreover obtain *τ1 σ1 σ1′ τ1′* **where** *b2*: *τ1* = (*map-seq-labels f τ*) ∧ *σ1* = (*map-seq-labels f σ*)
∧ (*σ1′* = *map-seq-labels f σ′*) ∧ (*τ1′* = *map-seq-labels f τ′*)
by *blast*
ultimately have *b3*: (*map-diag-labels f d*) = (*τ1*, *σ1*, *σ1′*, *τ1′*) **by** *simp*
have *b4*: *fst σ* = *fst τ* ∧ *lst σ* = *fst τ′* ∧ *lst τ* = *fst σ′* ∧ *lst σ′* = *lst τ′*
using *b1 a2* **unfolding** *Decreasing-Diagrams.diagram-def* **by** *simp*
have *b5*: *σ1* ∈ *Decreasing-Diagrams.seq lrs′* ∧ *τ1* ∈ *Decreasing-Diagrams.seq lrs′*

∧ *σ1′* ∈ *Decreasing-Diagrams.seq lrs′* ∧ *τ1′* ∈ *Decreasing-Diagrams.seq lrs′*

using *a1 a2 b1 b2 lem-map-seq*[*of lrs′ f*] **by** (*simp add*: *Decreasing-Diagrams.diagram-def*)
moreover have *fst σ1* = *fst τ1* **using** *b2 b4* **unfolding** *map-seq-labels-def* **by** *simp*
moreover have *lst σ1* = *fst τ1′* ∧ *lst τ1* = *fst σ1′* **using** *b4*
by (*simp add*: *b2 map-seq-labels-def lst-def*, *metis* (*no-types*, *lifting*) *case-prod-beta last-map snd-conv*)
moreover have *lst σ1′* = *lst τ1′* **using** *b4*
by (*simp add*: *b2 map-seq-labels-def lst-def*, *metis* (*no-types*, *lifting*) *case-prod-beta last-map snd-conv*)

ultimately show *diagram lrs'* (*map-diag-labels f d*) **using** *b3 b5* **unfolding** *Decreasing-Diagrams.diagram-def* **by** *simp*
qed

lemma *lem-map-D-loc*:
fixes *cmp cmp' s1 s2 s3 s4 f*
assumes *a1*: *Decreasing-Diagrams.D cmp s1 s2 s3 s4*
and *a2*: *trans cmp* **and** *a3*: *irrefl cmp* **and** *a4*: *inj-on f* (*Field cmp*)
and *a5*: *cmp'* = {(*a'*,*b'*). ∃ *a b*. *a'* = *f a* ∧ *b'* = *f b* ∧ (*a*,*b*) ∈ *cmp*}
and *a6*: *length s1* = *1* **and** *a7*: *length s2* = *1*
shows *Decreasing-Diagrams.D cmp'* (*map f s1*) (*map f s2*) (*map f s3*) (*map f s4*)
proof −
obtain *α* **where** *b1*: *s2* = [*α*] **using** *a7* **by** (*metis One-nat-def Suc-length-conv length-0-conv*)
moreover obtain *β* **where** *b2*: *s1* = [*β*] **using** *a6* **by** (*metis One-nat-def Suc-length-conv length-0-conv*)
ultimately have *b3*: *Decreasing-Diagrams.D cmp* [*β*] [*α*] *s3 s4* **using** *a1* **by** *blast*
then obtain *σ1 σ2 σ3 τ1 τ2 τ3* **where** *b4*: *s3* = *σ1*@*σ2*@*σ3* **and** *b5*: *s4* = *τ1*@*τ2*@*τ3* **and** *b6*: *LD' cmp β α σ1 σ2 σ3 τ1 τ2 τ3*
using *Decreasing-Diagrams.proposition3-4-inv*[*of cmp β α s3 s4*] *a2 a3* **by** *blast*
obtain *σ1' σ2' σ3'* **where** *b7*: *σ1'* = *map f σ1* ∧ *σ2'* = *map f σ2* ∧ *σ3'* = *map f σ3* **by** *blast*
obtain *τ1' τ2' τ3'* **where** *b8*: *τ1'* = *map f τ1* ∧ *τ2'* = *map f τ2* ∧ *τ3'* = *map f τ3* **by** *blast*
obtain *s3' s4'* **where** *b9*: *s3'* = *map f s3* **and** *b10*: *s4'* = *map f s4* **by** *blast*
have *trans cmp'* **using** *a2 a4 a5 lem-tr-inj-im-tr* **by** *blast*
moreover have *irrefl cmp'* **using** *a3 a4 a5 lem-irr-inj-im-irr* **by** *blast*
moreover have *s3'* = *σ1'*@*σ2'*@*σ3'* **using** *b4 b7 b9* **by** *simp*
moreover have *s4'* = *τ1'*@*τ2'*@*τ3'* **using** *b5 b8 b10* **by** *simp*
moreover have *LD' cmp'* (*f β*) (*f α*) *σ1' σ2' σ3' τ1' τ2' τ3'*
proof −
have *c1*: *LD-1' cmp β α σ1 σ2 σ3* **and** *c2*: *LD-1' cmp α β τ1 τ2 τ3*
using *b6* **unfolding** *Decreasing-Diagrams.LD'-def* **by** *blast+*
have *LD-1' cmp'* (*f β*) (*f α*) *σ1' σ2' σ3'*
using *c1* **unfolding** *Decreasing-Diagrams.LD-1'-def Decreasing-Diagrams.ds-def* **by** (*simp add*: *a5 b7*, *blast*)
moreover have *LD-1' cmp'* (*f α*) (*f β*) *τ1' τ2' τ3'*
using *c2* **unfolding** *Decreasing-Diagrams.LD-1'-def Decreasing-Diagrams.ds-def* **by** (*simp add*: *a5 b8*, *blast*)
ultimately show *LD' cmp'* (*f β*) (*f α*) *σ1' σ2' σ3' τ1' τ2' τ3'* **unfolding** *Decreasing-Diagrams.LD'-def* **by** *blast*
qed
ultimately have *Decreasing-Diagrams.D cmp'* [*f β*] [*f α*] *s3' s4'* **using** *Decreasing-Diagrams.proposition3-4*[*of cmp'*] **by** *blast*
moreover have (*map f s1*) = [*f β*] ∧ (*map f s2*) = [*f α*] **using** *b1 b2* **by** *simp*
ultimately show *Decreasing-Diagrams.D cmp'* (*map f s1*) (*map f s2*) (*map f s3*) (*map f s4*) **using** *b9 b10* **by** *simp*
qed

lemma *lem-map-DD-loc*:
fixes *lrs*::('*a*,'*b*) *lars* **and** *cmp*::'*b rel* **and** *lrs*'::('*a*,'*c*) *lars* **and** *cmp*'::'*c rel* **and** *f*::'*b* ⇒ '*c*
assumes *a1*: *trans cmp* **and** *a2*: *irrefl cmp* **and** *a3*: *inj-on f* (*Field cmp*)
  and *a4*: *cmp*' = {(*a*',*b*'). ∃ *a b*. *a*' = *f a* ∧ *b*' = *f b* ∧ (*a*,*b*) ∈ *cmp*}
  and *a5*: *lrs*' = {(*a*,*l*',*b*). ∃ *l*. *l*' = *f l* ∧ (*a*,*l*,*b*) ∈ *lrs* }
  and *a6*: *length* (*snd* (*fst d*)) = *1* **and** *a7*: *length* (*snd* (*fst* (*snd d*))) = *1*
  and *a8*: *DD lrs cmp d*
shows *DD lrs*' *cmp*' (*map-diag-labels f d*)
proof −
 have *diagram lrs*' (*map-diag-labels f d*) **using** *a4 a5 a8 lem-map-diag* **unfolding** *Decreasing-Diagrams.DD-def* **by** *blast*
 moreover have *D2 cmp*' (*map-diag-labels f d*)
 proof −
  obtain *τ σ σ*' *τ*' **where** *c1*: *d* = (*τ*,*σ*,*σ*',*τ*') **by** (*metis prod-cases3*)
  obtain *s1 s2 s3 s4* **where** *c2*: *s1* = *labels τ* ∧ *s2* = *labels σ* ∧ *s3* = *labels σ*' ∧ *s4* = *labels τ*' **by** *blast*
  have *Decreasing-Diagrams.D cmp s1 s2 s3 s4*
  using *a8 c1 c2* **unfolding** *Decreasing-Diagrams.DD-def Decreasing-Diagrams.D2-def* **by** *simp*
  moreover have *length s1* = *1* ∧ *length s2* = *1* **using** *a6 a7 c1 c2* **unfolding** *labels-def* **by** *simp*
  ultimately have *Decreasing-Diagrams.D cmp*' (*map f s1*) (*map f s2*) (*map f s3*) (*map f s4*)
   using *a1 a2 a3 a4 lem-map-D-loc* **by** *blast*
  moreover have *labels* (*map-seq-labels f τ*) = (*map f s1*)
     and *labels* (*map-seq-labels f σ*) = (*map f s2*)
     and *labels* (*map-seq-labels f σ*') = (*map f s3*)
     and *labels* (*map-seq-labels f τ*') = (*map f s4*)
   using *c2* **unfolding** *map-seq-labels-def Decreasing-Diagrams.labels-def* **by** *force+*
  ultimately have *D2 cmp*' ((*map-seq-labels f τ*), (*map-seq-labels f σ*), (*map-seq-labels f σ*'), (*map-seq-labels f τ*'))
   unfolding *Decreasing-Diagrams.D2-def* **by** *simp*
   then show *D2 cmp*' (*map-diag-labels f d*) **using** *c1* **unfolding** *Decreasing-Diagrams.D2-def* **by** *simp*
 qed
 ultimately show *DD lrs*' *cmp*' (*map-diag-labels f d*) **unfolding** *Decreasing-Diagrams.DD-def* **by** *blast*
qed

lemma *lem-ddseq-mon*: *lrs1* ⊆ *lrs2* ⟹ *Decreasing-Diagrams.seq lrs1* ⊆ *Decreasing-Diagrams.seq lrs2*
proof −
 assume *a1*: *lrs1* ⊆ *lrs2*
 show *Decreasing-Diagrams.seq lrs1* ⊆ *Decreasing-Diagrams.seq lrs2*
 proof
  fix *a s*

assume *b1*: (*a*,*s*) ∈ *Decreasing-Diagrams.seq lrs1*
show (*a*,*s*) ∈ *Decreasing-Diagrams.seq lrs2*
by (*rule Decreasing-Diagrams.seq.induct*[*of - - lrs1*],
simp only: *b1*, *simp only*: *seq.intros*(*1*), *meson a1 contra-subsetD seq.intros*(*2*))
qed
qed

lemma *lem-dd-D-mon*:
fixes *cmp1 cmp2* α β *s1 s2*
assumes *a1*: *trans cmp1* ∧ *irrefl cmp1* **and** *a2*: *trans cmp2* ∧ *irrefl cmp2* **and** *a3*: *cmp1* ⊆ *cmp2*
and *a4*: *Decreasing-Diagrams.D cmp1* [α] [β] *s1 s2*
shows *Decreasing-Diagrams.D cmp2* [α] [β] *s1 s2*
proof −
obtain $\sigma 1$ $\sigma 2$ $\sigma 3$ $\tau 1$ $\tau 2$ $\tau 3$
where *b1*: *s1* = $\sigma 1$@$\sigma 2$@$\sigma 3$ ∧ *s2* = $\tau 1$@$\tau 2$@$\tau 3$ **and** *b2*: *LD′ cmp1* α β $\sigma 1$ $\sigma 2$ $\sigma 3$ $\tau 1$ $\tau 2$ $\tau 3$
using *a1 a4 Decreasing-Diagrams.proposition3-4-inv*[*of cmp1* α β *s1 s2*] **by** *blast*
then have *b3*: *LD-1′ cmp1* α β $\sigma 1$ $\sigma 2$ $\sigma 3$ **and** *b4*: *LD-1′ cmp1* β α $\tau 1$ $\tau 2$ $\tau 3$
unfolding *Decreasing-Diagrams.LD′-def* **by** *blast+*
have *LD-1′ cmp2* α β $\sigma 1$ $\sigma 2$ $\sigma 3$
using *a3 b3* **unfolding** *Decreasing-Diagrams.LD-1′-def Decreasing-Diagrams.ds-def* **by** *blast*
moreover have *LD-1′ cmp2* β α $\tau 1$ $\tau 2$ $\tau 3$
using *a3 b4* **unfolding** *Decreasing-Diagrams.LD-1′-def Decreasing-Diagrams.ds-def* **by** *blast*
ultimately show *Decreasing-Diagrams.D cmp2* [α] [β] *s1 s2*
using *Decreasing-Diagrams.proposition3-4*[*of cmp2* α β] **by** (*simp add*: *a2 b1 LD′-def*)
qed

1.4.3 Result

The next lemma has the following meaning: every ARS in the finite DCR hierarchy has the LD property.

lemma *lem-dcr-to-ld*:
fixes *n*::*nat* **and** *r*::*′U rel*
assumes *DCR n r*
shows *LD* (*UNIV*::*nat set*) *r*
proof −
obtain *g*::*nat* ⇒ *′U rel* **where**
b1: *DCR-generating g* **and** *b3*: $r = \bigcup \{ r'.\ \exists\ \alpha'.\ \alpha' < n \wedge r' = g\ \alpha' \}$
using *assms* **unfolding** *DCR-def* **by** *blast*
obtain *lrs*::(*′U*, *nat*) *lars* **where** *b4*: $lrs = \{(a,\alpha',b).\ \alpha' < n \wedge (a,b) \in g\ \alpha'\}$ **by** *blast*
obtain *cmp*::*nat rel* **where** *b5*: $cmp = \{(\alpha, \beta).\ \alpha < \beta \}$ **by** *blast*
have *r* = *unlabel lrs* **using** *b3 b4* **unfolding** *unlabel-def* **by** *blast*
moreover have *b6*: *trans cmp* **using** *b5* **unfolding** *trans-def* **by** *force*

moreover have *b7*: *wf cmp*
proof −
have *cmp* = ({(*x*::*nat*, *y*::*nat*). *x* < *y*})
unfolding *b5 lex-prod-def* **by** *fastforce*
moreover have *wf* {(*x*::*nat*, *y*::*nat*). *x* < *y*} **using** *wf-less* **by** *blast*
ultimately show *?thesis* **using** *wf-lex-prod* **by** *blast*
qed
moreover have ∀ *P*. *local-peak lrs P* ⟶ (∃ *σ′ τ′*. *DD lrs cmp* (*fst P*,*snd P*,*σ′*,*τ′*))
proof (*intro allI impI*)
fix *P*
assume *c1*: *local-peak lrs P*
moreover obtain *τ σ* **where** *c2*: *P* = (*τ*, *σ*) **using** *surjective-pairing* **by** *blast*
ultimately obtain *a b c α β*
where *c3*: (*a*,*α*,*b*) ∈ *lrs* ∧ (*a*,*β*,*c*) ∈ *lrs*
and *c4*: *σ* = (*a*,[(*α*,*b*)]) ∧ *τ* = (*a*,[(*β*,*c*)]) **using** *lem-lpeak-expr*[*of lrs*] **by** *blast*
then have *c5*: *α* < *n* ∧ *β* < *n* **and** *c6*: (*a*,*b*) ∈ (*g α*) ∧ (*a*,*c*) ∈ (*g β*) **using** *b4* **by** *blast*+
obtain *b′ b″ c′ c″ d* **where**
c7: (*b*,*b′*) ∈ (𝔏*1 g α*)^* ∧ (*b′*,*b″*) ∈ (*g β*)^= ∧ (*b″*,*d*) ∈ (𝔏*v g α β*)^*
and *c8*: (*c*,*c′*) ∈ (𝔏*1 g β*)^* ∧ (*c′*,*c″*) ∈ (*g α*)^= ∧ (*c″*,*d*) ∈ (𝔏*v g β α*)^*
using *b1 c6* **unfolding** *DCR-generating-def* 𝔇*-def* **by** (*metis* (*no-types*, *lifting*) *mem-Collect-eq old.prod.case*)
obtain *pn1* **where** (*b*,*b′*) ∈ (𝔏*1 g α*)^^*pn1* **using** *c7* **by** *fastforce*
then obtain *ph1* **where** *pc9*: *ph1 0* = *b* ∧ *ph1 pn1* = *b′* **and** ∀ *i*::*nat*. *i* < *pn1* ⟶ (*ph1 i*, *ph1* (*Suc i*)) ∈ (𝔏*1 g α*)
using *relpow-fun-conv* **by** *metis*
then have ∀ *i*::*nat*. *i*<*pn1* ⟶ (∃ *α′*. *α′* < *α* ∧ (*ph1 i*, *ph1* (*Suc i*)) ∈ *g α′*) **unfolding** 𝔏*1-def* **by** *blast*
then obtain *pαi1*::*nat* ⇒ *nat*
where *pc10*: ∀ *i*::*nat*. *i*<*pn1* ⟶ (*pαi1 i*) < *α* ∧ (*ph1 i*, *ph1* (*Suc i*)) ∈ *g* (*pαi1 i*) **by** *metis*
let *?pf1* = *λi*. (*pαi1 i*, *ph1* (*Suc i*))
obtain *pls1* **where** *pc11*: *pls1* = (*f-to-ls ?pf1 pn1*) **by** *blast*
obtain *n1* **where** (*b″*,*d*) ∈ (𝔏*v g α β*)^^*n1* **using** *c7* **by** *fastforce*
then obtain *h1* **where** *c9*: *h1 0* = *b″* ∧ *h1 n1* = *d* **and** ∀ *i*::*nat*. *i* < *n1* ⟶ (*h1 i*, *h1* (*Suc i*)) ∈ (𝔏*v g α β*)
using *relpow-fun-conv* **by** *metis*
then have ∀ *i*::*nat*. *i* < *n1* ⟶ (∃ *α′*. (*α′* < *α* ∨ *α′* < *β*) ∧ (*h1 i*, *h1* (*Suc i*)) ∈ *g α′*) **unfolding** 𝔏*v-def* **by** *blast*
then obtain *αi1*::*nat* ⇒ *nat*
where *c10*: ∀ *i*::*nat*. *i*<*n1* ⟶ ((*αi1 i*) < *α* ∨ (*αi1 i*) < *β*) ∧ (*h1 i*, *h1* (*Suc i*)) ∈ *g* (*αi1 i*) **by** *metis*
let *?f1* = *λi*. (*αi1 i*, *h1* (*Suc i*))
obtain *ls1* **where** *c11*: *ls1* = (*f-to-ls ?f1 n1*) **by** *blast*
obtain *τ″* **where** *qc12*: *τ″* = (*if b′* = *b″ then* (*b″*, *ls1*) *else* (*b′*, (*β*, *b″*) # *ls1*)) **by** *blast*
obtain *τ′* **where** *c12*: *τ′* = (*b*, *pls1* @ (*snd τ″*)) **by** *blast*

obtain *pn2* **where** $(c,c') \in (\mathfrak{L}1\ g\ \beta)$^^*pn2* **using** *c8* **by** *fastforce*
then obtain *ph2* **where** *pc13*: $ph2\ 0 = c \wedge ph2\ pn2 = c'$ **and** $\forall\ i{::}nat.\ i < pn2 \longrightarrow (ph2\ i,\ ph2\ (Suc\ i)) \in (\mathfrak{L}1\ g\ \beta)$
using *relpow-fun-conv* **by** *metis*
then have $\forall\ i{::}nat.\ i{<}pn2 \longrightarrow (\exists\ \alpha'.\ \alpha' < \beta \wedge (ph2\ i,\ ph2\ (Suc\ i)) \in g\ \alpha')$ **unfolding** *𝔏1-def* **by** *blast*
then obtain $p\alpha i2{::}nat \Rightarrow nat$
where *pc14*: $\forall\ i{::}nat.\ i{<}pn2 \longrightarrow (p\alpha i2\ i) < \beta \wedge (ph2\ i,\ ph2\ (Suc\ i)) \in g\ (p\alpha i2\ i)$ **by** *metis*
let $?pf2 = \lambda i.\ (\ p\alpha i2\ i,\ ph2\ (Suc\ i)\ )$
obtain *pls2* **where** *pc15*: $pls2 = (f\text{-}to\text{-}ls\ ?pf2\ pn2)$ **by** *blast*
have $\mathfrak{L}v\ g\ \beta\ \alpha = \mathfrak{L}v\ g\ \alpha\ \beta$ **unfolding** *𝔏v-def* **by** *blast*
then have $(c'',d) \in (\mathfrak{L}v\ g\ \alpha\ \beta)$^* **using** *c8* **by** *simp*
then obtain *n2* **where** $(c'',d) \in (\mathfrak{L}v\ g\ \alpha\ \beta)$^^*n2* **using** *c8* **by** *fastforce*
then obtain *h2* **where** *c13*: $h2\ 0 = c'' \wedge h2\ n2 = d$ **and** $\forall\ i{::}nat.\ i < n2 \longrightarrow (h2\ i,\ h2\ (Suc\ i)) \in (\mathfrak{L}v\ g\ \alpha\ \beta)$
using *relpow-fun-conv* **by** *metis*
then have $\forall\ i{::}nat.\ i{<}n2 \longrightarrow (\exists\ \alpha'.\ (\alpha' < \alpha \vee \alpha' < \beta) \wedge (h2\ i,\ h2\ (Suc\ i)) \in g\ \alpha')$ **unfolding** *𝔏v-def* **by** *blast*
then obtain $\alpha i2{::}nat \Rightarrow nat$
where *c14*: $\forall\ i{::}nat.\ i{<}n2 \longrightarrow ((\alpha i2\ i) < \alpha \vee (\alpha i2\ i) < \beta) \wedge (h2\ i,\ h2\ (Suc\ i)) \in g\ (\alpha i2\ i)$ **by** *metis*
let $?f2 = \lambda i.\ (\ \alpha i2\ i,\ h2\ (Suc\ i)\ )$
obtain *ls2* **where** *c15*: $ls2 = (f\text{-}to\text{-}ls\ ?f2\ n2)$ **by** *blast*
obtain σ'' **where** *qc16*: $\sigma'' = (if\ c' = c''\ then\ (c'',\ ls2)\ else\ (c',\ (\alpha,\ c'')\ \#\ ls2))$ **by** *blast*
obtain σ' **where** *c16*: $\sigma' = (c,\ pls2\ @\ (snd\ \sigma''))$ **by** *blast*
have $DD\ lrs\ cmp\ (\tau,\ \sigma,\ \sigma',\ \tau')$
proof −
have *d1'*: $\forall\ k.\ k < pn1 \longrightarrow (ph1\ k,\ p\alpha i1\ k,\ ph1\ (Suc\ k)) \in lrs$
proof (*intro allI impI*)
fix *k*
assume $k < pn1$
moreover then have $(ph1\ k,\ ph1\ (Suc\ k)) \in g\ (p\alpha i1\ k) \wedge (p\alpha i1\ k < n)$
using *c5 pc10* **by** *force*
ultimately show $(ph1\ k,\ p\alpha i1\ k,\ ph1\ (Suc\ k)) \in lrs$ **using** *b4* **by** *blast*
qed
have *d1*: $\forall\ k.\ k < n1 \longrightarrow (h1\ k,\ \alpha i1\ k,\ h1\ (Suc\ k)) \in lrs$
proof (*intro allI impI*)
fix *k*
assume $k < n1$
moreover then have $(h1\ k,\ h1\ (Suc\ k)) \in g\ (\alpha i1\ k) \wedge \alpha i1\ k < n$
using *c5 c10* **by** *force*
ultimately show $(h1\ k,\ \alpha i1\ k,\ h1\ (Suc\ k)) \in lrs$ **using** *b4* **by** *blast*
qed
have *d2'*: $\forall\ k.\ k < pn2 \longrightarrow (ph2\ k,\ p\alpha i2\ k,\ ph2\ (Suc\ k)) \in lrs$
proof (*intro allI impI*)
fix *k*
assume $k < pn2$

moreover then have $(ph2\ k,\ ph2\ (Suc\ k)) \in g\ (p\alpha i2\ k) \wedge p\alpha i2\ k < n$
using *c5 pc14* **by** *force*
ultimately show $(ph2\ k,\ p\alpha i2\ k,\ ph2\ (Suc\ k)) \in lrs$ **using** *b4* **by** *blast*
qed
have *d2*: $\forall\ k.\ k < n2 \longrightarrow (h2\ k,\ \alpha i2\ k,\ h2\ (Suc\ k)) \in lrs$
proof (*intro allI impI*)
fix *k*
assume $k < n2$
moreover then have $(h2\ k,\ h2\ (Suc\ k)) \in g\ (\alpha i2\ k) \wedge \alpha i2\ k < n$
using *c5 c14* **by** *force*
ultimately show $(h2\ k,\ \alpha i2\ k,\ h2\ (Suc\ k)) \in lrs$ **using** *b4* **by** *blast*
qed
have *d3*: $\tau'' \in$ *Decreasing-Diagrams.seq lrs*
proof −
have $\forall\ k.\ k \leq n1 \longrightarrow (b'',\ (f\text{-}to\text{-}ls\ ?f1\ k)) \in$ *Decreasing-Diagrams.seq lrs*
proof
fix *k0*
show $k0 \leq n1 \longrightarrow (b'',\ (f\text{-}to\text{-}ls\ ?f1\ k0)) \in$ *Decreasing-Diagrams.seq lrs*
proof (*induct k0*)
show $0 \leq n1 \longrightarrow (b'',\ f\text{-}to\text{-}ls\ ?f1\ 0) \in$ *Decreasing-Diagrams.seq lrs*
using *Decreasing-Diagrams.seq.intros*(*1*)[*of - lrs*] **by** *simp*
next
fix *k*
assume *g1*: $k \leq n1 \longrightarrow (b'',\ f\text{-}to\text{-}ls\ ?f1\ k) \in$ *Decreasing-Diagrams.seq lrs*
show $Suc\ k \leq n1 \longrightarrow (b'',\ f\text{-}to\text{-}ls\ ?f1\ (Suc\ k)) \in$ *Decreasing-Diagrams.seq lrs*
proof
assume *h1*: $Suc\ k \leq n1$
then have *h2*: $(b'',\ f\text{-}to\text{-}ls\ ?f1\ k) \in$ *Decreasing-Diagrams.seq lrs* **using** *g1* **by** *simp*
obtain *s* **where** *h3*: $s = (h1\ k,\ [(\alpha i1\ k,\ h1\ (Suc\ k))])$ **by** *blast*
then have $s \in$ *Decreasing-Diagrams.seq lrs*
using *h1 d1 Decreasing-Diagrams.seq.intros*(*2*)[*of h1 k* α*i1 k*] *Decreasing-Diagrams.seq.intros*(*1*)[*of - lrs*] **by** *simp*
moreover have $lst\ (b'',\ f\text{-}to\text{-}ls\ ?f1\ k) = fst\ s$
using *c9 h3* **unfolding** *lst-def* **by** (*cases k*, *simp+*)
ultimately show $(b'',\ f\text{-}to\text{-}ls\ ?f1\ (Suc\ k)) \in$ *Decreasing-Diagrams.seq lrs*
using *h2 h3 Decreasing-Diagrams.seq-concat-helper*[*of* b'' *f-to-ls ?f1 k lrs s*] **by** *simp*
qed
qed
qed
then have $(b'',\ ls1) \in$ *Decreasing-Diagrams.seq lrs* **using** *c11* **by** *blast*
moreover then have $b' \neq b'' \longrightarrow (b',\ (\beta,\ b'')\ \#\ ls1) \in$ *Decreasing-Diagrams.seq lrs*
using *b4 c5 c7 Decreasing-Diagrams.seq.intros*(*2*)[*of* $b'\ \beta\ b''$] **by** *fastforce*
ultimately show $\tau'' \in$ *Decreasing-Diagrams.seq lrs* **using** *qc12* **by** *simp*
qed

```
have d4: σ'' ∈ Decreasing-Diagrams.seq lrs
proof −
  have ∀ k. k ≤ n2 ⟶ (c'', (f-to-ls ?f2 k)) ∈ Decreasing-Diagrams.seq lrs
  proof
    fix k0
    show k0 ≤ n2 ⟶ (c'', (f-to-ls ?f2 k0)) ∈ Decreasing-Diagrams.seq lrs
    proof (induct k0)
      show 0 ≤ n2 ⟶ (c'', f-to-ls ?f2 0) ∈ Decreasing-Diagrams.seq lrs
        using Decreasing-Diagrams.seq.intros(1)[of - lrs] by simp
    next
      fix k
     assume g1: k ≤ n2 ⟶ (c'', f-to-ls ?f2 k) ∈ Decreasing-Diagrams.seq lrs
    show Suc k ≤ n2 ⟶ (c'', f-to-ls ?f2 (Suc k)) ∈ Decreasing-Diagrams.seq
lrs
      proof
        assume h1: Suc k ≤ n2
        then have h2: (c'', f-to-ls ?f2 k) ∈ Decreasing-Diagrams.seq lrs using
g1 by simp
        obtain s where h3: s = (h2 k, [(αi2 k, h2 (Suc k))]) by blast
        then have s ∈ Decreasing-Diagrams.seq lrs
                using h1 d2 Decreasing-Diagrams.seq.intros(2)[of h2 k αi2 k]
Decreasing-Diagrams.seq.intros(1)[of - lrs] by simp
        moreover have lst (c'', f-to-ls ?f2 k) = fst s
          using c13 h3 unfolding lst-def by (cases k, simp+)
        ultimately show (c'', f-to-ls ?f2 (Suc k)) ∈ Decreasing-Diagrams.seq
lrs
          using h2 h3 Decreasing-Diagrams.seq-concat-helper[of c'' f-to-ls ?f2 k
lrs s] by simp
      qed
    qed
  qed
  then have (c'', ls2) ∈ Decreasing-Diagrams.seq lrs using c15 by blast
      moreover then have c' ≠ c'' ⟶ (c', (α, c'') # ls2) ∈ Decreas-
ing-Diagrams.seq lrs
    using b4 c5 c8 Decreasing-Diagrams.seq.intros(2)[of c' α c''] by fastforce
  ultimately show σ'' ∈ Decreasing-Diagrams.seq lrs using qc16 by simp
qed
  have σ ∈ Decreasing-Diagrams.seq lrs by (simp add: c3 c4 seq.intros(1)
seq.intros(2))
   moreover have τ ∈ Decreasing-Diagrams.seq lrs by (simp add: c3 c4
seq.intros(1) seq.intros(2))
moreover have d5: σ' ∈ Decreasing-Diagrams.seq lrs ∧ lst σ' = lst σ''
proof −
  have (c, pls2) ∈ Decreasing-Diagrams.seq lrs
  proof −
   have ∀ k. k ≤ pn2 ⟶ (c, (f-to-ls ?pf2 k)) ∈ Decreasing-Diagrams.seq lrs
    proof
      fix k0
     show k0 ≤ pn2 ⟶ (c, (f-to-ls ?pf2 k0)) ∈ Decreasing-Diagrams.seq lrs
```

proof (*induct k0*)
show *0* ≤ *pn2* ⟶ (*c*, *f-to-ls ?pf2 0*) ∈ *Decreasing-Diagrams.seq lrs*
using *Decreasing-Diagrams.seq.intros*(*1*)[*of - lrs*] **by** *simp*
next
fix *k*
assume *g1*: *k* ≤ *pn2* ⟶ (*c*, *f-to-ls ?pf2 k*) ∈ *Decreasing-Diagrams.seq lrs*
show *Suc k* ≤ *pn2* ⟶ (*c*, *f-to-ls ?pf2* (*Suc k*)) ∈ *Decreasing-Diagrams.seq lrs*
proof
assume *h1*: *Suc k* ≤ *pn2*
then have *h2*: (*c*, *f-to-ls ?pf2 k*) ∈ *Decreasing-Diagrams.seq lrs* **using** *g1* **by** *simp*
obtain *s* **where** *h3*: *s* = (*ph2 k*, [(*pαi2 k*, *ph2* (*Suc k*))]) **by** *blast*
then have *s* ∈ *Decreasing-Diagrams.seq lrs*
using *h1 d2′ Decreasing-Diagrams.seq.intros*(*2*)[*of ph2 k pαi2 k*] *Decreasing-Diagrams.seq.intros*(*1*)[*of - lrs*] **by** *simp*
moreover have *lst* (*c*, *f-to-ls ?pf2 k*) = *fst s*
using *pc13 h3* **unfolding** *lst-def* **by** (*cases k*, *simp+*)
ultimately show (*c*, *f-to-ls ?pf2* (*Suc k*)) ∈ *Decreasing-Diagrams.seq lrs*
using *h2 h3 Decreasing-Diagrams.seq-concat-helper*[*of c f-to-ls ?pf2 k lrs s*] **by** *simp*
qed
qed
qed
then show *?thesis* **using** *pc15* **by** *blast*
qed
moreover have *lst* (*c*, *pls2*) = *fst σ′′*
proof −
have *lst* (*c*, *pls2*) = *c′* **using** *pc13 pc15* **unfolding** *lst-def* **by** (*cases pn2*, *simp+*)
then show *?thesis* **unfolding** *qc16* **by** *simp*
qed
ultimately show *?thesis* **using** *d4*
unfolding *c16* **using** *Decreasing-Diagrams.seq-concat-helper*[*of c pls2 lrs σ′′*] **by** *blast*
qed
moreover have *d6*: *τ′* ∈ *Decreasing-Diagrams.seq lrs* ∧ *lst τ′* = *lst τ′′*
proof −
have (*b*, *pls1*) ∈ *Decreasing-Diagrams.seq lrs*
proof −
have ∀ *k*. *k* ≤ *pn1* ⟶ (*b*, (*f-to-ls ?pf1 k*)) ∈ *Decreasing-Diagrams.seq lrs*
proof
fix *k0*
show *k0* ≤ *pn1* ⟶ (*b*, (*f-to-ls ?pf1 k0*)) ∈ *Decreasing-Diagrams.seq lrs*
proof (*induct k0*)
show *0* ≤ *pn1* ⟶ (*b*, *f-to-ls ?pf1 0*) ∈ *Decreasing-Diagrams.seq lrs*
using *Decreasing-Diagrams.seq.intros*(*1*)[*of - lrs*] **by** *simp*

next
fix *k*
assume *g1*: $k \le pn1 \longrightarrow (b, \textit{f-to-ls ?pf1 } k) \in \textit{Decreasing-Diagrams.seq}$ *lrs*
show $Suc\ k \le pn1 \longrightarrow (b, \textit{f-to-ls ?pf1 } (Suc\ k)) \in \textit{Decreasing-Diagrams.seq}$ *lrs*
proof
assume *h1*: $Suc\ k \le pn1$
then have *h2*: $(b, \textit{f-to-ls ?pf1 } k) \in \textit{Decreasing-Diagrams.seq lrs}$ **using** *g1* **by** *simp*
obtain *s* **where** *h3*: $s = (ph1\ k, [(p\alpha i1\ k, ph1\ (Suc\ k))])$ **by** *blast*
then have $s \in \textit{Decreasing-Diagrams.seq lrs}$
using *h1 d1′ Decreasing-Diagrams.seq.intros*(*2*)[*of ph1 k pαi1 k*] *Decreasing-Diagrams.seq.intros*(*1*)[*of - lrs*] **by** *simp*
moreover have $lst\ (b, \textit{f-to-ls ?pf1 } k) = fst\ s$
using *pc9 h3* **unfolding** *lst-def* **by** (*cases k*, *simp+*)
ultimately show $(b, \textit{f-to-ls ?pf1 } (Suc\ k)) \in \textit{Decreasing-Diagrams.seq}$ *lrs*
using *h2 h3 Decreasing-Diagrams.seq-concat-helper*[*of b f-to-ls ?pf1 k lrs s*] **by** *simp*
qed
qed
qed
then show *?thesis* **using** *pc11* **by** *blast*
qed
moreover have $lst\ (b, pls1) = fst\ \tau''$
proof −
have $lst\ (b, pls1) = b'$ **using** *pc9 pc11* **unfolding** *lst-def* **by** (*cases pn1*, *simp+*)
then show *?thesis* **unfolding** *qc12* **by** *simp*
qed
ultimately show *?thesis* **using** *d3*
unfolding *c12* **using** *Decreasing-Diagrams.seq-concat-helper*[*of b pls1 lrs* τ''] **by** *blast*
qed
moreover have $fst\ \sigma = fst\ \tau$ **using** *c4* **by** *simp*
moreover have $lst\ \sigma = fst\ \tau'$ **using** *c4 c12* **unfolding** *lst-def* **by** *simp*
moreover have $lst\ \tau = fst\ \sigma'$ **using** *c4 c16* **unfolding** *lst-def* **by** *simp*
moreover have $lst\ \sigma' = lst\ \tau'$
proof −
have $lst\ \tau'' = d$
proof ($cases\ n1 = 0$)
assume $n1 = 0$
then show $lst\ \tau'' = d$ **using** *c9 c11 qc12* **unfolding** *lst-def* **by** *force*
next
assume $n1 \neq 0$
moreover then have $last\ ls1 = (\ \alpha i1\ (n1-1),\ h1\ n1\ )$ **using** *c11* **by** (*cases n1*, *simp+*)
ultimately show $lst\ \tau'' = d$ **using** *c9 c11 qc12 lem-ftofs-len* **unfolding**

lst-def
by (*smt last-ConsR list.distinct*(*1*) *list.size*(*3*) *snd-conv*)
qed
moreover have *lst* σ'' = *d*
proof (*cases n2* = *0*)
assume *n2* = *0*
then show *lst* σ'' = *d* **using** *c13 c15 qc16* **unfolding** *lst-def* **by** *force*
next
assume *n2* $\neq$ *0*
moreover then have *last ls2* = (α*i2* (*n2*−*1*), *h2 n2*) **using** *c15* **by** (*cases n2*, *simp*+)
ultimately show *lst* σ'' = *d* **using** *c13 c15 qc16 lem-ftofs-len* **unfolding** *lst-def*
by (*smt last-ConsR list.distinct*(*1*) *list.size*(*3*) *snd-conv*)
qed
moreover have *lst* τ' = *lst* τ'' $\wedge$ *lst* σ' = *lst* σ'' **using** *d5 d6* **by** *blast*
ultimately show *?thesis* **by** *metis*
qed
moreover have *Decreasing-Diagrams.D cmp* (*labels* τ) (*labels* σ) (*labels* σ') (*labels* τ')
proof −
obtain σ*1* **where** *e01*: σ*1* = (*f-to-ls p*α*i2 pn2*) **by** *blast*
obtain σ*2* **where** *e1*: σ*2* = (*if c*$'$ = *c*$''$ *then* [] *else* [α]) **by** *blast*
obtain σ*3* **where** *e2*: σ*3* = (*f-to-ls* α*i2 n2*) **by** *blast*
obtain τ*1* **where** *e02*: τ*1* = (*f-to-ls p*α*i1 pn1*) **by** *blast*
obtain τ*2* **where** *e3*: τ*2* = (*if b*$'$ = *b*$''$ *then* [] *else* [β]) **by** *blast*
obtain τ*3* **where** *e4*: τ*3* = (*f-to-ls* α*i1 n1*) **by** *blast*
have *labels* τ = [β] $\wedge$ *labels* σ = [α] **using** *c4* **unfolding** *labels-def* **by** *simp*
moreover have *labels* σ' = σ*1* @ σ*2* @ σ*3*
proof −
have *labels* σ'' = σ*2* @ σ*3*
proof −
have $\forall$ *k*. *k* $\leq$ *n2* $\longrightarrow$ *map fst* (*f-to-ls ?f2 k*) = *f-to-ls* α*i2 k*
proof
fix *k*
show *k* $\leq$ *n2* $\longrightarrow$ *map fst* (*f-to-ls ?f2 k*) = *f-to-ls* α*i2 k* **by** (*induct k*, *simp*+)
qed
then show *?thesis* **using** *c15 qc16 e1 e2* **unfolding** *labels-def* **by** *simp*
qed
moreover have *labels* σ' = σ*1* @ *labels* σ''
proof −
have $\forall$ *k*. *k* $\leq$ *pn2* $\longrightarrow$ *map fst* (*f-to-ls ?pf2 k*) = *f-to-ls p*α*i2 k*
proof
fix *k*
show *k* $\leq$ *pn2* $\longrightarrow$ *map fst* (*f-to-ls ?pf2 k*) = *f-to-ls p*α*i2 k* **by** (*induct k*, *simp*+)
qed
then have *map fst pls2* = σ*1* **unfolding** *pc15 e01* **by** *blast*

then show *?thesis* **unfolding** *c16 labels-def* **by** *simp*
qed
ultimately show *?thesis* **by** *simp*
qed
moreover have *labels* τ' = $\tau 1$ @ $\tau 2$ @ $\tau 3$
proof −
have *labels* τ'' = $\tau 2$ @ $\tau 3$
proof −
have ∀ *k*. *k* ≤ *n1* ⟶ *map fst* (*f-to-ls ?f1 k*) = *f-to-ls* α*i1 k*
proof
fix *k*
show *k* ≤ *n1* ⟶ *map fst* (*f-to-ls ?f1 k*) = *f-to-ls* α*i1 k* **by** (*induct k*, *simp+*)
qed
then show *?thesis* **using** *c11 qc12 e3 e4* **unfolding** *labels-def* **by** *simp*
qed
moreover have *labels* τ' = $\tau 1$ @ *labels* τ''
proof −
have ∀ *k*. *k* ≤ *pn1* ⟶ *map fst* (*f-to-ls ?pf1 k*) = *f-to-ls p*α*i1 k*
proof
fix *k*
show *k* ≤ *pn1* ⟶ *map fst* (*f-to-ls ?pf1 k*) = *f-to-ls p*α*i1 k* **by** (*induct k*, *simp+*)
qed
then have *map fst pls1* = $\tau 1$ **unfolding** *pc11 e02* **by** *blast*
then show *?thesis* **unfolding** *c12 labels-def* **by** *simp*
qed
ultimately show *?thesis* **by** *simp*
qed
moreover have *LD'* *cmp* β α $\sigma 1$ $\sigma 2$ $\sigma 3$ $\tau 1$ $\tau 2$ $\tau 3$
proof −
let *?dn* = {α' . (α',α) ∈ *cmp* ∨ (α',β) ∈ *cmp*}
have *pf1*: *set* $\sigma 1$ ⊆ {*y*. (*y*, β) ∈ *cmp*}
proof −
have ∀ *k*. *k* ≤ *pn2* ⟶ *set* (*f-to-ls p*α*i2 k*) ⊆ {*y*. (*y*, β) ∈ *cmp*}
proof
fix *k*
show *k* ≤ *pn2* ⟶ *set* (*f-to-ls p*α*i2 k*) ⊆ {*y*. (*y*, β) ∈ *cmp*} **using** *b5 pc14* **by** (*induct k*, *simp+*)
qed
then show *?thesis* **using** *e01* **by** *blast*
qed
have *pf2*: *set* $\tau 1$ ⊆ {*y*. (*y*, α) ∈ *cmp*}
proof −
have ∀ *k*. *k* ≤ *pn1* ⟶ *set* (*f-to-ls p*α*i1 k*) ⊆ {*y*. (*y*, α) ∈ *cmp*}
proof
fix *k*
show *k* ≤ *pn1* ⟶ *set* (*f-to-ls p*α*i1 k*) ⊆ {*y*. (*y*, α) ∈ *cmp*} **using** *b5 pc10* **by** (*induct k*, *simp+*)

```
          qed
          then show ?thesis using e02 by blast
        qed
        have f1: set σ3 ⊆ ?dn
        proof −
          have ∀ k. k ≤ n2 ⟶ set (f-to-ls αi2 k) ⊆ ?dn
          proof
            fix k
             show k ≤ n2 ⟶ set (f-to-ls αi2 k) ⊆ ?dn using b5 c14 by (induct
k, simp+)
          qed
          then show ?thesis using e2 by blast
        qed
        have f2: set τ3 ⊆ ?dn
        proof −
          have ∀ k. k ≤ n1 ⟶ set (f-to-ls αi1 k) ⊆ ?dn
          proof
            fix k
             show k ≤ n1 ⟶ set (f-to-ls αi1 k) ⊆ ?dn using b5 c10 by (induct
k, simp+)
          qed
          then show ?thesis using e4 by blast
        qed
        have LD-1' cmp β α σ1 σ2 σ3 using pf1 f1 e1 e2 unfolding LD-1'-def
Decreasing-Diagrams.ds-def by simp
        moreover have LD-1' cmp α β τ1 τ2 τ3 using pf2 f2 e3 e4 unfolding
LD-1'-def Decreasing-Diagrams.ds-def by force
        ultimately show ?thesis unfolding LD'-def by blast
      qed
      moreover have trans cmp ∧ wf cmp using b6 b7 by blast
      moreover then have irrefl cmp using irrefl-def by fastforce
       ultimately show ?thesis using proposition3-4[of cmp β α σ1 σ2 σ3 τ1
τ2 τ3] by simp
    qed
    ultimately show ?thesis unfolding DD-def diagram-def D2-def by simp
  qed
  then show ∃ σ' τ'. DD lrs cmp (fst P,snd P,σ',τ') using c2 by fastforce
 qed
 ultimately show ?thesis unfolding LD-def by blast
qed
```

2 Main theorem

The next theorem has the following meaning: if the cardinality of a binary relation r does not exceed the first uncountable cardinal (*cardSuc* $|UNIV::nat\ set|$), then the following two conditions are equivalent:

1. r is confluent (*Abstract-Rewriting.CR* r)

2. r can be proven confluent using the decreasing diagrams method with natural numbers as labels (*Decreasing-Diagrams.LD* (*UNIV::nat set*) *r*).

theorem *N1-completeness*:
fixes *r::'a rel*
assumes $|r| \leq o$ *cardSuc* |*UNIV::nat set*|
shows *Abstract-Rewriting.CR r* = *Decreasing-Diagrams.LD* (*UNIV::nat set*) *r*
proof
  assume *b0*: *CR r*
  have *b1*: $|r| \leq o$ *cardSuc* |*UNIV::nat set*| **using** *assms* **by** *simp*
  obtain κ **where** *b2*: κ = *cardSuc* |*UNIV::nat set*| **by** *blast*
  have |*Field r*| $\leq o$ *cardSuc* |*UNIV::nat set*|
  proof (*cases finite r*)
    assume *finite r*
    then show *?thesis* **using** *b2 lem-fin-fl-rel* **by** (*metis Field-card-of Field-natLeq cardSuc-ordLeq-ordLess*
    card-of-card-order-on card-of-mono2 finite-iff-ordLess-natLeq ordLess-imp-ordLeq)
  next
    assume ¬ *finite r*
    then show *?thesis* **using** *b1 b2 lem-rel-inf-fld-card* **using** *ordIso-ordLeq-trans* **by** *blast*
  qed
  moreover have *confl-rel r* **using** *b0* **unfolding** *confl-rel-def Abstract-Rewriting.CR-on-def* **by** *blast*
  ultimately show *LD* (*UNIV::nat set*) *r* **using** *lem-dc3-confl-lewsuc*[*of r*] *lem-dcr-to-ld* **by** *blast*
next
  assume *LD* (*UNIV::nat set*) *r*
  then show *CR r* **using** *Decreasing-Diagrams.sound* **by** *blast*
qed

end

References

[1] I. Ivanov. Formal proof of completeness of the decreasing diagrams method for proving confluence of relations of the least uncountable cardinality, 2024. https://doi.org/10.5281/zenodo.14254256, Formal proof development.

[2] I. Ivanov. Formalization of an abstract rewriting system in the class $DCR_3 \backslash DCR_2$, 2024. https://doi.org/10.5281/zenodo.11571490, Formal proof development.

[3] I. Ivanov. On non-triviality of the hierarchy of decreasing Church-Rosser abstract rewriting systems. In *Proceedings of the 13th International Workshop on Confluence*, pages 30–35, 2024.

[4] I. Ivanov. Formalization of a confluent abstract rewriting system of the least uncountable cardinality outside of the class DCR_2, 2025. https://doi.org/10.5281/zenodo.14740062, Formal proof development.

[5] I. Ivanov. Modified version of a formal proof of completeness of the decreasing diagrams method for proving confluence of relations of the least uncountable cardinality, 2025. https://doi.org/10.5281/zenodo.15190469, Formal proof development.

[6] C. Sternagel and R. Thiemann. Abstract rewriting. *Archive of Formal Proofs*, June 2010. https://isa-afp.org/entries/Abstract-Rewriting.html, Formal proof development.

[7] V. Van Oostrom. Confluence by decreasing diagrams. *Theoretical computer science*, 126(2):259–280, 1994.

[8] H. Zankl. Decreasing diagrams. *Archive of Formal Proofs*, November 2013. https://isa-afp.org/entries/Decreasing-Diagrams.html, Formal proof development.