Countable Ordinals

Brian Huffman

October 13, 2025

Abstract

This development defines a well-ordered type of countable ordinals. It includes notions of continuous and normal functions, recursively defined functions over ordinals, least fixed-points, and derivatives. Much of ordinal arithmetic is formalized, including exponentials and logarithms. The development concludes with formalizations of Cantor Normal Form and Veblen hierarchies over normal functions.

Contents

1 Definition of Ordinals

theory *OrdinalDef*
  imports *Main*
begin

1.1 Preliminary datatype for ordinals

datatype $ord0 = ord0\text{-}Zero \mid ord0\text{-}Lim\ nat \Rightarrow ord0$

subterm ordering on ord0

definition
  $ord0\text{-}prec :: (ord0 \times ord0)\ set$ **where**
  $ord0\text{-}prec = (\bigcup f\ i.\ \{(f\ i,\ ord0\text{-}Lim\ f)\})$

lemma *wf-ord0-prec*: $wf\ ord0\text{-}prec$
⟨*proof*⟩

lemmas $ord0\text{-}prec\text{-}induct = wf\text{-}induct[OF\ wf\text{-}trancl[OF\ wf\text{-}ord0\text{-}prec]]$

less-than-or-equal ordering on ord0

inductive-set $ord0\text{-}leq :: (ord0 \times ord0)\ set$ **where**
  $[\![\forall a.\ (a,x) \in ord0\text{-}prec^+ \longrightarrow (\exists b.\ (b,y) \in ord0\text{-}prec^+ \wedge (a,b) \in ord0\text{-}leq)]\!]$
  $\Longrightarrow (x,y) \in ord0\text{-}leq$

lemma *ord0-leqI*:
  $[\![\forall a.\ (a,x) \in ord0\text{-}prec^+ \longrightarrow (a,y) \in ord0\text{-}leq\ O\ ord0\text{-}prec^+]\!]$
 $\Longrightarrow (x,y) \in ord0\text{-}leq$
  ⟨*proof*⟩

lemma *ord0-leqD*:
  $[\![(x,y) \in ord0\text{-}leq;\ (a,x) \in ord0\text{-}prec^+]\!] \Longrightarrow (a,y) \in ord0\text{-}leq\ O\ ord0\text{-}prec^+$
  ⟨*proof*⟩

lemma *ord0-leq-refl*: $(x,\ x) \in ord0\text{-}leq$
  ⟨*proof*⟩

lemma *ord0-leq-trans*:
  $(x,y) \in ord0\text{-}leq \Longrightarrow (y,z) \in ord0\text{-}leq \Longrightarrow (x,z) \in ord0\text{-}leq$
⟨*proof*⟩

lemma *wf-ord0-leq*: $wf\ (ord0\text{-}leq\ O\ ord0\text{-}prec^+)$
  ⟨*proof*⟩

ordering on ord0

instantiation *ord0* :: *ord*
begin

definition

ord0-less-def: $x < y \longleftrightarrow (x,y) \in$ *ord0-leq O ord0-prec*$^+$

definition
ord0-le-def: $x \leq y \longleftrightarrow (x,y) \in$ *ord0-leq*

instance ⟨*proof*⟩

end

lemma *ord0-order-refl*[*simp*]: $(x::ord0) \leq x$
⟨*proof*⟩

lemma *ord0-order-trans*: $[\![(x::ord0) \leq y;\ y \leq z]\!] \Longrightarrow x \leq z$
⟨*proof*⟩

lemma *ord0-wf*: *wf* $\{(x,y::ord0).\ x < y\}$
⟨*proof*⟩

lemmas *ord0-less-induct* = *wf-induct*[*OF ord0-wf*]

lemma *ord0-leI*: $[\![\forall\, a::ord0.\ a < x \longrightarrow a < y]\!] \Longrightarrow x \leq y$
⟨*proof*⟩

lemma *ord0-less-le-trans*: $[\![(x::ord0) < y;\ y \leq z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *ord0-le-less-trans*:
$[\![(x::ord0) \leq y;\ y < z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *rev-ord0-le-less-trans*:
$[\![(y::ord0) < z;\ x \leq y]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *ord0-less-trans*: $[\![(x::ord0) < y;\ y < z]\!] \Longrightarrow x < z$
⟨*proof*⟩

lemma *ord0-less-imp-le*: $(x::ord0) < y \Longrightarrow x \leq y$
⟨*proof*⟩

lemma *ord0-linear-lemma*:
fixes m :: *ord0* **and** n :: *ord0*
shows $m < n \lor n < m \lor (m \leq n \land n \leq m)$
⟨*proof*⟩

lemma *ord0-linear*: $(x::ord0) \leq y \lor y \leq x$
⟨*proof*⟩

lemma *ord0-order-less-le*: $(x::ord0) < y \longleftrightarrow (x \leq y \land \neg\ y \leq x)$ (**is** *?L=?R*)

⟨*proof*⟩

1.2 Ordinal type

definition
ord0rel :: (*ord0* × *ord0*) *set* **where**
ord0rel = {(*x*,*y*). *x* ≤ *y* ∧ *y* ≤ *x*}

typedef *ordinal* = (*UNIV*::*ord0 set*) // *ord0rel*
⟨*proof*⟩

theorem *Abs-ordinal-cases2* [*case-names Abs-ordinal*, *cases type*: *ordinal*]:
(⋀*z*. *x* = *Abs-ordinal* (*ord0rel* ‘‘ {*z*}) ⟹ *P*) ⟹ *P*
⟨*proof*⟩

instantiation *ordinal* :: *ord*
begin

definition
ordinal-less-def: *x* < *y* ⟷ (∀ *a*∈*Rep-ordinal x*. ∀ *b*∈*Rep-ordinal y*. *a* < *b*)

definition
ordinal-le-def: *x* ≤ *y* ⟷ (∀ *a*∈*Rep-ordinal x*. ∀ *b*∈*Rep-ordinal y*. *a* ≤ *b*)

instance ⟨*proof*⟩

end

lemma *Rep-Abs-ord0rel* [*simp*]:
Rep-ordinal (*Abs-ordinal* (*ord0rel* ‘‘ {*x*})) = (*ord0rel* ‘‘ {*x*})
⟨*proof*⟩

lemma *mem-ord0rel-Image* [*simp*, *intro*!]: *x* ∈ *ord0rel* ‘‘ {*x*}
⟨*proof*⟩

lemma *equiv-ord0rel*: *equiv UNIV ord0rel*
⟨*proof*⟩

lemma *Abs-ordinal-eq*[*simp*]:
(*Abs-ordinal* (*ord0rel* ‘‘ {*x*}) = *Abs-ordinal* (*ord0rel* ‘‘ {*y*})) = (*x* ≤ *y* ∧ *y* ≤ *x*)
⟨*proof*⟩

lemma *Abs-ordinal-le*[*simp*]:
Abs-ordinal (*ord0rel* ‘‘ {*x*}) ≤ *Abs-ordinal* (*ord0rel* ‘‘ {*y*}) ⟷ (*x* ≤ *y*) (**is** *?L*=*?R*)
⟨*proof*⟩

lemma *Abs-ordinal-less*[*simp*]:

$Abs\text{-}ordinal\ (ord0rel\ ``\ \{x\}) < Abs\text{-}ordinal\ (ord0rel\ ``\ \{y\}) \longleftrightarrow (x < y)$ (**is** $?L=?R$)
⟨*proof*⟩

instance *ordinal* :: *linorder*
⟨*proof*⟩

instance *ordinal* :: *wellorder*
⟨*proof*⟩

lemma *ordinal-linear*: $(x::ordinal) \le y \lor y \le x$
⟨*proof*⟩

lemma *ordinal-wf*: $wf\ \{(x,y::ordinal).\ x < y\}$
⟨*proof*⟩

1.3 Induction over ordinals

zero and strict limits

definition
oZero :: *ordinal* **where**
$oZero = Abs\text{-}ordinal\ (ord0rel\ ``\ \{ord0\text{-}Zero\})$

definition
oStrictLimit :: $(nat \Rightarrow ordinal) \Rightarrow ordinal$ **where**
$oStrictLimit\ f = Abs\text{-}ordinal$
$(ord0rel\ ``\ \{ord0\text{-}Lim\ (\lambda n.\ SOME\ x.\ x \in Rep\text{-}ordinal\ (f\ n))\})$

induction over ordinals

lemma *ord0relD*: $(x,y) \in ord0rel \Longrightarrow x \le y \land y \le x$
⟨*proof*⟩

lemma *ord0-precD*: $(x,y) \in ord0\text{-}prec \Longrightarrow \exists f\ n.\ x = f\ n \land y = ord0\text{-}Lim\ f$
⟨*proof*⟩

lemma *less-ord0-LimI*: $f\ n < ord0\text{-}Lim\ f$
⟨*proof*⟩

lemma *less-ord0-LimD*:
assumes $x < ord0\text{-}Lim\ f$ **shows** $\exists n.\ x \le f\ n$
⟨*proof*⟩

lemma *some-ord0rel*: $(x,\ SOME\ y.\ (x,y) \in ord0rel) \in ord0rel$
⟨*proof*⟩

lemma *ord0-Lim-le*: $\forall n.\ f\ n \le g\ n \Longrightarrow ord0\text{-}Lim\ f \le ord0\text{-}Lim\ g$
⟨*proof*⟩

lemma *ord0-Lim-ord0rel*:

$\forall\, n.\ (f\ n,\ g\ n) \in ord0rel \Longrightarrow (ord0\text{-}Lim\ f,\ ord0\text{-}Lim\ g) \in ord0rel$
⟨*proof*⟩

lemma *Abs-ordinal-oStrictLimit*:
$Abs\text{-}ordinal\ (ord0rel\ ``\ \{ord0\text{-}Lim\ f\})$
$= oStrictLimit\ (\lambda n.\ Abs\text{-}ordinal\ (ord0rel\ ``\ \{f\ n\}))$
⟨*proof*⟩

lemma *oStrictLimit-induct*:
assumes *base*: $P\ oZero$
assumes *step*: $\bigwedge f.\ \forall\, n.\ P\ (f\ n) \Longrightarrow P\ (oStrictLimit\ f)$
shows $P\ a$
⟨*proof*⟩

order properties of 0 and strict limits

lemma *oZero-least*: $oZero \le x$
⟨*proof*⟩

lemma *oStrictLimit-ub*: $f\ n < oStrictLimit\ f$
⟨*proof*⟩

lemma *oStrictLimit-lub*:
assumes $\forall\, n.\ f\ n < x$ **shows** $oStrictLimit\ f \le x$
⟨*proof*⟩

lemma *less-oStrictLimitD*: $x < oStrictLimit\ f \Longrightarrow \exists\, n.\ x \le f\ n$
⟨*proof*⟩

end

2 Ordinal Induction

theory *OrdinalInduct*
imports *OrdinalDef*
begin

2.1 Zero and successor ordinals

definition
$oSuc :: ordinal \Rightarrow ordinal$ **where**
$oSuc\ x = oStrictLimit\ (\lambda n.\ x)$

lemma *less-oSuc*[*iff*]: $x < oSuc\ x$
⟨*proof*⟩

lemma *oSuc-leI*: $x < y \Longrightarrow oSuc\ x \le y$
⟨*proof*⟩

instantiation $ordinal :: \{zero,\ one\}$

begin

definition
ordinal-zero-def: $(0::ordinal) = oZero$

definition
ordinal-one-def [*simp*]: $(1::ordinal) = oSuc\ 0$

instance ⟨*proof*⟩

end

2.1.1 Derived properties of 0 and oSuc

lemma *less-oSuc-eq-le*: $(x < oSuc\ y) = (x \leq y)$
⟨*proof*⟩

lemma *ordinal-0-le* [*iff*]: $0 \leq (x::ordinal)$
⟨*proof*⟩

lemma *ordinal-not-less-0* [*iff*]: $\neg\ (x::ordinal) < 0$
⟨*proof*⟩

lemma *ordinal-le-0* [*iff*]: $(x \leq 0) = (x = (0::ordinal))$
⟨*proof*⟩

lemma *ordinal-neq-0* [*iff*]: $(x \neq 0) = (0 < (x::ordinal))$
⟨*proof*⟩

lemma *ordinal-not-0-less* [*iff*]: $(\neg\ 0 < x) = (x = (0::ordinal))$
⟨*proof*⟩

lemma *oSuc-le-eq-less*: $(oSuc\ x \leq y) = (x < y)$
⟨*proof*⟩

lemma *zero-less-oSuc* [*iff*]: $0 < oSuc\ x$
⟨*proof*⟩

lemma *oSuc-not-0* [*iff*]: $oSuc\ x \neq 0$
⟨*proof*⟩

lemma *less-oSuc0* [*iff*]: $(x < oSuc\ 0) = (x = 0)$
⟨*proof*⟩

lemma *oSuc-less-oSuc* [*iff*]: $(oSuc\ x < oSuc\ y) = (x < y)$
⟨*proof*⟩

lemma *oSuc-eq-oSuc* [*iff*]: $(oSuc\ x = oSuc\ y) = (x = y)$
⟨*proof*⟩

lemma *oSuc-le-oSuc* [*iff*]: $(oSuc\ x \le oSuc\ y) = (x \le y)$
⟨*proof*⟩

lemma *le-oSucE*:
$[\![x \le oSuc\ y;\ x \le y \Longrightarrow R;\ x = oSuc\ y \Longrightarrow R]\!] \Longrightarrow R$
⟨*proof*⟩

lemma *less-oSucE*:
$[\![x < oSuc\ y;\ x < y \Longrightarrow P;\ x = y \Longrightarrow P]\!] \Longrightarrow P$
⟨*proof*⟩

2.2 Strict monotonicity

locale *strict-mono* =
fixes f
assumes *strict-mono*: $A < B \Longrightarrow f\ A < f\ B$

lemmas *strict-monoI* = *strict-mono.intro*
and *strict-monoD* = *strict-mono.strict-mono*

lemma *strict-mono-natI*:
fixes $f :: nat \Rightarrow 'a::order$
shows $(\bigwedge n.\ f\ n < f\ (Suc\ n)) \Longrightarrow strict\text{-}mono\ f$
⟨*proof*⟩

lemma *mono-natI*:
fixes $f :: nat \Rightarrow 'a::order$
shows $(\bigwedge n.\ f\ n \le f\ (Suc\ n)) \Longrightarrow mono\ f$
⟨*proof*⟩

lemma *strict-mono-mono*:
fixes $f :: 'a::order \Rightarrow 'b::order$
shows $strict\text{-}mono\ f \Longrightarrow mono\ f$
⟨*proof*⟩

lemma *strict-mono-monoD*:
fixes $f :: 'a::order \Rightarrow 'b::order$
shows $[\![strict\text{-}mono\ f;\ A \le B]\!] \Longrightarrow f\ A \le f\ B$
⟨*proof*⟩

lemma *strict-mono-cancel-eq*:
fixes $f :: 'a::linorder \Rightarrow 'b::linorder$
shows $strict\text{-}mono\ f \Longrightarrow (f\ x = f\ y) = (x = y)$
⟨*proof*⟩

lemma *strict-mono-cancel-less*:
fixes $f :: 'a::linorder \Rightarrow 'b::linorder$
shows $strict\text{-}mono\ f \Longrightarrow (f\ x < f\ y) = (x < y)$

⟨*proof*⟩

lemma *strict-mono-cancel-le*:
fixes $f :: 'a::linorder \Rightarrow 'b::linorder$
shows $strict\text{-}mono\ f \Longrightarrow (f\ x \leq f\ y) = (x \leq y)$
⟨*proof*⟩

2.3 Limit ordinals

definition
$oLimit :: (nat \Rightarrow ordinal) \Rightarrow ordinal$ **where**
$oLimit\ f = (LEAST\ k.\ \forall n.\ f\ n \leq k)$

lemma *oLimit-leI*: $\forall n.\ f\ n \leq x \Longrightarrow oLimit\ f \leq x$
⟨*proof*⟩

lemma *le-oLimit* [*iff*]: $f\ n \leq oLimit\ f$
⟨*proof*⟩

lemma *le-oLimitI*: $x \leq f\ n \Longrightarrow x \leq oLimit\ f$
⟨*proof*⟩

lemma *less-oLimitI*: $x < f\ n \Longrightarrow x < oLimit\ f$
⟨*proof*⟩

lemma *less-oLimitD*: $x < oLimit\ f \Longrightarrow \exists n.\ x < f\ n$
⟨*proof*⟩

lemma *less-oLimitE*: $[\![x < oLimit\ f;\ \bigwedge n.\ x < f\ n \Longrightarrow P]\!] \Longrightarrow P$
⟨*proof*⟩

lemma *le-oLimitE*:
$[\![x \leq oLimit\ f;\ \bigwedge n.\ x \leq f\ n \Longrightarrow R;\ x = oLimit\ f \Longrightarrow R]\!] \Longrightarrow R$
⟨*proof*⟩

lemma *oLimit-const* [*simp*]: $oLimit\ (\lambda n.\ x) = x$
⟨*proof*⟩

lemma *strict-mono-less-oLimit*: $strict\text{-}mono\ f \Longrightarrow f\ n < oLimit\ f$
⟨*proof*⟩

lemma *oLimit-eqI*:
$[\![\bigwedge n.\ \exists m.\ f\ n \leq g\ m;\ \bigwedge n.\ \exists m.\ g\ n \leq f\ m]\!] \Longrightarrow oLimit\ f = oLimit\ g$
⟨*proof*⟩

lemma *oLimit-Suc*:
$f\ 0 < oLimit\ f \Longrightarrow oLimit\ (\lambda n.\ f\ (Suc\ n)) = oLimit\ f$
⟨*proof*⟩

lemma *oLimit-shift*:
$\forall n.\ f\ n < oLimit\ f \Longrightarrow oLimit\ (\lambda n.\ f\ (n + k)) = oLimit\ f$
$\langle proof \rangle$

lemma *oLimit-shift-mono*:
$mono\ f \Longrightarrow oLimit\ (\lambda n.\ f\ (n + k)) = oLimit\ f$
$\langle proof \rangle$

limit ordinal predicate

definition
limit-ordinal :: *ordinal* $\Rightarrow$ *bool* **where**
$limit\text{-}ordinal\ x \longleftrightarrow (x \neq 0) \wedge (\forall y.\ x \neq oSuc\ y)$

lemma *limit-ordinal-not-0* [*simp*]: $\neg\ limit\text{-}ordinal\ 0$
$\langle proof \rangle$

lemma *zero-less-limit-ordinal* [*simp*]: $limit\text{-}ordinal\ x \Longrightarrow 0 < x$
$\langle proof \rangle$

lemma *limit-ordinal-not-oSuc* [*simp*]: $\neg\ limit\text{-}ordinal\ (oSuc\ p)$
$\langle proof \rangle$

lemma *oSuc-less-limit-ordinal*:
$limit\text{-}ordinal\ x \Longrightarrow (oSuc\ w < x) = (w < x)$
$\langle proof \rangle$

lemma *limit-ordinal-oLimitI*:
$\forall n.\ f\ n < oLimit\ f \Longrightarrow limit\text{-}ordinal\ (oLimit\ f)$
$\langle proof \rangle$

lemma *strict-mono-limit-ordinal*:
$strict\text{-}mono\ f \Longrightarrow limit\text{-}ordinal\ (oLimit\ f)$
$\langle proof \rangle$

lemma *limit-ordinalI*:
$[\![0 < z;\ \forall x{<}z.\ oSuc\ x < z]\!] \Longrightarrow limit\text{-}ordinal\ z$
$\langle proof \rangle$

2.3.1 Making strict monotonic sequences

primrec *make-mono* :: $(nat \Rightarrow ordinal) \Rightarrow nat \Rightarrow nat$
where
$\quad make\text{-}mono\ f\ 0 \quad = 0$
$|\ make\text{-}mono\ f\ (Suc\ n) = (LEAST\ x.\ f\ (make\text{-}mono\ f\ n) < f\ x)$

lemma *f-make-mono-less*:
$\forall n.\ f\ n < oLimit\ f \Longrightarrow f\ (make\text{-}mono\ f\ n) < f\ (make\text{-}mono\ f\ (Suc\ n))$
$\langle proof \rangle$

lemma *strict-mono-f-make-mono*:
$\forall n.\ f\ n < oLimit\ f \Longrightarrow strict\text{-}mono\ (\lambda n.\ f\ (make\text{-}mono\ f\ n))$
⟨*proof*⟩

lemma *le-f-make-mono*:
$[\![\forall n.\ f\ n < oLimit\ f;\ m \le make\text{-}mono\ f\ n]\!] \Longrightarrow f\ m \le f\ (make\text{-}mono\ f\ n)$
⟨*proof*⟩

lemma *make-mono-less*:
$\forall n.\ f\ n < oLimit\ f \Longrightarrow make\text{-}mono\ f\ n < make\text{-}mono\ f\ (Suc\ n)$
⟨*proof*⟩

declare *make-mono.simps* [*simp del*]

lemma *oLimit-make-mono-eq*:
assumes $\forall n.\ f\ n < oLimit\ f$ **shows** $oLimit\ (\lambda n.\ f\ (make\text{-}mono\ f\ n)) = oLimit\ f$
⟨*proof*⟩

2.4 Induction principle for ordinals

lemma *oLimit-le-oStrictLimit*: $oLimit\ f \le oStrictLimit\ f$
⟨*proof*⟩

lemma *oLimit-induct* [*case-names zero suc lim*]:
assumes *zero*: $P\ 0$
and *suc*: $\bigwedge x.\ P\ x \Longrightarrow P\ (oSuc\ x)$
and *lim*: $\bigwedge f.\ [\![strict\text{-}mono\ f;\ \forall n.\ P\ (f\ n)]\!] \Longrightarrow P\ (oLimit\ f)$
shows $P\ a$
⟨*proof*⟩

lemma *ordinal-cases* [*case-names zero suc lim*]:
assumes *zero*: $a = 0 \Longrightarrow P$
and *suc*: $\bigwedge x.\ a = oSuc\ x \Longrightarrow P$
and *lim*: $\bigwedge f.\ [\![strict\text{-}mono\ f;\ a = oLimit\ f]\!] \Longrightarrow P$
shows P
⟨*proof*⟩

end

3 Continuity

theory *OrdinalCont*
imports *OrdinalInduct*
begin

3.1 Continuous functions

locale *continuous* =
fixes $F :: ordinal \Rightarrow ordinal$

assumes *cont*: F (*oLimit* f) $=$ *oLimit* ($\lambda n.$ F (f n))

lemmas *continuousD* $=$ *continuous.cont*

lemma (**in** *continuous*) *monoD*: **assumes** $x \leq y$ **shows** $F\ x \leq F\ y$
⟨*proof*⟩

lemma (**in** *continuous*) *mono*: *mono* F
⟨*proof*⟩

lemma *continuousI*:
assumes *lim*: $\bigwedge f.$ *strict-mono* $f \Longrightarrow F$ (*oLimit* f) $=$ *oLimit* ($\lambda n.$ F (f n))
assumes *suc*: $\bigwedge x.$ $F\ x \leq F$ (*oSuc* x)
shows *continuous* F
⟨*proof*⟩

3.2 Normal functions

locale *normal* $=$ *continuous* $+$
assumes *strict*: *strict-mono* F

lemma (**in** *normal*) *mono*: *mono* F
⟨*proof*⟩

lemma (**in** *normal*) *continuous*: *continuous* F
⟨*proof*⟩

lemma (**in** *normal*) *monoD*: $x \leq y \Longrightarrow F\ x \leq F\ y$
⟨*proof*⟩

lemma (**in** *normal*) *strict-monoD*: $x < y \Longrightarrow F\ x < F\ y$
⟨*proof*⟩

lemma (**in** *normal*) *cancel-eq*: $(F\ x = F\ y) = (x = y)$
⟨*proof*⟩

lemma (**in** *normal*) *cancel-less*: $(F\ x < F\ y) = (x < y)$
⟨*proof*⟩

lemma (**in** *normal*) *cancel-le*: $(F\ x \leq F\ y) = (x \leq y)$
⟨*proof*⟩

lemma (**in** *normal*) *oLimit*: F (*oLimit* f) $=$ *oLimit* ($\lambda n.$ F (f n))
⟨*proof*⟩

lemma (**in** *normal*) *increasing*: $x \leq F\ x$
⟨*proof*⟩

lemma *normalI*:

assumes *lim*: $\bigwedge f.\ strict\text{-}mono\ f \Longrightarrow F\ (oLimit\ f) = oLimit\ (\lambda n.\ F\ (f\ n))$
assumes *suc*: $\bigwedge x.\ F\ x < F\ (oSuc\ x)$
shows *normal F*
⟨*proof*⟩

lemma *normal-range-le*:
assumes *nml*: *normal F normal G* **and** *range G* $\subseteq$ *range F*
shows $F\ x \leq G\ x$
⟨*proof*⟩

lemma *normal-range-eq*:
$[\![normal\ F;\ normal\ G;\ range\ F = range\ G]\!] \Longrightarrow F = G$
⟨*proof*⟩

end

4 Recursive Definitions

theory *OrdinalRec*
imports *OrdinalCont*
begin

definition
oPrec :: *ordinal* $\Rightarrow$ *ordinal* **where**
$oPrec\ x = (THE\ p.\ x = oSuc\ p)$

lemma *oPrec-oSuc* [*simp*]: $oPrec\ (oSuc\ x) = x$
⟨*proof*⟩

lemma *oPrec-less*: $\exists\, p.\ x = oSuc\ p \Longrightarrow oPrec\ x < x$
⟨*proof*⟩

definition
ordinal-rec0 ::
$['a,\ ordinal \Rightarrow 'a \Rightarrow 'a,\ (nat \Rightarrow 'a) \Rightarrow 'a,\ ordinal] \Rightarrow 'a$ **where**
$ordinal\text{-}rec0\ z\ s\ l \equiv wfrec\ \{(x,y).\ x < y\}\ (\lambda F\ x.$
$if\ x = 0\ then\ z\ else$
$if\ (\exists\, p.\ x = oSuc\ p)\ then\ s\ (oPrec\ x)\ (F\ (oPrec\ x))\ else$
$(THE\ y.\ \forall f.\ (\forall\, n.\ f\ n < oLimit\ f) \land oLimit\ f = x$
$\longrightarrow l\ (\lambda n.\ F\ (f\ n)) = y))$

lemma *ordinal-rec0-0* [*simp*]: $ordinal\text{-}rec0\ z\ s\ l\ 0 = z$
⟨*proof*⟩

lemma *ordinal-rec0-oSuc*: $ordinal\text{-}rec0\ z\ s\ l\ (oSuc\ x) = s\ x\ (ordinal\text{-}rec0\ z\ s\ l\ x)$
⟨*proof*⟩

lemma *limit-ordinal-not-0*: $limit\text{-}ordinal\ x \Longrightarrow x \neq 0$ **and** *limit-ordinal-not-oSuc*:
$limit\text{-}ordinal\ x \Longrightarrow x \neq oSuc\ p$

$\langle proof \rangle$

lemma *ordinal-rec0-limit-ordinal*:
limit-ordinal $x \Longrightarrow$ *ordinal-rec0* $z\ s\ l\ x =$
$(THE\ y.\ \forall f.\ (\forall n.\ f\ n < oLimit\ f) \wedge oLimit\ f = x \longrightarrow$
$l\ (\lambda n.\ ordinal\text{-}rec0\ z\ s\ l\ (f\ n)) = y)$
$\langle proof \rangle$

4.1 Partial orders

locale *porder* =
fixes *le* :: $'a \Rightarrow 'a \Rightarrow bool$ (**infixl** ‹<<› *55*)
assumes *po-refl*: $\bigwedge x.\ x << x$
and *po-trans*: $\bigwedge x\ y\ z.\ [\![x << y;\ y << z]\!] \Longrightarrow x << z$
and *po-antisym*: $\bigwedge x\ y.\ [\![x << y;\ y << x]\!] \Longrightarrow x = y$

lemma *porder-order*: *porder* $((\leq) :: 'a::order \Rightarrow 'a \Rightarrow bool)$
$\langle proof \rangle$

lemma (**in** *porder*) *flip*: *porder* $(\lambda x\ y.\ y << x)$
$\langle proof \rangle$

locale *omega-complete* = *porder* +
fixes *lub* :: $(nat \Rightarrow 'a) \Rightarrow 'a$
assumes *is-ub-lub*: $\bigwedge f\ n.\ f\ n << lub\ f$
assumes *is-lub-lub*: $\bigwedge f\ x.\ \forall n.\ f\ n << x \Longrightarrow lub\ f << x$

lemma (**in** *omega-complete*) *lub-cong-lemma*:
$[\![\forall n.\ f\ n < oLimit\ f;\ \forall m.\ g\ m < oLimit\ g;\ oLimit\ f \leq oLimit\ g;$
$\forall y{<}oLimit\ g.\ \forall x{\leq}y.\ F\ x << F\ y]\!]$
$\Longrightarrow lub\ (\lambda n.\ F\ (f\ n)) << lub\ (\lambda n.\ F\ (g\ n))$
$\langle proof \rangle$

lemma (**in** *omega-complete*) *lub-cong*:
$[\![\forall n.\ f\ n < oLimit\ f;\ \forall m.\ g\ m < oLimit\ g;\ oLimit\ f = oLimit\ g;$
$\forall y{<}oLimit\ g.\ \forall x{\leq}y.\ F\ x << F\ y]\!]$
$\Longrightarrow lub\ (\lambda n.\ F\ (f\ n)) = lub\ (\lambda n.\ F\ (g\ n))$
$\langle proof \rangle$

lemma (**in** *omega-complete*) *ordinal-rec0-mono*:
assumes *s*: $\forall p\ x.\ x << s\ p\ x$ **and** $x \leq y$
shows *ordinal-rec0* $z\ s\ lub\ x <<$ *ordinal-rec0* $z\ s\ lub\ y$
$\langle proof \rangle$

lemma (**in** *omega-complete*) *ordinal-rec0-oLimit*:
assumes *s*: $\forall p\ x.\ x << s\ p\ x$
shows *ordinal-rec0* $z\ s\ lub\ (oLimit\ f) =$

    lub ($\lambda n.$ *ordinal-rec0 z s lub* (f n))
⟨*proof*⟩

4.2 Recursive definitions for *ordinal* ⇒ *ordinal*

definition
  ordinal-rec ::
    [*ordinal*, *ordinal* ⇒ *ordinal* ⇒ *ordinal*, *ordinal*] ⇒ *ordinal* **where**
  ordinal-rec z s = *ordinal-rec0 z s oLimit*

lemma *omega-complete-oLimit*: *omega-complete* (≤) *oLimit*
  ⟨*proof*⟩

lemma *ordinal-rec-0* [*simp*]: *ordinal-rec z s 0* = *z*
  ⟨*proof*⟩

lemma *ordinal-rec-oSuc* [*simp*]:
  ordinal-rec z s (*oSuc x*) = *s x* (*ordinal-rec z s x*)
  ⟨*proof*⟩

lemma *ordinal-rec-oLimit*:
  assumes *s*: ∀ *p x*. *x* ≤ *s p x*
  shows *ordinal-rec z s* (*oLimit f*) = *oLimit* ($\lambda n.$ *ordinal-rec z s* (f n))
  ⟨*proof*⟩

lemma *continuous-ordinal-rec*:
  assumes *s*: ∀ *p x*. *x* ≤ *s p x*
  shows *continuous* (*ordinal-rec z s*)
  ⟨*proof*⟩

lemma *mono-ordinal-rec*:
  assumes *s*: ∀ *p x*. *x* ≤ *s p x*
  shows *mono* (*ordinal-rec z s*)
  ⟨*proof*⟩

lemma *normal-ordinal-rec*:
  assumes *s*: ∀ *p x*. *x* < *s p x*
  shows *normal* (*ordinal-rec z s*)
  ⟨*proof*⟩

end

5 Ordinal Arithmetic

theory *OrdinalArith*
imports *OrdinalRec*
begin

5.1 Addition

instantiation *ordinal* :: *plus*
begin

definition
$(+) = (\lambda x.\ \mathit{ordinal\text{-}rec}\ x\ (\lambda p.\ \mathit{oSuc}))$

instance $\langle \mathit{proof} \rangle$

end

lemma *normal-plus*: $\mathit{normal}\ ((+)\ x)$
$\langle \mathit{proof} \rangle$

lemma *ordinal-plus-0* [*simp*]: $x + \mathit{0} = (x{::}\mathit{ordinal})$
$\langle \mathit{proof} \rangle$

lemma *ordinal-plus-oSuc* [*simp*]: $x + \mathit{oSuc}\ y = \mathit{oSuc}\ (x + y)$
$\langle \mathit{proof} \rangle$

lemma *ordinal-plus-oLimit* [*simp*]: $x + \mathit{oLimit}\ f = \mathit{oLimit}\ (\lambda n.\ x + f\ n)$
$\langle \mathit{proof} \rangle$

lemma *ordinal-0-plus* [*simp*]: $\mathit{0} + x = (x{::}\mathit{ordinal})$
$\langle \mathit{proof} \rangle$

lemma *ordinal-plus-assoc*: $(x + y) + z = x + (y + z{::}\mathit{ordinal})$
$\langle \mathit{proof} \rangle$

lemma *ordinal-plus-monoL* [*rule-format*]:
$\forall x\ x'.\ x \le x' \longrightarrow x + y \le x' + (y{::}\mathit{ordinal})$
$\langle \mathit{proof} \rangle$

lemma *ordinal-plus-monoR*: $y \le y' \Longrightarrow x + y \le x + (y'{::}\mathit{ordinal})$
$\langle \mathit{proof} \rangle$

lemma *ordinal-plus-mono*:
$[\![x \le x';\ y \le y']\!] \Longrightarrow x + y \le x' + (y'{::}\mathit{ordinal})$
$\langle \mathit{proof} \rangle$

lemma *ordinal-plus-strict-monoR*: $y < y' \Longrightarrow x + y < x + (y'{::}\mathit{ordinal})$
$\langle \mathit{proof} \rangle$

lemma *ordinal-le-plusL* [*simp*]: $y \le x + (y{::}\mathit{ordinal})$
$\langle \mathit{proof} \rangle$

lemma *ordinal-le-plusR* [*simp*]: $x \le x + (y{::}\mathit{ordinal})$
$\langle \mathit{proof} \rangle$

lemma *ordinal-less-plusR*: $0 < y \Longrightarrow x < x + (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-plus-left-cancel* [*simp*]:
$(w + x = w + y) = (x = (y::ordinal))$
⟨*proof*⟩

lemma *ordinal-plus-left-cancel-le* [*simp*]:
$(w + x \leq w + y) = (x \leq (y::ordinal))$
⟨*proof*⟩

lemma *ordinal-plus-left-cancel-less* [*simp*]:
$(w + x < w + y) = (x < (y::ordinal))$
⟨*proof*⟩

lemma *ordinal-plus-not-0*: $(0 < x + y) = (0 < x \vee 0 < (y::ordinal))$
⟨*proof*⟩

lemma *not-inject*: $(\neg P) = (\neg Q) \Longrightarrow P = Q$
⟨*proof*⟩

lemma *ordinal-plus-eq-0*:
$((x::ordinal) + y = 0) = (x = 0 \wedge y = 0)$
⟨*proof*⟩

5.2 Subtraction

instantiation *ordinal* :: *minus*
begin

definition
minus-ordinal-def:
$x - y = \mathit{ordinal\text{-}rec}\ 0\ (\lambda p\ w.\ \mathit{if}\ y \leq p\ \mathit{then}\ \mathit{oSuc}\ w\ \mathit{else}\ w)\ x$

instance ⟨*proof*⟩

end

lemma *continuous-minus*: *continuous* $(\lambda x.\ x - y)$
⟨*proof*⟩

lemma *ordinal-0-minus* [*simp*]: $0 - x = (0::ordinal)$
⟨*proof*⟩

lemma *ordinal-oSuc-minus* [*simp*]: $y \leq x \Longrightarrow \mathit{oSuc}\ x - y = \mathit{oSuc}\ (x - y)$
⟨*proof*⟩

lemma *ordinal-oLimit-minus* [*simp*]: $\mathit{oLimit}\ f - y = \mathit{oLimit}\ (\lambda n.\ f\ n - y)$
⟨*proof*⟩

lemma *ordinal-minus-0* [*simp*]: $x - 0 = (x::ordinal)$
⟨*proof*⟩

lemma *ordinal-oSuc-minus2*: $x < y \Longrightarrow oSuc\ x - y = x - y$
⟨*proof*⟩

lemma *ordinal-minus-eq-0* [*rule-format*, *simp*]:
$x \leq y \longrightarrow x - y = (0::ordinal)$
⟨*proof*⟩

lemma *ordinal-plus-minus1* [*simp*]: $(x + y) - x = (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-plus-minus2* [*simp*]: $x \leq y \Longrightarrow x + (y - x) = (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-minusI*: $x = y + z \Longrightarrow x - y = (z::ordinal)$
⟨*proof*⟩

lemma *ordinal-minus-less-eq* [*simp*]:
$(y::ordinal) \leq x \Longrightarrow (x - y < z) = (x < y + z)$
⟨*proof*⟩

lemma *ordinal-minus-le-eq* [*simp*]: $(x - y \leq z) = (x \leq y + (z::ordinal))$
⟨*proof*⟩

lemma *ordinal-minus-monoL*: $x \leq y \Longrightarrow x - z \leq y - (z::ordinal)$
⟨*proof*⟩

lemma *ordinal-minus-monoR*: $x \leq y \Longrightarrow z - y \leq z - (x::ordinal)$
⟨*proof*⟩

5.3 Multiplication

instantiation *ordinal* :: *times*
begin

definition
times-ordinal-def: $(*) = (\lambda x.\ ordinal\text{-}rec\ 0\ (\lambda p\ w.\ w + x))$

instance ⟨*proof*⟩

end

lemma *continuous-times*: *continuous* $((*)\ x)$
⟨*proof*⟩

lemma *normal-times*: $0 < x \Longrightarrow normal\ ((*)\ x)$

$\langle proof \rangle$

lemma *ordinal-times-0* [*simp*]: $x * 0 = (0::ordinal)$
$\langle proof \rangle$

lemma *ordinal-times-oSuc* [*simp*]: $x * oSuc\ y = (x * y) + x$
$\langle proof \rangle$

lemma *ordinal-times-oLimit* [*simp*]: $x * oLimit\ f = oLimit\ (\lambda n.\ x * f\ n)$
$\langle proof \rangle$

lemma *ordinal-0-times* [*simp*]: $0 * x = (0::ordinal)$
$\langle proof \rangle$

lemma *ordinal-1-times* [*simp*]: $oSuc\ 0 * x = (x::ordinal)$
$\langle proof \rangle$

lemma *ordinal-times-1* [*simp*]: $x * oSuc\ 0 = (x::ordinal)$
$\langle proof \rangle$

lemma *ordinal-times-distrib*:
$x * (y + z) = (x * y) + (x * z::ordinal)$
$\langle proof \rangle$

lemma *ordinal-times-assoc*:
$(x * y::ordinal) * z = x * (y * z)$
$\langle proof \rangle$

lemma *ordinal-times-monoL* [*rule-format*]:
$\forall x\ x'.\ x \leq x' \longrightarrow x * y \leq x' * (y::ordinal)$
$\langle proof \rangle$

lemma *ordinal-times-monoR*: $y \leq y' \Longrightarrow x * y \leq x * (y'::ordinal)$
$\langle proof \rangle$

lemma *ordinal-times-mono*:
$[\![x \leq x';\ y \leq y']\!] \Longrightarrow x * y \leq x' * (y'::ordinal)$
$\langle proof \rangle$

lemma *ordinal-times-strict-monoR*:
$[\![y < y';\ 0 < x]\!] \Longrightarrow x * y < x * (y'::ordinal)$
$\langle proof \rangle$

lemma *ordinal-le-timesL* [*simp*]: $0 < x \Longrightarrow y \leq x * (y::ordinal)$
$\langle proof \rangle$

lemma *ordinal-le-timesR* [*simp*]: $0 < y \Longrightarrow x \leq x * (y::ordinal)$
$\langle proof \rangle$

lemma *ordinal-less-timesR*: $[\![0 < x;\ oSuc\ 0 < y]\!] \Longrightarrow x < x * (y{::}ordinal)$
⟨*proof*⟩

lemma *ordinal-times-left-cancel* [*simp*]:
$0 < w \Longrightarrow (w * x = w * y) = (x = (y{::}ordinal))$
⟨*proof*⟩

lemma *ordinal-times-left-cancel-le* [*simp*]:
$0 < w \Longrightarrow (w * x \leq w * y) = (x \leq (y{::}ordinal))$
⟨*proof*⟩

lemma *ordinal-times-left-cancel-less* [*simp*]:
$0 < w \Longrightarrow (w * x < w * y) = (x < (y{::}ordinal))$
⟨*proof*⟩

lemma *ordinal-times-eq-0*:
$((x{::}ordinal) * y = 0) = (x = 0 \vee y = 0)$
⟨*proof*⟩

lemma *ordinal-times-not-0* [*simp*]:
$((0{::}ordinal) < x * y) = (0 < x \wedge 0 < y)$
⟨*proof*⟩

5.4 Exponentiation

definition
exp-ordinal :: [*ordinal*, *ordinal*] ⇒ *ordinal* (**infixr** ‹∗∗› *75*) **where**
$(**) = (\lambda x.\ if\ 0 < x\ then\ ordinal\text{-}rec\ 1\ (\lambda p\ w.\ w * x)$
$else\ (\lambda y.\ if\ y = 0\ then\ 1\ else\ 0))$

lemma *continuous-exp*: $0 < x \Longrightarrow continuous\ ((**)\ x)$
⟨*proof*⟩

lemma *ordinal-exp-0* [*simp*]: $x ** 0 = (1{::}ordinal)$
⟨*proof*⟩

lemma *ordinal-exp-oSuc* [*simp*]: $x ** oSuc\ y = (x ** y) * x$
⟨*proof*⟩

lemma *ordinal-exp-oLimit* [*simp*]:
$0 < x \Longrightarrow x ** oLimit\ f = oLimit\ (\lambda n.\ x ** f\ n)$
⟨*proof*⟩

lemma *ordinal-0-exp* [*simp*]: $0 ** x = (if\ x = 0\ then\ 1\ else\ 0)$
⟨*proof*⟩

lemma *ordinal-1-exp* [*simp*]: $oSuc\ 0 ** x = oSuc\ 0$
⟨*proof*⟩

lemma *ordinal-exp-1* [*simp*]: $x ** oSuc\ 0 = x$
⟨*proof*⟩

lemma *ordinal-exp-distrib*:
$x ** (y + z) = (x ** y) * (x ** (z{::}ordinal))$
⟨*proof*⟩

lemma *ordinal-exp-not-0* [*simp*]: $(0 < x ** y) = (0 < x \vee y = 0)$
⟨*proof*⟩

lemma *ordinal-exp-eq-0* [*simp*]: $(x ** y = 0) = (x = 0 \wedge 0 < y)$
⟨*proof*⟩

lemma *ordinal-exp-assoc*:
$(x ** y) ** z = x ** (y * z)$
⟨*proof*⟩

lemma *ordinal-exp-monoL* [*rule-format*]:
$\forall x\ x'.\ x \leq x' \longrightarrow x ** y \leq x' ** (y{::}ordinal)$
⟨*proof*⟩

lemma *normal-exp*: $oSuc\ 0 < x \Longrightarrow normal\ ((**)\ x)$
⟨*proof*⟩

lemma *ordinal-exp-monoR*:
$[\![0 < x;\ y \leq y']\!] \Longrightarrow x ** y \leq x ** (y'{::}ordinal)$
⟨*proof*⟩

lemma *ordinal-exp-mono*:
$[\![0 < x';\ x \leq x';\ y \leq y']\!] \Longrightarrow x ** y \leq x' ** (y'{::}ordinal)$
⟨*proof*⟩

lemma *ordinal-exp-strict-monoR*:
$[\![oSuc\ 0 < x;\ y < y']\!] \Longrightarrow x ** y < x ** (y'{::}ordinal)$
⟨*proof*⟩

lemma *ordinal-le-expR* [*simp*]: $0 < y \Longrightarrow x \leq x ** (y{::}ordinal)$
⟨*proof*⟩

lemma *ordinal-exp-left-cancel* [*simp*]:
$oSuc\ 0 < w \Longrightarrow (w ** x = w ** y) = (x = y)$
⟨*proof*⟩

lemma *ordinal-exp-left-cancel-le* [*simp*]:
$oSuc\ 0 < w \Longrightarrow (w ** x \leq w ** y) = (x \leq y)$
⟨*proof*⟩

lemma *ordinal-exp-left-cancel-less* [*simp*]:
$oSuc\ 0 < w \Longrightarrow (w ** x < w ** y) = (x < y)$

$\langle proof \rangle$

end

6 Inverse Functions

theory *OrdinalInverse*
imports *OrdinalArith*
begin

lemma (**in** *normal*) *oInv-ex*:
assumes $F\ 0 \leq a$ **shows** $\exists\, q.\ F\ q \leq a \wedge a < F\ (oSuc\ q)$
$\langle proof \rangle$

lemma *oInv-uniq*:
assumes $mono\ (F{::}ordinal \Rightarrow ordinal)\ F\ x \leq a\ a < F\ (oSuc\ x)\ F\ y \leq a\ a < F\ (oSuc\ y)$
shows $x = y$
$\langle proof \rangle$

definition
$oInv :: (ordinal \Rightarrow ordinal) \Rightarrow ordinal \Rightarrow ordinal$ **where**
$oInv\ F\ a = (if\ F\ 0 \leq a\ then\ (THE\ x.\ F\ x \leq a \wedge a < F\ (oSuc\ x))\ else\ 0)$

lemma (**in** *normal*) *oInv-bounds*: $F\ 0 \leq a \Longrightarrow F\ (oInv\ F\ a) \leq a \wedge a < F\ (oSuc\ (oInv\ F\ a))$
$\langle proof \rangle$

lemma (**in** *normal*) *oInv-bound1*:
$F\ 0 \leq a \Longrightarrow F\ (oInv\ F\ a) \leq a$
$\langle proof \rangle$

lemma (**in** *normal*) *oInv-bound2*: $a < F\ (oSuc\ (oInv\ F\ a))$
$\langle proof \rangle$

lemma (**in** *normal*) *oInv-equality*: $[\![F\ x \leq a;\ a < F\ (oSuc\ x)]\!] \Longrightarrow oInv\ F\ a = x$
$\langle proof \rangle$

lemma (**in** *normal*) *oInv-inverse*: $oInv\ F\ (F\ x) = x$
$\langle proof \rangle$

lemma (**in** *normal*) *oInv-equality'*: $a = F\ x \Longrightarrow oInv\ F\ a = x$
$\langle proof \rangle$

lemma (**in** *normal*) *oInv-eq-0*: $a \leq F\ 0 \Longrightarrow oInv\ F\ a = 0$
$\langle proof \rangle$

lemma (**in** *normal*) *oInv-less*: $[\![F\ 0 \leq a;\ a < F\ z]\!] \Longrightarrow oInv\ F\ a < z$
$\langle proof \rangle$

lemma (**in** *normal*) *le-oInv*: $F\ z \leq a \Longrightarrow z \leq oInv\ F\ a$
⟨*proof*⟩

lemma (**in** *normal*) *less-oInvD*: $x < oInv\ F\ a \Longrightarrow F\ (oSuc\ x) \leq a$
⟨*proof*⟩

lemma (**in** *normal*) *oInv-le*: $a < F\ (oSuc\ x) \Longrightarrow oInv\ F\ a \leq x$
⟨*proof*⟩

lemma (**in** *normal*) *mono-oInv*: *mono* (*oInv F*)
⟨*proof*⟩

lemma (**in** *normal*) *oInv-decreasing*: $F\ 0 \leq x \Longrightarrow oInv\ F\ x \leq x$
⟨*proof*⟩

6.1 Division

instantiation *ordinal* :: *modulo*
begin

definition
div-ordinal-def:
$x\ div\ y = (if\ 0 < y\ then\ oInv\ ((*)\ y)\ x\ else\ 0)$

definition
mod-ordinal-def:
$x\ mod\ y = ((x::ordinal) - y * (x\ div\ y))$

instance ⟨*proof*⟩

end

lemma *ordinal-divI*: $[\![x = y * q + r; r < y]\!] \Longrightarrow x\ div\ y = (q::ordinal)$
⟨*proof*⟩

lemma *ordinal-times-div-le*: $y * (x\ div\ y) \leq (x::ordinal)$
⟨*proof*⟩

lemma *ordinal-less-times-div-plus*: $0 < y \Longrightarrow x < y * (x\ div\ y) + (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-modI*: $[\![x = y * q + r; r < y]\!] \Longrightarrow x\ mod\ y = (r::ordinal)$
⟨*proof*⟩

lemma *ordinal-mod-less*: $0 < y \Longrightarrow x\ mod\ y < (y::ordinal)$
⟨*proof*⟩

lemma *ordinal-div-plus-mod*: $y * (x\ div\ y) + (x\ mod\ y) = (x::ordinal)$

$\langle proof \rangle$

lemma *ordinal-div-less*: $x < y * z \Longrightarrow x \ div \ y < (z::ordinal)$
$\langle proof \rangle$

lemma *ordinal-le-div*: $[\![0 < y; y * z \leq x]\!] \Longrightarrow (z::ordinal) \leq x \ div \ y$
$\langle proof \rangle$

lemma *ordinal-mono-div*: $mono \ (\lambda x. \ x \ div \ y::ordinal)$
$\langle proof \rangle$

lemma *ordinal-div-monoL*: $x \leq x' \Longrightarrow x \ div \ y \leq x' \ div \ (y::ordinal)$
$\langle proof \rangle$

lemma *ordinal-div-decreasing*: $(x::ordinal) \ div \ y \leq x$
$\langle proof \rangle$

lemma *ordinal-div-0*: $x \ div \ 0 = (0::ordinal)$
$\langle proof \rangle$

lemma *ordinal-mod-0*: $x \ mod \ 0 = (x::ordinal)$
$\langle proof \rangle$

6.2 Derived properties of division

lemma *ordinal-div-1* [*simp*]: $x \ div \ oSuc \ 0 = x$
$\langle proof \rangle$

lemma *ordinal-mod-1* [*simp*]: $x \ mod \ oSuc \ 0 = 0$
$\langle proof \rangle$

lemma *ordinal-div-self* [*simp*]: $0 < x \Longrightarrow x \ div \ x = (1::ordinal)$
$\langle proof \rangle$

lemma *ordinal-mod-self* [*simp*]: $x \ mod \ x = (0::ordinal)$
$\langle proof \rangle$

lemma *ordinal-div-greater* [*simp*]: $x < y \Longrightarrow x \ div \ y = (0::ordinal)$
$\langle proof \rangle$

lemma *ordinal-mod-greater* [*simp*]: $x < y \Longrightarrow x \ mod \ y = (x::ordinal)$
$\langle proof \rangle$

lemma *ordinal-0-div* [*simp*]: $0 \ div \ x = (0::ordinal)$
$\langle proof \rangle$

lemma *ordinal-0-mod* [*simp*]: $0 \ mod \ x = (0::ordinal)$
$\langle proof \rangle$

lemma *ordinal-1-dvd* [*simp*]: $oSuc\ 0\ dvd\ x$
⟨*proof*⟩

lemma *ordinal-dvd-mod*: $y\ dvd\ x = (x\ mod\ y = (0::ordinal))$
⟨*proof*⟩

lemma *ordinal-dvd-times-div*: $y\ dvd\ x \Longrightarrow y * (x\ div\ y) = (x::ordinal)$
⟨*proof*⟩

lemma *ordinal-dvd-oLimit*:
assumes $\forall n.\ x\ dvd\ f\ n$ **shows** $x\ dvd\ oLimit\ f$
⟨*proof*⟩

6.3 Logarithms

definition
$oLog :: ordinal \Rightarrow ordinal \Rightarrow ordinal$ **where**
$oLog\ b = (\lambda x.\ if\ 1 < b\ then\ oInv\ ((**)\ b)\ x\ else\ 0)$

lemma *ordinal-oLogI*:
assumes $b ** y \le x\ x < b ** y * b$ **shows** $oLog\ b\ x = y$
⟨*proof*⟩

lemma *ordinal-exp-oLog-le*: $[\![0 < x;\ oSuc\ 0 < b]\!] \Longrightarrow b ** (oLog\ b\ x) \le x$
⟨*proof*⟩

lemma *ordinal-less-exp-oLog*: $oSuc\ 0 < b \Longrightarrow x < b ** (oLog\ b\ x) * b$
⟨*proof*⟩

lemma *ordinal-oLog-less*: $[\![0 < x;\ oSuc\ 0 < b;\ x < b ** y]\!] \Longrightarrow oLog\ b\ x < y$
⟨*proof*⟩

lemma *ordinal-le-oLog*:
$[\![oSuc\ 0 < b;\ b ** y \le x]\!] \Longrightarrow y \le oLog\ b\ x$
⟨*proof*⟩

lemma *ordinal-oLogI2*:
assumes $oSuc\ 0 < b\ x = b ** y * q + r\ 0 < q\ q < b\ r < b ** y$
shows $oLog\ b\ x = y$
⟨*proof*⟩

lemma *ordinal-div-exp-oLog-less*: $oSuc\ 0 < b \Longrightarrow x\ div\ (b ** oLog\ b\ x) < b$
⟨*proof*⟩

lemma *ordinal-oLog-base-0*: $oLog\ 0\ x = 0$
⟨*proof*⟩

lemma *ordinal-oLog-base-1*: $oLog\ (oSuc\ 0)\ x = 0$
⟨*proof*⟩

lemma *ordinal-oLog-0*: $oLog\ b\ 0 = 0$
⟨*proof*⟩

lemma *ordinal-oLog-exp*: $oSuc\ 0 < b \Longrightarrow oLog\ b\ (b ** x) = x$
⟨*proof*⟩

lemma *ordinal-oLog-self*: $oSuc\ 0 < b \Longrightarrow oLog\ b\ b = oSuc\ 0$
⟨*proof*⟩

lemma *ordinal-mono-oLog*: $mono\ (oLog\ b)$
⟨*proof*⟩

lemma *ordinal-oLog-monoR*: $x \le y \Longrightarrow oLog\ b\ x \le oLog\ b\ y$
⟨*proof*⟩

lemma *ordinal-oLog-decreasing*: $oLog\ b\ x \le x$
⟨*proof*⟩

end

7 Fixed-points

theory *OrdinalFix*
imports *OrdinalInverse*
begin

primrec $iter :: nat \Rightarrow ('a \Rightarrow 'a) \Rightarrow ('a \Rightarrow 'a)$
where
$iter\ 0\ \ \ \ \ \ F\ x = x$
$|\ iter\ (Suc\ n)\ F\ x = F\ (iter\ n\ F\ x)$

definition
$oFix :: (ordinal \Rightarrow ordinal) \Rightarrow ordinal \Rightarrow ordinal$ **where**
$oFix\ F\ a = oLimit\ (\lambda n.\ iter\ n\ F\ a)$

lemma *oFix-fixed*:
assumes $continuous\ F\ a \le F\ a$
shows $F\ (oFix\ F\ a) = oFix\ F\ a$
⟨*proof*⟩

lemma *oFix-least*:
assumes $mono\ F\ F\ x = x\ a \le x$ **shows** $oFix\ F\ a \le x$
⟨*proof*⟩

lemma *mono-oFix*:
assumes $mono\ F$ **shows** $mono\ (oFix\ F)$
⟨*proof*⟩

lemma *less-oFixD*: $[\![x < oFix\ F\ a;\ mono\ F;\ F\ x = x]\!] \Longrightarrow x < a$
  $\langle proof \rangle$

lemma *less-oFixI*: $a < F\ a \Longrightarrow a < oFix\ F\ a$
  $\langle proof \rangle$

lemma *le-oFix*: $a \le oFix\ F\ a$
  $\langle proof \rangle$

lemma *le-oFix1*: $F\ a \le oFix\ F\ a$
  $\langle proof \rangle$

lemma *less-oFix-0D*:
  assumes $x < oFix\ F\ 0\ mono\ F$ **shows** $x < F\ x$
$\langle proof \rangle$

lemma *zero-less-oFix-eq*: $(0 < oFix\ F\ 0) = (0 < F\ 0)$
$\langle proof \rangle$

lemma *oFix-eq-self*:
  assumes $F\ a = a$ **shows** $oFix\ F\ a = a$
$\langle proof \rangle$

7.1 Derivatives of ordinal functions

The derivative of F enumerates all the fixed-points of F

definition
  $oDeriv :: (ordinal \Rightarrow ordinal) \Rightarrow ordinal \Rightarrow ordinal$ **where**
  $oDeriv\ F = ordinal\text{-}rec\ (oFix\ F\ 0)\ (\lambda p\ x.\ oFix\ F\ (oSuc\ x))$

lemma *oDeriv-0* [*simp*]:
  $oDeriv\ F\ 0 = oFix\ F\ 0$
  $\langle proof \rangle$

lemma *oDeriv-oSuc* [*simp*]:
  $oDeriv\ F\ (oSuc\ x) = oFix\ F\ (oSuc\ (oDeriv\ F\ x))$
  $\langle proof \rangle$

lemma *oDeriv-oLimit* [*simp*]:
  $oDeriv\ F\ (oLimit\ f) = oLimit\ (\lambda n.\ oDeriv\ F\ (f\ n))$
  $\langle proof \rangle$

lemma *oDeriv-fixed*:
  assumes $normal\ F$ **shows** $F\ (oDeriv\ F\ n) = oDeriv\ F\ n$
$\langle proof \rangle$

lemma *oDeriv-fixedD*: $[\![oDeriv\ F\ x = x;\ normal\ F]\!] \Longrightarrow F\ x = x$
  $\langle proof \rangle$

lemma *normal-oDeriv*: $normal\ (oDeriv\ F)$
⟨*proof*⟩

lemma *oDeriv-increasing*:
assumes *continuous F* **shows** $F\ n \leq oDeriv\ F\ n$
⟨*proof*⟩

lemma *oDeriv-total*:
assumes *normal F* $F\ x = x$ **shows** $\exists\, n.\ x = oDeriv\ F\ n$
⟨*proof*⟩

lemma *range-oDeriv*: $normal\ F \Longrightarrow range\ (oDeriv\ F) = \{x.\ F\ x = x\}$
⟨*proof*⟩

end

8 Omega

theory *OrdinalOmega*
imports *OrdinalFix*
begin

8.1 Embedding naturals in the ordinals

primrec *ordinal-of-nat* :: $nat \Rightarrow ordinal$
where
ordinal-of-nat 0 = *0*
| *ordinal-of-nat* (*Suc n*) = *oSuc* (*ordinal-of-nat n*)

lemma *strict-mono-ordinal-of-nat*: *strict-mono ordinal-of-nat*
⟨*proof*⟩

lemma *not-limit-ordinal-nat*: ¬ *limit-ordinal* (*ordinal-of-nat n*)
⟨*proof*⟩

lemma *ordinal-of-nat-eq* [*simp*]:
(*ordinal-of-nat x* = *ordinal-of-nat y*) = ($x = y$)
⟨*proof*⟩

lemma *ordinal-of-nat-less* [*simp*]:
(*ordinal-of-nat x* < *ordinal-of-nat y*) = ($x < y$)
⟨*proof*⟩

lemma *ordinal-of-nat-le* [*simp*]:
(*ordinal-of-nat x* ≤ *ordinal-of-nat y*) = ($x \leq y$)
⟨*proof*⟩

lemma *ordinal-of-nat-plus* [*simp*]:
ordinal-of-nat x + *ordinal-of-nat y* = *ordinal-of-nat* ($x + y$)

$\langle proof \rangle$

lemma *ordinal-of-nat-times* [*simp*]:
$ordinal\text{-}of\text{-}nat\ x * ordinal\text{-}of\text{-}nat\ y = ordinal\text{-}of\text{-}nat\ (x * y)$
$\langle proof \rangle$

lemma *ordinal-of-nat-exp* [*simp*]:
$ordinal\text{-}of\text{-}nat\ x ** ordinal\text{-}of\text{-}nat\ y = ordinal\text{-}of\text{-}nat\ (x \mathbin{\hat{}} y)$
$\langle proof \rangle$

lemma *oSuc-plus-ordinal-of-nat*:
$oSuc\ x + ordinal\text{-}of\text{-}nat\ n = oSuc\ (x + ordinal\text{-}of\text{-}nat\ n)$
$\langle proof \rangle$

lemma *less-ordinal-of-nat*:
$(x < ordinal\text{-}of\text{-}nat\ n) = (\exists\, m.\ x = ordinal\text{-}of\text{-}nat\ m \wedge m < n)$
$\langle proof \rangle$

lemma *le-ordinal-of-nat*:
$(x \leq ordinal\text{-}of\text{-}nat\ n) = (\exists\, m.\ x = ordinal\text{-}of\text{-}nat\ m \wedge m \leq n)$
$\langle proof \rangle$

8.2 Omega, the least limit ordinal

definition
$omega :: ordinal$ (‹ω›) **where**
$omega = oLimit\ ordinal\text{-}of\text{-}nat$

lemma *less-omegaD*: $x < \omega \Longrightarrow \exists\, n.\ x = ordinal\text{-}of\text{-}nat\ n$
$\langle proof \rangle$

lemma *omega-leI*: $\forall\, n.\ ordinal\text{-}of\text{-}nat\ n \leq x \Longrightarrow \omega \leq x$
$\langle proof \rangle$

lemma *nat-le-omega* [*simp*]: $ordinal\text{-}of\text{-}nat\ n \leq \omega$
$\langle proof \rangle$

lemma *nat-less-omega* [*simp*]: $ordinal\text{-}of\text{-}nat\ n < \omega$
$\langle proof \rangle$

lemma *zero-less-omega* [*simp*]: $0 < \omega$
$\langle proof \rangle$

lemma *limit-ordinal-omega*: $limit\text{-}ordinal\ \omega$
$\langle proof \rangle$

lemma *Least-limit-ordinal*: $(LEAST\ x.\ limit\text{-}ordinal\ x) = \omega$
$\langle proof \rangle$

lemma *range f* = *range ordinal-of-nat* $\Longrightarrow$ *oLimit f* = ω
⟨*proof*⟩

8.3 Arithmetic properties of ω

lemma *oSuc-less-omega* [*simp*]: (*oSuc* $x < \omega$) = ($x < \omega$)
⟨*proof*⟩

lemma *oSuc-plus-omega* [*simp*]: *oSuc* $x + \omega = x + \omega$
⟨*proof*⟩

lemma *ordinal-of-nat-plus-omega* [*simp*]:
ordinal-of-nat $n + \omega = \omega$
⟨*proof*⟩

lemma *ordinal-of-nat-times-omega* [*simp*]:
assumes $k > 0$ **shows** *ordinal-of-nat* $k * \omega = \omega$
⟨*proof*⟩

lemma *ordinal-plus-times-omega*: $x + x * \omega = x * \omega$
⟨*proof*⟩

lemma *ordinal-plus-absorb*: $x * \omega \leq y \Longrightarrow x + y = y$
⟨*proof*⟩

lemma *ordinal-less-plusL*:
assumes $y < x * \omega$ **shows** $y < x + y$
⟨*proof*⟩

lemma *ordinal-plus-absorb-iff*: $(x + y = y) = (x * \omega \leq y)$
⟨*proof*⟩

lemma *ordinal-less-plusL-iff*: $(y < x + y) = (y < x * \omega)$
⟨*proof*⟩

8.4 Additive principal ordinals

locale *additive-principal* =
fixes *a* :: *ordinal*
assumes *not-0*: $0 < a$
assumes *sum-eq*: $\bigwedge b.\ b < a \Longrightarrow b + a = a$

lemma (**in** *additive-principal*) *sum-less*:
$[\![x < a;\ y < a]\!] \Longrightarrow x + y < a$
⟨*proof*⟩

lemma (**in** *additive-principal*) *times-nat-less*:
$x < a \Longrightarrow x *$ *ordinal-of-nat* $n < a$
⟨*proof*⟩

lemma *not-additive-principal-0*: $\neg$ *additive-principal 0*
⟨*proof*⟩

lemma *additive-principal-oSuc*:
additive-principal (*oSuc a*) = (*a* = *0*)
⟨*proof*⟩

lemma *additive-principal-intro2* [*rule-format*]:
assumes *not-0*: *0* < *a* **and** *lessa*: ($\forall x<a.\ \forall y<a.\ x + y < a$)
shows *additive-principal a*
⟨*proof*⟩

lemma *additive-principal-1*: *additive-principal* (*oSuc 0*)
⟨*proof*⟩

lemma *additive-principal-omega*: *additive-principal* ω
⟨*proof*⟩

lemma *additive-principal-times-omega*:
assumes *0* < *x* **shows** *additive-principal* ($x * \omega$)
⟨*proof*⟩

lemma *additive-principal-oLimit*:
assumes $\forall n.$ *additive-principal* (*f n*)
shows *additive-principal* (*oLimit f*)
⟨*proof*⟩

lemma *additive-principal-omega-exp*: *additive-principal* (ω ** *x*)
⟨*proof*⟩

lemma (**in** *additive-principal*) *omega-exp*: $\exists x.\ a = \omega$ ** *x*
⟨*proof*⟩

lemma *additive-principal-iff*:
additive-principal a = ($\exists x.\ a = \omega$ ** *x*)
⟨*proof*⟩

lemma *absorb-omega-exp*:
x < ω ** *a* $\Longrightarrow$ *x* + ω ** *a* = ω ** *a*
⟨*proof*⟩

lemma *absorb-omega-exp2*: *a* < *b* $\Longrightarrow$ ω ** *a* + ω ** *b* = ω ** *b*
⟨*proof*⟩

8.5 Cantor normal form

lemma *cnf-lemma*: *x* > *0* $\Longrightarrow$ *x* − ω ** *oLog* ω *x* < *x*
⟨*proof*⟩

primrec *from-cnf* **where**
  from-cnf [] $= 0$
| *from-cnf* $(x \# xs) = \omega ** x +$ *from-cnf xs*

function *to-cnf* **where**
  [*simp del*]: *to-cnf* $x = ($*if* $x = 0$ *then* [] *else*
    oLog ω x # *to-cnf* $(x - \omega ** $ *oLog* ω $x))$
  ⟨*proof*⟩

termination ⟨*proof*⟩

lemma *to-cnf-0* [*simp*]: *to-cnf* 0 = []
  ⟨*proof*⟩

lemma *to-cnf-not-0*:
  $0 < x \Longrightarrow$ *to-cnf* $x =$ *oLog* ω x # *to-cnf* $(x - \omega **$ *oLog* ω $x)$
  ⟨*proof*⟩

lemma *to-cnf-eq-Cons*: *to-cnf* $x = a$ # *list* $\Longrightarrow a =$ *oLog* ω x
  ⟨*proof*⟩

lemma *to-cnf-inverse*: *from-cnf* (*to-cnf* x) $= x$
  ⟨*proof*⟩

primrec *normalize-cnf* **where**
  normalize-cnf-Nil: *normalize-cnf* [] = []
| *normalize-cnf-Cons*: *normalize-cnf* $(x \# xs) =$
    (*case xs of* [] ⇒ [x] | *y* # *ys* ⇒
      (*if* $x < y$ *then* [] *else* [x]) @ *normalize-cnf xs*)

lemma *from-cnf-normalize-cnf*: *from-cnf* (*normalize-cnf xs*) = *from-cnf xs*
⟨*proof*⟩

lemma *normalize-cnf-to-cnf*: *normalize-cnf* (*to-cnf* x) = *to-cnf* x
  ⟨*proof*⟩

alternate form of CNF

lemma *cnf2-lemma*:
  $0 < x \Longrightarrow x$ *mod* $\omega **$ *oLog* ω $x < x$
  ⟨*proof*⟩

primrec *from-cnf2* **where**
  from-cnf2 [] $= 0$
| *from-cnf2* $(x \# xs) = \omega **$ *fst* x $*$ *ordinal-of-nat* (*snd* x) + *from-cnf2 xs*

function *to-cnf2* **where**
  [*simp del*]: *to-cnf2* $x = ($*if* $x = 0$ *then* [] *else*

(*oLog* ω *x*, *inv ordinal-of-nat* (*x div* (ω ** *oLog* ω *x*)))
to-cnf2 (*x mod* (ω ** *oLog* ω *x*)))
⟨*proof*⟩

termination ⟨*proof*⟩

lemma *to-cnf2-0* [*simp*]: *to-cnf2 0* = []
⟨*proof*⟩

lemma *to-cnf2-not-0*:
$0 < x \Longrightarrow$ *to-cnf2 x* = (*oLog* ω *x*, *inv ordinal-of-nat* (*x div* (ω ** *oLog* ω *x*)))
to-cnf2 (*x mod* (ω ** *oLog* ω *x*))
⟨*proof*⟩

lemma *to-cnf2-eq-Cons*: *to-cnf2 x* = (*a*,*b*) # *list* $\Longrightarrow$ *a* = *oLog* ω *x*
⟨*proof*⟩

lemma *ordinal-of-nat-of-ordinal*:
$x < \omega \Longrightarrow$ *ordinal-of-nat* (*inv ordinal-of-nat x*) = *x*
⟨*proof*⟩

lemma *to-cnf2-inverse*: *from-cnf2* (*to-cnf2 x*) = *x*
⟨*proof*⟩

primrec *is-normalized2* **where**
is-normalized2-Nil: *is-normalized2* [] = *True*
| *is-normalized2-Cons*: *is-normalized2* (*x* # *xs*) =
(*case xs of* [] ⇒ *True* | *y* # *ys* ⇒ *fst y* < *fst x* ∧ *is-normalized2 xs*)

lemma *is-normalized2-to-cnf2*: *is-normalized2* (*to-cnf2 x*)
⟨*proof*⟩

8.6 Epsilon 0

definition *epsilon0* :: *ordinal* (‹ε_0›) **where**
epsilon0 = *oFix* ((**) ω) *0*

lemma *less-omega-exp*: $x < \varepsilon_0 \Longrightarrow x < \omega$ ** x
⟨*proof*⟩

lemma *omega-exp-epsilon0*: ω ** $\varepsilon_0 = \varepsilon_0$
⟨*proof*⟩

lemma *oLog-omega-less*: $[\![0 < x; x < \varepsilon_0]\!] \Longrightarrow$ *oLog* ω $x < x$
⟨*proof*⟩

end

9 Veblen Hierarchies

theory *OrdinalVeblen*
imports *OrdinalOmega*
begin

9.1 Closed, unbounded sets

locale *normal-set* =
fixes A :: *ordinal set*
assumes *closed*: $\bigwedge g.\ \forall n.\ g\ n \in A \Longrightarrow oLimit\ g \in A$
  and *unbounded*: $\bigwedge x.\ \exists y{\in}A.\ x < y$

lemma (**in** *normal-set*) *less-next*: $x < (LEAST\ z.\ z \in A \wedge x < z)$
  $\langle proof \rangle$

lemma (**in** *normal-set*) *mem-next*: $(LEAST\ z.\ z \in A \wedge x < z) \in A$
  $\langle proof \rangle$

lemma (**in** *normal*) *normal-set-range*: *normal-set* (*range* F)
$\langle proof \rangle$

lemma *oLimit-mem-INTER*:
  assumes *norm*: $\forall n.\ normal\text{-}set\ (A\ n)$
    and A: $\forall n.\ A\ (Suc\ n) \subseteq A\ n\ \forall n.\ f\ n \in A\ n$ **and** *mono* f
  shows $oLimit\ f \in (\bigcap n.\ A\ n)$
$\langle proof \rangle$

lemma *normal-set-INTER*:
  assumes *norm*: $\forall n.\ normal\text{-}set\ (A\ n)$ **and** A: $\forall n.\ A\ (Suc\ n) \subseteq A\ n$
  shows $normal\text{-}set\ (\bigcap n.\ A\ n)$
$\langle proof \rangle$

9.2 Ordering functions

There is a one-to-one correspondence between closed, unbounded sets of ordinals and normal functions on ordinals.

definition
  ordering :: (*ordinal set*) $\Rightarrow$ (*ordinal* $\Rightarrow$ *ordinal*) **where**
  $ordering\ A = ordinal\text{-}rec\ (LEAST\ z.\ z \in A)\ (\lambda p\ x.\ LEAST\ z.\ z \in A \wedge x < z)$

lemma *ordering-0*:
  $ordering\ A\ 0 = (LEAST\ z.\ z \in A)$
  $\langle proof \rangle$

lemma *ordering-oSuc*:
  $ordering\ A\ (oSuc\ x) = (LEAST\ z.\ z \in A \wedge ordering\ A\ x < z)$
  $\langle proof \rangle$

lemma (**in** *normal-set*) *normal-ordering*: *normal* (*ordering A*)
⟨*proof*⟩

lemma (**in** *normal-set*) *ordering-oLimit*: *ordering A* (*oLimit f*) = *oLimit* ($\lambda n.$ *ordering A* (*f n*))
⟨*proof*⟩

lemma (**in** *normal*) *ordering-range*: *ordering* (*range F*) = *F*
⟨*proof*⟩

lemma (**in** *normal-set*) *ordering-mem*: *ordering A x* $\in$ *A*
⟨*proof*⟩

lemma (**in** *normal-set*) *range-ordering*: *range* (*ordering A*) = *A*
⟨*proof*⟩

lemma *ordering-INTER-0*:
assumes *norm*: $\forall n.$ *normal-set* (*A n*) **and** *A*: $\forall n.$ *A* (*Suc n*) $\subseteq$ *A n*
shows *ordering* ($\bigcap n.$ *A n*) *0* = *oLimit* ($\lambda n.$ *ordering* (*A n*) *0*)
⟨*proof*⟩

9.3 Critical ordinals

definition
critical-set :: *ordinal set* $\Rightarrow$ *ordinal* $\Rightarrow$ *ordinal set* **where**
critical-set A =
ordinal-rec0 A ($\lambda p\ x.$ *x* $\cap$ *range* (*oDeriv* (*ordering x*))) ($\lambda f.$ $\bigcap n.$ *f n*)

lemma *critical-set-0* [*simp*]: *critical-set A 0* = *A*
⟨*proof*⟩

lemma *critical-set-oSuc-lemma*:
critical-set A (*oSuc n*) = *critical-set A n* $\cap$ *range* (*oDeriv* (*ordering* (*critical-set A n*)))
⟨*proof*⟩

lemma *omega-complete-INTER*: *omega-complete* ($\lambda x\ y.$ $y \subseteq x$) ($\lambda f.$ $\bigcap$ (*range f*))
⟨*proof*⟩

lemma *critical-set-oLimit*: *critical-set A* (*oLimit f*) = ($\bigcap n.$ *critical-set A* (*f n*))
⟨*proof*⟩

lemma *critical-set-mono*: $x \leq y \Longrightarrow$ *critical-set A y* $\subseteq$ *critical-set A x*
⟨*proof*⟩

lemma (**in** *normal-set*) *range-oDeriv-subset*: *range* (*oDeriv* (*ordering A*)) $\subseteq$ *A*
⟨*proof*⟩

lemma *normal-set-critical-set*: *normal-set A* $\Longrightarrow$ *normal-set* (*critical-set A x*)

⟨*proof*⟩

lemma *critical-set-oSuc*:
normal-set A $\Longrightarrow$ *critical-set A* (*oSuc x*) = *range* (*oDeriv* (*ordering* (*critical-set A x*)))
⟨*proof*⟩

9.4 Veblen hierarchy over a normal function

definition
oVeblen :: (*ordinal* $\Rightarrow$ *ordinal*) $\Rightarrow$ *ordinal* $\Rightarrow$ *ordinal* $\Rightarrow$ *ordinal* **where**
oVeblen F = (λ*x*. *ordering* (*critical-set* (*range F*) *x*))

lemma (**in** *normal*) *oVeblen-0*: *oVeblen F 0* = *F*
⟨*proof*⟩

lemma (**in** *normal*) *oVeblen-oSuc*: *oVeblen F* (*oSuc x*) = *oDeriv* (*oVeblen F x*)
⟨*proof*⟩

lemma (**in** *normal*) *oVeblen-oLimit*:
oVeblen F (*oLimit f*) = *ordering* ($\bigcap$*n*. *range* (*oVeblen F* (*f n*)))
⟨*proof*⟩

lemma (**in** *normal*) *normal-oVeblen*: *normal* (*oVeblen F x*)
⟨*proof*⟩

lemma (**in** *normal*) *continuous-oVeblen-0*: *continuous* (λ*x*. *oVeblen F x 0*)
⟨*proof*⟩

lemma (**in** *normal*) *oVeblen-oLimit-0*:
oVeblen F (*oLimit f*) *0* = *oLimit* (λ*n*. *oVeblen F* (*f n*) *0*)
⟨*proof*⟩

lemma (**in** *normal*) *normal-oVeblen-0*:
assumes *0* < *F 0* **shows** *normal* (λ*x*. *oVeblen F x 0*)
⟨*proof*⟩

lemma (**in** *normal*) *range-oVeblen*:
range (*oVeblen F x*) = *critical-set* (*range F*) *x*
⟨*proof*⟩

lemma (**in** *normal*) *range-oVeblen-subset*:
$x \le y \Longrightarrow$ *range* (*oVeblen F y*) $\subseteq$ *range* (*oVeblen F x*)
⟨*proof*⟩

lemma (**in** *normal*) *oVeblen-fixed*:
assumes $x<y$
shows *oVeblen F x* (*oVeblen F y a*) = *oVeblen F y a*
⟨*proof*⟩

lemma (**in** *normal*) *critical-set-fixed*:
  assumes $0 < z$
  shows $range\ (oVeblen\ F\ z) = \{x.\ \forall y{<}z.\ oVeblen\ F\ y\ x = x\}$ (**is** $?L = ?R$)
⟨*proof*⟩

9.5 Veblen hierarchy over $\lambda x.\ 1 + x$

lemma *oDeriv-id*: $oDeriv\ id = id$
⟨*proof*⟩

lemma *oFix-plus*: $oFix\ (\lambda x.\ a + x)\ 0 = a * \omega$
⟨*proof*⟩

lemma *oDeriv-plus*: $oDeriv\ ((+)\ a) = ((+)\ (a * \omega))$
⟨*proof*⟩

lemma *oVeblen-1-plus*: $oVeblen\ ((+)\ 1)\ x = ((+)\ (\omega ** x))$
  ⟨*proof*⟩

end