

# Reducing Rewrite Properties to Properties on Ground Terms\*

Alexander Lochmann

August 7, 2022

## Abstract

This AFP entry relates important rewriting properties between the set of terms and the set of ground terms induced by a given signature. The properties considered are confluence, strong/local confluence, the normal form property, unique normal forms with respect to reduction and conversion, commutation, conversion equivalence, and normalization equivalence.

## Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Preliminaries</b>	<b>4</b>
2.1	Additional operations on terms and positions . . . . .	4
2.1.1	Linearity . . . . .	4
2.1.2	Positions induced by contexts, by variables and by given subterms . . . . .	4
2.1.3	Replacing functions symbols that aren't specified in the signature by variables . . . . .	4
2.1.4	Replace term at a given position in contexts . . . . .	5
2.1.5	Multihole context closure of a term relation as inductive set . . . . .	5
2.2	Destruction and introduction of <i>all-ctxt-closed</i> . . . . .	5
2.3	Lemmas for <i>poss</i> and ordering of positions . . . . .	6
2.4	Lemmas for $( -)$ and <i>replace-term-at</i> . . . . .	7
2.5	<i>term-to-sig</i> invariants and distributions . . . . .	10
2.6	Misc . . . . .	11

---

\*Supported by FWF (Austrian Science Fund) projects P30301.

<b>3</b>	<b>Rewriting</b>	<b>14</b>
3.1	Basic rewrite definitions . . . . .	14
3.1.1	Rewrite steps with implicit signature declaration (en- coded in the type) . . . . .	14
3.1.2	Restrict relations to terms induced by a given signature	14
3.1.3	Rewriting under a given signature/restricted to ground terms . . . . .	14
3.1.4	Rewriting sequences involving a root step . . . . .	14
3.2	Monotonicity laws . . . . .	14
3.3	Introduction, elimination, and destruction rules for <i>sig-step</i> , <i>rstep</i> , <i>rrstep</i> , <i>srrstep</i> , and <i>srstep</i> . . . . .	15
3.3.1	Transitive and reflexive closure distribution over <i>sig-step</i>	17
3.3.2	Distributivity laws . . . . .	19
3.4	Substitution closure of <i>srstep</i> . . . . .	20
3.5	Context closure of <i>srstep</i> . . . . .	21
3.6	Removing/Replacing constants in a rewrite sequence that do not appear in the rewrite system . . . . .	23
3.6.1	Removal lemma applied to various rewrite relations . . . . .	28
<b>4</b>	<b>Confluence related rewriting properties</b>	<b>31</b>
4.1	Confluence related ARS properties . . . . .	31
4.2	Signature closure of relation to model multihole context closure	32
4.3	Specific results about rewriting under a linear variable-separated system . . . . .	49
4.4	Specific results about rewriting under a ground system . . . . .	59
4.5	funas . . . . .	61
<b>5</b>	<b>Reducing Rewrite Properties to Properties on Ground Terms over Left-Linear Right-Ground Systems</b>	<b>62</b>
<b>6</b>	<b>LLRG results</b>	<b>62</b>
6.1	Specialized for monadic signature . . . . .	68
<b>7</b>	<b>Reducing Rewrite Properties to Properties on Ground Terms over Ground Systems</b>	<b>71</b>
<b>8</b>	<b>Reducing Rewrite Properties to Properties on Ground Terms over Linear Variable-Separated Systems</b>	<b>77</b>
<b>9</b>	<b>Linear-variable separated results</b>	<b>78</b>
9.1	Specialized for monadic signature . . . . .	87

# 1 Introduction

Rewriting is an abstract model of computation. Among other things, it studies important properties including the following:

CR:	$\forall s \forall t \forall u (s \rightarrow^* t \wedge s \rightarrow^* u \implies t \downarrow u)$	confluence
SCR:	$\forall s \forall t \forall u (s \rightarrow t \wedge s \rightarrow u \implies \exists v (t \rightarrow^= v \wedge u \rightarrow^* v))$	strong confluence
WCR:	$\forall s \forall t \forall u (s \rightarrow t \wedge s \rightarrow u \implies t \downarrow u)$	local confluence
NFP:	$\forall s \forall t \forall u (s \rightarrow^* t \wedge s \rightarrow^! u \implies t \rightarrow^! u)$	normal form property
UNR:	$\forall s \forall t \forall u (s \rightarrow^! t \wedge s \rightarrow^! u \implies t = u)$	unique normal forms with respect to reduction
UNC:	$\forall t \forall u (t \leftrightarrow^* u \wedge \text{NF}(t) \wedge \text{NF}(u) \implies t = u)$	unique normal forms with respect to conversion

We also consider the following properties involving two TRSs  $\mathcal{R}$  and  $\mathcal{S}$ :

COM:	$\forall s \forall t \forall u (s \rightarrow_{\mathcal{R}}^* t \wedge s \rightarrow_{\mathcal{S}}^* u \implies \exists v (t \rightarrow_{\mathcal{S}}^* v \wedge u \rightarrow_{\mathcal{R}}^* v))$	commutation
CE:	$\forall s \forall t (s \leftrightarrow_{\mathcal{R}}^* t \iff s \leftrightarrow_{\mathcal{S}}^* t)$	conversion equivalence
NE:	$\forall s \forall t (s \rightarrow_{\mathcal{R}}^! t \iff s \rightarrow_{\mathcal{S}}^! t)$	normalization equivalence

An interesting observation is that for each of these properties there exists a rewrite system that satisfies the property when restricted to ground terms but not when arbitrary terms are allowed. Consider the left-linear right-ground TRS  $\mathcal{R}$  consisting of the rules

$$a \rightarrow b \qquad f(a, x) \rightarrow b \qquad f(b, b) \rightarrow b$$

over the signature  $\mathcal{F} = \{a, b, f\}$ . It is ground-confluent because every ground term in  $\mathcal{T}(\mathcal{F})$  rewrites to  $b$ . Confluence does not hold; the term  $f(a, x)$  rewrites to the different normal forms  $b$  and  $f(b, x)$ .

In this AFP entry, properties on arbitrary terms are reduced to the corresponding properties on ground terms, for left-linear right-ground rewrite systems and for linear variable-separated systems. To do this, I formalized fundamental term rewriting operations that include the root step and the one step rewriting relations. Also, I added definitions for conversion equivalence, normalization equivalence, strong confluence and the normal form property extending the list of important rewriting properties of the AFP entry ‘‘Abstract Rewriting’’ [1].

Rewrite sequences that contain a root step play an important role in the formalization. The table of contents should give the reader a good overview of the content of this entry.

## 2 Preliminaries

**theory** *Terms-Positions*  
**imports** *Regular-Tree-Relations.Ground-Terms*  
**begin**

### 2.1 Additional operations on terms and positions

#### 2.1.1 Linearity

**fun** *linear-term* :: ('f, 'v) term  $\Rightarrow$  bool **where**  
*linear-term* (Var -) = True |  
*linear-term* (Fun f ts) = (is-partition (map vars-term ts)  $\wedge$  ( $\forall t \in \text{set } ts.$  *linear-term* t))  
**abbreviation** *linear-sys*  $\mathcal{R} \equiv \forall (l, r) \in \mathcal{R}. \text{linear-term } l \wedge \text{linear-term } r$

#### 2.1.2 Positions induced by contexts, by variables and by given subterms

**definition** *possc* C = {p | p t. p  $\in$  poss C(t)}  
**definition** *varposs* s = {p | p. p  $\in$  poss s  $\wedge$  is-Var (s |- p)}  
**definition** *poss-of-term* u t = {p. p  $\in$  poss t  $\wedge$  t |- p = u}

#### 2.1.3 Replacing functions symbols that aren't specified in the signature by variables

**definition** *funas-rel*  $\mathcal{R} = (\bigcup (l, r) \in \mathcal{R}. \text{funas-term } l \cup \text{funas-term } r)$

**fun** *term-to-sig* **where**  
*term-to-sig*  $\mathcal{F}$  v (Var x) = Var x  
| *term-to-sig*  $\mathcal{F}$  v (Fun f ts) =  
(if (f, length ts)  $\in$   $\mathcal{F}$  then Fun f (map (term-to-sig  $\mathcal{F}$  v) ts) else Var v)

**fun** *ctxt-well-def-hole-path* **where**  
*ctxt-well-def-hole-path*  $\mathcal{F}$  Hole  $\longleftrightarrow$  True  
| *ctxt-well-def-hole-path*  $\mathcal{F}$  (More f ss C ts)  $\longleftrightarrow$  (f, Suc (length ss + length ts))  $\in$   $\mathcal{F} \wedge \text{ctxt-well-def-hole-path } \mathcal{F} C$

**fun** *inv-const-ctxt* **where**  
*inv-const-ctxt*  $\mathcal{F}$  v Hole = Hole  
| *inv-const-ctxt*  $\mathcal{F}$  v ((More f ss C ts))  
= (More f (map (term-to-sig  $\mathcal{F}$  v) ss) (inv-const-ctxt  $\mathcal{F}$  v C) (map (term-to-sig  $\mathcal{F}$  v) ts))

**fun** *inv-const-ctxt'* **where**  
*inv-const-ctxt'*  $\mathcal{F}$  v Hole = Var v  
| *inv-const-ctxt'*  $\mathcal{F}$  v ((More f ss C ts))  
= (if (f, Suc (length ss + length ts))  $\in$   $\mathcal{F}$  then Fun f (map (term-to-sig  $\mathcal{F}$  v) ss @ inv-const-ctxt'  $\mathcal{F}$  v C # map (term-to-sig  $\mathcal{F}$  v) ts) else Var v)

### 2.1.4 Replace term at a given position in contexts

**fun** *replace-term-context-at* :: ('f, 'v) *ctxt*  $\Rightarrow$  *pos*  $\Rightarrow$  ('f, 'v) *term*  $\Rightarrow$  ('f, 'v) *ctxt*  
 (-[-  $\leftarrow$  -]<sub>C</sub> [1000, 0] 1000) **where**  
*replace-term-context-at*  $\square$  *p u* =  $\square$   
 | *replace-term-context-at* (More *f ss C ts*) (*i* # *ps*) *u* =  
   (if *i* < length *ss* then More *f* (*ss*[*i* := (*ss* ! *i*)[*ps*  $\leftarrow$  *u*]]) *C ts*  
   else if *i* = length *ss* then More *f ss* (*replace-term-context-at* *C ps u*) *ts*  
   else More *f ss C* (*ts*[(*i* - Suc (length *ss*)) := (*ts* ! (*i* - Suc (length *ss*)))] [*ps*  $\leftarrow$  *u*]))

**abbreviation** *constT c*  $\equiv$  *Fun c* []

### 2.1.5 Multihole context closure of a term relation as inductive set

**definition** *all-ctxt-closed* **where**

*all-ctxt-closed* *F r*  $\iff$  ( $\forall$  *f ts ss*. (*f*, length *ss*)  $\in$  *F*  $\longrightarrow$  length *ts* = length *ss*  $\longrightarrow$   
 ( $\forall$  *i*. *i* < length *ts*  $\longrightarrow$  (*ts* ! *i*, *ss* ! *i*)  $\in$  *r*)  $\longrightarrow$   
 ( $\forall$  *i*. *i* < length *ts*  $\longrightarrow$  funas-term (*ts* ! *i*)  $\cup$  funas-term (*ss* ! *i*)  $\subseteq$  *F*)  $\longrightarrow$  (*Fun*  
*f ts*, *Fun f ss*)  $\in$  *r*)  $\wedge$   
 ( $\forall$  *x*. (*Var x*, *Var x*)  $\in$  *r*)

## 2.2 Destruction and introduction of *all-ctxt-closed*

**lemma** *all-ctxt-closedD*: *all-ctxt-closed* *F r*  $\implies$  (*f*, length *ss*)  $\in$  *F*  $\implies$  length *ts* = length *ss*

$\implies$  [ $\wedge$  *i*. *i* < length *ts*  $\implies$  (*ts* ! *i*, *ss* ! *i*)  $\in$  *r* ]  
 $\implies$  [ $\wedge$  *i*. *i* < length *ts*  $\implies$  funas-term (*ts* ! *i*)  $\subseteq$  *F* ]  
 $\implies$  [ $\wedge$  *i*. *i* < length *ts*  $\implies$  funas-term (*ss* ! *i*)  $\subseteq$  *F* ]  
 $\implies$  (*Fun f ts*, *Fun f ss*)  $\in$  *r*

**unfolding** *all-ctxt-closed-def* **by** *auto*

**lemma** *trans-ctxt-sig-imp-all-ctxt-closed*: **assumes** *tran*: *trans r*

**and** *refl*:  $\wedge$  *t*. funas-term *t*  $\subseteq$  *F*  $\implies$  (*t*, *t*)  $\in$  *r*

**and** *ctxt*:  $\wedge$  *C s t*. funas-ctxt *C*  $\subseteq$  *F*  $\implies$  funas-term *s*  $\subseteq$  *F*  $\implies$  funas-term *t*  $\subseteq$  *F*  
 $\implies$  (*s*, *t*)  $\in$  *r*  $\implies$  (*C*  $\langle$  *s*  $\rangle$ , *C*  $\langle$  *t*  $\rangle$ )  $\in$  *r*

**shows** *all-ctxt-closed* *F r* **unfolding** *all-ctxt-closed-def*

**proof** (*rule*, *intro* *allI impI*)

**fix** *f ts ss*

**assume** *f*: (*f*, length *ss*)  $\in$  *F* **and**

*l*: length *ts* = length *ss* **and**

*steps*:  $\forall$  *i* < length *ts*. (*ts* ! *i*, *ss* ! *i*)  $\in$  *r* **and**

*sig*:  $\forall$  *i* < length *ts*. funas-term (*ts* ! *i*)  $\cup$  funas-term (*ss* ! *i*)  $\subseteq$  *F*

**from** *sig* **have** *sig-ts*:  $\wedge$  *t*. *t*  $\in$  set *ts*  $\implies$  funas-term *t*  $\subseteq$  *F* **unfolding** *set-conv-nth*  
**by** *auto*

**let** *?p* =  $\lambda$  *ss*. (*Fun f ts*, *Fun f ss*)  $\in$  *r*  $\wedge$  funas-term (*Fun f ss*)  $\subseteq$  *F*

**let** *?r* =  $\lambda$  *xsi ysi*. (*xsi*, *ysi*)  $\in$  *r*  $\wedge$  funas-term *ysi*  $\subseteq$  *F*

**have** *init*: *?p ts* **by** (*rule* *conjI*[*OF refl*], *insert f sig-ts l*, *auto*)

**have** *?p ss*

```

proof (rule parallel-list-update[where  $p = ?p$  and  $r = ?r$ , OF - HOL.refl init
l[symmetric]])
  fix  $xs\ i\ y$ 
  assume  $len: length\ xs = length\ ts$ 
    and  $i: i < length\ ts$ 
    and  $r: ?r\ (xs\ !\ i)\ y$ 
    and  $p: ?p\ xs$ 
  let  $?C = More\ f\ (take\ i\ xs)\ Hole\ (drop\ (Suc\ i)\ xs)$ 
  have  $id1: Fun\ f\ xs = ?C\ \langle\ xs\ !\ i\ \rangle$  using id-take-nth-drop[OF\ i[folded\ len]] by
simp
  have  $id2: Fun\ f\ (xs[i := y]) = ?C\ \langle\ y\ \rangle$  using upd-conv-take-nth-drop[OF\ i[folded\ len]] by simp
  from  $p[unfolded\ id1]$  have  $C: funas-ctxt\ ?C \subseteq F$  and  $xi: funas-term\ (xs\ !\ i) \subseteq F$  by auto
  from  $r$  have  $funas-term\ y \subseteq F\ (xs\ !\ i, y) \in r$  by auto
  with  $ctxt[OF\ C\ xi\ this]\ C$  have  $r: (Fun\ f\ xs, Fun\ f\ (xs[i := y])) \in r$ 
    and  $f: funas-term\ (Fun\ f\ (xs[i := y])) \subseteq F$  unfolding  $id1\ id2$  by auto
  from  $p\ r\ tran$  have  $(Fun\ f\ ts, Fun\ f\ (xs[i := y])) \in r$  unfolding trans-def by
auto
  with  $f$ 
  show  $?p\ (xs[i := y])$  by auto
  qed (insert\ sig\ steps, auto)
  then show  $(Fun\ f\ ts, Fun\ f\ ss) \in r ..$ 
qed (insert\ refl, auto)

```

### 2.3 Lemmas for *poss* and ordering of positions

```

lemma subst-poss-mono:  $poss\ s \subseteq poss\ (s \cdot \sigma)$ 
by (induct\ s) force+

```

```

lemma par-pos-prefix [simp]:
   $(i \# p) \perp (i \# q) \implies p \perp q$ 
by (simp\ add: par-Cons-iff)

```

```

lemma pos-diff-itself [simp]:  $p -_p p = []$ 
by (simp\ add: pos-diff-def)

```

```

lemma pos-less-eq-append-diff [simp]:
   $p \leq_p q \implies p @ (q -_p p) = q$ 
by (metis\ option.sel\ pos-diff-def\ position-less-eq-def\ remove-prefix-append)

```

```

lemma pos-diff-append-itself [simp]:  $(p @ q) -_p p = q$ 
by (simp\ add: pos-diff-def\ remove-prefix-append)

```

```

lemma poss-pos-diffI:
   $p \leq_p q \implies q \in poss\ s \implies q -_p p \in poss\ (s \mid- p)$ 
using poss-append-poss by fastforce

```

```

lemma less-eq-poss-append-itself [simp]:  $p \leq_p (p @ q)$ 

```

**using** *position-less-eq-def* **by** *blast*

**lemma** *poss-ctxt-apply* [*simp*]:  
 $hole\text{-}pos\ C\ @\ p \in\ poss\ C\langle s \rangle \longleftrightarrow p \in\ poss\ s$   
**by** (*induct C*) *auto*

**lemma** *pos-replace-at-pres*:  
 $p \in\ poss\ s \implies p \in\ poss\ s[p \leftarrow t]$   
**proof** (*induct p arbitrary: s*)  
  **case** (*Cons i p*)  
  **show** *?case using Cons(1)[of args s ! i] Cons(2-)*  
  **by** (*cases s*) *auto*  
**qed** *auto*

**lemma** *par-pos-replace-pres*:  
 $p \in\ poss\ s \implies p \perp q \implies p \in\ poss\ s[q \leftarrow t]$   
**proof** (*induct p arbitrary: s q*)  
  **case** (*Cons i p*)  
  **show** *?case using Cons(1)[of args s ! i tl q] Cons(2-)*  
  **by** (*cases s; cases q*) (*auto simp add: nth-list-update par-Cons-iff*)  
**qed** *auto*

**lemma** *poss-of-termE* [*elim*]:  
  **assumes**  $p \in\ poss\text{-of-term}\ u\ s$   
  **and**  $p \in\ poss\ s \implies s \mid\text{-}\ p = u \implies P$   
  **shows**  $P$  **using** *assms unfolding poss-of-term-def*  
  **by** *blast*

**lemma** *poss-of-term-Cons*:  
 $i \# p \in\ poss\text{-of-term}\ u\ (Fun\ f\ ts) \implies p \in\ poss\text{-of-term}\ u\ (ts\ !\ i)$   
**unfolding** *poss-of-term-def* **by** *auto*

**lemma** *poss-of-term-const-ctxt-apply*:  
  **assumes**  $p \in\ poss\text{-of-term}\ (constT\ c)\ C\langle s \rangle$   
  **shows**  $p \perp (hole\text{-}pos\ C) \vee (hole\text{-}pos\ C) \leq_p p$  **using** *assms*  
**proof** (*induct p arbitrary: C*)  
  **case** *Nil* **then show** *?case*  
  **by** (*cases C*) *auto*  
**next**  
  **case** (*Cons i p*) **then show** *?case*  
  **by** (*cases C*) (*fastforce simp add: par-Cons-iff dest!: poss-of-term-Cons*)  
**qed**

## 2.4 Lemmas for $(\mid\text{-})$ and *replace-term-at*

**lemma** *subt-at-append-dist*:  
 $p\ @\ q \in\ poss\ s \implies s \mid\text{-}\ (p\ @\ q) = (s \mid\text{-}\ p) \mid\text{-}\ q$   
**proof** (*induct p arbitrary: s*)  
  **case** (*Cons i p*) **then show** *?case*

by (cases s) auto  
qed auto

**lemma** *ctxt-apply-term-subst-at-hole-pos* [simp]:  
 $C\langle s \rangle \mid- (\text{hole-pos } C \text{ @ } q) = s \mid- q$   
 by (induct C) auto

**lemma** *subst-subst-at-dist*:  
 $p \in \text{poss } s \implies s \cdot \sigma \mid- p = s \mid- p \cdot \sigma$   
**proof** (induct p arbitrary: s)  
 case (Cons i p) then show ?case  
 by (cases s) auto  
 qed auto

**lemma** *replace-term-at-subst-at-id* [simp]:  $s[p \leftarrow (s \mid- p)] = s$   
**proof** (induct p arbitrary: s)  
 case (Cons i p) then show ?case  
 by (cases s) auto  
 qed auto

**lemma** *replace-term-at-same-pos* [simp]:  
 $s[p \leftarrow u][p \leftarrow t] = s[p \leftarrow t]$   
 using *position-less-refl replace-term-at-above* by blast

— Replacement at under substitution

**lemma** *subt-at-vars-term*:  
 $p \in \text{poss } s \implies s \mid- p = \text{Var } x \implies x \in \text{vars-term } s$   
 by (metis UnCI ctxt-at-pos-subst-at-id term.set-intros(3) vars-term-ctxt-apply)

**lemma** *linear-term-varposs-subst-replace-term*:  
 $\text{linear-term } s \implies p \in \text{varposs } s \implies p \leq_p q \implies$   
 $(s \cdot \sigma)[q \leftarrow u] = s \cdot (\lambda x. \text{if } \text{Var } x = s \mid- p \text{ then } (\sigma x)[q \leftarrow_p p \leftarrow u] \text{ else } (\sigma x))$   
**proof** (induct q arbitrary: s p)  
 case (Cons i q)  
 show ?case using Cons(1)[of args s ! i tl p] Cons(2-)  
 by (cases s) (auto simp: varposs-def nth-list-update term-subst-eq-conv  
 is-partition-alt is-partition-alt-def disjoint-iff subt-at-vars-term intro!: nth-equalityI)  
 qed (auto simp: varposs-def)

— Replacement at context parallel to the hole position

**lemma** *par-hole-pos-replace-term-context-at*:  
 $p \perp \text{hole-pos } C \implies C\langle s \rangle[p \leftarrow u] = (C[p \leftarrow u]_C)\langle s \rangle$   
**proof** (induct p arbitrary: C)  
 case (Cons i p)  
 from Cons(2) obtain f ss D ts where [simp]:  $C = \text{More } f \text{ ss } D \text{ ts}$  by (cases C)  
 auto  
 show ?case using Cons(1)[of D] Cons(2)  
 by (auto simp: list-update-append nth-append-Cons minus-nat.simps(2) split:



*nat.splits*)  
**qed** *auto*

**lemma** *par-pos-replace-term-at*:  
 $p \in \text{poss } s \implies p \perp q \implies s[q \leftarrow t] \mid\!-\! p = s \mid\!-\! p$   
**proof** (*induct p arbitrary: s q*)  
  **case** (*Cons i p*)  
  **show** *?case using Cons(1)[of args s ! i tl q] Cons(2-)*  
  **by** (*cases s; cases q*) (*auto, metis nth-list-update par-Cons-iff*)  
**qed** *auto*

**lemma** *less-eq-subt-at-replace*:  
 $p \in \text{poss } s \implies p \leq_p q \implies s[q \leftarrow t] \mid\!-\! p = (s \mid\!-\! p)[q \!-\!_p p \leftarrow t]$   
**proof** (*induct p arbitrary: s q*)  
  **case** (*Cons i p*)  
  **show** *?case using Cons(1)[of args s ! i tl q] Cons(2-)*  
  **by** (*cases s; cases q*) *auto*  
**qed** *auto*

**lemma** *greater-eq-subt-at-replace*:  
 $p \in \text{poss } s \implies q \leq_p p \implies s[q \leftarrow t] \mid\!-\! p = t \mid\!-\! (p \!-\!_p q)$   
**proof** (*induct p arbitrary: s q*)  
  **case** (*Cons i p*)  
  **show** *?case using Cons(1)[of args s ! i tl q] Cons(2-)*  
  **by** (*cases s; cases q*) *auto*  
**qed** *auto*

**lemma** *replace-subterm-at-itself [simp]*:  
 $s[p \leftarrow (s \mid\!-\! p)[q \leftarrow t]] = s[p \text{ @ } q \leftarrow t]$   
**proof** (*induct p arbitrary: s*)  
  **case** (*Cons i p*)  
  **show** *?case using Cons(1)[of args s ! i]*  
  **by** (*cases s*) *auto*  
**qed** *auto*

**lemma** *hole-pos-replace-term-at [simp]*:  
 $\text{hole-pos } C \leq_p p \implies C\langle s \rangle[p \leftarrow u] = C\langle s[p \!-\!_p \text{hole-pos } C \leftarrow u] \rangle$   
**proof** (*induct C arbitrary: p*)  
  **case** (*More f ss C ts*) **then show** *?case*  
  **by** (*cases p*) *auto*  
**qed** *auto*

**lemma** *ctxt-of-pos-term-apply-replace-at-ident*:  
**assumes**  $p \in \text{poss } s$   
**shows**  $(\text{ctxt-at-pos } s \text{ } p)\langle t \rangle = s[p \leftarrow t]$   
**using** *assms*

**proof** (*induct p arbitrary: s*)  
**case** (*Cons i p*)  
**show** *?case using Cons(1)[of args s ! i] Cons(2-)*  
**by** (*cases s*) (*auto simp: nth-append-Cons intro!: nth-equalityI*)  
**qed** *auto*

**lemma** *ctxt-apply-term-replace-term-hole-pos [simp]:*  
 $C\langle s \rangle[\text{hole-pos } C @ q \leftarrow u] = C\langle s[q \leftarrow u] \rangle$   
**by** (*simp add: pos-diff-def position-less-eq-def remove-prefix-append*)

**lemma** *ctxt-apply-subst-at-hole-pos [simp]:*  $C\langle s \rangle \mid\text{- hole-pos } C = s$   
**by** (*induct C*) *auto*

**lemma** *subst-at-imp-supteq':*  
**assumes**  $p \in \text{poss } s$  **and**  $s \mid\text{-}p = t$  **shows**  $s \supseteq t$  **using** *assms*  
**proof** (*induct p arbitrary: s*)  
**case** (*Cons i p*)  
**from** *Cons(2-)* **show** *?case using Cons(1)[of args s ! i]*  
**by** (*cases s*) *force+*  
**qed** *auto*

**lemma** *subst-at-imp-supteq:*  
**assumes**  $p \in \text{poss } s$  **shows**  $s \supseteq s \mid\text{-}p$   
**proof** –  
**have**  $s \mid\text{-}p = s \mid\text{-}p$  **by** *auto*  
**with** *assms* **show** *?thesis* **by** (*rule subst-at-imp-supteq'*)  
**qed**

## 2.5 term-to-sig invariants and distributions

**lemma** *funas-term-term-to-sig [simp]:*  $\text{funas-term } (\text{term-to-sig } \mathcal{F} \ v \ t) \subseteq \mathcal{F}$   
**by** (*induct t*) *auto*

**lemma** *term-to-sig-id [simp]:*  
 $\text{funas-term } t \subseteq \mathcal{F} \implies \text{term-to-sig } \mathcal{F} \ v \ t = t$   
**by** (*induct t*) (*auto simp add: UN-subset-iff map-idI*)

**lemma** *term-to-sig-subst-sig [simp]:*  
 $\text{funas-term } t \subseteq \mathcal{F} \implies \text{term-to-sig } \mathcal{F} \ v \ (t \cdot \sigma) = t \cdot (\lambda x. \text{term-to-sig } \mathcal{F} \ v \ (\sigma \ x))$   
**by** (*induct t*) *auto*

**lemma** *funas-ctxt-ctxt-inv-const-ctxt-ind [simp]:*  
 $\text{funas-ctxt } C \subseteq \mathcal{F} \implies \text{inv-const-ctxt } \mathcal{F} \ v \ C = C$   
**by** (*induct C*) (*auto simp add: UN-subset-iff intro!: nth-equalityI*)

**lemma** *term-to-sig-ctxt-apply [simp]:*  
 $\text{ctxt-well-def-hole-path } \mathcal{F} \ C \implies \text{term-to-sig } \mathcal{F} \ v \ C\langle s \rangle = (\text{inv-const-ctxt } \mathcal{F} \ v \ C)\langle \text{term-to-sig } \mathcal{F} \ v \ s \rangle$   
**by** (*induct C*) *auto*

**lemma** *term-to-sig-ctxt-apply'* [*simp*]:  
 $\neg \text{ctxt-well-def-hole-path } \mathcal{F} \ C \implies \text{term-to-sig } \mathcal{F} \ v \ C \langle s \rangle = \text{inv-const-ctxt}' \ \mathcal{F} \ v \ C$   
**by** (*induct C*) *auto*

**lemma** *funas-ctxt-ctxt-well-def-hole-path*:  
 $\text{funas-ctxt } C \subseteq \mathcal{F} \implies \text{ctxt-well-def-hole-path } \mathcal{F} \ C$   
**by** (*induct C*) *auto*

## 2.6 Misc

**lemma** *funas-term-subt-at*:  
 $(f, n) \in \text{funas-term } t \implies (\exists \ p \ ts. \ p \in \text{poss } t \wedge t \mid\!-\ p = \text{Fun } f \ ts \wedge \text{length } ts = n)$   
**proof** (*induct t*)  
**case** (*Fun g ts*) **note** *IH = this*  
**show** *?case*  
**proof** (*cases g = f \wedge \text{length } ts = n*)  
**case** *False*  
**then obtain** *i* **where** *i: i < \text{length } ts* (*f, n*)  $\in \text{funas-term } (ts \ ! \ i)$  **using** *IH(2)*  
**using** *in-set-idx* **by** *force*  
**from** *IH(1)[OF nth-mem[OF this(1)] this(2)]* **show** *?thesis* **using** *i(1)*  
**by** (*metis poss-Cons-poss subt-at.simps(2) term.sel(4)*)  
**qed** *auto*  
**qed** *simp*

**lemma** *finite-poss: finite (poss s)*  
**proof** (*induct s*)  
**case** (*Fun f ts*)  
**have**  $\text{poss } (\text{Fun } f \ ts) = \text{insert } [] \ (\bigcup \ (\text{set } (\text{map2 } (\lambda \ i \ p. \ ((\#) \ i) \ 'p) \ [0..< \text{length } ts]) \ (\text{map } \text{poss } ts))))$   
**by** (*auto simp: image-iff set-zip split: prod.splits*)  
**then show** *?case* **using** *Fun*  
**by** (*auto simp del: poss.simps dest!: set-zip-rightD*)  
**qed** *simp*

**lemma** *finite-varposs: finite (varposs s)*  
**by** (*intro finite-subset[of varposs s poss s]*) (*auto simp: varposs-def finite-poss*)

**lemma** *ground-linear* [*simp*]:  $\text{ground } t \implies \text{linear-term } t$   
**by** (*induct t*) (*auto simp: is-partition-alt is-partition-alt-def*)

**declare** *ground-substI*[*intro, simp*]

**lemma** *ground-ctxt-substI*:  
 $(\bigwedge \ x. \ x \in \text{vars-ctxt } C \implies \text{ground } (\sigma \ x)) \implies \text{ground-ctxt } (C \cdot_c \sigma)$   
**by** (*induct C*) *auto*

**lemma** *funas-ctxt-subst-apply-ctxt*:  
 $\text{funas-ctxt } (C \cdot_c \sigma) = \text{funas-ctxt } C \cup (\bigcup \ (\text{funas-term } \ ' \sigma \ ' \text{vars-ctxt } C))$

**proof** (*induct C*)  
**case** (*More f ss C ts*)  
**then show** *?case*  
**by** (*fastforce simp add: funas-term-subst*)  
**qed** *simp*

**lemma** *varposs-Var[simp]*:  
 $varposs (Var x) = \{\}\}$   
**by** (*auto simp: varposs-def*)

**lemma** *varposs-Fun[simp]*:  
 $varposs (Fun f ts) = \{ i \# p \mid i p. i < length\ ts \wedge p \in varposs (ts\ !\ i) \}$   
**by** (*auto simp: varposs-def*)

**lemma** *vars-term-varposs-iff*:  
 $x \in vars-term\ s \iff (\exists p \in varposs\ s. s \mid- p = Var\ x)$   
**proof** (*induct s*)  
**case** (*Fun f ts*)  
**show** *?case using Fun[OF nth-mem]*  
**by** (*force simp: in-set-conv-nth Bex-def*)  
**qed** *auto*

**lemma** *vars-term-empty-ground*:  
 $vars-term\ s = \{\} \implies ground\ s$   
**by** (*metis equals0D ground-substI subst-ident*)

**lemma** *ground-subst-apply*:  $ground\ t \implies t \cdot \sigma = t$   
**by** (*induct t*) (*auto intro: nth-equalityI*)

**lemma** *varposs-imp-poss*:  
 $p \in varposs\ s \implies p \in poss\ s$  **by** (*auto simp: varposs-def*)

**lemma** *varposs-empty-gound*:  
 $varposs\ s = \{\} \iff ground\ s$   
**by** (*induct s*) (*fastforce simp: in-set-conv-nth*)+

**lemma** *funas-term-subterm-atI [intro]*:  
 $p \in poss\ s \implies funas-term\ s \subseteq \mathcal{F} \implies funas-term\ (s \mid- p) \subseteq \mathcal{F}$   
**by** (*metis ctxt-at-pos-subt-at-id funas-ctxt-apply le-sup-iff*)

**lemma** *varposs-ground-replace-at*:  
 $p \in varposs\ s \implies ground\ u \implies varposs\ s[p \leftarrow u] = varposs\ s - \{p\}$   
**proof** (*induct p arbitrary: s*)  
**case** *Nil* **then show** *?case*  
**by** (*cases s*) (*auto simp: varposs-empty-gound*)  
**next**  
**case** (*Cons i p*)  
**from** *Cons(2)* **obtain** *f ts* **where** *[simp]: s = Fun f ts* **by** (*cases s*) *auto*  
**from** *Cons(2)* **have** *var: p \in varposs (ts ! i)* **by** *auto*

**from**  $\text{Cons}(1)[\text{OF var Cons}(3)]$  **have**  $j < \text{length } ts \implies \{j \# q \mid q. q \in \text{varposs } (ts[i := (ts ! i)[p \leftarrow u]] ! j)\} = \{j \# q \mid q. q \in \text{varposs } (ts ! j)\} - \{i \# p\}$  **for**  $j$   
**by**  $(\text{cases } j = i)$   $(\text{auto simp add: nth-list-update})$   
**then show**  $?case$  **by**  $\text{auto blast}$   
**qed**

**lemma**  $\text{funas-term-replace-at-upper}$ :  
 $\text{funas-term } s[p \leftarrow t] \subseteq \text{funas-term } s \cup \text{funas-term } t$   
**proof**  $(\text{induct } p \text{ arbitrary: } s)$   
**case**  $(\text{Cons } i p)$   
**show**  $?case$  **using**  $\text{Cons}(1)[\text{of args } s ! i]$   
**by**  $(\text{cases } s)$   $(\text{fastforce simp: in-set-conv-nth nth-list-update split!: if-splits})+$   
**qed**  $\text{simp}$

**lemma**  $\text{funas-term-replace-at-lower}$ :  
 $p \in \text{poss } s \implies \text{funas-term } t \subseteq \text{funas-term } (s[p \leftarrow t])$   
**proof**  $(\text{induct } p \text{ arbitrary: } s)$   
**case**  $(\text{Cons } i p)$   
**show**  $?case$  **using**  $\text{Cons}(1)[\text{of args } s ! i]$   $\text{Cons}(2-)$   
**by**  $(\text{cases } s)$   $(\text{fastforce simp: in-set-conv-nth nth-list-update split!: if-splits})+$   
**qed**  $\text{simp}$

**lemma**  $\text{poss-of-term-possI}$   $[\text{intro!}]$ :  
 $p \in \text{poss } s \implies s \mid- p = u \implies p \in \text{poss-of-term } u s$   
**unfolding**  $\text{poss-of-term-def}$  **by**  $\text{blast}$

**lemma**  $\text{poss-of-term-replace-term-at}$ :  
 $p \in \text{poss } s \implies p \in \text{poss-of-term } u s[p \leftarrow u]$   
**proof**  $(\text{induct } p \text{ arbitrary: } s)$   
**case**  $(\text{Cons } i p)$  **then show**  $?case$   
**by**  $(\text{cases } s)$   $(\text{auto simp: poss-of-term-def})$   
**qed**  $\text{auto}$

**lemma**  $\text{constT-nfunas-term-poss-of-term-empty}$ :  
 $(c, 0) \notin \text{funas-term } t \iff \text{poss-of-term } (\text{constT } c) t = \{\}$   
**unfolding**  $\text{poss-of-term-def}$   
**using**  $\text{funas-term-subt-at}[\text{of } c \ 0 \ t]$   
**using**  $\text{funas-term-subterm-atI}[\text{where } ?\mathcal{F} = \text{funas-term } t \text{ and } ?s = t, \text{ THEN subsetD}]$   
**by**  $\text{auto}$

**lemma**  $\text{poss-of-term-poss-emptyD}$ :  
**assumes**  $\text{poss-of-term } u s = \{\}$   
**shows**  $p \in \text{poss } s \implies s \mid- p \neq u$  **using**  $\text{assms}$   
**unfolding**  $\text{poss-of-term-def}$  **by**  $\text{blast}$

**lemma**  $\text{possc-subt-at-ctxt-apply}$ :  
 $p \in \text{possc } C \implies p \perp \text{hole-pos } C \implies C\langle s \rangle \mid- p = C\langle t \rangle \mid- p$

```

proof (induct p arbitrary: C)
  case (Cons i p)
  have [dest]: length ss # p ∈ possc (More f ss D ts) ⇒ p ∈ possc D for f ss D ts
    by (auto simp: possc-def)
  show ?case using Cons
    by (cases C) (auto simp: nth-append-Cons)
qed simp

end

```

### 3 Rewriting

```

theory Rewriting
  imports Terms-Positions
begin

```

#### 3.1 Basic rewrite definitions

##### 3.1.1 Rewrite steps with implicit signature declaration (encoded in the type)

```

inductive-set rstep :: ('f, 'v) term rel ⇒ ('f, 'v) term rel for  $\mathcal{R}$  where
  [intro]: (l, r) ∈  $\mathcal{R}$  ⇒ (l ·  $\sigma$ , r ·  $\sigma$ ) ∈ rstep  $\mathcal{R}$ 

```

```

inductive-set rstep :: ('f, 'v) term rel ⇒ ('f, 'v) term rel for  $\mathcal{R}$  where
  (s, t) ∈ rstep  $\mathcal{R}$  ⇒ (C⟨s⟩, C⟨t⟩) ∈ rstep  $\mathcal{R}$ 

```

##### 3.1.2 Restrict relations to terms induced by a given signature

```

definition sig-step  $\mathcal{F}$   $\mathcal{R}$  = Restr  $\mathcal{R}$  (Collect (λ s. funas-term s ⊆  $\mathcal{F}$ ))

```

##### 3.1.3 Rewriting under a given signature/restricted to ground terms

```

abbreviation srrstep  $\mathcal{F}$   $\mathcal{R}$  ≡ sig-step  $\mathcal{F}$  (rstep  $\mathcal{R}$ )

```

```

abbreviation srstep  $\mathcal{F}$   $\mathcal{R}$  ≡ sig-step  $\mathcal{F}$  (rstep  $\mathcal{R}$ )

```

```

abbreviation gsrstep  $\mathcal{F}$   $\mathcal{R}$  ≡ Restr (sig-step  $\mathcal{F}$  (rstep  $\mathcal{R}$ )) (Collect ground)

```

##### 3.1.4 Rewriting sequences involving a root step

```

abbreviation (input) relto :: 'a rel ⇒ 'a rel ⇒ 'a rel where

```

```

  relto R S ≡ S~* O R O S~*

```

```

definition srsteps-with-root-step  $\mathcal{F}$   $\mathcal{R}$  ≡ relto (sig-step  $\mathcal{F}$  (rstep  $\mathcal{R}$ )) (srstep  $\mathcal{F}$   $\mathcal{R}$ )

```

### 3.2 Monotonicity laws

```

lemma Restr-mono: Restr r A ⊆ r by auto

```

```

lemma Restr-transl-mono-set: (Restr r A)+ ⊆ A × A

```

by (simp add: trancl-subset-Sigma)

**lemma** *rrstep-rstep-mono*:  $rrstep \mathcal{R} \subseteq rstep \mathcal{R}$   
by (auto intro: rstep.intros[where ?C =  $\square$ , simplified])

**lemma** *sig-step-mono*:  
 $\mathcal{F} \subseteq \mathcal{G} \implies sig\text{-step } \mathcal{F} \mathcal{R} \subseteq sig\text{-step } \mathcal{G} \mathcal{R}$   
by (auto simp: sig-step-def)

**lemma** *sig-step-mono2*:  
 $\mathcal{R} \subseteq \mathcal{L} \implies sig\text{-step } \mathcal{F} \mathcal{R} \subseteq sig\text{-step } \mathcal{F} \mathcal{L}$   
by (auto simp: sig-step-def)

**lemma** *srrstep-monp*:  
 $\mathcal{F} \subseteq \mathcal{G} \implies srrstep \mathcal{F} \mathcal{R} \subseteq srrstep \mathcal{G} \mathcal{R}$   
by (simp add: sig-step-mono)

**lemma** *srstep-monp*:  
 $\mathcal{F} \subseteq \mathcal{G} \implies srstep \mathcal{F} \mathcal{R} \subseteq srstep \mathcal{G} \mathcal{R}$   
by (simp add: sig-step-mono)

**lemma** *srsteps-monp*:  
 $\mathcal{F} \subseteq \mathcal{G} \implies (srstep \mathcal{F} \mathcal{R})^+ \subseteq (srstep \mathcal{G} \mathcal{R})^+$   
by (simp add: sig-step-mono trancl-mono-set)

**lemma** *srsteps-eq-monp*:  
 $\mathcal{F} \subseteq \mathcal{G} \implies (srstep \mathcal{F} \mathcal{R})^* \subseteq (srstep \mathcal{G} \mathcal{R})^*$   
by (meson rtrancl-mono sig-step-mono subrelI subsetD trancl-into-rtrancl)

**lemma** *srsteps-with-root-step-sig-mono*:  
 $\mathcal{F} \subseteq \mathcal{G} \implies srsteps\text{-with-root-step } \mathcal{F} \mathcal{R} \subseteq srsteps\text{-with-root-step } \mathcal{G} \mathcal{R}$   
**unfolding** *srsteps-with-root-step-def*  
by (simp add: relcomp-mono srrstep-monp srsteps-eq-monp)

### 3.3 Introduction, elimination, and destruction rules for *sig-step*, *rstep*, *rrstep*, *srrstep*, and *srstep*

**lemma** *sig-stepE* [*elim*, *consumes 1*]:  
 $(s, t) \in sig\text{-step } \mathcal{F} \mathcal{R} \implies [(s, t) \in \mathcal{R} \implies funas\text{-term } s \subseteq \mathcal{F} \implies funas\text{-term } t \subseteq \mathcal{F} \implies P] \implies P$   
by (auto simp: sig-step-def)

**lemma** *sig-stepI* [*intro*]:  
 $funas\text{-term } s \subseteq \mathcal{F} \implies funas\text{-term } t \subseteq \mathcal{F} \implies (s, t) \in \mathcal{R} \implies (s, t) \in sig\text{-step } \mathcal{F} \mathcal{R}$   
by (auto simp: sig-step-def)

**lemma** *rrstep-subst* [*elim*, *consumes 1*]:  
**assumes**  $(s, t) \in rrstep \mathcal{R}$

**obtains**  $l r \sigma$  **where**  $(l, r) \in \mathcal{R} \ s = l \cdot \sigma \ t = r \cdot \sigma$  **using** *assms*  
**by** (*meson rstep.simps*)

**lemma** *rstep-imp-C-s-r*:  
**assumes**  $(s, t) \in \text{rstep } \mathcal{R}$   
**shows**  $\exists C \ l \ r \ \sigma. (l, r) \in \mathcal{R} \wedge s = C\langle l \cdot \sigma \rangle \wedge t = C\langle r \cdot \sigma \rangle$  **using** *assms*  
**by** (*metis rstep.cases rstep.simps*)

**lemma** *rstep-imp-C-s-r'* [*elim, consumes 1*]:  
**assumes**  $(s, t) \in \text{rstep } \mathcal{R}$   
**obtains**  $C \ l \ r \ \sigma$  **where**  $(l, r) \in \mathcal{R} \ s = C\langle l \cdot \sigma \rangle \ t = C\langle r \cdot \sigma \rangle$  **using** *assms*  
**using** *rstep-imp-C-s-r* **by** *blast*

**lemma** *rrstep-basicI* [*intro*]:  
 $(l, r) \in \mathcal{R} \implies (l, r) \in \text{rrstep } \mathcal{R}$   
**by** (*metis rrstepp.intros rrstepp-rrstep-eq subst-apply-term-empty*)

**lemma** *rstep-ruleI* [*intro*]:  
 $(l, r) \in \mathcal{R} \implies (l, r) \in \text{rstep } \mathcal{R}$   
**using** *rrstep-rstep-mono* **by** *blast*

**lemma** *rstepI* [*intro*]:  
 $(l, r) \in \mathcal{R} \implies s = C\langle l \cdot \sigma \rangle \implies t = C\langle r \cdot \sigma \rangle \implies (s, t) \in \text{rstep } \mathcal{R}$   
**by** (*simp add: rrstep.intros rstep.intros*)

**lemma** *rstep-substI* [*intro*]:  
 $(s, t) \in \text{rstep } \mathcal{R} \implies (s \cdot \sigma, t \cdot \sigma) \in \text{rstep } \mathcal{R}$   
**by** (*auto elim!: rstep-imp-C-s-r' simp flip: subst-subst-compose*)

**lemma** *rstep-ctxtI* [*intro*]:  
 $(s, t) \in \text{rstep } \mathcal{R} \implies (C\langle s \rangle, C\langle t \rangle) \in \text{rstep } \mathcal{R}$   
**by** (*auto elim!: rstep-imp-C-s-r' simp flip: ctxt-ctxt-compose*)

**lemma** *srrstepD*:  
 $(s, t) \in \text{srrstep } \mathcal{F} \ \mathcal{R} \implies (s, t) \in \text{rrstep } \mathcal{R} \wedge \text{funas-term } s \subseteq \mathcal{F} \wedge \text{funas-term } t \subseteq \mathcal{F}$   
**by** (*auto simp: sig-step-def*)

**lemma** *srstepD*:  
 $(s, t) \in (\text{srstep } \mathcal{F} \ \mathcal{R}) \implies (s, t) \in \text{rstep } \mathcal{R} \wedge \text{funas-term } s \subseteq \mathcal{F} \wedge \text{funas-term } t \subseteq \mathcal{F}$   
**by** (*auto simp: sig-step-def*)

**lemma** *srstepsD*:  
 $(s, t) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^+ \implies (s, t) \in (\text{rstep } \mathcal{R})^+ \wedge \text{funas-term } s \subseteq \mathcal{F} \wedge \text{funas-term } t \subseteq \mathcal{F}$   
**unfolding** *sig-step-def* **using** *trancl-mono-set[OF Restr-mono]*  
**by** (*auto simp: sig-step-def dest: subsetD[OF Restr-trancl-mono-set]*)



### 3.3.1 Transitive and reflexive closure distribution over *sig-step*

**lemma** *funas-rel-converse*:

*funas-rel*  $\mathcal{R} \subseteq \mathcal{F} \implies \text{funas-rel } (\mathcal{R}^{-1}) \subseteq \mathcal{F}$  **unfolding** *funas-rel-def*  
**by** *auto*

**lemma** *rstep-term-to-sig-r*:

**assumes**  $(s, t) \in \text{rstep } \mathcal{R}$  **and** *funas-rel*  $\mathcal{R} \subseteq \mathcal{F}$  **and** *funas-term*  $s \subseteq \mathcal{F}$   
**shows**  $(s, \text{term-to-sig } \mathcal{F} \ v \ t) \in \text{rstep } \mathcal{R}$

**proof** –

**from** *assms(1)* **obtain**  $C \ l \ r \ \sigma$  **where**

$*: s = C\langle l \cdot \sigma \rangle \ t = C\langle r \cdot \sigma \rangle \ (l, r) \in \mathcal{R}$  **by** *auto*

**from** *assms(2, 3)*  $*(3)$  **have** *funas-ctxt*  $C \subseteq \mathcal{F}$  *funas-term*  $l \subseteq \mathcal{F}$  *funas-term*  $r \subseteq \mathcal{F}$

**by** (*auto simp: \*(1) funas-rel-def funas-term-subst subset-eq*)

**then have**  $(\text{term-to-sig } \mathcal{F} \ v \ s, \text{term-to-sig } \mathcal{F} \ v \ t) \in \text{rstep } \mathcal{R}$  **using**  $*(3)$

**by** (*auto simp: \*(1, 2) funas-ctxt-ctxt-well-def-hole-path*)

**then show** *?thesis* **using** *assms(3)* **by** *auto*

**qed**

**lemma** *rstep-term-to-sig-l*:

**assumes**  $(s, t) \in \text{rstep } \mathcal{R}$  **and** *funas-rel*  $\mathcal{R} \subseteq \mathcal{F}$  **and** *funas-term*  $t \subseteq \mathcal{F}$   
**shows**  $(\text{term-to-sig } \mathcal{F} \ v \ s, t) \in \text{rstep } \mathcal{R}$

**proof** –

**from** *assms(1)* **obtain**  $C \ l \ r \ \sigma$  **where**

$*: s = C\langle l \cdot \sigma \rangle \ t = C\langle r \cdot \sigma \rangle \ (l, r) \in \mathcal{R}$  **by** *auto*

**from** *assms(2, 3)*  $*(3)$  **have** *funas-ctxt*  $C \subseteq \mathcal{F}$  *funas-term*  $l \subseteq \mathcal{F}$  *funas-term*  $r \subseteq \mathcal{F}$

**by** (*auto simp: \*(2) funas-rel-def funas-term-subst subset-eq*)

**then have**  $(\text{term-to-sig } \mathcal{F} \ v \ s, \text{term-to-sig } \mathcal{F} \ v \ t) \in \text{rstep } \mathcal{R}$  **using**  $*(3)$

**by** (*auto simp: \*(1, 2) funas-ctxt-ctxt-well-def-hole-path*)

**then show** *?thesis* **using** *assms(3)* **by** *auto*

**qed**

**lemma** *rstep-trancl-sig-step-r*:

**assumes**  $(s, t) \in (\text{rstep } \mathcal{R})^+$  **and** *funas-rel*  $\mathcal{R} \subseteq \mathcal{F}$  **and** *funas-term*  $s \subseteq \mathcal{F}$   
**shows**  $(s, \text{term-to-sig } \mathcal{F} \ v \ t) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^+$  **using** *assms*

**proof** (*induct*)

**case** (*base t*)

**then show** *?case* **using** *subsetD[OF funas-term-term-to-sig, of - \mathcal{F} \ v]*

**by** (*auto simp: rstep-term-to-sig-r sig-step-def intro!: r-into-trancl*)

**next**

**case** (*step t u*)

**then have** *st*:  $(s, \text{term-to-sig } \mathcal{F} \ v \ t) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^+$  **by** *auto*

**from** *step(2)* **obtain**  $C \ l \ r \ \sigma$  **where**

$*: t = C\langle l \cdot \sigma \rangle \ u = C\langle r \cdot \sigma \rangle \ (l, r) \in \mathcal{R}$  **by** *auto*

**show** *?case*

**proof** (*cases ctxt-well-def-hole-path \mathcal{F} \ C*)

**case** *True*

**from**  $*(3)$  *step(4)* **have** *funas-term*  $l \subseteq \mathcal{F}$  *funas-term*  $r \subseteq \mathcal{F}$  **by** (*auto simp:*

*funas-rel-def*  
**then have**  $(\text{term-to-sig } \mathcal{F} \ v \ t, \text{term-to-sig } \mathcal{F} \ v \ u) \in \text{rstep } \mathcal{R}$   
**using** *True step(2) \*(3) unfolding \**  
**by auto**  
**then have**  $(\text{term-to-sig } \mathcal{F} \ v \ t, \text{term-to-sig } \mathcal{F} \ v \ u) \in \text{srstep } \mathcal{F} \ \mathcal{R}$   
**by** *(auto simp:- sig-step-def)*  
**then show** *?thesis using st by auto*  
**next**  
**case** *False*  
**then have**  $\text{term-to-sig } \mathcal{F} \ v \ t = \text{term-to-sig } \mathcal{F} \ v \ u$  **unfolding \*** **by auto**  
**then show** *?thesis using st by auto*  
**qed**  
**qed**

**lemma** *rstep-trancl-sig-step-l*:  
**assumes**  $(s, t) \in (\text{rstep } \mathcal{R})^+$  **and**  $\text{funas-rel } \mathcal{R} \subseteq \mathcal{F}$  **and**  $\text{funas-term } t \subseteq \mathcal{F}$   
**shows**  $(\text{term-to-sig } \mathcal{F} \ v \ s, t) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^+$  **using** *assms*  
**proof** *(induct rule: converse-trancl-induct)*  
**case** *(base t)*  
**then show** *?case using subsetD[OF funas-term-term-to-sig, of - \mathcal{F} v]*  
**by** *(auto simp: rstep-term-to-sig-l sig-step-def intro!: r-into-trancl)*  
**next**  
**case** *(step s u)*  
**then have**  $st: (\text{term-to-sig } \mathcal{F} \ v \ u, t) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^+$  **by auto**  
**from** *step(1)* **obtain**  $C \ l \ r \ \sigma$  **where**  
 $*$ :  $s = C \langle l \cdot \sigma \rangle \ u = C \langle r \cdot \sigma \rangle \ (l, r) \in \mathcal{R}$  **by auto**  
**show** *?case*  
**proof** *(cases ctxt-well-def-hole-path \mathcal{F} C)*  
**case** *True*  
**from**  $*$ (3) *step(4)* **have**  $\text{funas-term } l \subseteq \mathcal{F} \ \text{funas-term } r \subseteq \mathcal{F}$  **by** *(auto simp: funas-rel-def)*  
**then have**  $(\text{term-to-sig } \mathcal{F} \ v \ s, \text{term-to-sig } \mathcal{F} \ v \ u) \in \text{rstep } \mathcal{R}$   
**using** *True step(2) \*(3) unfolding \**  
**by auto**  
**then have**  $(\text{term-to-sig } \mathcal{F} \ v \ s, \text{term-to-sig } \mathcal{F} \ v \ u) \in \text{srstep } \mathcal{F} \ \mathcal{R}$   
**by** *(auto simp:- sig-step-def)*  
**then show** *?thesis using st by auto*  
**next**  
**case** *False*  
**then have**  $\text{term-to-sig } \mathcal{F} \ v \ s = \text{term-to-sig } \mathcal{F} \ v \ u$  **unfolding \*** **by auto**  
**then show** *?thesis using st by auto*  
**qed**  
**qed**

**lemma** *rstep-srstepI [intro]*:  
 $\text{funas-rel } \mathcal{R} \subseteq \mathcal{F} \implies \text{funas-term } s \subseteq \mathcal{F} \implies \text{funas-term } t \subseteq \mathcal{F} \implies (s, t) \in \text{rstep } \mathcal{R} \implies (s, t) \in \text{srstep } \mathcal{F} \ \mathcal{R}$   
**by** *blast*

**lemma** *rsteps-srstepsI* [*intro*]:  
*funas-rel*  $\mathcal{R} \subseteq \mathcal{F} \implies \text{funas-term } s \subseteq \mathcal{F} \implies \text{funas-term } t \subseteq \mathcal{F} \implies (s, t) \in (\text{rstep } \mathcal{R})^+ \implies (s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$   
**using** *rstep-trancl-sig-step-r*[*of s t R F*]  
**by** *auto*

**lemma** *rsteps-eq-srsteps-eqI* [*intro*]:  
*funas-rel*  $\mathcal{R} \subseteq \mathcal{F} \implies \text{funas-term } s \subseteq \mathcal{F} \implies \text{funas-term } t \subseteq \mathcal{F} \implies (s, t) \in (\text{rstep } \mathcal{R})^* \implies (s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   
**by** (*auto simp add: rtrancl-eq-or-trancl*)

**lemma** *rsteps-eq-relcomp-srsteps-eq-relcompI* [*intro*]:

**assumes** *funas-rel*  $\mathcal{R} \subseteq \mathcal{F}$  *funas-rel*  $\mathcal{S} \subseteq \mathcal{F}$   
**and** *funas*: *funas-term*  $s \subseteq \mathcal{F}$  *funas-term*  $t \subseteq \mathcal{F}$   
**and** *steps*:  $(s, t) \in (\text{rstep } \mathcal{R})^* \ O (\text{rstep } \mathcal{S})^*$   
**shows**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^* \ O (\text{srstep } \mathcal{F} \mathcal{S})^*$

**proof** –

**from** *steps* **obtain** *u* **where**  $(s, u) \in (\text{rstep } \mathcal{R})^*$   $(u, t) \in (\text{rstep } \mathcal{S})^*$  **by** *auto*  
**then** **have**  $(s, \text{term-to-sig } \mathcal{F} \ v \ u) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^*$   $(\text{term-to-sig } \mathcal{F} \ v \ u, t) \in (\text{srstep } \mathcal{F} \ \mathcal{S})^*$

**using** *rstep-trancl-sig-step-l*[*OF - assms(2) funas(2), of u v*]

**using** *rstep-trancl-sig-step-r*[*OF - assms(1) funas(1), of u v*] *funas*

**by** (*auto simp: rtrancl-eq-or-trancl*)

**then** **show** *?thesis* **by** *auto*

**qed**

### 3.3.2 Distributivity laws

**lemma** *rstep-smycl-dist*:  
 $(\text{rstep } \mathcal{R})^{\leftrightarrow} = \text{rstep } (\mathcal{R}^{\leftrightarrow})$   
**by** (*auto simp: sig-step-def*)

**lemma** *sig-step-symcl-dist*:  
 $(\text{sig-step } \mathcal{F} \ \mathcal{R})^{\leftrightarrow} = \text{sig-step } \mathcal{F} \ (\mathcal{R}^{\leftrightarrow})$   
**by** (*auto simp: sig-step-def*)

**lemma** *srstep-symcl-dist*:  
 $(\text{srstep } \mathcal{F} \ \mathcal{R})^{\leftrightarrow} = \text{srstep } \mathcal{F} \ (\mathcal{R}^{\leftrightarrow})$   
**by** (*auto simp: sig-step-def*)

**lemma** *Restr-smycl-dist*:  
 $(\text{Restr } \mathcal{R} \ \mathcal{A})^{\leftrightarrow} = \text{Restr } (\mathcal{R}^{\leftrightarrow}) \ \mathcal{A}$   
**by** *auto*

**lemmas** *rew-symcl-inwards* = *rstep-smycl-dist sig-step-symcl-dist srstep-symcl-dist Restr-smycl-dist*

**lemmas** *rew-symcl-outwards* = *rew-symcl-inwards*[*symmetric*]

**lemma** *rstep-converse-dist*:  
 $(rstep \mathcal{R})^{-1} = rstep (\mathcal{R}^{-1})$   
**by** *auto*

**lemma** *srrstep-converse-dist*:  
 $(srrstep \mathcal{F} \mathcal{R})^{-1} = srrstep \mathcal{F} (\mathcal{R}^{-1})$   
**by** (*fastforce simp: sig-step-def*)

**lemma** *sig-step-converse-rstep*:  
 $(srstep \mathcal{F} \mathcal{R})^{-1} = sig-step \mathcal{F} ((rstep \mathcal{R})^{-1})$   
**by** (*meson converse.simps set-eq-subset sig-stepE(1) sig-stepE sig-stepI subrelI*)

**lemma** *srstep-converse-dist*:  
 $(srstep \mathcal{F} \mathcal{R})^{-1} = srstep \mathcal{F} (\mathcal{R}^{-1})$   
**by** (*auto simp: sig-step-def*)

**lemma** *Restr-converse*:  $(Restr \mathcal{R} A)^{-1} = Restr (\mathcal{R}^{-1}) A$   
**by** *auto*

**lemmas** *rew-converse-inwards* = *rstep-converse-dist srrstep-converse-dist sig-step-converse-rstep srstep-converse-dist Restr-converse trancl-converse[symmetric] rtrancl-converse[symmetric]*  
**lemmas** *rew-converse-outwards* = *rew-converse-inwards[symmetric]*

**lemma** *sig-step-rsteps-dist*:  
 $funas-rel \mathcal{R} \subseteq \mathcal{F} \implies sig-step \mathcal{F} ((rstep \mathcal{R})^+) = (srstep \mathcal{F} \mathcal{R})^+$   
**by** (*auto elim!: sig-stepE dest: srstepsD*)

**lemma** *sig-step-rsteps-eq-dist*:  
 $funas-rel \mathcal{R} \subseteq \mathcal{F} \implies sig-step \mathcal{F} ((rstep \mathcal{R})^+) \cup Id = (srstep \mathcal{F} \mathcal{R})^*$   
**by** (*auto simp: rtrancl-eq-or-trancl sig-step-rsteps-dist*)

**lemma** *sig-step-conversion-dist*:  
 $(srstep \mathcal{F} \mathcal{R})^{\leftrightarrow*} = (srstep \mathcal{F} (\mathcal{R}^{\leftrightarrow}))^*$   
**by** (*auto simp: rtrancl-eq-or-trancl sig-step-rsteps-dist conversion-def srstep-symcl-dist*)

**lemma** *gsrstep-conversion-dist*:  
 $(gsrstep \mathcal{F} \mathcal{R})^{\leftrightarrow*} = (gsrstep \mathcal{F} (\mathcal{R}^{\leftrightarrow}))^*$   
**by** (*auto simp: conversion-def rew-symcl-inwards*)

**lemma** *sig-step-grstep-dist*:  
 $gsrstep \mathcal{F} \mathcal{R} = sig-step \mathcal{F} (Restr (rstep \mathcal{R}) (Collect ground))$   
**by** (*auto simp: sig-step-def*)

### 3.4 Substitution closure of *srstep*

**lemma** *srstep-subst-closed*:  
**assumes**  $(s, t) \in srstep \mathcal{F} \mathcal{R} \wedge x. funas-term (\sigma x) \subseteq \mathcal{F}$   
**shows**  $(s \cdot \sigma, t \cdot \sigma) \in srstep \mathcal{F} \mathcal{R}$  **using** *assms*  
**by** (*auto simp: sig-step-def funas-term-subst*)

**lemma** *srsteps-subst-closed*:  
**assumes**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+ \wedge x. \text{funas-term } (\sigma x) \subseteq \mathcal{F}$   
**shows**  $(s \cdot \sigma, t \cdot \sigma) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$  **using** *assms(1)*  
**proof** (*induct rule: trancl.induct*)  
**case** (*r-into-trancl s t*) **show** *?case*  
**using** *srstep-subst-closed[OF r-into-trancl assms(2)]*  
**by** *auto*  
**next**  
**case** (*trancl-into-trancl s t u*)  
**from** *trancl-into-trancl(2)* **show** *?case*  
**using** *srstep-subst-closed[OF trancl-into-trancl(3) assms(2)]*  
**by** (*meson rtrancl-into-trancl1 trancl-into-rtrancl*)  
**qed**

**lemma** *srsteps-eq-subst-closed*:  
**assumes**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^* \wedge x. \text{funas-term } (\sigma x) \subseteq \mathcal{F}$   
**shows**  $(s \cdot \sigma, t \cdot \sigma) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$  **using** *assms srsteps-subst-closed*  
**by** (*metis rtrancl-eq-or-trancl*)

**lemma** *srsteps-eq-subst-relcomp-closed*:  
**assumes**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^* O (\text{srstep } \mathcal{F} \mathcal{S})^* \wedge x. \text{funas-term } (\sigma x) \subseteq \mathcal{F}$   
**shows**  $(s \cdot \sigma, t \cdot \sigma) \in (\text{srstep } \mathcal{F} \mathcal{R})^* O (\text{srstep } \mathcal{F} \mathcal{S})^*$   
**proof** –  
**from** *assms(1)* **obtain** *u* **where**  $(s, u) \in (\text{srstep } \mathcal{F} \mathcal{R})^* (u, t) \in (\text{srstep } \mathcal{F} \mathcal{S})^*$   
**by** *auto*  
**then** **have**  $(s \cdot \sigma, u \cdot \sigma) \in (\text{srstep } \mathcal{F} \mathcal{R})^* (u \cdot \sigma, t \cdot \sigma) \in (\text{srstep } \mathcal{F} \mathcal{S})^*$   
**using** *assms srsteps-eq-subst-closed*  
**by** *metis+*  
**then** **show** *?thesis* **by** *auto*  
**qed**

### 3.5 Context closure of *srstep*

**lemma** *srstep-ctxt-closed*:  
**assumes** *funas-ctxt C*  $C \subseteq \mathcal{F}$  **and**  $(s, t) \in \text{srstep } \mathcal{F} \mathcal{R}$   
**shows**  $(C\langle s \rangle, C\langle t \rangle) \in \text{srstep } \mathcal{F} \mathcal{R}$  **using** *assms*  
**by** (*intro sig-stepI*) (*auto dest: srstepD*)

**lemma** *srsteps-ctxt-closed*:  
**assumes** *funas-ctxt C*  $C \subseteq \mathcal{F}$  **and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$   
**shows**  $(C\langle s \rangle, C\langle t \rangle) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$  **using** *assms(2) srstep-ctxt-closed[OF assms(1)]*  
**by** (*induct*) *force+*

**lemma** *srsteps-eq-ctxt-closed*:  
**assumes** *funas-ctxt C*  $C \subseteq \mathcal{F}$  **and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   
**shows**  $(C\langle s \rangle, C\langle t \rangle) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$  **using** *srsteps-ctxt-closed[OF assms(1)]*  
*assms(2)*

by (*metis rtrancl-eq-or-trancl*)

**lemma** *sig-steps-join-ctxt-closed*:

**assumes** *funas-ctxt*  $C \subseteq \mathcal{F}$  **and**  $(s, t) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^\downarrow$

**shows**  $(C\langle s \rangle, C\langle t \rangle) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^\downarrow$  **using** *srsteps-eq-ctxt-closed*[*OF assms(1)*]  
*assms(2)*

**unfolding** *join-def rew-converse-inwards*

**by** *auto*

The following lemma shows that every rewrite sequence either contains a root step or is root stable

**lemma** *nsrsteps-with-root-step-step-on-args*:

**assumes**  $(s, t) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^+$   $(s, t) \notin \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{R}$

**shows**  $\exists f \ ss \ ts. s = \text{Fun } f \ ss \wedge t = \text{Fun } f \ ts \wedge \text{length } ss = \text{length } ts \wedge$

$(\forall i < \text{length } ts. (ss ! i, ts ! i) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^*)$  **using** *assms*

**proof** (*induct*)

**case** (*base t*)

**obtain**  $C \ l \ r \ \sigma$  **where** [*simp*]:  $s = C\langle l \cdot \sigma \rangle \ t = C\langle r \cdot \sigma \rangle$  **and**  $r: (l, r) \in \mathcal{R}$

**using** *base(1)* **unfolding** *sig-step-def*

**by** *blast*

**then have** *funas*: *funas-ctxt*  $C \subseteq \mathcal{F}$  *funas-term*  $(l \cdot \sigma) \subseteq \mathcal{F}$  *funas-term*  $(r \cdot \sigma) \subseteq \mathcal{F}$

**using** *base(1)* **by** (*auto simp: sig-step-def*)

**from** *funas(2-)* *r* **have**  $(l \cdot \sigma, r \cdot \sigma) \in \text{srrstep } \mathcal{F} \ \mathcal{R}$

**by** (*auto simp: sig-step-def*)

**then have**  $C = \text{Hole} \implies \text{False}$  **using** *base(2)* *r*

**by** (*auto simp: srsteps-with-root-step-def*)

**then obtain**  $f \ ss \ D \ ts$  **where** [*simp*]:  $C = \text{More } f \ ss \ D \ ts$  **by** (*cases C*) *auto*

**have**  $(D\langle l \cdot \sigma \rangle, D\langle r \cdot \sigma \rangle) \in (\text{srstep } \mathcal{F} \ \mathcal{R})$  **using** *base(1)* *r* *funas*

**by** (*auto simp: sig-step-def*)

**then show** *?case* **using** *funas* **by** (*auto simp: nth-append-Cons*)

**next**

**case** (*step t u*) **show** *?case*

**proof** (*cases*  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{R} \vee (t, u) \in \text{sig-step } \mathcal{F} \ (\text{rrstep } \mathcal{R})$ )

**case** *True* **then show** *?thesis* **using** *step(1, 2, 4)*

**by** (*auto simp add: relcomp3-I rtrancl.rtrancl-into-rtrancl srsteps-with-root-step-def*)

**next**

**case** *False*

**obtain**  $C \ l \ r \ \sigma$  **where** \**[simp]*:  $t = C\langle l \cdot \sigma \rangle \ u = C\langle r \cdot \sigma \rangle$  **and**  $r: (l, r) \in \mathcal{R}$

**using** *step(2)* **unfolding** *sig-step-def* **by** *blast*

**then have** *funas*: *funas-ctxt*  $C \subseteq \mathcal{F}$  *funas-term*  $(l \cdot \sigma) \subseteq \mathcal{F}$  *funas-term*  $(r \cdot \sigma) \subseteq \mathcal{F}$

**using** *step(2)* **by** (*auto simp: sig-step-def*)

**from** *False* **have**  $C \neq \text{Hole}$  **using** *funas r* **by** (*force simp: sig-step-def*)

**then obtain**  $f \ ss \ D \ ts$  **where** *c*[*simp*]:  $C = \text{More } f \ ss \ D \ ts$  **by** (*cases C*) *auto*

**from** *step(3, 1)* *False* **obtain**  $g \ sss \ tss$  **where**

\*\**[simp]*:  $s = \text{Fun } g \ sss \ t = \text{Fun } g \ tss$  **and**  $l: \text{length } sss = \text{length } tss$  **and**

*inv*:  $\forall i < \text{length } tss. (sss ! i, tss ! i) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^*$

```

    by auto
    have [simp]:  $g = f$  and  $lc: \text{Suc} (\text{length } ss + \text{length } ts) = \text{length } sss$ 
    using  $l$  *(1) unfolding  $c$  using *(2) by auto
    then have  $\forall i < \text{Suc} (\text{length } ss + \text{length } ts). ((ss @ D\langle l \cdot \sigma \rangle \# ts) ! i, (ss @ D\langle r \cdot \sigma \rangle \# ts) ! i) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$ 
    using * funas  $r$  by (auto simp: nth-append-Cons r-into-rtrancl rstep.intros rstepI sig-stepI)
    then have  $i < \text{length } tss \implies (sss ! i, (ss @ D\langle r \cdot \sigma \rangle \# ts) ! i) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$  for  $i$ 
    using inv *  $l$   $lc$  funas **
    by (auto simp: nth-append-Cons simp del: ** * split!: if-splits)
    then show ?thesis using inv  $l$   $lc$  * unfolding  $c$ 
    by auto
  qed
qed

```

```

lemma rstep-to-pos-replace:
  assumes  $(s, t) \in \text{rstep } \mathcal{R}$ 
  shows  $\exists p \ l \ r \ \sigma. p \in \text{poss } s \wedge (l, r) \in \mathcal{R} \wedge s \mid- p = l \cdot \sigma \wedge t = s[p \leftarrow r \cdot \sigma]$ 
proof -
  from assms obtain  $C \ l \ r \ \sigma$  where  $st: (l, r) \in \mathcal{R} \ s = C\langle l \cdot \sigma \rangle \ t = C\langle r \cdot \sigma \rangle$ 
  using rstep-imp-C-s-r by fastforce
  from  $st(2, 3)$  have *:  $t = s[\text{hole-pos } C \leftarrow r \cdot \sigma]$  by simp
  from this  $st$  show ?thesis unfolding *
  by (intro exI[of - hole-pos C]) auto
qed

```

```

lemma pos-replace-to-rstep:
  assumes  $p \in \text{poss } s \ (l, r) \in \mathcal{R}$ 
  and  $s \mid- p = l \cdot \sigma \ t = s[p \leftarrow r \cdot \sigma]$ 
  shows  $(s, t) \in \text{rstep } \mathcal{R}$ 
  using assms(1, 3-) replace-term-at-subst-at-id [of  $s \ p$ ]
  by (intro rstepI[OF assms(2), of  $s \ \text{ctxt-at-pos } s \ p \ \sigma$ ])
  (auto simp add: ctxt-of-pos-term-apply-replace-at-ident)

```

```

end
theory Replace-Constant
  imports Rewriting
begin

```

### 3.6 Removing/Replacing constants in a rewrite sequence that do not appear in the rewrite system

```

lemma funas-term-const-subst-conv:
   $(c, 0) \notin \text{funas-term } l \iff \neg (l \supseteq \text{constT } c)$ 
proof (induct  $l$ )
  case (Fun  $f \ ts$ ) then show ?case
  by auto (metis Fun-supt supseq-supt-conv term.inject(2))+
qed (auto simp add: supseq-var-imp-eq)

```

**lemma** *fresh-const-single-step-replace*:

**assumes** *lin*: linear-sys  $\mathcal{R}$  **and** *fresh*:  $(c, 0) \notin \text{funas-rel } \mathcal{R}$   
**and** *occ*:  $p \in \text{poss-of-term } (\text{constT } c) \ s$  **and** *step*:  $(s, t) \in \text{rstep } \mathcal{R}$   
**shows**  $(s[p \leftarrow u], t) \in \text{rstep } \mathcal{R} \vee$   
 $(\exists q. q \in \text{poss-of-term } (\text{constT } c) \ t \wedge (s[p \leftarrow u], t[q \leftarrow u]) \in \text{rstep } \mathcal{R})$

**proof** –

**from** *occ* **have** *const*:  $p \in \text{poss } s \wedge s \mid\!-\! p = \text{constT } c$  **by** *auto*  
**from** *step* **obtain**  $C \ l \ r \ \sigma$  **where**  $t \ [\text{simp}]: s = C(l \cdot \sigma) \ t = C(r \cdot \sigma)$   
**and** *rule*:  $(l, r) \in \mathcal{R}$  **by** *blast*  
**from** *rule* *lin* **have** *lin*: linear-term  $l$  linear-term  $r$  **by** *fastforce+*  
**from** *fresh* *rule* **have** *nt-lhs*:  $(c, 0) \notin \text{funas-term } l$  **by**  $(\text{auto } \text{simp}: \text{funas-rel-def})$   
**consider**  $(\text{par}) \ p \perp (\text{hole-pos } C) \mid (\text{below}) \ \text{hole-pos } C \leq_p p$  **using** *occ*  
**by**  $(\text{auto } \text{dest}: \text{poss-of-term-const-ctxt-apply})$   
**then show** *?thesis*  
**proof** *cases*  
**case** *par*  
**then have** *possc*:  $p \in \text{possc } C$  **using** *const t possc-def* **by** *blast*  
**then have**  $p \in \text{poss-of-term } (\text{constT } c) \ t \ (s[p \leftarrow u], t[p \leftarrow u]) \in \text{rstep } \mathcal{R}$   
**using** *const par-hole-pos-replace-term-context-at[OF par]*  
**using** *possc-subt-at-ctxt-apply[OF possc par, of r \cdot \sigma l \cdot \sigma]* *rule*  
**by** *auto*  $(\text{metis } \text{par } \text{par-pos-replace-pres } \text{replace-at-hole-pos})$   
**then show** *?thesis* **by** *blast*  
**next**  
**case** *below*  
**then obtain**  $q$  **where**  $[simp]: p = \text{hole-pos } C \ @ \ q$  **and** *poss*:  $q \in \text{poss } (l \cdot \sigma)$   
**using** *const position-less-eq-def*  
**by**  $(\text{metis } (\text{full-types}) \ \text{ctxt-at-pos-hole-pos } \text{ctxt-at-pos-subt-at-pos } \text{poss-append-poss } t(1))$   
**have** *const*:  $l \cdot \sigma \mid\!-\! q = \text{constT } c$  **using** *const* **by** *auto*  
**from** *nt-lhs* **have**  $\exists r. r \in \text{varposs } l \wedge r \leq_p q$  **using** *const poss*  
**proof**  $(\text{induct } l \ \text{arbitrary}: q)$   
**case**  $(\text{Var } x)$   
**then show** *?case* **by** *auto*  
**next**  
**case**  $(\text{Fun } f \ ts)$   
**from**  $\text{Fun}(1)[\text{OF } \text{nth-mem}, \text{of hd } q \ \text{tl } q]$   $\text{Fun}(2-)$  **obtain**  $r$  **where**  
 $r \in \text{varposs } (ts \ ! \ \text{hd } q) \wedge r \leq_p \ \text{tl } q$   
**by**  $(\text{cases } q) \ \text{auto}$   
**then show** *?case* **using**  $\text{Fun}(2- \ 4)$   
**by**  $(\text{intro } \text{exI}[\text{of } - \ \text{hd } q \ \# \ r]) \ \text{auto}$   
**qed**  
**then obtain**  $x \ v$  **where**  $\text{varposs}: v \in \text{varposs } l \ v \leq_p q \ l \mid\!-\! v = \text{Var } x$   
**unfolding** *varposs-def* **by** *blast*  
**let**  $? \tau = \lambda x. \text{if } \text{Var } x = l \mid\!-\! v \ \text{then } (\sigma \ x)[q \ -_p \ v \leftarrow u] \ \text{else } \sigma \ x$   
**show** *?thesis*  
**proof**  $(\text{cases } x \in \text{vars-term } r)$   
**case** *True*  
**then obtain**  $q'$  **where**  $\text{varposs-r}: q' \in \text{varposs } r \ r \mid\!-\! q' = \text{Var } x$



```

    by (metis vars-term-varposs-iff)
  have (s[p ← u], t[(hole-pos C) @ q' @ (q -p v) ← u]) ∈ rstep  $\mathcal{R}$ 
    using lin varposs rule varposs-r
    by (auto simp: linear-term-varposs-subst-replace-term intro!: rstep-ctxI)
      (smt (verit, ccfv-SIG) pos-diff-append-itself rrstep.intros rrstep-rstep-mono
subset-eq term-subst-eq)
  moreover have (hole-pos C) @ q' @ q -p v ∈ poss-of-term (constT c) t
    using varposs-r varposs poss const poss-pos-diffI[OF varposs(2) poss]
    using subst-at-append-dist[of q' q -p v r · σ]
    by (auto simp: poss-append-poss varposs-imp-poss[THEN subst-subst-at-dist]
varposs-imp-poss[THEN subsetD[OF subst-poss-mono]])
      (metis pos-les-eq-append-diff subst-apply-term.simps(1) subst-subst-at-dist
subst-at-append-dist varposs-imp-poss)
    ultimately show ?thesis by auto
  next
  case False
  then have [simp]: r · σ = r · ?τ using varposs
    by (auto simp add: term-subst-eq-conv)
  have (s[p ← u], t) ∈ rstep  $\mathcal{R}$  using rule varposs lin
    by (auto simp: linear-term-varposs-subst-replace-term)
  then show ?thesis by auto
qed
qed
qed

```

lemma fresh-const-steps-replace:

```

  assumes lin: linear-sys  $\mathcal{R}$  and fresh: (c, 0) ∉ funas-rel  $\mathcal{R}$ 
    and occ: p ∈ poss-of-term (constT c) s and steps: (s, t) ∈ (rstep  $\mathcal{R}$ )+
  shows (s[p ← u], t) ∈ (rstep  $\mathcal{R}$ )+ ∨
    (∃ q. q ∈ poss-of-term (constT c) t ∧ (s[p ← u], t[q ← u]) ∈ (rstep  $\mathcal{R}$ )+)
    using steps occ
  proof (induct arbitrary: p rule: converse-trancl-induct)
    case (base s)
    from fresh-const-single-step-replace[OF lin fresh base(2, 1)] show ?case
      by (meson r-into-trancl')
  next
    case (step s t)
    from fresh-const-single-step-replace[OF lin fresh step(4, 1)]
    consider (a) (s[p ← u], t) ∈ rstep  $\mathcal{R}$  | (b) ∃ q. q ∈ poss-of-term (constT c) t ∧
(s[p ← u], t[q ← u]) ∈ rstep  $\mathcal{R}$  by blast
    then show ?case
  proof cases
    case a then show ?thesis using step(2)
      by auto
  next
    case b
    then obtain q where q ∈ poss-of-term (constT c) t (s[p ← u], t[q ← u]) ∈
rstep  $\mathcal{R}$  by blast
    from step(3)[OF this(1)] this(2) show ?thesis

```

by (*metis trancl-into-trancl2*)  
qed  
qed

**lemma** *remove-const-lhs-steps*:

assumes *lin*: *linear-sys*  $\mathcal{R}$  and *fresh*:  $(c, 0) \notin \text{funas-rel } \mathcal{R}$   
and *const*:  $(c, 0) \notin \text{funas-term } t$   
and *pos*:  $p \in \text{poss-of-term } (\text{constT } c) s$   
and *steps*:  $(s, t) \in (\text{rstep } \mathcal{R})^+$   
shows  $(s[p \leftarrow u], t) \in (\text{rstep } \mathcal{R})^+$  using *steps pos const fresh-const-steps-replace*  
by (*metis fresh funas-term-const-subst-conv lin poss-of-termE subt-at-imp-supteq*)

Now we can show that we may remove a constant substitution

**definition** *const-replace-closed* where

*const-replace-closed*  $c U = (\forall s t u p.$   
 $p \in \text{poss-of-term } (\text{constT } c) s \longrightarrow (s, t) \in U \longrightarrow$   
 $(\exists q. q \in \text{poss-of-term } (\text{constT } c) t \wedge (s[p \leftarrow u], t[q \leftarrow u]) \in U) \vee (s[p \leftarrow u],$   
 $t) \in U)$

**lemma** *const-replace-closedD*:

assumes *const-replace-closed*  $c U$   $p \in \text{poss-of-term } (\text{constT } c) s$   $(s, t) \in U$   
shows  $(s[p \leftarrow u], t) \in U \vee (\exists q. q \in \text{poss-of-term } (\text{constT } c) t \wedge (s[p \leftarrow u], t[q$   
 $\leftarrow u]) \in U)$  using *assms*  
unfolding *const-replace-closed-def* by *blast*

**lemma** *const-replace-closedI*:

assumes  $\bigwedge s t u p. p \in \text{poss-of-term } (\text{constT } c) s \implies (s, t) \in U \implies$   
 $(\exists q. q \in \text{poss-of-term } (\text{constT } c) t \wedge (s[p \leftarrow u], t[q \leftarrow u]) \in U) \vee (s[p \leftarrow$   
 $u], t) \in U$   
shows *const-replace-closed*  $c U$  using *assms*  
unfolding *const-replace-closed-def*  
by *auto*

**abbreviation** *const-subst* ::  $'f \Rightarrow 'v \Rightarrow ('f, 'v) \text{Term.term}$  where

*const-subst*  $c \equiv (\lambda x. \text{Fun } c \ [])$

**lemma** *lin-fresh-rstep-const-replace-closed*:

*linear-sys*  $\mathcal{R} \implies (c, 0) \notin \text{funas-rel } \mathcal{R} \implies \text{const-replace-closed } c (\text{rstep } \mathcal{R})$   
using *fresh-const-single-step-replace*[of  $\mathcal{R}$   $c$ ]  
by (*intro const-replace-closedI*) (*auto simp: constT-nfunas-term-poss-of-term-empty,*  
*blast*)

**lemma** *const-replace-closed-symcl*:

*const-replace-closed*  $c U \implies \text{const-replace-closed } c (U^=)$   
unfolding *const-replace-closed-def*  
by (*metis Un-iff pair-in-Id-conv*)

**lemma** *const-replace-closed-trancl*:

*const-replace-closed*  $c U \implies \text{const-replace-closed } c (U^+)$

**proof** (*intro const-replace-closedI*)  
**fix**  $s t u p$   
**assume**  $const: const\text{-replace-closed } c \ U$  **and**  $wit: p \in poss\text{-of-term } (constT \ c) \ s$   
**and**  $steps : (s, t) \in U^+$   
**show**  $(\exists q. q \in poss\text{-of-term } (constT \ c) \ t \wedge (s[p \leftarrow u], t[q \leftarrow u]) \in U^+) \vee (s[p \leftarrow u], t) \in U^+$  **using**  $steps \ wit$   
**proof** (*induct arbitrary: p rule: converse-trancl-induct*)  
**case** (*base s*)  
**show**  $?case$  **using**  $const\text{-replace-closedD}[OF \ const \ base(2, 1)]$   
**by**  $blast$   
**next**  
**case** (*step s v*)  
**from**  $const\text{-replace-closedD}[OF \ const \ step(4, 1)]$   
**consider**  $(a) (s[p \leftarrow u], v) \in U \mid (b) \exists q. q \in poss\text{-of-term } (constT \ c) \ v \wedge (s[p \leftarrow u], v[q \leftarrow u]) \in U$  **by**  $auto$   
**then show**  $?case$   
**proof** *cases*  
**case**  $a$  **then show**  $?thesis$  **using**  $step(2)$   
**by** (*meson trancl-into-trancl2*)  
**next**  
**case**  $b$   
**then show**  $?thesis$  **using**  $step(3, 4)$  **by** (*meson trancl-into-trancl2*)  
**qed**  
**qed**  
**qed**

**lemma** *const-replace-closed-rtrancl*:  
 $const\text{-replace-closed } c \ U \implies const\text{-replace-closed } c \ (U^*)$   
**proof** (*intro const-replace-closedI*)  
**fix**  $s t u p$   
**assume**  $const: const\text{-replace-closed } c \ U$  **and**  $wit: p \in poss\text{-of-term } (constT \ c) \ s$   
**and**  $steps : (s, t) \in U^*$   
**show**  $(\exists q. q \in poss\text{-of-term } (constT \ c) \ t \wedge (s[p \leftarrow u], t[q \leftarrow u]) \in U^*) \vee (s[p \leftarrow u], t) \in U^*$   
**using**  $const\text{-replace-closed-trancl}[OF \ const] \ wit \ steps$   
**by** (*metis const-replace-closedD rtrancl-eq-or-trancl*)  
**qed**

**lemma** *const-replace-closed-relcomp*:  
 $const\text{-replace-closed } c \ U \implies const\text{-replace-closed } c \ V \implies const\text{-replace-closed } c \ (U \ O \ V)$   
**proof** (*intro const-replace-closedI*)  
**fix**  $s t u p$   
**assume**  $const: const\text{-replace-closed } c \ U \ const\text{-replace-closed } c \ V$   
**and**  $wit: p \in poss\text{-of-term } (constT \ c) \ s$  **and**  $step: (s, t) \in U \ O \ V$   
**from**  $step$  **obtain**  $w$  **where**  $w: (s, w) \in U \ (w, t) \in V$  **by**  $auto$   
**from**  $const\text{-replace-closedD}[OF \ const(1) \ wit \ this(1)]$   
**consider**  $(a) (s[p \leftarrow u], w) \in U \mid (b) (\exists q. q \in poss\text{-of-term } (constT \ c) \ w \wedge (s[p \leftarrow u], w[q \leftarrow u]) \in U)$

by *auto*  
**then show**  $(\exists q. q \in \text{poss-of-term } (\text{constT } c) \ t \wedge (s[p \leftarrow u], t[q \leftarrow u]) \in U \ O \ V) \vee (s[p \leftarrow u], t) \in U \ O \ V$   
**proof cases**  
   **case a**  
     **then show** *?thesis using w(2) by auto*  
 next  
   **case b**  
     **then show** *?thesis using const-replace-closedD[OF const(2) - w(2)]*  
     by (*meson relcomp.simps*)  
**qed**  
**qed**

*const-replace-closed* allow the removal of a fresh constant substitution

**lemma** *const-replace-closed-remove-subst-lhs*:  
**assumes** *replc: const-replace-closed c U*  
**and** *const: (c, 0) \notin funas-term t*  
**and** *steps: (s \cdot const-subst c, t) \in U*  
**shows**  $(s, t) \in U$  **using** *steps*  
**proof** (*induct card (varposs s) arbitrary: s*)  
**case** (*Suc n*)  
**obtain** *p ps* **where** *vl: varposs s = insert p ps p \notin ps using Suc(2)*  
**by** (*metis card-le-Suc-iff dual-order.refl*)  
**let** *?s = s[p \leftarrow Fun c []]* **have** *vp: p \in varposs s using vl by auto*  
**then have** [*simp*]:  $?s \cdot \text{const-subst } c = s \cdot \text{const-subst } c$   
**by** (*induct s arbitrary: p*) (*auto simp: nth-list-update map-update intro!: nth-equalityI*)  
**have** *varposs ?s = ps using vl varposs-ground-replace-at[of p s constT c]*  
**by auto**  
**then have**  $n = \text{card } (\text{varposs } ?s)$  **using** *vl Suc(2)* **by** (*auto simp: card-insert-if finite-varposs*)  
**from** *Suc(1)[OF this] have IH: (s[p \leftarrow constT c], t) \in U p \in \text{poss-of-term } (\text{constT } c) s[p \leftarrow constT c]*  
**using** *Suc(2, 3) vl poss-of-term-replace-term-at varposs-imp-poss vp*  
**using**  $\langle s[p \leftarrow \text{constT } c] \cdot \text{const-subst } c = s \cdot \text{const-subst } c \rangle$   
**by fastforce+**  
**show** *?case using const-replace-closedD[OF replc] const IH(2, 1)*  
**by** (*metis constT-nfunas-term-poss-of-term-empty empty-iff replace-term-at-same-pos replace-term-at-subt-at-id*)  
**qed** (*auto simp: ground-subst-apply card-eq-0-iff finite-varposs varposs-empty-gound*)

### 3.6.1 Removal lemma applied to various rewrite relations

**lemma** *remove-const-subst-step-lhs*:  
**assumes** *lin: linear-sys R* **and** *fresh: (c, 0) \notin funas-rel R*  
**and** *const: (c, 0) \notin funas-term t*  
**and** *step: (s \cdot const-subst c, t) \in (rstep R)*  
**shows**  $(s, t) \in (rstep R)$   
**using** *lin-fresh-rstep-const-replace-closed[OF lin fresh, THEN const-replace-closed-remove-subst-lhs]*  
*const step*  
**by blast**

**lemma** *remove-const-subst-steps-lhs*:

**assumes** *lin*: linear-sys  $\mathcal{R}$  **and** *fresh*:  $(c, 0) \notin \text{funas-rel } \mathcal{R}$   
**and** *const*:  $(c, 0) \notin \text{funas-term } t$   
**and** *steps*:  $(s \cdot \text{const-subst } c, t) \in (\text{rstep } \mathcal{R})^+$   
**shows**  $(s, t) \in (\text{rstep } \mathcal{R})^+$   
**using** *lin-fresh-rstep-const-replace-closed*[*THEN const-replace-closed-trancl*,  
*OF lin fresh*, *THEN const-replace-closed-remove-subst-lhs*]  
**using** *const steps*  
**by** *blast*

**lemma** *remove-const-subst-steps-eq-lhs*:

**assumes** *lin*: linear-sys  $\mathcal{R}$  **and** *fresh*:  $(c, 0) \notin \text{funas-rel } \mathcal{R}$   
**and** *const*:  $(c, 0) \notin \text{funas-term } t$   
**and** *steps*:  $(s \cdot \text{const-subst } c, t) \in (\text{rstep } \mathcal{R})^*$   
**shows**  $(s, t) \in (\text{rstep } \mathcal{R})^*$  **using** *steps const*  
**by** (*cases s = t*) (*auto simp: rtrancl-eq-or-trancl funas-term-subst ground-subst-apply vars-term-empty-ground*  
*dest: remove-const-subst-steps-lhs*[*OF lin fresh const*] *split: if-splits*)

**lemma** *remove-const-subst-steps-rhs*:

**assumes** *lin*: linear-sys  $\mathcal{R}$  **and** *fresh*:  $(c, 0) \notin \text{funas-rel } \mathcal{R}$   
**and** *const*:  $(c, 0) \notin \text{funas-term } s$   
**and** *steps*:  $(s, t \cdot \text{const-subst } c) \in (\text{rstep } \mathcal{R})^+$   
**shows**  $(s, t) \in (\text{rstep } \mathcal{R})^+$   
**proof** –  
**from** *steps* **have** *revs*:  $(t \cdot \text{const-subst } c, s) \in (\text{rstep } (\mathcal{R}^{-1}))^+$   
**unfolding** *rew-converse-outwards* **by** *auto*  
**have**  $(t, s) \in (\text{rstep } (\mathcal{R}^{-1}))^+$  **using** *assms*  
**by** (*intro remove-const-subst-steps-lhs*[*OF - - revs*]) (*auto simp: funas-rel-def*)  
**then show** *?thesis* **unfolding** *rew-converse-outwards* **by** *auto*  
**qed**

**lemma** *remove-const-subst-steps-eq-rhs*:

**assumes** *lin*: linear-sys  $\mathcal{R}$  **and** *fresh*:  $(c, 0) \notin \text{funas-rel } \mathcal{R}$   
**and** *const*:  $(c, 0) \notin \text{funas-term } s$   
**and** *steps*:  $(s, t \cdot \text{const-subst } c) \in (\text{rstep } \mathcal{R})^*$   
**shows**  $(s, t) \in (\text{rstep } \mathcal{R})^*$   
**using** *steps const*  
**by** (*cases s = t*) (*auto simp: rtrancl-eq-or-trancl funas-term-subst ground-subst-apply vars-term-empty-ground*  
*dest!: remove-const-subst-steps-rhs*[*OF lin fresh const*] *split: if-splits*)

Main lemmas

**lemma** *const-subst-eq-ground-eq*:

**assumes**  $s \cdot \text{const-subst } c = t \cdot \text{const-subst } d$   $c \neq d$   
**and**  $(c, 0) \notin \text{funas-term } t$   $(d, 0) \notin \text{funas-term } s$   
**shows**  $s = t$  **using** *assms*  
**proof** (*induct s arbitrary: t*)

```

  case (Var x) then show ?case by (cases t) auto
next
case (Fun f ts)
from Fun(2-) obtain g us where [simp]: t = Fun g us by (cases t) auto
have [simp]: g = f and l: length ts = length us using Fun(2)
  by (auto intro: map-eq-imp-length-eq)
have i < length ts  $\implies$  ts ! i = us ! i for i
  using Fun(1)[OF nth-mem, of i us ! i for i] Fun(2-) l
  by (auto simp: map-eq-conv')
then show ?case using l
  by (auto intro: nth-equalityI)
qed

```

**lemma** *remove-const-subst-steps*:

```

  assumes linear-sys  $\mathcal{R}$  and (c, 0)  $\notin$  funas-rel  $\mathcal{R}$  and (d, 0)  $\notin$  funas-rel  $\mathcal{R}$ 
  and c  $\neq$  d (c, 0)  $\notin$  funas-term t (d, 0)  $\notin$  funas-term s
  and (s · const-subst c, t · const-subst d)  $\in$  (rstep  $\mathcal{R}$ )*
  shows (s, t)  $\in$  (rstep  $\mathcal{R}$ )*
proof (cases s · const-subst c = t · const-subst d)
  case True
  from const-subst-eq-ground-eq[OF this] assms(4 - 6) show ?thesis by auto
next
  case False
  then have step: (s · const-subst c, t · const-subst d)  $\in$  (rstep  $\mathcal{R}$ )+ using assms(7)
  by (auto simp: rtrancl-eq-or-trancl)
  then have (s, t · const-subst d)  $\in$  (rstep  $\mathcal{R}$ )+ using assms
  by (intro remove-const-subst-steps-lhs[OF - - - step]) (auto simp: funas-term-subst)
  from remove-const-subst-steps-rhs[OF - - - this] show ?thesis using assms
  by auto
qed

```

**lemma** *remove-const-subst-relcomp-lhs*:

```

  assumes sys: linear-sys  $\mathcal{R}$  linear-sys  $\mathcal{S}$ 
  and fr: (c, 0)  $\notin$  funas-rel  $\mathcal{R}$  and fs:(c, 0)  $\notin$  funas-rel  $\mathcal{S}$ 
  and funas: (c, 0)  $\notin$  funas-term t
  and seq: (s · const-subst c, t)  $\in$  (rstep  $\mathcal{R}$ )* O (rstep  $\mathcal{S}$ )*
  shows (s, t)  $\in$  (rstep  $\mathcal{R}$ )* O (rstep  $\mathcal{S}$ )* using seq
  using lin-fresh-rstep-const-replace-closed[OF sys(1) fr, THEN const-replace-closed-rtrancl]
  using lin-fresh-rstep-const-replace-closed[OF sys(2) fs, THEN const-replace-closed-rtrancl]
  using const-replace-closed-relcomp
  by (intro const-replace-closed-remove-subst-lhs[OF - funas seq]) force

```

**lemma** *remove-const-subst-relcomp-rhs*:

```

  assumes sys: linear-sys  $\mathcal{R}$  linear-sys  $\mathcal{S}$ 
  and fr: (c, 0)  $\notin$  funas-rel  $\mathcal{R}$  and fs:(c, 0)  $\notin$  funas-rel  $\mathcal{S}$ 
  and funas: (c, 0)  $\notin$  funas-term s
  and seq: (s, t · const-subst c)  $\in$  (rstep  $\mathcal{R}$ )* O (rstep  $\mathcal{S}$ )*
  shows (s, t)  $\in$  (rstep  $\mathcal{R}$ )* O (rstep  $\mathcal{S}$ )*

```

**proof** –  
**from** *seq* **have**  $(t \cdot \text{const-subst } c, s) \in ((\text{rstep } \mathcal{R})^* \ O \ (\text{rstep } \mathcal{S})^*)^{-1}$   
**by** *auto*  
**then** **have**  $(t \cdot \text{const-subst } c, s) \in ((\text{rstep } \mathcal{S})^*)^{-1} \ O \ ((\text{rstep } \mathcal{R})^*)^{-1}$   
**using** *converse-relcomp by blast*  
**note** *seq = this[unfolded rtrancl-converse[symmetric] rew-converse-inwards]*  
**from** *sys fr fs* **have** *linear-sys*  $(\mathcal{S}^{-1})$  *linear-sys*  $(\mathcal{R}^{-1})$   $(c, 0) \notin \text{funas-rel } (\mathcal{S}^{-1})$   
 $(c, 0) \notin \text{funas-rel } (\mathcal{R}^{-1})$   
**by** *(auto simp: funas-rel-def)*  
**from** *remove-const-subst-relcomp-lhs[OF this funas seq]*  
**have**  $(t, s) \in (\text{rstep } (\mathcal{S}^{-1}))^* \ O \ (\text{rstep } (\mathcal{R}^{-1}))^*$  **by** *simp*  
**then** **show** *?thesis*  
**unfolding** *rew-converse-outwards converse-relcomp[symmetric]*  
**by** *simp*  
**qed**

**lemma** *remove-const-subst-relcomp:*

**assumes** *sys: linear-sys*  $\mathcal{R}$  *linear-sys*  $\mathcal{S}$   
**and** *fr:*  $(c, 0) \notin \text{funas-rel } \mathcal{R}$   $(d, 0) \notin \text{funas-rel } \mathcal{R}$   
**and** *fs:*  $(c, 0) \notin \text{funas-rel } \mathcal{S}$   $(d, 0) \notin \text{funas-rel } \mathcal{S}$   
**and** *diff:*  $c \neq d$  **and** *funas:*  $(c, 0) \notin \text{funas-term } t$   $(d, 0) \notin \text{funas-term } s$   
**and** *seq:*  $(s \cdot \text{const-subst } c, t \cdot \text{const-subst } d) \in (\text{rstep } \mathcal{R})^* \ O \ (\text{rstep } \mathcal{S})^*$   
**shows**  $(s, t) \in (\text{rstep } \mathcal{R})^* \ O \ (\text{rstep } \mathcal{S})^*$

**proof** –

**from** *diff funas(1)* **have**  $*$ :  $(c, 0) \notin \text{funas-term } (t \cdot \text{const-subst } d)$   
**by** *(auto simp: funas-term-subst)*  
**show** *?thesis* **using** *remove-const-subst-relcomp-rhs[OF sys fr(2) fs(2) funas(2)*  
*remove-const-subst-relcomp-lhs[OF sys fr(1) fs(1) \* seq]]*  
**by** *blast*

**qed**

**end**

## 4 Confluence related rewriting properties

**theory** *Rewriting-Properties*

**imports** *Rewriting*

*Abstract-Rewriting.Abstract-Rewriting*

**begin**

### 4.1 Confluence related ARS properties

**definition** *SCR-on*  $r \ A \equiv (\forall a \in A. \forall b \ c. (a, b) \in r \wedge (a, c) \in r \longrightarrow (\exists d. (b, d) \in r^= \wedge (c, d) \in r^*))$

**abbreviation** *SCR*  $:: 'a \ \text{rel} \Rightarrow \text{bool}$  **where** *SCR*  $r \equiv \text{SCR-on } r \ \text{UNIV}$

**definition** *NFP-on*  $:: 'a \ \text{rel} \Rightarrow 'a \ \text{set} \Rightarrow \text{bool}$  **where**

$NFP\text{-on } r A \longleftrightarrow (\forall a \in A. \forall b c. (a, b) \in r^* \wedge (a, c) \in r^! \longrightarrow (b, c) \in r^*)$

**abbreviation**  $NFP :: 'a \text{ rel} \Rightarrow \text{bool}$  **where**  $NFP r \equiv NFP\text{-on } r UNIV$

**definition**  $CE\text{-on} :: 'a \text{ rel} \Rightarrow 'a \text{ rel} \Rightarrow 'a \text{ set} \Rightarrow \text{bool}$  **where**  
 $CE\text{-on } r s A \longleftrightarrow (\forall a \in A. \forall b. (a, b) \in r^{\leftrightarrow*} \longleftrightarrow (a, b) \in s^{\leftrightarrow*})$

**abbreviation**  $CE :: 'a \text{ rel} \Rightarrow 'a \text{ rel} \Rightarrow \text{bool}$  **where**  $CE r s \equiv CE\text{-on } r s UNIV$

**definition**  $NE\text{-on} :: 'a \text{ rel} \Rightarrow 'a \text{ rel} \Rightarrow 'a \text{ set} \Rightarrow \text{bool}$  **where**  
 $NE\text{-on } r s A \longleftrightarrow (\forall a \in A. \forall b. (a, b) \in r^! \longleftrightarrow (a, b) \in s^!)$

**abbreviation**  $NE :: 'a \text{ rel} \Rightarrow 'a \text{ rel} \Rightarrow \text{bool}$  **where**  $NE r s \equiv NE\text{-on } r s UNIV$

## 4.2 Signature closure of relation to model multihole context closure

**lemma**  $all\text{-ctxt-closed-sig-rsteps}$  [intro]:

**fixes**  $\mathcal{R} :: ('f, 'v) \text{ term rel}$

**shows**  $all\text{-ctxt-closed } \mathcal{F} ((srstep \mathcal{F} \mathcal{R})^*)$  (**is**  $all\text{-ctxt-closed} - (?R^*)$ )

**proof** ( $rule \text{trans-ctxt-sig-imp-all-ctxt-closed}$ )

**fix**  $C :: ('f, 'v) \text{ ctxt}$  **and**  $s t :: ('f, 'v) \text{ term}$

**assume**  $C: funas\text{-ctxt } C \subseteq \mathcal{F}$

**and**  $s: funas\text{-term } s \subseteq \mathcal{F}$

**and**  $t: funas\text{-term } t \subseteq \mathcal{F}$

**and**  $steps: (s, t) \in ?R^*$

**from**  $steps$

**show**  $(C \langle s \rangle, C \langle t \rangle) \in ?R^*$

**proof** ( $induct$ )

**case** ( $step t u$ )

**from**  $step(2)$  **have**  $tu: (t, u) \in rstep \mathcal{R}$  **and**  $t: funas\text{-term } t \subseteq \mathcal{F}$  **and**  $u: funas\text{-term } u \subseteq \mathcal{F}$

**by** ( $auto \text{ dest: } srstepD$ )

**have**  $(C \langle t \rangle, C \langle u \rangle) \in ?R$  **by** ( $rule \text{sig-stepI}[OF - - rstep\text{-ctxtI}[OF tu]]$ ,  $insert C t u, auto$ )

**with**  $step(3)$  **show**  $?case$  **by**  $auto$

**qed**  $auto$

**qed** ( $auto \text{ intro: } trans\text{-rtrancl}$ )

**lemma**  $sigstep\text{-trancl-funass}$ :

$(s, t) \in (srstep \mathcal{F} \mathcal{S})^* \Longrightarrow s \neq t \Longrightarrow funas\text{-term } s \subseteq \mathcal{F}$

$(s, t) \in (srstep \mathcal{F} \mathcal{S})^* \Longrightarrow s \neq t \Longrightarrow funas\text{-term } t \subseteq \mathcal{F}$

**by** ( $auto \text{ simp: } rtrancl\text{-eq-or-trancl} \text{ dest: } srstepsD$ )

**lemma**  $srrstep\text{-to-srstep}$ :

$(s, t) \in srrstep \mathcal{F} \mathcal{R} \Longrightarrow (s, t) \in srstep \mathcal{F} \mathcal{R}$

**by** ( $meson \text{ in-mono } rstep\text{-rstep-mono } sig\text{-step-mono2}$ )

**lemma**  $srsteps\text{-with-root-step-srstepsD}$ :



$(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R} \implies (s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$   
**by** (*auto dest: srrstep-to-srstep simp: srsteps-with-root-step-def*)

**lemma** *srsteps-with-root-step-srsteps-eqD*:

$(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R} \implies (s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   
**by** (*auto dest: srrstep-to-srstep simp: srsteps-with-root-step-def*)

**lemma** *symcl-srstep-conversion*:

$(s, t) \in \text{srstep } \mathcal{F} (\mathcal{R}^{\leftrightarrow}) \implies (s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^{\leftrightarrow*}$   
**by** (*simp add: conversion-def rstep-converse-dist srstep-symcl-dist*)

**lemma** *symcl-srsteps-conversion*:

$(s, t) \in (\text{srstep } \mathcal{F} (\mathcal{R}^{\leftrightarrow}))^* \implies (s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^{\leftrightarrow*}$   
**by** (*simp add: conversion-def rstep-converse-dist srstep-symcl-dist*)

**lemma** *NF-srstep-args*:

**assumes**  $\text{Fun } f \text{ } ss \in \text{NF } (\text{srstep } \mathcal{F} \mathcal{R}) \text{ funas-term } (\text{Fun } f \text{ } ss) \subseteq \mathcal{F} \text{ } i < \text{length } ss$   
**shows**  $ss ! i \in \text{NF } (\text{srstep } \mathcal{F} \mathcal{R})$

**proof** (*rule ccontr*)

**assume**  $ss ! i \notin \text{NF } (\text{srstep } \mathcal{F} \mathcal{R})$

**then obtain**  $t$  **where**  $\text{step: } (ss ! i, t) \in \text{rstep } \mathcal{R} \text{ funas-term } t \subseteq \mathcal{F}$

**by** (*auto simp: NF-def sig-step-def*)

**from** *assms*(3) **have** [*simp*]:  $\text{Suc } (\text{length } ss - \text{Suc } 0) = \text{length } ss$  **by** *auto*

**from** *rstep-ctxtI*[*OF step*(1), **where**  $?C = \text{ctxt-at-pos } (\text{Fun } f \text{ } ss)[i]$ ]

**have**  $(\text{Fun } f \text{ } ss, \text{Fun } f \text{ } (ss[i := t])) \in \text{srstep } \mathcal{F} \mathcal{R}$  **using** *step*(2) *assms*(2, 3)

**by** (*auto simp: sig-step-def upd-conv-take-nth-drop min-def UN-subset-iff*  
*dest: in-set-takeD in-set-dropD simp flip: id-take-nth-drop*)

**then show** *False* **using** *assms*(1)

**by** (*auto simp: NF-def*)

**qed**

**lemma** *all-ctxt-closed-srstep-conversions* [*simp*]:

*all-ctxt-closed*  $\mathcal{F} ((\text{srstep } \mathcal{F} \mathcal{R})^{\leftrightarrow*})$

**by** (*simp add: all-ctxt-closed-sig-rsteps sig-step-conversion-dist*)

**lemma** *NFP-stepD*:

$\text{NFP } r \implies (a, b) \in r^* \implies (a, c) \in r^* \implies c \in \text{NF } r \implies (b, c) \in r^*$

**by** (*auto simp: NFP-on-def*)

**lemma** *NE-symmetric*:  $\text{NE } r \text{ } s \implies \text{NE } s \text{ } r$

**unfolding** *NE-on-def* **by** *auto*

**lemma** *CE-symmetric*:  $\text{CE } r \text{ } s \implies \text{CE } s \text{ } r$

**unfolding** *CE-on-def* **by** *auto*

Reducing the quantification over rewrite sequences for properties *CR* ...

to rewrite sequences containing at least one root step

**lemma** *all-ctxt-closed-sig-reflE*:

*all-ctxt-closed*  $\mathcal{F} \mathcal{R} \implies \text{funas-term } t \subseteq \mathcal{F} \implies (t, t) \in \mathcal{R}$

**proof** (*induct t*)

**case** (*Fun f ts*)

**from** *Fun(1)[OF nth-mem Fun(2)] Fun(3)*

**have**  $i < \text{length } ts \implies \text{funas-term } (ts ! i) \subseteq \mathcal{F} \ i < \text{length } ts \implies (ts ! i, ts ! i) \in \mathcal{R}$  **for**  $i$

**by** (*auto simp: SUP-le-iff*)

**then show** *?case using all-ctxt-closedD[OF Fun(2)] Fun(3)*

**by** *simp*

**qed** (*simp add: all-ctxt-closed-def*)

**lemma** *all-ctxt-closed-relcomp [intro]*:

$(\bigwedge s t. (s, t) \in \mathcal{R} \implies s \neq t \implies \text{funas-term } s \subseteq \mathcal{F} \wedge \text{funas-term } t \subseteq \mathcal{F}) \implies$

$(\bigwedge s t. (s, t) \in \mathcal{S} \implies s \neq t \implies \text{funas-term } s \subseteq \mathcal{F} \wedge \text{funas-term } t \subseteq \mathcal{F}) \implies$

*all-ctxt-closed*  $\mathcal{F} \mathcal{R} \implies \text{all-ctxt-closed } \mathcal{F} \mathcal{S} \implies \text{all-ctxt-closed } \mathcal{F} (\mathcal{R} \circ \mathcal{S})$

**proof** –

**assume** *funas*: $(\bigwedge s t. (s, t) \in \mathcal{R} \implies s \neq t \implies \text{funas-term } s \subseteq \mathcal{F} \wedge \text{funas-term } t \subseteq \mathcal{F})$

$(\bigwedge s t. (s, t) \in \mathcal{S} \implies s \neq t \implies \text{funas-term } s \subseteq \mathcal{F} \wedge \text{funas-term } t \subseteq \mathcal{F})$

**and** *ctxt-cl*: *all-ctxt-closed*  $\mathcal{F} \mathcal{R}$  *all-ctxt-closed*  $\mathcal{F} \mathcal{S}$

**{fix**  $f ss ts$  **assume** *ass*:  $(f, \text{length } ss) \in \mathcal{F} \ \text{length } ss = \text{length } ts \wedge i. i < \text{length } ts \implies (ss ! i, ts ! i) \in (\mathcal{R} \circ \mathcal{S})$

$\bigwedge i. i < \text{length } ts \implies \text{funas-term } (ts ! i) \subseteq \mathcal{F} \wedge i. i < \text{length } ts \implies \text{funas-term } (ss ! i) \subseteq \mathcal{F}$

**from** *ass(2, 3)* **obtain**  $us$  **where**  $us: \text{length } us = \text{length } ts \wedge i. i < \text{length } ts \implies (ss ! i, us ! i) \in \mathcal{R}$

$\bigwedge i. i < \text{length } ts \implies (us ! i, ts ! i) \in \mathcal{S}$

**using** *Ex-list-of-length-P[of length ts λ x i. (ss ! i, x) ∈ R ∧ (x, ts ! i) ∈ S]*

**by** *auto*

**from** *funas* **have**  $fu: \bigwedge i. i < \text{length } us \implies \text{funas-term } (us ! i) \subseteq \mathcal{F}$  **using** *us ass(4, 5)*

**by** (*auto simp: funas-rel-def*) (*metis in-mono*)

**have**  $(\text{Fun } f ss, \text{Fun } f us) \in \mathcal{R}$  **using** *ass(1, 2, 5) us(1, 2) fu*

**by** (*intro all-ctxt-closedD[OF ctxt-cl(1), of f]*) *auto*

**moreover have**  $(\text{Fun } f us, \text{Fun } f ts) \in \mathcal{S}$  **using** *ass(1, 2, 4) us(1, 3) fu*

**by** (*intro all-ctxt-closedD[OF ctxt-cl(2), of f]*) *auto*

**ultimately have**  $(\text{Fun } f ss, \text{Fun } f ts) \in \mathcal{R} \circ \mathcal{S}$  **by** *auto*}

**moreover**

**{fix**  $x$  **have**  $(\text{Var } x, \text{Var } x) \in \mathcal{R} \ (\text{Var } x, \text{Var } x) \in \mathcal{S}$  **using** *ctxt-cl*

**by** (*auto simp: all-ctxt-closed-def*)

**then have**  $(\text{Var } x, \text{Var } x) \in \mathcal{R} \circ \mathcal{S}$  **by** *auto*}

**ultimately show** *?thesis* **by** (*auto simp: all-ctxt-closed-def*)

**qed**

**abbreviation** *prop-to-rel*  $P \equiv \{(s, t) \mid s t. P s t\}$

**abbreviation**  $\text{prop-mctxt-cl } \mathcal{F} P \equiv \text{all-ctxt-closed } \mathcal{F} (\text{prop-to-rel } P)$

**lemma**  $\text{prop-mctxt-cl-Var}$ :

$\text{prop-mctxt-cl } \mathcal{F} P \implies P (\text{Var } x) (\text{Var } x)$   
**by** ( $\text{simp add: all-ctxt-closed-def}$ )

**lemma**  $\text{prop-mctxt-cl-refl-on}$ :

$\text{prop-mctxt-cl } \mathcal{F} P \implies \text{funas-term } t \subseteq \mathcal{F} \implies P t t$   
**using**  $\text{all-ctxt-closed-sig-reflE}$  **by**  $\text{blast}$

**lemma**  $\text{prop-mctxt-cl-reflcl-on}$ :

$\text{prop-mctxt-cl } \mathcal{F} P \implies \text{funas-term } s \subseteq \mathcal{F} \implies P s s$   
**using**  $\text{all-ctxt-closed-sig-reflE}$  **by**  $\text{blast}$

**lemma**  $\text{reduction-relations-to-root-step}$ :

**assumes**  $\bigwedge s t. (s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R} \implies P s t$   
**and**  $\text{cl: prop-mctxt-cl } \mathcal{F} P$   
**and**  $\text{well: funas-term } s \subseteq \mathcal{F} \text{ funas-term } t \subseteq \mathcal{F}$   
**and**  $\text{steps: } (s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   
**shows**  $P s t$  **using**  $\text{steps well}$

**proof** ( $\text{induct } s \text{ arbitrary: } t$ )

**case**  $(\text{Var } x)$

**have**  $(\text{Var } x, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+ \implies (\text{Var } x, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R}$   
**using**  $\text{nsrsteps-with-root-step-step-on-args}$  **by**  $\text{blast}$   
**from**  $\text{assms}(1)[\text{OF this}]$  **show**  $?case$  **using**  $\text{Var cl}$   
**by** ( $\text{auto simp: rtrancl-eq-or-trancl dest: all-ctxt-closed-sig-reflE}$ )

**next**

**case**  $(\text{Fun } f ss)$  **note**  $\text{IH} = \text{this}$  **show**  $?case$

**proof** ( $\text{cases } \text{Fun } f ss = t$ )

**case**  $\text{True}$  **show**  $?thesis$  **using**  $\text{IH}(2, 4)$  **unfolding**  $\text{True}$   
**by** ( $\text{intro prop-mctxt-cl-reflcl-on}[\text{OF cl}]$ )  $\text{auto}$

**next**

**case**  $\text{False}$

**then have**  $\text{step: } (\text{Fun } f ss, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+ \text{ using } \text{IH}(2)$   
**by** ( $\text{auto simp: refl rtrancl-eq-or-trancl}$ )

**show**  $?thesis$

**proof** ( $\text{cases } (\text{Fun } f ss, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R}$ )

**case**  $\text{False}$

**from**  $\text{nsrsteps-with-root-step-step-on-args}[\text{OF step this}]$  **obtain**  $ts$

**where**  $*[\text{simp}]: t = \text{Fun } f ts$  **and**  $\text{inv: length } ss = \text{length } ts$   
 $\forall i < \text{length } ts. (ss ! i, ts ! i) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$

**by**  $\text{auto}$

**have**  $\text{funas: } (f, \text{length } ts) \in \mathcal{F} \forall i < \text{length } ts. \text{funas-term } (ss ! i) \subseteq \mathcal{F} \wedge$   
 $\text{funas-term } (ts ! i) \subseteq \mathcal{F}$

**using**  $\text{IH}(3, 4)$   $\text{step inv}(1)$  **by** ( $\text{auto simp: UN-subset-iff}$ )

**then have**  $t: \forall i < \text{length } ts. P (ss ! i) (ts ! i)$

**using**  $\text{prop-mctxt-cl-reflcl-on}[\text{OF cl}]$   $\text{IH}(1)$   $\text{inv}$

**by** ( $\text{auto simp: rtrancl-eq-or-trancl}$ )

```

    then show ?thesis unfolding * using funas inv(1) all-ctxt-closedD[OF cl]
    by auto
  qed (auto simp add: assms(1))
qed
qed

```

**abbreviation**  $comp\text{-}rrstep\text{-}rel\ \mathcal{F}\ \mathcal{R}\ \mathcal{S} \equiv srsteps\text{-}with\text{-}root\text{-}step\ \mathcal{F}\ \mathcal{R}\ O\ (srstep\ \mathcal{F}\ \mathcal{S})^* \cup (srstep\ \mathcal{F}\ \mathcal{R})^* O\ srsteps\text{-}with\text{-}root\text{-}step\ \mathcal{F}\ \mathcal{S}$

**abbreviation**  $comp\text{-}rrstep\text{-}rel'\ \mathcal{F}\ \mathcal{R}\ \mathcal{S} \equiv srsteps\text{-}with\text{-}root\text{-}step\ \mathcal{F}\ \mathcal{R}\ O\ (srstep\ \mathcal{F}\ \mathcal{S})^+ \cup (srstep\ \mathcal{F}\ \mathcal{R})^+ O\ srsteps\text{-}with\text{-}root\text{-}step\ \mathcal{F}\ \mathcal{S}$

**lemma** *reduction-join-relations-to-root-step:*

```

assumes  $\bigwedge s\ t. (s, t) \in comp\text{-}rrstep\text{-}rel\ \mathcal{F}\ \mathcal{R}\ \mathcal{S} \implies P\ s\ t$ 
and cl: prop-mctxt-cl  $\mathcal{F}\ P$ 
and well: funas-term  $s \subseteq \mathcal{F}$  funas-term  $t \subseteq \mathcal{F}$ 
and steps:  $(s, t) \in (srstep\ \mathcal{F}\ \mathcal{R})^* O\ (srstep\ \mathcal{F}\ \mathcal{S})^*$ 
shows  $P\ s\ t$  using steps well
proof (induct s arbitrary: t)
  case (Var x)
  have f:  $(Var\ x, t) \in (srstep\ \mathcal{F}\ \mathcal{R})^+ \implies (Var\ x, t) \in comp\text{-}rrstep\text{-}rel\ \mathcal{F}\ \mathcal{R}\ \mathcal{S}$ 
  using nsrsteps-with-root-step-step-on-args[of Var x -  $\mathcal{F}\ \mathcal{R}$ ] unfolding srsteps-with-root-step-def
  by (metis (no-types, lifting) Term.term.simps(4) UnI1 relcomp.relcompI rtrancl-eq-or-trancl)
  have s:  $(Var\ x, t) \in (srstep\ \mathcal{F}\ \mathcal{S})^+ \implies (Var\ x, t) \in comp\text{-}rrstep\text{-}rel\ \mathcal{F}\ \mathcal{R}\ \mathcal{S}$ 
  using nsrsteps-with-root-step-step-on-args[of Var x -  $\mathcal{F}\ \mathcal{S}$ ] unfolding srsteps-with-root-step-def
  by (metis (no-types, lifting) Term.term.simps(4) UnI2 relcomp.simps rtrancl.simps)
  have t:  $(Var\ x, u) \in (srstep\ \mathcal{F}\ \mathcal{R})^+ \implies (u, t) \in (srstep\ \mathcal{F}\ \mathcal{S})^+ \implies (Var\ x, t)$ 
 $\in comp\text{-}rrstep\text{-}rel\ \mathcal{F}\ \mathcal{R}\ \mathcal{S}$  for u
  using nsrsteps-with-root-step-step-on-args[of Var x u  $\mathcal{F}\ \mathcal{R}$ ] unfolding srsteps-with-root-step-def
  by auto (meson relcomp.simps trancl-into-rtrancl)
  show ?case using Var f[THEN assms(1)] s[THEN assms(1)] t[THEN assms(1)]
cl
  by (auto simp: rtrancl-eq-or-trancl prop-mctxt-cl-Var)
next
  case (Fun f ss) note IH = this show ?case
  proof (cases Fun f ss = t)
    case True show ?thesis using IH(2, 3, 4) cl
    by (auto simp: True prop-mctxt-cl-refl-on)
  next
    case False
  obtain u where  $u: (Fun\ f\ ss, u) \in (srstep\ \mathcal{F}\ \mathcal{R})^* (u, t) \in (srstep\ \mathcal{F}\ \mathcal{S})^*$  using
  IH(2) by auto
  show ?thesis
  proof (cases (Fun f ss, u)  $\in srsteps\text{-}with\text{-}root\text{-}step\ \mathcal{F}\ \mathcal{R}$ )
    case True

```

```

then have (Fun f ss, t) ∈ comp-rrstep-rel  $\mathcal{F} \mathcal{R} \mathcal{S}$  using u
  by (auto simp: srsteps-with-root-step-def)
from assms(1)[OF this] show ?thesis by simp
next
case False note ntfst = this show ?thesis
proof (cases (u, t) ∈ srsteps-with-root-step  $\mathcal{F} \mathcal{S}$ )
  case True
    then have (Fun f ss, t) ∈ comp-rrstep-rel  $\mathcal{F} \mathcal{R} \mathcal{S}$  using u unfolding
srsteps-with-root-step-def
    by blast
  from assms(1)[OF this] show ?thesis by simp
next
case False note no-root = False ntfst
show ?thesis
proof (cases Fun f ss = u ∨ u = t)
  case True
    from assms(1) have f:  $\bigwedge s t. (s, t) \in srsteps-with-root-step \mathcal{F} \mathcal{R} \implies P$ 
s t
    and s:  $\bigwedge s t. (s, t) \in srsteps-with-root-step \mathcal{F} \mathcal{S} \implies P s t$  unfolding
srsteps-with-root-step-def
    by blast+
  have u = t  $\implies$  ?thesis using u cl IH(3, 4)
  by (intro reduction-relations-to-root-step[OF f]) auto
  moreover have Fun f ss = u  $\implies$  ?thesis using u cl IH(3, 4)
  by (intro reduction-relations-to-root-step[OF s]) auto
  ultimately show ?thesis using True by auto
next
case False
  then have steps: (Fun f ss, u) ∈ (srstep  $\mathcal{F} \mathcal{R}$ )+ (u, t) ∈ (srstep  $\mathcal{F} \mathcal{S}$ )+
using u
  by (auto simp: rtrancl-eq-or-trancl)
  obtain ts us
    where [simp]: u = Fun f us and inv-u: length ss = length us  $\forall i <$ 
length ts. (ss ! i, us ! i) ∈ (srstep  $\mathcal{F} \mathcal{R}$ )*
    and [simp]: t = Fun f ts and inv-t: length us = length ts  $\forall i <$ 
length
ts. (us ! i, ts ! i) ∈ (srstep  $\mathcal{F} \mathcal{S}$ )*
    using nsrsteps-with-root-step-step-on-args[OF steps(1) no-root(2)]
    using nsrsteps-with-root-step-step-on-args[OF steps(2) no-root(1)]
    by auto
  from inv-u inv-t cl IH(3, 4) have t:  $\forall i <$  length ts. P (ss ! i) (ts ! i)
  by (auto simp: UN-subset-iff intro!: IH(1)[OF nth-mem, of i ts ! i for i])
  moreover have (f, length ts) ∈  $\mathcal{F}$  using IH(4) by auto
  ultimately show ?thesis using IH(3, 4) inv-u inv-t all-ctxt-closedD[OF
cl]
    by (auto simp: UN-subset-iff)
qed
qed
qed
qed

```

**qed**

— Reducing search space for *commute* to conversions involving root steps

**definition** *commute-redp*  $\mathcal{F} \mathcal{R} \mathcal{S} s t \iff (s, t) \in ((\text{srstep } \mathcal{F} \mathcal{S})^* O ((\text{srstep } \mathcal{F} \mathcal{R})^{-1})^*)$

**declare** *subsetI*[*rule del*]

**lemma** *commute-redp-mctxt-cl*:

*prop-mctxt-cl*  $\mathcal{F} (\text{commute-redp } \mathcal{F} \mathcal{R} \mathcal{S})$

**by** (*auto simp: commute-redp-def rew-converse-inwards*

*dest: sigstep-trancl-funas intro!: all-ctxt-closed-relcomp*)

**declare** *subsetI*[*intro!*]

**lemma** *commute-rrstep-intro*:

**assumes**  $\bigwedge s t. (s, t) \in \text{comp-rrstep-rel}' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{S} \implies \text{commute-redp } \mathcal{F} \mathcal{R} \mathcal{S} s t$

**shows** *commute* (*srstep*  $\mathcal{F} \mathcal{R}$ ) (*srstep*  $\mathcal{F} \mathcal{S}$ )

**proof** —

**have** [*simp*]:  $x \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{U} \implies x \in (\text{srstep } \mathcal{F} \mathcal{U})^* O \mathcal{L}^*$  **for**  $x \in \mathcal{U} \mathcal{L}$

**by** (*cases x*) (*auto dest!: srsteps-with-root-step-srsteps-eqD*)

**have** [*simp*]:  $x \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{U} \implies x \in \mathcal{L}^* O (\text{srstep } \mathcal{F} \mathcal{U})^*$  **for**  $x \in \mathcal{U} \mathcal{L}$

**by** (*cases x*) (*auto dest!: srsteps-with-root-step-srsteps-eqD*)

**have** *red*:  $\bigwedge s t. (s, t) \in \text{comp-rrstep-rel } \mathcal{F} (\mathcal{R}^{-1}) \mathcal{S} \implies \text{commute-redp } \mathcal{F} \mathcal{R} \mathcal{S} s t$  **using** *assms*

**unfolding** *commute-redp-def srstep-converse-dist*

**by** (*auto simp: rtrancl-eq-or-trancl blast+*)

**have** *comI*:  $(\bigwedge s t. (s, t) \in ((\text{srstep } \mathcal{F} (\mathcal{R}^{-1}))^*) O (\text{srstep } \mathcal{F} \mathcal{S})^* \implies \text{commute-redp } \mathcal{F} \mathcal{R} \mathcal{S} s t) \implies$

*commute* (*srstep*  $\mathcal{F} \mathcal{R}$ ) (*srstep*  $\mathcal{F} \mathcal{S}$ )

**by** (*auto simp: commute-redp-def commute-def subsetD rew-converse-inwards*)

**show** *?thesis*

**using** *reduction-join-relations-to-root-step*[*OF red commute-redp-mctxt-cl, of*  $\mathcal{R}^{-1} \mathcal{S}$ ]

**by** (*intro comI, auto*) (*metis (no-types, lifting) commute-redp-def relcompI rew-converse-inwards sigstep-trancl-funas srstep-converse-dist*)

**qed**

**lemma** *commute-to-rrstep*:

**assumes** *commute* (*srstep*  $\mathcal{F} \mathcal{R}$ ) (*srstep*  $\mathcal{F} \mathcal{S}$ )

**shows**  $\bigwedge s t. (s, t) \in \text{comp-rrstep-rel } \mathcal{F} (\mathcal{R}^{-1}) \mathcal{S} \implies \text{commute-redp } \mathcal{F} \mathcal{R} \mathcal{S} s t$  **using** *assms*

**unfolding** *commute-def commute-redp-def srstep-converse-dist*

**by** (*auto simp: srstep-converse-dist dest: srsteps-with-root-step-srsteps-eqD*)

— Reducing search space for *CR* to conversions involving root steps

**lemma** *CR-Aux*:

**assumes**  $\bigwedge s t. (s, t) \in (\text{srstep } \mathcal{F} (\mathcal{R}^{-1}))^* \text{ } O \text{ } \text{srsteps-with-root-step } \mathcal{F} \mathcal{R} \implies$   
*commute-redp*  $\mathcal{F} \mathcal{R} \mathcal{R} s t$   
**shows**  $\bigwedge s t. (s, t) \in \text{comp-rrstep-rel } \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R} \implies \text{commute-redp } \mathcal{F} \mathcal{R} \mathcal{R} s t$   
**proof** –  
**have** *sym*: *commute-redp*  $\mathcal{F} \mathcal{R} \mathcal{R} s t \implies \text{commute-redp } \mathcal{F} \mathcal{R} \mathcal{R} t s$  **for**  $s t$   
**by** (*auto simp*: *commute-redp-def*) (*metis converseI relcomp.relcompI rtrancl-converse*  
*rtrancl-converseD*)  
**{fix**  $s t$  **assume**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^* \text{ } O \text{ } \text{srsteps-with-root-step } \mathcal{F} (\mathcal{R}^{-1})$   
**then have** *commute-redp*  $\mathcal{F} \mathcal{R} \mathcal{R} s t$  **unfolding** *commute-redp-def*  
**by** (*auto simp*: *srsteps-with-root-step-def rew-converse-inwards dest!*: *srrstep-to-srstep*)  
**note**  $*$  = *this*  
**{fix**  $s t$  **assume** *ass*:  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} (\mathcal{R}^{-1}) \text{ } O \text{ } (\text{srstep } \mathcal{F} \mathcal{R})^*$   
**have** [*dest!*]:  $(u, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^* \implies (t, u) \in (\text{sig-step } \mathcal{F} ((\text{rstep } \mathcal{R})^{-1}))^*$   
**for**  $u$   
**by** (*metis rew-converse-outwards rtrancl-converseI srstep-converse-dist*)  
**from** *ass* **have**  $(t, s) \in (\text{srstep } \mathcal{F} (\mathcal{R}^{-1}))^* \text{ } O \text{ } \text{srsteps-with-root-step } \mathcal{F} \mathcal{R}$   
**unfolding** *srsteps-with-root-step-def rstep-converse-dist*  
**by** (*metis (mono-tags, lifting) O-assoc converse.simps converse-converse con-*  
*verse-inward(1) converse-relcomp rew-converse-outwards(1, 2) sig-step-converse-rstep*)  
**from** *assms*[*OF this*] **have** *commute-redp*  $\mathcal{F} \mathcal{R} \mathcal{R} s t$  **using** *sym* **by** *blast*  
**then show**  $\bigwedge s t. (s, t) \in \text{comp-rrstep-rel } \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R} \implies \text{commute-redp } \mathcal{F} \mathcal{R}$   
 $\mathcal{R} s t$  **unfolding** *srsteps-with-root-step-def*  
**by** (*metis UnE assms srsteps-with-root-step-def*)  
**qed**

**lemma** *CR-rrstep-intro*:

**assumes**  $\bigwedge s t. (s, t) \in (\text{srstep } \mathcal{F} (\mathcal{R}^{-1}))^+ \text{ } O \text{ } \text{srsteps-with-root-step } \mathcal{F} \mathcal{R} \implies$   
*commute-redp*  $\mathcal{F} \mathcal{R} \mathcal{R} s t$   
**shows** *CR*  $(\text{srstep } \mathcal{F} \mathcal{R})$   
**proof** –  
**{fix**  $s u$  **assume**  $(s, u) \in (\text{srstep } \mathcal{F} (\mathcal{R}^{-1}))^* \text{ } O \text{ } \text{srsteps-with-root-step } \mathcal{F} \mathcal{R}$   
**then obtain**  $t$  **where**  $a$ :  $(s, t) \in (\text{srstep } \mathcal{F} (\mathcal{R}^{-1}))^* (t, u) \in \text{srsteps-with-root-step}$   
 $\mathcal{F} \mathcal{R}$  **by** *blast*  
**have** *commute-redp*  $\mathcal{F} \mathcal{R} \mathcal{R} s u$   
**proof** (*cases*  $s = t$ )  
**case** [*simp*]: *True*  
**from** *srsteps-with-root-step-srstepsD*[*OF a(2)*] **show** *?thesis*  
**by** (*auto simp*: *commute-redp-def*)  
**next**  
**case** *False*  
**then have**  $(s, t) \in (\text{srstep } \mathcal{F} (\mathcal{R}^{-1}))^+ \text{ } \text{using } a(1)$  **unfolding** *rtrancl-eq-or-trancl*  
**by** *simp*  
**then show** *?thesis* **using** *assms a(2)* **by** *blast*  
**qed}**  
**from** *commute-rrstep-intro*[*OF CR-Aux*[*OF this*]]  
**show** *?thesis* **unfolding** *CR-iff-self-commute*  
**by** (*metis Un-iff reflcl-trancl relcomp-distrib relcomp-distrib2*)  
**qed**

**lemma** *CR-to-rrstep*:  
**assumes** *CR* (*srstep*  $\mathcal{F}$   $\mathcal{R}$ )  
**shows**  $\bigwedge s t. (s, t) \in \text{comp-rrstep-rel } \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R} \implies \text{commute-redp } \mathcal{F} \mathcal{R} \mathcal{R} s t$   
**using** *assms*  
**using** *commute-to-rrstep*[*OF* *assms*[*unfolded CR-iff-self-commute*]]  
**by** *simp*

— Reducing search space for *NFP* to conversions involving root steps

**definition** *NFP-redp* **where**

*NFP-redp*  $\mathcal{F} \mathcal{R} s t \longleftrightarrow t \in \text{NF } (\text{srstep } \mathcal{F} \mathcal{R}) \longrightarrow (s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$

**lemma** *prop-mctxt-cl-NFP-redp*:

*prop-mctxt-cl*  $\mathcal{F}$  (*NFP-redp*  $\mathcal{F}$   $\mathcal{R}$ )

**proof** —

**{fix** *f ts ss* **assume** *sig*: (*f*, *length ss*)  $\in \mathcal{F}$  *length ts* = *length ss*  
**and** *steps*:  $\forall i < \text{length } ss. ss ! i \in \text{NF } (\text{srstep } \mathcal{F} \mathcal{R}) \longrightarrow (ts ! i, ss ! i) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   
**and** *funas*:  $\forall i < \text{length } ss. \text{funas-term } (ts ! i) \subseteq \mathcal{F} \wedge \text{funas-term } (ss ! i) \subseteq \mathcal{F}$   
**and** *NF*: *Fun f ss*  $\in \text{NF } (\text{srstep } \mathcal{F} \mathcal{R})$   
**from** *steps* **have** *steps*:  $i < \text{length } ss \implies (ts ! i, ss ! i) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$  **for** *i*  
**using** *sig funas NF-srstep-args*[*OF* *NF*]  
**by** (*auto simp: UN-subset-iff*) (*metis in-set-idx*)  
**then** **have** (*Fun f ts*, *Fun f ss*)  $\in (\text{srstep } \mathcal{F} \mathcal{R})^*$  **using** *sig*  
**by** (*metis all-ctxt-closed-def all-ctxt-closed-sig-rsteps funas le-sup-iff*)}  
**then** **show** *?thesis*  
**by** (*auto simp: NFP-redp-def all-ctxt-closed-def*)

**qed**

**lemma** *NFP-rrstep-intro*:

**assumes**  $\bigwedge s t. (s, t) \in \text{comp-rrstep-rel}' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R} \implies \text{NFP-redp } \mathcal{F} \mathcal{R} s t$

**shows** *NFP* (*srstep*  $\mathcal{F}$   $\mathcal{R}$ )

**proof** —

**from** *assms* **have** *red*:  $\bigwedge t u. (t, u) \in \text{comp-rrstep-rel } \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R} \implies \text{NFP-redp } \mathcal{F} \mathcal{R} t u$

**apply** (*auto simp: NFP-redp-def rtrancl-eq-or-trancl*)

**apply** (*metis NF-no-trancl-step converseD srstep-converse-dist srsteps-with-root-step-srstepsD trancl-converse*)

**apply** *blast*

**apply** (*meson NF-no-trancl-step srsteps-with-root-step-srstepsD*)

**by** *blast*

**have**  $\bigwedge s t. (s, t) \in (\text{sig-step } \mathcal{F} ((\text{rstep } \mathcal{R})^{-1}))^* \text{O } (\text{srstep } \mathcal{F} \mathcal{R})^* \implies \text{NFP-redp } \mathcal{F} \mathcal{R} s t$

**using** *reduction-join-relations-to-root-step*[*OF* *red prop-mctxt-cl-NFP-redp*, *of*  $\mathcal{R}^{-1} \mathcal{R}$ ]

**by** (*auto simp: NFP-redp-def*) (*metis (no-types, lifting) relcomp.relcompI rstep-converse-dist rtranclD srstepsD*)



**then show** *?thesis unfolding NFP-on-def NFP-redp-def*  
**by** (*auto simp: normalizability-def*) (*metis meetI meet-def rstep-converse-dist srstep-converse-dist*)  
**qed**

**lemma** *NFP-lift-to-conversion:*  
**assumes** *NFP*  $r (s, t) \in (r^{\leftrightarrow})^*$  **and**  $t \in NF\ r$   
**shows**  $(s, t) \in r^*$  **using** *assms(2, 3)*  
**proof** (*induct rule: converse-rtrancl-induct*)  
**case** (*step s u*)  
**then have**  $(u, t) \in r^!$  **by** *auto*  
**then show** *?case using assms(1) step(1) unfolding NFP-on-def*  
**by** *auto*  
**qed** *simp*

**lemma** *NFP-to-rrstep:*  
**assumes** *NFP* (*srstep*  $\mathcal{F}$   $\mathcal{R}$ )  
**shows**  $\bigwedge s\ t. (s, t) \in srsteps\text{-with-root-step } \mathcal{F} (\mathcal{R}^{\leftrightarrow}) \implies NFP\text{-redp } \mathcal{F} \mathcal{R} s\ t$   
**using** *assms*  
**using** *NFP-lift-to-conversion[OF assms] unfolding NFP-redp-def srsteps-with-root-step-def*  
**by** *auto (metis (no-types, lifting) r-into-rtrancl rstep-converse-dist rtrancl-trans srrstep-to-srstep srstep-symcl-dist)*

— Reducing search space for *UNC* to conversions involving root steps

**definition** *UN-redp*  $\mathcal{F} \mathcal{R} s\ t \iff s \in NF (srstep\ \mathcal{F}\ \mathcal{R}) \wedge t \in NF (srstep\ \mathcal{F}\ \mathcal{R})$   
 $\longrightarrow s = t$

**lemma** *prop-mctx-cl-UN-redp:*  
*prop-mctx-cl*  $\mathcal{F}$  (*UN-redp*  $\mathcal{F}$   $\mathcal{R}$ )  
**proof** –  
**{fix**  $f\ ts\ ss$  **assume**  $sig: (f, length\ ss) \in \mathcal{F}$   $length\ ts = length\ ss$   
**and**  $steps: \forall i < length\ ss. ts\ !\ i \in NF (srstep\ \mathcal{F}\ \mathcal{R}) \wedge ss\ !\ i \in NF (srstep\ \mathcal{F}\ \mathcal{R}) \longrightarrow ts\ !\ i = ss\ !\ i$   
**and**  $funas: \forall i < length\ ss. funas\text{-term } (ts\ !\ i) \subseteq \mathcal{F} \wedge funas\text{-term } (ss\ !\ i) \subseteq \mathcal{F}$   
**and** *NF:*  $Fun\ f\ ts \in NF (srstep\ \mathcal{F}\ \mathcal{R})\ Fun\ f\ ss \in NF (srstep\ \mathcal{F}\ \mathcal{R})$   
**from**  $steps$  **have**  $steps: i < length\ ss \implies ts\ !\ i = ss\ !\ i$  **for**  $i$   
**using**  $sig\ funas\ NF\text{-srstep-args}[OF\ NF(1)]\ NF\text{-srstep-args}[OF\ NF(2)]$   
**by** (*auto simp: UN-subset-iff*) (*metis in-set-idx*)  
**then have**  $Fun\ f\ ts = Fun\ f\ ss$  **using**  $sig(2)$   
**by** (*simp add: nth-equalityI*)  
**then show** *?thesis*  
**by** (*auto simp: UN-redp-def all-cctx-closed-def*)  
**qed**

**lemma** *UNC-rrstep-intro:*  
**assumes**  $\bigwedge s\ t. (s, t) \in srsteps\text{-with-root-step } \mathcal{F} (\mathcal{R}^{\leftrightarrow}) \implies UN\text{-redp } \mathcal{F} \mathcal{R} s\ t$

**shows**  $UNC (srstep \mathcal{F} \mathcal{R})$   
**proof** –  
**have**  $\bigwedge s t. (s, t) \in (srstep \mathcal{F} (\mathcal{R}^{\leftrightarrow}))^* \implies UN-redp \mathcal{F} \mathcal{R} s t$   
**using**  $reduction-relations-to-root-step[OF assms(1) prop-mctxt-cl-UN-redp, of \mathcal{R}^{\leftrightarrow}]$   
**by**  $(auto simp: UN-redp-def) (meson rtranclD srstepsD)$   
**then show** *?thesis unfolding*  $UNC-def UN-redp-def$   
**by**  $(auto simp: sig-step-conversion-dist)$   
**qed**

**lemma**  $UNC-to-rrstep$ :  
**assumes**  $UNC (srstep \mathcal{F} \mathcal{R})$   
**shows**  $\bigwedge s t. (s, t) \in srsteps-with-root-step \mathcal{F} (\mathcal{R}^{\leftrightarrow}) \implies UN-redp \mathcal{F} \mathcal{R} s t$   
**using**  $assms unfolding UNC-def UN-redp-def srsteps-with-root-step-def$   
**by**  $(auto dest!: srrstep-to-srstep symcl-srstep-conversion symcl-srsteps-conversion)$   
 $(metis (no-types, opaque-lifting) conversion-def rtrancl-trans)$

— Reducing search space for  $UNF$  to conversions involving root steps

**lemma**  $UNF-rrstep-intro$ :  
**assumes**  $\bigwedge t u. (t, u) \in comp-rrstep-rel' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R} \implies UN-redp \mathcal{F} \mathcal{R} t u$   
**shows**  $UNF (srstep \mathcal{F} \mathcal{R})$   
**proof** –  
**from**  $assms$  **have**  $red: \bigwedge t u. (t, u) \in comp-rrstep-rel \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R} \implies UN-redp \mathcal{F} \mathcal{R} t u$   
**apply**  $(auto simp: UN-redp-def rtrancl-eq-or-trancl)$   
**apply**  $(metis NF-no-trancl-step converseD srstep-converse-dist srsteps-with-root-step-srstepsD trancl-converse)$   
**apply**  $blast$   
**apply**  $(meson NF-no-trancl-step srsteps-with-root-step-srstepsD)$   
**by**  $blast$   
**have**  $\bigwedge s t. (s, t) \in (sig-step \mathcal{F} ((rstep \mathcal{R})^{-1}))^* O (srstep \mathcal{F} \mathcal{R})^* \implies UN-redp \mathcal{F} \mathcal{R} s t$   
**using**  $reduction-join-relations-to-root-step[OF red prop-mctxt-cl-UN-redp, of \mathcal{R}^{-1} \mathcal{R}]$   
**by**  $(auto simp: UN-redp-def) (metis (no-types, lifting) relcomp.relcompI rstep-converse-dist rtranclD srstepsD)$   
**then show** *?thesis unfolding*  $UNF-on-def UN-redp-def$   
**by**  $(auto simp: normalizability-def) (metis meetI meet-def rstep-converse-dist srstep-converse-dist)$   
**qed**

**lemma**  $UNF-to-rrstep$ :  
**assumes**  $UNF (srstep \mathcal{F} \mathcal{R})$   
**shows**  $\bigwedge s t. (s, t) \in comp-rrstep-rel \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R} \implies UN-redp \mathcal{F} \mathcal{R} s t$   
**using**  $assms unfolding UNF-on-def UN-redp-def normalizability-def srsteps-with-root-step-def$   
**by**  $(auto simp flip: srstep-converse-dist dest!: srrstep-to-srstep)$   
 $(metis (no-types, lifting) rstep-converse-dist rtrancl.rtrancl-into-rtrancl rtrancl-converseD)$

*rtrancl-idemp srstep-converse-dist*)+

— Reducing search space for *CE* to conversions involving root steps

**lemma** *CE-rrstep-intro*:

**assumes**  $\bigwedge s t. (s, t) \in \text{srsteps-with-root-step } \mathcal{F} (\mathcal{R}^{\leftrightarrow}) \implies (s, t) \in (\text{srstep } \mathcal{F} \mathcal{S})^{\leftrightarrow*}$   
**and**  $\bigwedge s t. (s, t) \in \text{srsteps-with-root-step } \mathcal{F} (\mathcal{S}^{\leftrightarrow}) \implies (s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^{\leftrightarrow*}$   
**shows** *CE* (*srstep*  $\mathcal{F} \mathcal{R}$ ) (*srstep*  $\mathcal{F} \mathcal{S}$ )  
**using** *reduction-relations-to-root-step*[*OF* *assms*(1), **where**  $?s1 = \lambda s t. s$  **and**  $?t1 = \lambda s t. t$ , of  $\mathcal{F} \mathcal{R}^{\leftrightarrow}$ ]  
**using** *reduction-relations-to-root-step*[*OF* *assms*(2), **where**  $?s1 = \lambda s t. s$  **and**  $?t1 = \lambda s t. t$ , of  $\mathcal{F} \mathcal{S}^{\leftrightarrow}$ ]  
**by** (*auto simp: CE-on-def*)  
(*metis converseI conversion-converse rtrancl-eq-or-trancl sig-step-conversion-dist sigstep-trancl-funas*(1, 2))+

**lemma** *CE-to-rrstep*:

**assumes** *CE* (*srstep*  $\mathcal{F} \mathcal{R}$ ) (*srstep*  $\mathcal{F} \mathcal{S}$ )  
**shows**  $\bigwedge s t. (s, t) \in \text{srsteps-with-root-step } \mathcal{F} (\mathcal{R}^{\leftrightarrow}) \implies (s, t) \in (\text{srstep } \mathcal{F} \mathcal{S})^{\leftrightarrow*}$   
 $\bigwedge s t. (s, t) \in \text{srsteps-with-root-step } \mathcal{F} (\mathcal{S}^{\leftrightarrow}) \implies (s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^{\leftrightarrow*}$   
**using** *assms unfolding CE-on-def srsteps-with-root-step-def*  
**by** (*auto simp flip: srstep-converse-dist dest!: srrstep-to-srstep symcl-srsteps-conversion symcl-srstep-conversion*)  
(*metis converse-rtrancl-into-rtrancl conversion-rtrancl*)+

— Reducing search space for *NE* to conversions involving root steps

**definition** *NE-redp where*

*NE-redp*  $\mathcal{F} \mathcal{R} \mathcal{S} s t \longleftrightarrow t \in \text{NF} (\text{srstep } \mathcal{F} \mathcal{R}) \longrightarrow t \in \text{NF} (\text{srstep } \mathcal{F} \mathcal{R}) \longrightarrow (s, t) \in (\text{srstep } \mathcal{F} \mathcal{S})^*$

**lemma** *prop-mctx-cl-NE-redp*:

*prop-mctx-cl*  $\mathcal{F} (\text{NE-redp } \mathcal{F} \mathcal{R} \mathcal{S})$

**proof** —

**{fix**  $f ts ss$  **assume**  $\text{sig}: (f, \text{length } ss) \in \mathcal{F} \text{ length } ts = \text{length } ss$   
**and**  $\text{steps}: \forall i < \text{length } ss. ss ! i \in \text{NF} (\text{srstep } \mathcal{F} \mathcal{R}) \longrightarrow (ts ! i, ss ! i) \in (\text{srstep } \mathcal{F} \mathcal{S})^*$   
**and**  $\text{funas}: \forall i < \text{length } ss. \text{funas-term } (ts ! i) \subseteq \mathcal{F} \wedge \text{funas-term } (ss ! i) \subseteq \mathcal{F}$   
**and**  $\text{NF}: \text{Fun } f ss \in \text{NF} (\text{srstep } \mathcal{F} \mathcal{R})$   
**from**  $\text{steps}$  **have**  $\text{steps}: i < \text{length } ss \implies (ts ! i, ss ! i) \in (\text{srstep } \mathcal{F} \mathcal{S})^*$  **for**  $i$   
**using**  $\text{sig funas NF-srstep-args}$ [*OF* *NF*]  
**by** (*auto simp: UN-subset-iff*) (*metis in-set-idx*)  
**then** **have** ( $\text{Fun } f ts, \text{Fun } f ss$ )  $\in (\text{srstep } \mathcal{F} \mathcal{S})^*$  **using**  $\text{sig}$   
**by** (*metis all-ctx-closed-def all-ctx-closed-sig-rsteps funas le-sup-iff*)}  
**then** **show**  $?thesis$   
**by** (*auto simp: all-ctx-closed-def NE-redp-def*)

qed

**lemma** *NE-rrstep-intro*:

**assumes**  $\bigwedge s t. (s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R} \implies \text{NE-redp } \mathcal{F} \mathcal{R} \mathcal{S} s t$   
**and**  $\bigwedge s t. (s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{S} \implies \text{NE-redp } \mathcal{F} \mathcal{S} \mathcal{R} s t$   
**and**  $\text{NF } (\text{srstep } \mathcal{F} \mathcal{R}) = \text{NF } (\text{srstep } \mathcal{F} \mathcal{S})$   
**shows**  $\text{NE } (\text{srstep } \mathcal{F} \mathcal{R}) (\text{srstep } \mathcal{F} \mathcal{S})$   
**using** *assms(3)*  
**using** *reduction-relations-to-root-step[OF assms(1) prop-mctxt-cl-NE-redp, of  $\mathcal{R}$ ]*  
**using** *reduction-relations-to-root-step[OF assms(2) prop-mctxt-cl-NE-redp, of  $\mathcal{S}$ ]*  
**by** (*auto simp: NE-on-def NE-redp-def normalizability-def*)  
(*metis rtrancl.rtrancl-refl sigstep-trancl-funass*)+

**lemma** *NE-to-rrstep*:

**assumes**  $\text{NE } (\text{srstep } \mathcal{F} \mathcal{R}) (\text{srstep } \mathcal{F} \mathcal{S})$   
**shows**  $\bigwedge s t. (s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R} \implies \text{NE-redp } \mathcal{F} \mathcal{R} \mathcal{S} s t$   
 $\bigwedge s t. (s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{S} \implies \text{NE-redp } \mathcal{F} \mathcal{S} \mathcal{R} s t$   
**using** *assms unfolding NE-on-def NE-redp-def srsteps-with-root-step-def*  
**by** (*auto simp: normalizability-def simp flip: srstep-converse-dist*)  
*dest!: srrstep-to-srstep*) (*meson converse-rtrancl-into-rtrancl rtrancl-trans*)+

**lemma** *NE-NF-eq*:

$\text{NE } \mathcal{R} \mathcal{S} \implies \text{NF } \mathcal{R} = \text{NF } \mathcal{S}$   
**by** (*auto simp: NE-on-def NF-def normalizability-def*)

— Reducing search space for *SCR* and *WCR* involving root steps

**abbreviation**  $\text{SCRp } \mathcal{F} \mathcal{R} t u \equiv \exists v. (t, v) \in (\text{srstep } \mathcal{F} \mathcal{R})^= \wedge (u, v) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$

**lemma** *SCR-rrstep-intro*:

**assumes**  $\bigwedge s t u. (s, t) \in \text{sig-step } \mathcal{F} (\text{rrstep } \mathcal{R}) \implies (s, u) \in \text{srstep } \mathcal{F} \mathcal{R} \implies \text{SCRp } \mathcal{F} \mathcal{R} t u$   
**and**  $\bigwedge s t u. (s, t) \in \text{srstep } \mathcal{F} \mathcal{R} \implies (s, u) \in \text{sig-step } \mathcal{F} (\text{rrstep } \mathcal{R}) \implies \text{SCRp } \mathcal{F} \mathcal{R} t u$

**shows**  $\text{SCR } (\text{srstep } \mathcal{F} \mathcal{R})$

**proof** —

{**fix**  $s t u$  **assume**  $\text{step}: (s, t) \in \text{srstep } \mathcal{F} \mathcal{R} (s, u) \in \text{srstep } \mathcal{F} \mathcal{R}$   
**from**  $\text{step}(1)$  **obtain**  $p l r \sigma$  **where**  $st: p \in \text{poss } s (l, r) \in \mathcal{R} s \mid - p = l \cdot \sigma t = s[p \leftarrow r \cdot \sigma]$   
**using** *rstep-to-pos-replace[of  $s t \mathcal{R}$ ] unfolding sig-step-def by blast*  
**from**  $\text{step}(2)$  **obtain**  $q l2 r2 \sigma2$  **where**  $su: q \in \text{poss } s (l2, r2) \in \mathcal{R} s \mid - q = l2 \cdot \sigma2 u = s[q \leftarrow r2 \cdot \sigma2]$   
**using** *rstep-to-pos-replace[of  $s u \mathcal{R}$ ] unfolding sig-step-def by blast*  
**from**  $\text{step } st su$  **have**  $\text{funas}: \text{funas-term } s \subseteq \mathcal{F} \text{ funas-term } t \subseteq \mathcal{F} \text{ funas-term } u \subseteq \mathcal{F}$   
**by** (*auto dest: srstepD*)  
**have**  $\text{funas2} : \text{funas-term } (r2 \cdot \sigma2) \subseteq \mathcal{F}$  **using** *funas-term-replace-at-lower[OF*

```

su(1)]
  using funas(3) unfolding su(4) by blast
  consider (a)  $p \leq_p q$  | (b)  $q \leq_p p$  | (c)  $p \perp q$ 
  using position-par-def by blast
  then have SCRp  $\mathcal{F} \mathcal{R} t u$ 
  proof cases
  case a
  from a have up:  $p \in \text{poss } u$  using st(1) su(1) unfolding st(4) su(4)
    by (metis pos-replace-at-pres position-less-eq-def poss-append-poss)
  let ?C = ctxt-at-pos s p have fc: funas-ctxt ?C  $\subseteq \mathcal{F}$  using funas(1) st(1)
    by (metis ctxt-at-pos-subt-at-id funas-ctxt-apply le-sup-iff)
  from funas have funas: funas-term (s |- p)  $\subseteq \mathcal{F}$  funas-term (t |- p)  $\subseteq \mathcal{F}$ 
  funas-term (u |- p)  $\subseteq \mathcal{F}$ 
    using a st(1) pos-replace-at-pres[OF st(1)] up unfolding st(4) su(4)
    by (intro funas-term-subterm-atI, blast+)+
  have (s |- p, t |- p)  $\in \text{sig-step } \mathcal{F}$  (rrstep  $\mathcal{R}$ ) unfolding st(4) su(4) using
  st(1 - 3) su(1 - 3) funas
    by (metis poss-of-termE poss-of-term-replace-term-at rrstep.intros sig-stepI
  st(4))
  moreover have (s |- p, u |- p)  $\in \text{srstep } \mathcal{F} \mathcal{R}$  unfolding st(4) su(4) using
  st(1 - 3) su(1 - 3) funas
    by (smt (verit, best) a ctxt-at-pos-subt-at-pos ctxt-of-pos-term-apply-replace-at-ident
  position-less-eq-def
  poss-append-poss replace-subterm-at-itself replace-term-at-subt-at-id rstepI
  sig-stepI su(4))
  ultimately obtain v where (t |- p, v)  $\in (\text{srstep } \mathcal{F} \mathcal{R})^=$  (u |- p, v)  $\in (\text{srstep } \mathcal{F} \mathcal{R})^*$ 
    using assms(1) by blast
  from this(1) srsteps-eq-ctxt-closed[OF fc this(2)]
  show ?thesis using a st(1) su(1) srsteps-eq-ctxt-closed[OF fc] unfolding
  st(4) su(4)
    apply (intro exI[of - ?C(v)])
    apply (auto simp: ctxt-of-pos-term-apply-replace-at-ident less-eq-subt-at-replace)
    apply (metis ctxt-of-pos-term-apply-replace-at-ident fc srstep-ctxt-closed)
    done
  next
  case b
  then have up:  $q \in \text{poss } t$  using st(1) su(1) unfolding st(4) su(4)
    by (metis pos-replace-at-pres position-less-eq-def poss-append-poss)
  let ?C = ctxt-at-pos s q have fc: funas-ctxt ?C  $\subseteq \mathcal{F}$  using funas(1) su(1)
    by (metis Un-subset-iff ctxt-at-pos-subt-at-id funas-ctxt-apply)
  from funas have funas: funas-term (s |- q)  $\subseteq \mathcal{F}$  funas-term (t |- q)  $\subseteq \mathcal{F}$ 
  funas-term (u |- q)  $\subseteq \mathcal{F}$ 
    using su(1) pos-replace-at-pres[OF su(1)] up unfolding st(4) su(4)
    by (intro funas-term-subterm-atI, blast+)+
  have (s |- q, t |- q)  $\in \text{srstep } \mathcal{F} \mathcal{R}$  unfolding st(4) su(4) using st(1 - 3)
  su(1 - 3) funas
    by (smt (verit, del-insts) b ctxt-at-pos-subt-at-pos ctxt-of-pos-term-apply-replace-at-ident
  position-less-eq-def poss-append-poss replace-subterm-at-itself replace-term-at-subt-at-id

```

```

rstepI sig-stepI st(4)
  moreover have (s |- q, u |- q) ∈ sig-step F (rrstep R) unfolding st(4)
su(4) using st(1 - 3) su(1 - 3) funas
  by (metis poss-of-termE poss-of-term-replace-term-at rrstep.intros sig-stepI
su(4))
  ultimately obtain v where (t |- q, v) ∈ (srstep F R)= (u |- q, v) ∈ (srstep
F R)*
  using assms(2) by blast
  from this(1) srsteps-eq-ctxt-closed[OF fc this(2)]
  show ?thesis using b st(1) su(1) srsteps-eq-ctxt-closed[OF fc] unfolding
st(4) su(4)
  apply (intro exI[of - ?C⟨v⟩])
  apply (auto simp: ctxt-of-pos-term-apply-replace-at-ident less-eq-subt-at-replace)
  apply (smt (verit, best) ctxt-of-pos-term-apply-replace-at-ident fc less-eq-subt-at-replace
replace-term-at-above replace-term-at-subt-at-id srstep-ctxt-closed)
  done
next
case c
define v where v = t[q ← r2 · σ2]
have funasv: funas-term v ⊆ F using funas su(1) unfolding v-def su(4)
  using funas-term-replace-at-upper funas2 by blast
from c have *: v = u[p ← r · σ] unfolding v-def st(4) su(4) using st(1)
su(1)
  using parallel-replace-term-commute by blast
from c have (t, v) ∈ rstep R unfolding st(4) v-def
  using su(1 - 3) par-pos-replace-pres[OF su(1)]
  by (metis par-pos-replace-term-at pos-replace-to-rstep position-par-def)
moreover from c have (u, v) ∈ rstep R unfolding su(4) *
  using st(1 - 3) par-pos-replace-pres[OF st(1)]
  by (intro pos-replace-to-rstep[of - - l]) (auto simp: par-pos-replace-term-at)
ultimately show ?thesis using funas(2-) funasv
  by auto
qed}
then show ?thesis unfolding SCR-on-def
  by blast
qed

```

**lemma** *SCE-to-rrstep*:

```

  assumes SCR (srstep F R)
  shows  $\bigwedge s t u. (s, t) \in \text{sig-step } F \text{ (rrstep } R) \implies (s, u) \in \text{srstep } F R \implies \text{SCRp}$ 
 $F R t u$ 
   $\bigwedge s t u. (s, t) \in \text{srstep } F R \implies (s, u) \in \text{sig-step } F \text{ (rrstep } R) \implies \text{SCRp}$ 
 $F R t u$ 
  using assms unfolding SCR-on-def srsteps-with-root-step-def
  by (auto simp flip: srstep-converse-dist dest!: srrstep-to-srstep symcl-srsteps-conversion
symcl-srstep-conversion)

```

**lemma** *WCR-rrstep-intro*:

```

  assumes  $\bigwedge s t u. (s, t) \in \text{sig-step } F \text{ (rrstep } R) \implies (s, u) \in \text{srstep } F R \implies (t,$ 

```

$u) \in (srstep \mathcal{F} \mathcal{R})^\downarrow$   
**shows**  $WCR (srstep \mathcal{F} \mathcal{R})$   
**proof** –  
 {**fix**  $s t u$  **assume**  $step: (s, t) \in srstep \mathcal{F} \mathcal{R} (s, u) \in srstep \mathcal{F} \mathcal{R}$   
**from**  $step(1)$  **obtain**  $p l r \sigma$  **where**  $st: p \in poss s (l, r) \in \mathcal{R} s \mid - p = l \cdot \sigma t$   
 $= s[p \leftarrow r \cdot \sigma]$   
**using**  $rstep\text{-to-}\mathit{pos}\text{-replace}[of s t \mathcal{R}]$  **unfolding**  $sig\text{-step}\text{-def}$  **by**  $blast$   
**from**  $step(2)$  **obtain**  $q l2 r2 \sigma2$  **where**  $su: q \in poss s (l2, r2) \in \mathcal{R} s \mid - q =$   
 $l2 \cdot \sigma2 u = s[q \leftarrow r2 \cdot \sigma2]$   
**using**  $rstep\text{-to-}\mathit{pos}\text{-replace}[of s u \mathcal{R}]$  **unfolding**  $sig\text{-step}\text{-def}$  **by**  $blast$   
**from**  $step st su$  **have**  $funas: funas\text{-term } s \subseteq \mathcal{F} funas\text{-term } t \subseteq \mathcal{F} funas\text{-term } u$   
 $\subseteq \mathcal{F}$   
**by**  $(auto\ dest: srstepD)$   
**have**  $funas2 : funas\text{-term } (r2 \cdot \sigma2) \subseteq \mathcal{F}$  **using**  $funas\text{-term}\text{-replace}\text{-at}\text{-lower}[OF$   
 $su(1)]$   
**using**  $funas(3)$  **unfolding**  $su(4)$  **by**  $blast$   
**consider**  $(a) p \leq_p q \mid (b) q \leq_p p \mid (c) p \perp q$   
**using**  $position\text{-par}\text{-def}$  **by**  $blast$   
**then** **have**  $(t, u) \in (srstep \mathcal{F} \mathcal{R})^\downarrow$   
**proof**  $cases$   
**case**  $a$   
**then** **have**  $up: p \in poss u$  **using**  $st(1) su(1)$  **unfolding**  $st(4) su(4)$   
**by**  $(metis\ pos\text{-replace}\text{-at}\text{-pres}\ position\text{-less}\text{-eq}\text{-def}\ poss\text{-append}\text{-poss})$   
**let**  $?C = \mathit{ctxt}\text{-at}\text{-pos } s p$  **have**  $fc: funas\text{-ctxt } ?C \subseteq \mathcal{F}$  **using**  $funas(1) st(1)$   
**by**  $(metis\ Un\text{-subset}\text{-iff}\ \mathit{ctxt}\text{-at}\text{-pos}\text{-subt}\text{-at}\text{-id}\ funas\text{-ctxt}\text{-apply})$   
**from**  $funas$  **have**  $funas: funas\text{-term } (s \mid - p) \subseteq \mathcal{F} funas\text{-term } (t \mid - p) \subseteq \mathcal{F}$   
 $funas\text{-term } (u \mid - p) \subseteq \mathcal{F}$   
**using**  $a\ st(1)\ pos\text{-replace}\text{-at}\text{-pres}[OF\ st(1)]\ up$  **unfolding**  $st(4) su(4)$   
**by**  $(intro\ funas\text{-term}\text{-subterm}\text{-at}I, blast+)\ +$   
**have**  $(s \mid - p, t \mid - p) \in sig\text{-step } \mathcal{F} (rrstep \mathcal{R})$  **unfolding**  $st(4) su(4)$  **using**  
 $st(1 - 3) su(1 - 3) funas$   
**by**  $(metis\ poss\text{-of}\text{-term}E\ poss\text{-of}\text{-term}\text{-replace}\text{-term}\text{-at}\ rrstep.\mathit{intros}\ sig\text{-step}I$   
 $st(4))$   
**moreover** **have**  $(s \mid - p, u \mid - p) \in srstep \mathcal{F} \mathcal{R}$  **unfolding**  $st(4) su(4)$  **using**  
 $st(1 - 3) su(1 - 3) funas$   
**by**  $(smt\ (verit, ccfv\text{-threshold})\ a\ greater\text{-eq}\text{-subt}\text{-at}\text{-replace}\ less\text{-eq}\text{-subt}\text{-at}\text{-replace}$   
 $pos\text{-diff}\text{-append}\text{-itself}$   
 $pos\text{-replace}\text{-to}\text{-rstep}\ position\text{-less}\text{-eq}\text{-def}\ poss\text{-append}\text{-poss}\ replace\text{-term}\text{-at}\text{-subt}\text{-at}\text{-id}$   
 $sig\text{-step}I\ su(4))$   
**ultimately** **have**  $(t \mid - p, u \mid - p) \in (srstep \mathcal{F} \mathcal{R})^\downarrow$   
**using**  $assms(1)$  **by**  $blast$   
**from**  $sig\text{-steps}\text{-join}\text{-ctxt}\text{-closed}[OF\ fc\ this(1)]$   
**show**  $?thesis$  **using**  $a\ st(1) su(1) srstep\text{-ctxt}\text{-closed}[OF\ fc]$  **unfolding**  $st(4)$   
 $su(4)$   
**by**  $(auto\ simp: \mathit{ctxt}\text{-of}\text{-pos}\text{-term}\text{-apply}\text{-replace}\text{-at}\text{-ident}\ less\text{-eq}\text{-subt}\text{-at}\text{-replace})$   
**next**  
**case**  $b$   
**then** **have**  $up: q \in poss t$  **using**  $st(1) su(1)$  **unfolding**  $st(4) su(4)$   
**by**  $(metis\ pos\text{-les}\text{-eq}\text{-append}\text{-diff}\ pos\text{-replace}\text{-at}\text{-pres}\ poss\text{-append}\text{-poss})$

```

    let ?C = ctxt-at-pos s q have fc: funas-ctxt ?C ⊆ F using funas(1) su(1)
      by (metis Un-subset-iff ctxt-at-pos-subt-at-id funas-ctxt-apply)
    from funas have funas: funas-term (s |- q) ⊆ F funas-term (t |- q) ⊆ F
  funas-term (u |- q) ⊆ F
    using su(1) pos-replace-at-pres[OF su(1)] up unfolding st(4) su(4)
    by (intro funas-term-subterm-atI, blast+)+
    have (s |- q, t |- q) ∈ srstep F R unfolding st(4) su(4) using st(1 - 3)
  su(1 - 3) funas
    by (smt (verit, ccfv-SIG) b greater-eq-subt-at-replace less-eq-subt-at-replace
  pos-diff-append-itself
    pos-replace-to-rstep position-less-eq-def poss-append-poss replace-term-at-subt-at-id
  sig-stepI st(4))
    moreover have (s |- q, u |- q) ∈ sig-step F (rrstep R) unfolding st(4)
  su(4) using st(1 - 3) su(1 - 3) funas
    by (metis poss-of-termE poss-of-term-replace-term-at rrstep.intros sig-stepI
  su(4))
    ultimately have (t |- q, u |- q) ∈ (srstep F R)↓
    using assms(1) by blast
    from sig-steps-join-ctxt-closed[OF fc this(1)]
    show ?thesis using b st(1) su(1) srstep-ctxt-closed[OF fc] unfolding st(4)
  su(4)
    by (auto simp: ctxt-of-pos-term-apply-replace-at-ident less-eq-subt-at-replace)
  next
  case c
  define v where v = t[q ← r2 · σ2]
  have funasv: funas-term v ⊆ F using funas su(1) unfolding v-def su(4)
    using funas-term-replace-at-upper funas2 by blast
  from c have *: v = u[p ← r · σ] unfolding v-def st(4) su(4) using st(1)
  su(1)
    using parallel-replace-term-commute by blast
  from c have (t, v) ∈ rstep R unfolding st(4) v-def
    using su(1 - 3) par-pos-replace-pres[OF su(1)]
    by (metis par-pos-replace-term-at pos-replace-to-rstep position-par-def)
  moreover from c have (u, v) ∈ rstep R unfolding su(4) *
    using st(1 - 3) par-pos-replace-pres[OF st(1)]
    by (metis par-pos-replace-term-at pos-replace-to-rstep)
  ultimately show ?thesis using funas(2-) funasv
    by auto
  qed}
  then show ?thesis unfolding WCR-on-def
    by blast
qed

end
theory Rewriting-LLRG-LV-Mondaic
  imports Rewriting
    Replace-Constant
begin

```



### 4.3 Specific results about rewriting under a linear variable-separated system

**lemma** *card-varposs-ground*:

*card (varposs s) = 0*  $\longleftrightarrow$  *ground s*  
**by** (*simp add: finite-varposs varposs-empty-ground*)

**lemma** *poss-of-term-subst-apply-varposs*:

**assumes**  $p \in \text{poss-of-term } (\text{constT } c) (s \cdot \sigma) (c, 0) \notin \text{funas-term } s$   
**shows**  $\exists q. q \in \text{varposs } s \wedge q \leq_p p$  **using** *assms*

**proof** (*induct p arbitrary: s*)

**case** *Nil*

**then show** *?case by (cases s) (auto simp: poss-of-term-def)*

**next**

**case** (*Cons i p*)

**show** *?case using Cons(1)[of args s ! i] Cons(2-)*

**apply** (*cases s*)

**apply** (*auto simp: poss-of-term-def*)

**apply** (*metis position-less-eq-Cons*)**+**

**done**

**qed**

**lemma** *poss-of-term-hole-poss*:

**assumes**  $p \in \text{poss-of-term } t C \langle s \rangle$  **and** *hole-pos C*  $\leq_p p$

**shows**  $p \dashv_p \text{hole-pos } C \in \text{poss-of-term } t s$  **using** *assms*

**proof** (*induct C arbitrary: p*)

**case** (*More f ss C ts*)

**from** *More(3)* **obtain** *ps* **where** [*simp*]:  $p = \text{length } ss \# ps$  **and** *h*: *hole-pos C*  
 $\leq_p ps$

**by** (*metis append-Cons hole-pos.simps(2) less-eq-poss-append-itself pos-les-eq-append-diff*)

**show** *?case using More(1)[OF - h] More(2)*

**by** (*auto simp: poss-of-term-def*)

**qed** *auto*

**lemma** *remove-const-subst-from-match*:

**assumes**  $s \cdot \text{const-subst } c = C \langle l \cdot \sigma \rangle (c, 0) \notin \text{funas-term } l$  *linear-term l*

**shows**  $\exists D \tau. s = D \langle l \cdot \tau \rangle$  **using** *assms*

**proof** (*induct card (varposs s) arbitrary: s*)

**case** (*Suc x*)

**from** *Suc(2)* **obtain** *p ps* **where** *varposs: varposs s = insert p ps*  $p \notin ps$

**by** (*metis card-Suc-eq*)

**let**  $?s = s[p \leftarrow \text{Fun } c []]$  **have** *vp*:  $p \in \text{varposs } s$  **using** *varposs* **by** *auto*

**then have**  $*$ :  $?s \cdot \text{const-subst } c = s \cdot \text{const-subst } c$

**by** (*induct s arbitrary: p*) (*auto simp: nth-list-update map-update intro!: nth-equalityI*)

**have** *varposs*  $?s = ps$  **using** *varposs varposs-ground-replace-at[of p s constT c]*

**by** *auto*

**from** *Suc(1)[of ?s] Suc(2-)* *varposs* **obtain** *D*  $\tau$  **where** *split*:  $s[p \leftarrow \text{constT } c]$   
 $= D \langle l \cdot \tau \rangle$

**by** (*metis \* <varposs s[p < constT c] = ps> card-insert-if diff-Suc-1 finite-varposs*)

**have** *wit*:  $s = D \langle l \cdot \tau \rangle [p \leftarrow s \mid p]$  **unfolding** *arg-cong[OF split, of  $\lambda t. t[p \leftarrow$*

$s \mid - p]$ , *symmetric*]  
**using**  $vp$  **by** *simp*  
**from**  $vp$  *split* **have**  $cases: p \perp \text{hole-pos } D \vee \text{hole-pos } D \leq_p p$   
**by** *auto* (*metis* *poss-of-term-const-ctxt-apply* *poss-of-term-replace-term-at* *var-poss-imp-poss*)  
**show** *?case*  
**proof** ( $cases\ p \perp \text{hole-pos } D$ )  
**case** *True* **then show** *?thesis* **using** *wit*  
**by** (*auto* *simp: par-hole-pos-replace-term-context-at*)  
**next**  
**case** *False*  
**then have**  $\text{hole: hole-pos } D \leq_p p$  **using**  $cases$  **by** *auto*  
**from**  $vp$  *split* **have**  $p \in \text{poss-of-term } (\text{constT } c) \ s[p \leftarrow \text{constT } c]$   
**using** *poss-of-term-replace-term-at* *varposs-imp-poss* **by** *blast*  
**from** *poss-of-term-hole-poss*[*OF* *this*[*unfolded split*] *hole*]  
**have**  $p \ -_p \text{hole-pos } D \in \text{poss-of-term } (\text{constT } c) \ (l \cdot \tau)$   
**by** *simp*  
**from** *poss-of-term-subst-apply-varposs*[*OF* *this* *Suc(4)*] **obtain**  $q$  **where**  
 $q: q \in \text{varposs } l \ q \leq_p (p \ -_p (\text{hole-pos } D))$  **by** *blast*  
**show** *?thesis* **using** *wit* *Suc(5)* *hole*  
**using** *linear-term-varposs-subst-replace-term*[*OF* *Suc(5)*  $q$ , *of*  $\tau \ s \ \mid - p]$   
**by** *auto*  
**qed**  
**qed** (*auto* *simp: card-varposs-ground* *ground-subst-apply*)

**definition**  $llrg \ \mathcal{R} \longleftrightarrow (\forall (l, r) \in \mathcal{R}. \text{linear-term } l \wedge \text{ground } r)$

**definition**  $lv \ \mathcal{R} \longleftrightarrow (\forall (l, r) \in \mathcal{R}. \text{linear-term } l \wedge \text{linear-term } r \wedge \text{vars-term } l \cap \text{vars-term } r = \{\})$

**definition**  $monadic \ \mathcal{F} \longleftrightarrow (\forall (f, n) \in \mathcal{F}. n \leq \text{Suc } 0)$

— NF of ground terms

**lemma** *ground-NF-srstep-gsrstep*:

$\text{ground } s \implies s \in \text{NF } (\text{srstep } \mathcal{F} \ \mathcal{R}) \implies s \in \text{NF } (\text{gsrstep } \mathcal{F} \ \mathcal{R})$   
**by** *blast*

**lemma** *NF-to-fresh-const-subst-NF*:

**assumes** *lin: linear-sys*  $\mathcal{R}$  **and** *fresh-const: (c, 0) \notin \text{funas-rel } \mathcal{R}  $\text{funas-rel } \mathcal{R} \subseteq \mathcal{F}$*

**and** *nf-f: funas-term*  $s \subseteq \mathcal{F}$   $s \in \text{NF } (\text{srstep } \mathcal{F} \ \mathcal{R})$

**shows**  $s \cdot \text{const-subst } c \in \text{NF } (\text{gsrstep } \mathcal{H} \ \mathcal{R})$

**proof** (*rule ccontr*)

**assume**  $s \cdot \text{const-subst } c \notin \text{NF}$  (*Restr (srstep  $\mathcal{H} \mathcal{R}$ ) (Collect ground)*)  
**then obtain**  $C \ l \ r \ \sigma$  **where**  $\text{step}: (l, r) \in \mathcal{R} \ s \cdot \text{const-subst } c = C\langle l \cdot \sigma \rangle$  **by**  
*fastforce*  
**from**  $\text{step}(1)$  **have**  $l: (c, 0) \notin \text{funas-term } l \ \text{linear-term } l$  **using** *lin fresh-const*  
**by** (*auto simp: funas-rel-def*)  
**obtain**  $D \ \tau$  **where**  $s = D\langle l \cdot \tau \rangle$  **using** *remove-const-subst-from-match[OF step(2)*  
 $l]$  **by** *blast*  
**then show** *False* **using**  $\text{step}(1)$  *nf-f*  
**by** (*meson NF-no-trancl-step fresh-const(2) r-into-trancl' rstepI rstep-trancl-sig-step-r*)  
**qed**

**lemma** *fresh-const-subst-NF-pres:*

**assumes** *fresh-const:*  $(c, 0) \notin \text{funas-rel } \mathcal{R} \ \text{funas-rel } \mathcal{R} \subseteq \mathcal{F}$   
**and** *nf-f:*  $\text{funas-term } s \subseteq \mathcal{F} \ \mathcal{F} \subseteq \mathcal{H} \ (c, 0) \in \mathcal{H} \ s \cdot \text{const-subst } c \in \text{NF}$  (*gsrstep*  
 $\mathcal{H} \ \mathcal{R}$ )  
**shows**  $s \in \text{NF}$  (*srstep  $\mathcal{F} \ \mathcal{R}$* )  
**proof** (*rule ccontr*)  
**assume**  $s \notin \text{NF}$  (*srstep  $\mathcal{F} \ \mathcal{R}$* )  
**then obtain**  $C \ l \ r \ \sigma$  **where**  $\text{step}: (l, r) \in \mathcal{R} \ s = C\langle l \cdot \sigma \rangle$  **by** *fastforce*  
**let**  $?\tau = \lambda x. \text{if } x \in \text{vars-term } l \text{ then } (\sigma \ x) \cdot \text{const-subst } c \text{ else } \text{Fun } c \ \square$   
**define**  $D$  **where**  $D = (C \cdot_c \text{const-subst } c)$   
**have**  $s: s \cdot \text{const-subst } c = D\langle l \cdot ?\tau \rangle$  **unfolding** *D-def step(2)*  
**by** (*auto simp: subst-compose simp flip: subst-subst-compose intro!: term-subst-eq*)  
**have** *funas:*  $\text{funas-ctxt } D \subseteq \mathcal{H} \ \text{funas-term } (l \cdot ?\tau) \subseteq \mathcal{H} \ \text{funas-term } (r \cdot ?\tau) \subseteq \mathcal{H}$   
**using**  $\text{step nf-f}(1 - 3)$  *fresh-const(2) unfolding D-def*  
**by** (*auto simp: funas-ctxt-subst-apply-ctxt funas-term-subst funas-rel-def split:*  
*if-splits*)  
**moreover have** *ground-ctxt*  $D \ \text{ground } (l \cdot ?\tau) \ \text{ground } (r \cdot ?\tau)$  **using** *arg-cong[OF*  
 $s, \text{ of ground}] \ \text{unfolding } D\text{-def}$   
**by** (*auto intro!: ground-substI*)  
**ultimately have**  $(D\langle l \cdot ?\tau \rangle, D\langle r \cdot ?\tau \rangle) \in \text{gsrstep } \mathcal{H} \ \mathcal{R}$  **using**  $\text{step}(1)$   
**by** (*simp add: rstepI sig-stepI*)  
**then show** *False* **using**  $\text{nf-f}(4)$  **unfolding** *s[symmetric]*  
**by** *blast*  
**qed**

**lemma** *linear-sys-gNF-eq-NF-eq:*

**assumes** *lin:* *linear-sys*  $\mathcal{R} \ \text{linear-sys } \mathcal{S}$   
**and** *well:*  $\text{funas-rel } \mathcal{R} \subseteq \mathcal{F} \ \text{funas-rel } \mathcal{S} \subseteq \mathcal{F}$   
**and** *fresh:*  $(c, 0) \notin \text{funas-rel } \mathcal{R} \ (c, 0) \notin \text{funas-rel } \mathcal{S}$   
**and** *lift:*  $\mathcal{F} \subseteq \mathcal{H} \ (c, 0) \in \mathcal{H}$   
**and** *nf:*  $\text{NF}(\text{gsrstep } \mathcal{H} \ \mathcal{R}) = \text{NF}(\text{gsrstep } \mathcal{H} \ \mathcal{S})$   
**shows**  $\text{NF}(\text{srstep } \mathcal{F} \ \mathcal{R}) = \text{NF}(\text{srstep } \mathcal{F} \ \mathcal{S})$   
**proof** –  
**have** [*simp*]:  $\neg \text{funas-term } s \subseteq \mathcal{F} \implies s \in \text{NF}(\text{srstep } \mathcal{F} \ \mathcal{U})$  **for**  $s \ \mathcal{U}$  **by** (*meson*  
 $\text{NF-I sig-stepE}(1)$ )  
**have**  $d1: s \in \text{NF}(\text{gsrstep } \mathcal{H} \ \mathcal{R}) \implies s \in \text{NF}(\text{gsrstep } \mathcal{H} \ \mathcal{S})$  **for**  $s$  **using** *nf* **by**  
*auto*

**have**  $d2: s \in NF (gsrstep \mathcal{H} \mathcal{S}) \implies s \in NF (gsrstep \mathcal{H} \mathcal{R})$  **for**  $s$  **using**  $nf$  **by**  
*auto*  
**{fix**  $s$  **assume**  $n: s \in NF (srstep \mathcal{F} \mathcal{R})$  **then have**  $s \in NF (srstep \mathcal{F} \mathcal{S})$   
**using**  $NF\text{-to-fresh-const-subst-NF}[OF \text{ lin}(1) \text{ fresh}(1) \text{ well}(1) - n, THEN d1]$   
**using**  $\text{fresh-const-subst-NF-pres}[OF \text{ fresh}(2) \text{ well}(2) - \text{lift}, \text{ of } s]$   
**by**  $(\text{cases funas-term } s \subseteq \mathcal{F}) \text{ simp-all}$   
**moreover**  
**{fix**  $s$  **assume**  $n: s \in NF (srstep \mathcal{F} \mathcal{S})$  **then have**  $s \in NF (srstep \mathcal{F} \mathcal{R})$   
**using**  $NF\text{-to-fresh-const-subst-NF}[OF \text{ lin}(2) \text{ fresh}(2) \text{ well}(2) - n, THEN d2]$   
**using**  $\text{fresh-const-subst-NF-pres}[OF \text{ fresh}(1) \text{ well}(1) - \text{lift}, \text{ of } s]$   
**by**  $(\text{cases funas-term } s \subseteq \mathcal{F}) \text{ simp-all}$   
**ultimately show**  $?thesis$  **by**  $\text{blast}$   
**qed**

— Steps of ground

**lemma**  $gsrsteps\text{-to-srsteps}$ :  
 $(s, t) \in (gsrstep \mathcal{F} \mathcal{R})^+ \implies (s, t) \in (srstep \mathcal{F} \mathcal{R})^+$   
**by**  $(\text{meson inf-le1 trancl-mono})$

**lemma**  $gsrsteps\text{-eq-to-srsteps-eq}$ :  
 $(s, t) \in (gsrstep \mathcal{F} \mathcal{R})^* \implies (s, t) \in (srstep \mathcal{F} \mathcal{R})^*$   
**by**  $(\text{metis gsrsteps-to-srsteps rtrancl-eq-or-trancl})$

**lemma**  $gsrsteps\text{-to-rsteps}$ :  
 $(s, t) \in (gsrstep \mathcal{F} \mathcal{R})^+ \implies (s, t) \in (rstep \mathcal{R})^+$   
**using**  $gsrsteps\text{-to-srsteps srstepsD}$  **by**  $\text{blast}$

**lemma**  $gsrsteps\text{-eq-to-rsteps-eq}$ :  
 $(s, t) \in (gsrstep \mathcal{F} \mathcal{R})^* \implies (s, t) \in (rstep \mathcal{R})^*$   
**by**  $(\text{metis gsrsteps-eq-to-srsteps-eq rtrancl-eq-or-trancl srstepsD})$

**lemma**  $gsrsteps\text{-eq-relcomp-srsteps-relcompD}$ :  
 $(s, t) \in (gsrstep \mathcal{F} \mathcal{R})^* O (gsrstep \mathcal{F} \mathcal{S})^* \implies (s, t) \in (srstep \mathcal{F} \mathcal{R})^* O (srstep \mathcal{F} \mathcal{S})^*$   
**using**  $gsrsteps\text{-eq-to-srsteps-eq}$  **by**  $\text{blast}$

**lemma**  $gsrsteps\text{-eq-relcomp-to-rsteps-relcomp}$ :  
 $(s, t) \in (gsrstep \mathcal{F} \mathcal{R})^* O (gsrstep \mathcal{F} \mathcal{S})^* \implies (s, t) \in (rstep \mathcal{R})^* O (rstep \mathcal{S})^*$   
**using**  $gsrsteps\text{-eq-relcomp-srsteps-relcompD}$   
**using**  $gsrsteps\text{-eq-to-rsteps-eq}$  **by**  $\text{blast}$

**lemma**  $\text{ground-srsteps-gsrsteps}$ :  
**assumes**  $\text{ground } s \text{ ground } t$   
**and**  $(s, t) \in (srstep \mathcal{F} \mathcal{R})^+$   
**shows**  $(s, t) \in (gsrstep \mathcal{F} \mathcal{R})^+$

**proof** –  
**let**  $?\sigma = \lambda -. s$   
**from**  $assms(\beta)$  **have**  $f: funas-term\ s \subseteq \mathcal{F}$  **using**  $srstepsD$  **by**  $blast$   
**have**  $(s \cdot ?\sigma, t \cdot ?\sigma) \in (gsrstep\ \mathcal{F}\ \mathcal{R})^+$  **using**  $assms(\beta, 1)\ f$   
**proof** (*induct*)  
  **case** (*base*  $t$ )  
  **then have**  $(s \cdot ?\sigma, t \cdot ?\sigma) \in gsrstep\ \mathcal{F}\ \mathcal{R}$   
  **by** (*auto intro: srstep-subst-closed*)  
  **then show**  $?case$  **by** *auto*  
**next**  
  **case** (*step*  $t\ u$ )  
  **from**  $step(2, 4, 5)$  **have**  $(t \cdot ?\sigma, u \cdot ?\sigma) \in gsrstep\ \mathcal{F}\ \mathcal{R}$   
  **by** (*auto intro: srstep-subst-closed*)  
  **then show**  $?case$  **using**  $step(3 - 5)$   
  **by** (*meson Transitive-Closure.trancl-into-trancl*)  
**qed**  
**then show**  $?thesis$  **using**  $assms(1, 2)$   
**by** (*simp add: ground-subst-apply*)  
**qed**

**lemma** *ground-srsteps-eq-gsrsteps-eq*:  
**assumes**  $ground\ s\ ground\ t$   
  **and**  $(s, t) \in (srstep\ \mathcal{F}\ \mathcal{R})^*$   
**shows**  $(s, t) \in (gsrstep\ \mathcal{F}\ \mathcal{R})^*$   
**using** *ground-srsteps-gsrsteps*  
**by** (*metis assms rtrancl-eq-or-trancl*)

**lemma** *srsteps-eq-relcomp-gsrsteps-relcomp*:  
**assumes**  $(s, t) \in (srstep\ \mathcal{F}\ \mathcal{R})^* \ O\ (srstep\ \mathcal{F}\ \mathcal{S})^*$   
  **and**  $ground\ s\ ground\ t$   
**shows**  $(s, t) \in (gsrstep\ \mathcal{F}\ \mathcal{R})^* \ O\ (gsrstep\ \mathcal{F}\ \mathcal{S})^*$   
**proof** –  
**from**  $assms(1)$  **obtain**  $u$  **where**  $steps: (s, u) \in (srstep\ \mathcal{F}\ \mathcal{R})^* \ (u, t) \in (srstep\ \mathcal{F}\ \mathcal{S})^*$   
**by** *blast*  
**let**  $?\sigma = \lambda x. s$   
**have**  $(s \cdot ?\sigma, u \cdot ?\sigma) \in (srstep\ \mathcal{F}\ \mathcal{R})^* \ (u \cdot ?\sigma, t \cdot ?\sigma) \in (srstep\ \mathcal{F}\ \mathcal{S})^*$  **using**  
 $steps$   
  **using** *srsteps-eq-subst-closed[OF steps(1), of ?σ]*  
  **using** *srsteps-eq-subst-closed[OF steps(2), of ?σ]*  
  **by** (*metis rtrancl-eq-or-trancl srstepsD*)  
**then have**  $(s \cdot ?\sigma, u \cdot ?\sigma) \in (gsrstep\ \mathcal{F}\ \mathcal{R})^* \ (u \cdot ?\sigma, t \cdot ?\sigma) \in (gsrstep\ \mathcal{F}\ \mathcal{S})^*$   
  **using**  $assms(2)$   
  **by** (*auto intro: ground-srsteps-eq-gsrsteps-eq*)  
**then show**  $?thesis$  **using**  $assms(2, 3)$   
  **by** (*auto simp: ground-subst-apply*)  
**qed**

— Steps of llrg systems

**lemma** *llrg-ground-rhs*:

*llrg*  $\mathcal{R} \implies (l, r) \in \mathcal{R} \implies \text{ground } r$   
**unfolding** *llrg-def* **by** *auto*

**lemma** *llrg-rrsteps-groundness*:

**assumes** *llrg*  $\mathcal{R}$  **and**  $(s, t) \in (\text{srrstep } \mathcal{F} \mathcal{R})$   
**shows** *ground t* **using** *assms(2)* *ground-vars-term-empty*  
**by** (*fastforce simp: llrg-def sig-step-def dest!: llrg-ground-rhs[OF assms(1)] split: prod.splits*)

**lemma** *llrg-rsteps-pres-groundness*:

**assumes** *llrg*  $\mathcal{R}$  *ground s*  
**and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   
**shows** *ground t* **using** *assms(3, 2)*  
**proof** (*induct rule: rtrancl.induct*)  
**case** (*rtrancl-into-rtrancl s t u*)  
**then have** *ground t* **by** *auto*  
**then show** *?case* **using** *rtrancl-into-rtrancl(3)*  
**by** (*auto simp: sig-step-def vars-term-ctxt-apply ground-vars-term-empty ground-subst-apply dest!: llrg-ground-rhs[OF assms(1)] rstep-imp-C-s-r split: prod.splits*)  
**qed** *simp*

**lemma** *llrg-srsteps-with-root-step-ground*:

**assumes** *llrg*  $\mathcal{R}$  **and**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R}$   
**shows** *ground t* **using** *assms llrg-rrsteps-groundness llrg-rsteps-pres-groundness*  
**unfolding** *srsteps-with-root-step-def*  
**by** *blast*

**lemma** *llrg-srsteps-with-root-step-inv-ground*:

**assumes** *llrg*  $\mathcal{R}$  **and**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} (\mathcal{R}^{-1})$   
**shows** *ground s* **using** *assms llrg-rrsteps-groundness llrg-rsteps-pres-groundness*  
**unfolding** *srsteps-with-root-step-def*  
**by** (*metis (no-types, lifting) converseD relcomp.cases rtrancl-converseD srrstep-converse-dist srstep-converse-dist*)

**lemma** *llrg-funas-term-step-pres*:

**assumes** *llrg*  $\mathcal{R}$  **and**  $(s, t) \in (\text{rstep } \mathcal{R})$   
**shows** *funas-term t*  $\subseteq$  *funas-rel*  $\mathcal{R} \cup$  *funas-term s*  
**proof** –  
**have** [*simp*]:  $(l, r) \in \mathcal{R} \implies r \cdot \sigma = r$  **for**  $l \ r \ \sigma$  **using** *assms(1)* **unfolding** *llrg-def*  
**by** (*auto split: prod.splits intro: ground-subst-apply*)  
**show** *?thesis* **using** *assms*  
**by** (*auto simp: llrg-def funas-rel-def dest!: rstep-imp-C-s-r*)  
**qed**

**lemma** *llrg-funas-term-steps-pres*:

**assumes** *llrg*  $\mathcal{R}$  **and**  $(s, t) \in (\text{rstep } \mathcal{R})^*$

**shows**  $\text{funas-term } t \subseteq \text{funas-rel } \mathcal{R} \cup \text{funas-term } s$   
**using**  $\text{assms}(2)$   $\text{llrg-funas-term-step-pres}[OF \text{ assms}(1)]$   
**by**  $(\text{induct}) \text{ auto}$

— Steps of monadic llrg systems

**lemma**  $\text{monadic-ground-ctxt-apply}$ :  
 $\text{monadic } \mathcal{F} \implies \text{funas-ctxt } C \subseteq \mathcal{F} \implies \text{ground } r \implies \text{ground } C\langle r \rangle$   
**by**  $(\text{induct } C)$   $(\text{auto simp: monadic-def})$

**lemma**  $\text{llrg-monadic-rstep-pres-groundness}$ :

**assumes**  $\text{llrg } \mathcal{R} \text{ monadic } \mathcal{F}$   
**and**  $(s, t) \in \text{srstep } \mathcal{F} \mathcal{R}$   
**shows**  $\text{ground } t$  **using**  $\text{assms}(3)$

**proof** —

**from**  $\text{assms}(3)$  **obtain**  $C \ l \ r \ \sigma$  **where**  $r: (l, r) \in \mathcal{R}$  **and**  $t:t = C\langle r \cdot \sigma \rangle$   
**using**  $\text{rstep-imp-C-s-r unfolding sig-step-def}$  **by**  $\text{blast}$   
**from**  $\text{assms}(1, 3)$  **have**  $\text{funas: funas-term } t \subseteq \mathcal{F}$   $\text{ground } r$   
**by**  $(\text{auto simp: llrg-ground-rhs}[OF \text{ assms}(1) \ r(1)] \text{ dest: srstepD})$   
**then have**  $*$ :  $r \cdot \sigma = r$  **by**  $(\text{simp add: ground-subst-apply})$   
**show**  $?thesis$  **using**  $\text{funas assms}(2)$  **unfolding**  $t *$   
**by**  $(\text{intro monadic-ground-ctxt-apply}) \text{ auto}$

**qed**

**lemma**  $\text{llrg-monadic-rsteps-groundness}$ :

**assumes**  $\text{llrg } \mathcal{R} \text{ monadic } \mathcal{F}$   
**and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$   
**shows**  $\text{ground } t$  **using**  $\text{assms}(3)$   
**using**  $\text{llrg-monadic-rstep-pres-groundness}[OF \text{ assms}(1, 2)]$   
**by**  $(\text{induct rule: trancl.induct}) \text{ auto}$

— Steps in monadic lv system

**fun**  $\text{monadic-term}$  **where**

$\text{monadic-term } (\text{Var } x) = \text{True}$   
 $|\ \text{monadic-term } (\text{Fun } f \ []) = \text{True}$   
 $|\ \text{monadic-term } (\text{Fun } f \ ts) = (\text{length } ts = \text{Suc } 0 \wedge \text{monadic-term } (\text{hd } ts))$

**fun**  $\text{monadic-get-leave}$  **where**

$\text{monadic-get-leave } (\text{Var } x) = (\text{Var } x)$   
 $|\ \text{monadic-get-leave } (\text{Fun } f \ []) = \text{Fun } f \ []$   
 $|\ \text{monadic-get-leave } (\text{Fun } f \ ts) = \text{monadic-get-leave } (\text{hd } ts)$

**fun**  $\text{monadic-replace-leave}$  **where**

$\text{monadic-replace-leave } t \ (\text{Var } x) = t$   
 $|\ \text{monadic-replace-leave } t \ (\text{Fun } f \ []) = t$   
 $|\ \text{monadic-replace-leave } t \ (\text{Fun } f \ ts) = \text{Fun } f \ [\text{monadic-replace-leave } t \ (\text{hd } ts)]$

**lemma** *monadic-replace-leave-undo-const-subst*:  
**assumes** *monadic-term s*  
**shows** *monadic-replace-leave (monadic-get-leave s) (s · const-subst c) = s* **using** *assms*  
**proof** (*induct s*)  
**case** (*Fun f ts*) **then show** *?case*  
**by** (*cases ts*) *auto*  
**qed** *auto*

**lemma** *monadic-replace-leave-context*:  
**assumes** *monadic-term C⟨s⟩*  
**shows** *monadic-replace-leave t C⟨s⟩ = C⟨monadic-replace-leave t s⟩* **using** *assms*  
**proof** (*induct C*)  
**case** (*More f ss C ts*) **then show** *?case*  
**by** (*cases ss; cases ts*) *auto*  
**qed** *simp*

**lemma** *monadic-replace-leave-subst*:  
**assumes** *monadic-term (s · σ) ⊢ ground s*  
**shows** *monadic-replace-leave t (s · σ) = s · (λ x. monadic-replace-leave t (σ x))*  
**using** *assms*  
**proof** (*induct s*)  
**case** (*Fun f ts*) **then show** *?case*  
**by** (*cases ts*) *auto*  
**qed** *auto*

**lemma** *monadic-sig*:  
*monadic F ⟹ (f, length ts) ∈ F ⟹ length ts ≤ Suc 0*  
**by** (*auto simp: monadic-def*)

**lemma** *monadic-sig-funas-term-mt*:  
*monadic F ⟹ funas-term s ⊆ F ⟹ monadic-term s*  
**proof** (*induct s*)  
**case** (*Fun f ts*) **then show** *?case* **unfolding** *monadic-def*  
**by** (*cases ts*) *auto*  
**qed** *simp*

**lemma** *monadic-term-const-pres [intro]*:  
*monadic-term s ⟹ monadic-term (s · const-subst c)*  
**proof** (*induct s*)  
**case** (*Fun f ts*) **then show** *?case*  
**by** (*cases ts*) *auto*  
**qed** *simp*

**lemma** *remove-const-lv-mondaic-step-lhs*:  
**assumes** *lv: lv R* **and** *fresh: (c, 0) ∉ funas-rel R*  
**and** *mon: monadic F*  
**and** *step: (s · const-subst c, t) ∈ (srstep F R)*



```

shows  $(s, t) \in (srstep \mathcal{F} \mathcal{R})$ 
proof –
  from step obtain  $C \ l \ r \ \sigma$  where  $s: (l, r) \in \mathcal{R} \ s \cdot const\text{-subst} \ c = C \langle l \cdot \sigma \rangle \ t = C \langle r \cdot \sigma \rangle$ 
    by fastforce
    have  $lv: x \in vars\text{-term} \ l \implies x \notin vars\text{-term} \ r$  for  $x$  using  $s(1) \ lv$ 
    by (auto simp: lv-def)
    from  $s(1)$  fresh have  $cl: (c, 0) \notin funas\text{-term} \ l$  by (auto simp: funas-rel-def)
    have  $funas: funas\text{-term} \ s \subseteq \mathcal{F} \ (c, 0) \notin funas\text{-term} \ l \ funas\text{-term} \ t \subseteq \mathcal{F}$  using
 $s(1)$  fresh
    using step mon funas-term-subst unfolding funas-rel-def
    by (auto dest!: srstepD) blast
    then have  $mt: monadic\text{-term} \ s \ monadic\text{-term} \ (s \cdot const\text{-subst} \ c)$ 
    using monadic-sig-funas-term-mt[OF mon] by auto
    then have  $ml: monadic\text{-term} \ (l \cdot \sigma)$  unfolding  $s(2)$ 
    by (metis funas-ctxt-apply le-sup-iff step mon monadic-sig-funas-term-mt s(2))
sig-stepE(1)
    show ?thesis
    proof (cases ground s)
      case True then show ?thesis using step
        by (auto simp: ground-subst-apply)
    next
      case False note  $ng = this$ 
      then have  $cs: (c, 0) \in funas\text{-term} \ (s \cdot const\text{-subst} \ c)$ 
        by (auto simp: funas-term-subst vars-term-empty-ground)
      have  $ngrl: \neg ground \ l$  using  $s(2) \ cs \ mt \ ng$ 
      proof (induct s arbitrary: C)
        case (Var x) then show ?case using  $cl \ cs$ 
        by (cases C) (auto simp: funas-rel-def ground-subst-apply)
      next
        case (Fun f ts)
        from Fun(5-) obtain  $t$  where  $[simp]: ts = [t]$  by (cases ts) auto
        show ?case
        proof (cases C = Hole)
          case True then show ?thesis using Fun(2, 3) cl
          by (auto simp: ground-subst-apply)
        next
          case False
          from this Fun(2, 3) obtain  $D$  where  $[simp]: C = More \ f \ [] \ D \ []$ 
          by (cases C) (auto simp: Cons-eq-append-conv)
          show ?thesis using Fun(1)[of t D] Fun(2-)
          by simp
        qed
      qed
    let  $?\tau = \lambda x. \text{if } x \in vars\text{-term} \ l \text{ then } monadic\text{-replace-leave} \ (monadic\text{-get-leave} \ s) \ (\sigma \ x) \ \text{else } (\sigma \ x)$ 
    have  $C \langle l \cdot (\lambda x. monadic\text{-replace-leave} \ (monadic\text{-get-leave} \ s) \ (\sigma \ x)) \rangle = C \langle l \cdot$ 
 $\?\tau \rangle$ 
    by (auto intro: term-subst-eq)

```

**then have**  $s = C\langle l \cdot ?\tau \rangle$  **using** *arg-cong*[*OF*  $s(2)$ ], *of monadic-replace-leave*  
*(monadic-get-leave s)*,  
*unfolded monadic-replace-leave-undo-const-subst*[*OF*  $mt(1)$ ], *of c*],  
*unfolded monadic-replace-leave-context*[*OF*  $mt(2)$ ][*unfolded s(2)*]],  
*unfolded monadic-replace-leave-subst*[*OF*  $ml\ ngrl$ ]]  
**by** *presburger*  
**moreover have**  $t = C\langle r \cdot ?\tau \rangle$  **using** *lv unfolding*  $s(3)$   
**by** (*auto intro!*: *term-subst-eq*)  
**ultimately show** *?thesis* **using**  $s(1)$  *funas(1, 3)*  
**by** *blast*  
**qed**  
**qed**

**lemma** *remove-const-lv-mondaic-step-rhs*:  
**assumes** *lv*:  $lv\ \mathcal{R}$  **and** *fresh*:  $(c, 0) \notin \text{funas-rel}\ \mathcal{R}$   
**and** *mon*: *monadic*  $\mathcal{F}$   
**and** *step*:  $(s, t \cdot \text{const-subst}\ c) \in (\text{srstep}\ \mathcal{F}\ \mathcal{R})$   
**shows**  $(s, t) \in (\text{srstep}\ \mathcal{F}\ \mathcal{R})$   
**proof** –  
**have** *inv-v*:  $lv\ (\mathcal{R}^{-1})(c, 0) \notin \text{funas-rel}\ (\mathcal{R}^{-1})$  **using** *fresh lv*  
**by** (*auto simp*: *funas-rel-def lv-def*)  
**have**  $(t \cdot \text{const-subst}\ c, s) \in (\text{srstep}\ \mathcal{F}\ (\mathcal{R}^{-1}))$  **using** *step*  
**by** (*auto simp*: *rew-converse-outwards*)  
**from** *remove-const-lv-mondaic-step-lhs*[*OF inv-v mon this*]  
**have**  $(t, s) \in (\text{srstep}\ \mathcal{F}\ (\mathcal{R}^{-1}))$  **by** *simp*  
**then show** *?thesis* **by** (*auto simp*: *rew-converse-outwards*)  
**qed**

**lemma** *remove-const-lv-mondaic-steps-lhs*:  
**assumes** *lv*:  $lv\ \mathcal{R}$  **and** *fresh*:  $(c, 0) \notin \text{funas-rel}\ \mathcal{R}$   
**and** *mon*: *monadic*  $\mathcal{F}$   
**and** *steps*:  $(s \cdot \text{const-subst}\ c, t) \in (\text{srstep}\ \mathcal{F}\ \mathcal{R})^+$   
**shows**  $(s, t) \in (\text{srstep}\ \mathcal{F}\ \mathcal{R})^+$   
**using** *remove-const-lv-mondaic-step-lhs*[*OF lv fresh mon*] *steps*  
**by** (*meson converse-tranclE r-into-trancl trancl-into-trancl2*)

**lemma** *remove-const-lv-mondaic-steps-rhs*:  
**assumes** *lv*:  $lv\ \mathcal{R}$  **and** *fresh*:  $(c, 0) \notin \text{funas-rel}\ \mathcal{R}$   
**and** *mon*: *monadic*  $\mathcal{F}$   
**and** *steps*:  $(s, t \cdot \text{const-subst}\ c) \in (\text{srstep}\ \mathcal{F}\ \mathcal{R})^+$   
**shows**  $(s, t) \in (\text{srstep}\ \mathcal{F}\ \mathcal{R})^+$   
**using** *remove-const-lv-mondaic-step-rhs*[*OF lv fresh mon*] *steps*  
**by** (*meson trancl.simps*)

**lemma** *remove-const-lv-mondaic-steps*:  
**assumes** *lv*:  $lv\ \mathcal{R}$  **and** *fresh*:  $(c, 0) \notin \text{funas-rel}\ \mathcal{R}$   
**and** *mon*: *monadic*  $\mathcal{F}$   
**and** *steps*:  $(s \cdot \text{const-subst}\ c, t \cdot \text{const-subst}\ c) \in (\text{srstep}\ \mathcal{F}\ \mathcal{R})^+$

**shows**  $(s, t) \in (\text{srrstep } \mathcal{F} \ \mathcal{R})^+$   
**using** *remove-const-lv-mondaic-steps-rhs*[*OF lv fresh mon remove-const-lv-mondaic-steps-lhs*[*OF*  
*assms*]]  
**by** *simp*

— Steps on lv trs

**lemma** *lv-root-step-idep-subst*:

**assumes** *lv*  $\mathcal{R}$   
**and**  $(s, t) \in \text{srrstep } \mathcal{F} \ \mathcal{R}$   
**and** *well*:  $\bigwedge x. \text{funas-term } (\sigma \ x) \subseteq \mathcal{F} \ \bigwedge x. \text{funas-term } (\tau \ x) \subseteq \mathcal{F}$   
**shows**  $(s \cdot \sigma, t \cdot \tau) \in \text{srrstep } \mathcal{F} \ \mathcal{R}$   
**proof** –  
**from** *assms*(2) **obtain**  $l \ r \ \gamma$  **where** *mid*:  $s = l \cdot \gamma \ t = r \cdot \gamma \ (l, r) \in \mathcal{R}$   
**by** (*auto simp: sig-step-def*)  
**from** *mid*(3) *assms*(1) **have** *vs*:  $x \in \text{vars-term } l \implies x \notin \text{vars-term } r$  **for**  $x$   
**by** (*auto simp: lv-def*)  
**let**  $?\sigma = \lambda x. \text{if } x \in \text{vars-term } l \text{ then } (\gamma \ x) \cdot \sigma \text{ else } (\gamma \ x) \cdot \tau$   
**have** *subst*:  $s \cdot \sigma = l \cdot ?\sigma \ t \cdot \tau = r \cdot ?\sigma$   
**unfolding** *mid subst-subst-compose*[*symmetric*]  
**unfolding** *term-subst-eq-conv*  
**by** (*auto simp: subst-compose-def vs*)  
**then show** *?thesis* **unfolding** *subst*  
**using** *assms*(2) *mid*(3) *well* **unfolding** *mid*(1, 2)  
**by** (*auto simp: sig-step-def funas-term-subst*)  
**qed**

**lemma** *lv-srsteps-with-root-step-idep-subst*:

**assumes** *lv*  $\mathcal{R}$   
**and**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{R}$   
**and** *well*:  $\bigwedge x. \text{funas-term } (\sigma \ x) \subseteq \mathcal{F} \ \bigwedge x. \text{funas-term } (\tau \ x) \subseteq \mathcal{F}$   
**shows**  $(s \cdot \sigma, t \cdot \tau) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{R}$  **using** *assms*(2)  
**using** *lv-root-step-idep-subst*[*OF assms*(1) - *well*, **where**  $?x1 = \text{id}$  **and**  $?x2 = \text{id}$ ]  
**using** *srsteps-eq-subst-closed*[*OF - well*(1), **where**  $?x1 = \text{id}$  **and**  $?R = \mathcal{R}$ ]  
**using** *srsteps-eq-subst-closed*[*OF - well*(2), **where**  $?x1 = \text{id}$  **and**  $?R = \mathcal{R}$ ]  
**by** (*auto simp: srsteps-with-root-step-def*) (*metis* (*full-types*) *relcomp3-I*)

**end**

**theory** *Rewriting-GTRS*

**imports** *Rewriting*  
*Replace-Constant*

**begin**

#### 4.4 Specific results about rewriting under a ground system

**abbreviation** *ground-sys*  $\mathcal{R} \equiv (\forall (s, t) \in \mathcal{R}. \text{ground } s \wedge \text{ground } t)$

**lemma** *srrstep-ground*:  
**assumes** *ground-sys*  $\mathcal{R}$   
**and**  $(s, t) \in \text{srrstep } \mathcal{F} \mathcal{R}$   
**shows** *ground s ground t using* *assms*  
**by** (*auto simp: sig-step-def ground-subst-apply vars-term-subst elim!: rstep-subst*)

**lemma** *srstep-pres-ground-l*:  
**assumes** *ground-sys*  $\mathcal{R}$  *ground s*  
**and**  $(s, t) \in \text{srstep } \mathcal{F} \mathcal{R}$   
**shows** *ground t using* *assms*  
**by** (*auto simp: sig-step-def ground-subst-apply dest!: rstep-imp-C-s-r*)

**lemma** *srstep-pres-ground-r*:  
**assumes** *ground-sys*  $\mathcal{R}$  *ground t*  
**and**  $(s, t) \in \text{srstep } \mathcal{F} \mathcal{R}$   
**shows** *ground s using* *assms*  
**by** (*auto simp: ground-vars-term-empty vars-term-subst sig-step-def vars-term-ctxt-apply ground-subst-apply dest!: rstep-imp-C-s-r*)

**lemma** *srsteps-pres-ground-l*:  
**assumes** *ground-sys*  $\mathcal{R}$  *ground s*  
**and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$   
**shows** *ground t using* *assms*(3, 2) *srstep-pres-ground-l*[*OF* *assms*(1)]  
**by** (*induct rule: converse-trancl-induct*) *auto*

**lemma** *srsteps-pres-ground-r*:  
**assumes** *ground-sys*  $\mathcal{R}$  *ground t*  
**and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$   
**shows** *ground s using* *assms*(3, 2) *srstep-pres-ground-r*[*OF* *assms*(1)]  
**by** (*induct rule: converse-trancl-induct*) *auto*

**lemma** *srsteps-eq-pres-ground-l*:  
**assumes** *ground-sys*  $\mathcal{R}$  *ground s*  
**and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   
**shows** *ground t using* *srsteps-pres-ground-l*[*OF* *assms*(1, 2)] *assms*(2, 3)  
**by** (*auto simp: rtrancl-eq-or-trancl*)

**lemma** *srsteps-eq-pres-ground-r*:  
**assumes** *ground-sys*  $\mathcal{R}$  *ground t*  
**and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   
**shows** *ground s using* *srsteps-pres-ground-r*[*OF* *assms*(1, 2)] *assms*(2, 3)  
**by** (*auto simp: rtrancl-eq-or-trancl*)

**lemma** *srsteps-with-root-step-ground*:  
**assumes** *ground-sys*  $\mathcal{R}$   
**and**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R}$   
**shows** *ground s ground t using* *srrstep-ground*[*OF* *assms*(1)]  
**using** *srsteps-eq-pres-ground-l*[*OF* *assms*(1)]

**using** *srsteps-eq-pres-ground-r*[*OF assms(1)*]  
**using** *assms(2)* **unfolding** *srsteps-with-root-step-def*  
**by** (*meson relcomp.cases*)<sup>+</sup>

## 4.5 funas

**lemma** *srrstep-funas*:

**assumes** *ground-sys*  $\mathcal{R}$   
**and**  $(s, t) \in \text{srrstep } \mathcal{F} \mathcal{R}$   
**shows** *funas-term*  $s \subseteq \text{funas-rel } \mathcal{R}$  *funas-term*  $t \subseteq \text{funas-rel } \mathcal{R}$  **using** *assms*  
**by** (*auto simp: sig-step-def funas-term-subst ground-vars-term-empty funas-rel-def*  
*split: prod.splits elim!: rstep-subst*)

**lemma** *srstep-funas-l*:

**assumes** *ground-sys*  $\mathcal{R}$   
**and**  $(s, t) \in \text{srstep } \mathcal{F} \mathcal{R}$   
**shows** *funas-term*  $t \subseteq \text{funas-term } s \cup \text{funas-rel } \mathcal{R}$  **using** *assms*  
**by** (*auto simp: ground-vars-term-empty vars-term-subst sig-step-def vars-term-ctxt-apply*  
*funas-term-subst funas-rel-def split: prod.splits dest!: rstep-imp-C-s-r*)

**lemma** *srstep-funas-r*:

**assumes** *ground-sys*  $\mathcal{R}$   
**and**  $(s, t) \in \text{srstep } \mathcal{F} \mathcal{R}$   
**shows** *funas-term*  $s \subseteq \text{funas-term } t \cup \text{funas-rel } \mathcal{R}$  **using** *assms*  
**by** (*auto simp: ground-vars-term-empty vars-term-subst sig-step-def vars-term-ctxt-apply*  
*funas-term-subst funas-rel-def split: prod.splits dest!: rstep-imp-C-s-r*)

**lemma** *srsteps-funas-l*:

**assumes** *ground-sys*  $\mathcal{R}$   
**and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$   
**shows** *funas-term*  $t \subseteq \text{funas-term } s \cup \text{funas-rel } \mathcal{R}$  **using** *assms(2)*  
**by** (*induct rule: converse-trancl-induct*) (*auto dest: srstep-funas-l[OF assms(1)]*)

**lemma** *srsteps-funas-r*:

**assumes** *ground-sys*  $\mathcal{R}$   
**and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$   
**shows** *funas-term*  $s \subseteq \text{funas-term } t \cup \text{funas-rel } \mathcal{R}$  **using** *assms(2)*  
**by** (*induct rule: converse-trancl-induct*) (*auto dest: srstep-funas-r[OF assms(1)]*)

**lemma** *srsteps-eq-funas-l*:

**assumes** *ground-sys*  $\mathcal{R}$   
**and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   
**shows** *funas-term*  $t \subseteq \text{funas-term } s \cup \text{funas-rel } \mathcal{R}$  **using** *srsteps-funas-l[OF*  
*assms(1)] assms(2)*  
**by** (*auto simp: rtrancl-eq-or-trancl*)

**lemma** *srsteps-eq-funas-r*:

**assumes** *ground-sys*  $\mathcal{R}$   
**and**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$

**shows**  $\text{funas-term } s \subseteq \text{funas-term } t \cup \text{funas-rel } \mathcal{R}$  **using**  $\text{srsteps-funas-r}[OF \text{ assms}(1)] \text{ assms}(2)$   
**by** (*auto simp: rtrancl-eq-or-trancl*)

**lemma** *srsteps-with-root-step-funas*:  
**assumes** *ground-sys*  $\mathcal{R}$   
**and**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R}$   
**shows**  $\text{funas-term } s \subseteq \text{funas-rel } \mathcal{R} \text{ funas-term } t \subseteq \text{funas-rel } \mathcal{R}$   
**using** *srrstep-funas*[ $OF \text{ assms}(1)$ ]  
**using** *srsteps-eq-funas-l*[ $OF \text{ assms}(1)$ ]  
**using** *srsteps-eq-funas-r*[ $OF \text{ assms}(1)$ ]  
**using**  $\text{assms}(2)$  **unfolding** *srsteps-with-root-step-def*  
**by** (*metis relcompEpair sup-absorb2*)**+**

**end**

## 5 Reducing Rewrite Properties to Properties on Ground Terms over Left-Linear Right-Ground Systems

**theory** *Ground-Reduction-on-LLRG*  
**imports**  
*Rewriting-Properties*  
*Rewriting-LLRG-LV-Mondaic*  
**begin**

**lemma** *llrg-linear-sys*:  
 $\text{llrg } \mathcal{R} \implies \text{linear-sys } \mathcal{R}$   
**by** (*auto simp: llrg-def*)

## 6 LLRG results

**lemma** *llrg-commute*:  
**assumes**  $\text{sig: funas-rel } \mathcal{R} \subseteq \mathcal{F} \text{ funas-rel } \mathcal{S} \subseteq \mathcal{F}$   
**and**  $\text{fresh: } (c, 0) \notin \mathcal{F}$   
**and**  $\text{llrg: llrg } \mathcal{R} \text{ llrg } \mathcal{S}$   
**and**  $\text{com: commute } (\text{gsrstep } (\text{insert } (c, 0) \mathcal{F}) \mathcal{R}) (\text{gsrstep } (\text{insert } (c, 0) \mathcal{F}) \mathcal{S})$   
**shows**  $\text{commute } (\text{srstep } \mathcal{F} \mathcal{R}) (\text{srstep } \mathcal{F} \mathcal{S})$   
**proof** –  
**let**  $? \sigma = \lambda -. \text{Fun } c \ []$  **let**  $? \mathcal{H} = \text{insert } (c, 0) \mathcal{F}$   
**from**  $\text{fresh}$  **have**  $\text{fresh-sys: } (c, 0) \notin \text{funas-rel } \mathcal{S} \text{ } (c, 0) \notin \text{funas-rel } (\mathcal{R}^{-1})$  **using**  
 $\text{sig}$   
**by** (*auto simp: funas-rel-def*)  
**have**  $\text{linear: linear-sys } \mathcal{S} \text{ linear-sys } (\mathcal{R}^{-1})$  **using** *llrg* **unfolding** *llrg-def* **by** *auto*  
**have**  $\text{sig': funas-rel } \mathcal{S} \subseteq \mathcal{F} \text{ funas-rel } (\mathcal{R}^{-1}) \subseteq \mathcal{F}$  **using**  $\text{sig}$  **by** (*auto simp: funas-rel-def*)  
**have**  $\text{mono: } \mathcal{F} \subseteq ? \mathcal{H}$  **by** *auto*

```

{fix s t assume ass: (s, t) ∈ comp-rrstep-rel'  $\mathcal{F}$  ( $\mathcal{R}^{-1}$ )  $\mathcal{S}$ 
  from ass have funas: funas-term s ⊆  $\mathcal{F}$  funas-term t ⊆  $\mathcal{F}$ 
  by (metis Un-iff relcomp.cases srstepsD srsteps-with-root-step-srstepsD)+
  from ass have m: (s, t) ∈ (srstep ? $\mathcal{H}$  ( $\mathcal{R}^{-1}$ ))* O (srstep ? $\mathcal{H}$   $\mathcal{S}$ )*
  using rtrancl-mono[OF sig-step-mono[OF mono]]
  by (auto dest!: srsteps-with-root-step-srsteps-eqD trancl-into-rtrancl)
  have gr: ground s ∨ ground t using ass llrg
  unfolding srstep-converse-dist trancl-converse
  by (auto simp: llrg-srsteps-with-root-step-inv-ground llrg-srsteps-with-root-step-ground)
  have gstep: (s · ?σ, t · ?σ) ∈ (gsrstep ? $\mathcal{H}$   $\mathcal{S}$ )* O (gsrstep ? $\mathcal{H}$  ( $\mathcal{R}^{-1}$ ))*
  using srsteps-eq-subst-relcomp-closed[OF m, THEN srsteps-eq-relcomp-gsrsteps-relcomp,
of ?σ,
  THEN subsetD[OF com[unfolding commute-def], unfolded rew-converse-inwards]]
  by (auto simp: ground-substI)
  have [simp]: ground s ⇒ (c, 0) ∉ funas-term s ground t ⇒ (c, 0) ∉ funas-term
t
  using funas fresh by auto
  have (s, t) ∈ (rstep  $\mathcal{S}$ )* O (rstep ( $\mathcal{R}^{-1}$ ))*
  using remove-const-subst-relcomp-lhs[OF linear fresh-sys, of t s]
  using gsrsteps-eq-relcomp-to-rsteps-relcomp[OF gstep] gr
  using remove-const-subst-relcomp-lhs[OF linear fresh-sys, of t s]
  using remove-const-subst-relcomp-rhs[OF linear fresh-sys, of s t]
  by (cases ground s) (simp-all add: ground-subst-apply)
  from rsteps-eq-relcomp-srsteps-eq-relcompI[OF sig' funas this]
  have commute-redp  $\mathcal{F}$   $\mathcal{R}$   $\mathcal{S}$  s t unfolding commute-redp-def
  by (simp add: rew-converse-inwards)}
then show ?thesis by (intro commute-rrstep-intro) simp
qed

```

lemma llrg-CR:

```

assumes sig: funas-rel  $\mathcal{R}$  ⊆  $\mathcal{F}$ 
  and fresh: (c, 0) ∉  $\mathcal{F}$ 
  and llrg: llrg  $\mathcal{R}$ 
  and com: CR (gsrstep (insert (c, 0)  $\mathcal{F}$ )  $\mathcal{R}$ )
shows CR (srstep  $\mathcal{F}$   $\mathcal{R}$ )
using assms llrg-commute unfolding CR-iff-self-commute
by metis

```

lemma llrg-SCR:

```

assumes sig: funas-rel  $\mathcal{R}$  ⊆  $\mathcal{F}$  and fresh: (c, 0) ∉  $\mathcal{F}$ 
  and llrg: llrg  $\mathcal{R}$ 
  and scr: SCR (gsrstep (insert (c, 0)  $\mathcal{F}$ )  $\mathcal{R}$ )
shows SCR (srstep  $\mathcal{F}$   $\mathcal{R}$ )

```

proof –

```

let ?σ = const-subst c let ? $\mathcal{H}$  = insert (c, 0)  $\mathcal{F}$ 
from fresh have fresh-sys: (c, 0) ∉ funas-rel  $\mathcal{R}$  using sig
  by (auto simp: funas-rel-def)
have mono:  $\mathcal{F}$  ⊆ ? $\mathcal{H}$  by auto note sig-trans = subsetD[OF srstep-monp[OF

```

mono]]

```

{fix s t u assume ass: (s, t) ∈ srstep  $\mathcal{F}$   $\mathcal{R}$  (s, u) ∈ srstep  $\mathcal{F}$   $\mathcal{R}$ 
and root: (s, t) ∈ sig-step  $\mathcal{F}$  (rrstep  $\mathcal{R}$ ) ∨ (s, u) ∈ sig-step  $\mathcal{F}$  (rrstep  $\mathcal{R}$ )
from ass have funas: funas-term s ⊆  $\mathcal{F}$  funas-term t ⊆  $\mathcal{F}$  funas-term u ⊆  $\mathcal{F}$ 
by (force dest: sig-stepE)+
from root have gr: ground t ∨ ground u using ass llrg llrg-rrsteps-groundness
unfolding srstep-converse-dist trancl-converse by blast
have *: ground (s · ?σ) ground (t · ?σ) ground (u · ?σ) by auto
from this scr obtain v where v: (t · ?σ, v) ∈ (gsrstep ? $\mathcal{H}$   $\mathcal{R}$ )= ∧ (u · ?σ, v)
∈ (gsrstep ? $\mathcal{H}$   $\mathcal{R}$ )*
using srstep-subst-closed[OF sig-trans[OF ass(1)], of ?σ]
using srstep-subst-closed[OF sig-trans[OF ass(2)], of ?σ]
using ass unfolding SCR-on-def
by auto (metis (no-types, lifting) *)
then have fv: funas-term v ⊆  $\mathcal{F}$  using gr llrg-funas-term-step-pres[OF llrg, of
- v]
using llrg-funas-term-steps-pres[OF llrg, of - v] funas sig
by (auto simp: ground-subst-apply elim!: sig-stepE dest!: gsrsteps-eq-to-rsteps-eq)
blast+
then have c-free: (c, 0) ∉ funas-term v using fresh by blast
then have v = t · const-subst c ⇒ ground t using arg-cong[of v t · ?σ
funas-term]
by (auto simp: funas-term-subst vars-term-empty-ground split: if-splits)
from this v have (t, v) ∈ (srstep  $\mathcal{F}$   $\mathcal{R}$ )= (u, v) ∈ (srstep  $\mathcal{F}$   $\mathcal{R}$ )*
using remove-const-subst-steps-eq-lhs[OF llrg-linear-sys[OF llrg] fresh-sys
c-free, of u,
THEN rsteps-eq-srsteps-eqI[OF sig funas(3) fv]]
using remove-const-subst-step-lhs[OF llrg-linear-sys[OF llrg] fresh-sys c-free,
of t,
THEN sig-stepI[OF funas(2) fv]]
by (auto simp: ground-subst-apply dest: srstepD gsrsteps-eq-to-rsteps-eq)
then have SCRp  $\mathcal{F}$   $\mathcal{R}$  t u by blast}
then show ?thesis by (intro SCR-rrstep-intro) (metis srrstep-to-srstep)+
qed

```

**lemma llrg-WCR:**

```

assumes sig: funas-rel  $\mathcal{R}$  ⊆  $\mathcal{F}$  and fresh: (c, 0) ∉  $\mathcal{F}$ 
and llrg: llrg  $\mathcal{R}$ 
and wcr: WCR (gsrstep (insert (c, 0)  $\mathcal{F}$ )  $\mathcal{R}$ )
shows WCR (srstep  $\mathcal{F}$   $\mathcal{R}$ )
proof -
let ?σ = const-subst c let ? $\mathcal{H}$  = insert (c, 0)  $\mathcal{F}$ 
from fresh have fresh-sys: (c, 0) ∉ funas-rel  $\mathcal{R}$  (c, 0) ∉ funas-rel ( $\mathcal{R}^{-1}$ ) using
sig
by (auto simp: funas-rel-def)
have lin: linear-sys  $\mathcal{R}$  linear-sys ( $\mathcal{R}^{-1}$ ) using llrg unfolding llrg-def by auto
have mono:  $\mathcal{F}$  ⊆ ? $\mathcal{H}$  by auto note sig-trans = subsetD[OF srstep-monp[OF
mono]]
{fix s t u assume ass: (s, t) ∈ sig-step  $\mathcal{F}$  (rrstep  $\mathcal{R}$ ) (s, u) ∈ srstep  $\mathcal{F}$   $\mathcal{R}$ 

```



**from** *ass* **have** *funas*: *funas-term*  $s \subseteq \mathcal{F}$  *funas-term*  $t \subseteq \mathcal{F}$  *funas-term*  $u \subseteq \mathcal{F}$   
**by** *blast+*  
**then** **have** *c-free*:  $(c, 0) \notin \text{funas-term } t$  **using** *fresh* **by** *blast*  
**from** *ass* **have** *gr*: *ground*  $t$  **using** *ass llrg llrg-rrsteps-groundness* **by** *blast*  
**have**  $*$ : *ground*  $(s \cdot ?\sigma)$  *ground*  $(u \cdot ?\sigma)$  **by** *auto*  
**from** *this wcr* **have**  $w$ :  $(t, u \cdot ?\sigma) \in (\text{gsrstep } ?\mathcal{H} \mathcal{R})^\downarrow$   
**using** *srstep-subst-closed*[*OF sig-trans*[*OF srrstep-to-srstep*[*OF ass(1)*]], *of*  
 $?\sigma]$   
**using** *srstep-subst-closed*[*OF sig-trans*[*OF ass(2)*]], *of*  $?\sigma]$   
**using** *ass unfolding WCR-on-def*  
**by** *auto* (*metis \* gr ground-subst-apply*)  
**have**  $(t, u) \in (\text{srstep } \mathcal{F} \mathcal{R})^\downarrow$  **unfolding** *join-def*  
**using** *remove-const-subst-relcomp-rhs*[*OF lin fresh-sys c-free*  
*gsrsteps-eq-relcomp-to-rsteps-relcomp*[*OF w*[*unfolded join-def rew-converse-inwards*]],  
*THEN rsteps-eq-relcomp-srsteps-eq-relcompI*[*OF sig funas-rel-converse*[*OF*  
*sig*] *funas(2-)*]]  
**by** (*metis* (*no-types*, *lifting*) *srstep-converse-dist*)  
**then** **show** *?thesis* **by** (*intro WCR-rrstep-intro simp*  
*qed*)

**lemma** *llrg-UNF*:

**assumes** *sig*: *funas-rel*  $\mathcal{R} \subseteq \mathcal{F}$  **and** *fresh*:  $(c, 0) \notin \mathcal{F}$   
**and** *llrg*: *llrg*  $\mathcal{R}$   
**and** *unf*: *UNF* (*gsrstep* (*insert*  $(c, 0)$   $\mathcal{F}$ )  $\mathcal{R}$ )  
**shows** *UNF* (*srstep*  $\mathcal{F} \mathcal{R}$ )  
**proof** –  
**let**  $?\sigma = \text{const-subst } c$  **let**  $?\mathcal{H} = \text{insert } (c, 0)$   $\mathcal{F}$   
**from** *fresh* **have** *fresh-sys*:  $(c, 0) \notin \text{funas-rel } \mathcal{R}$  **using** *sig*  
**by** (*auto simp: funas-rel-def*)  
**have** *mono*:  $\mathcal{F} \subseteq ?\mathcal{H}$  **by** *auto*  
**{fix**  $s$   $t$  **assume** *ass*:  $(s, t) \in \text{comp-rrstep-rel}' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R}$   
**from** *ass* **have** *funas*: *funas-term*  $s \subseteq \mathcal{F}$  *funas-term*  $t \subseteq \mathcal{F}$   
**by** (*metis Un-iff relcomp.cases srstepsD srsteps-with-root-step-srstepsD*)  
**from** *ass* **have**  $m$ :  $(s, t) \in (\text{srstep } ?\mathcal{H} (\mathcal{R}^{-1}))^* O (\text{srstep } ?\mathcal{H} \mathcal{R})^*$   
**using** *rtrancl-mono*[*OF sig-step-mono*[*OF mono*]]  
**by** (*auto dest!: srsteps-with-root-step-srsteps-eqD trancl-into-rtrancl*)  
**have** *gr*: *ground*  $s \vee \text{ground } t$  **using** *ass llrg*  
**unfolding** *srstep-converse-dist trancl-converse*  
**by** (*auto simp: llrg-srsteps-with-root-step-inv-ground llrg-srsteps-with-root-step-ground*)  
**have** *wit*:  $s \cdot ?\sigma \in \text{NF} (\text{gsrstep } ?\mathcal{H} \mathcal{R}) \implies t \cdot ?\sigma \in \text{NF} (\text{gsrstep } ?\mathcal{H} \mathcal{R}) \implies s$   
 $\cdot ?\sigma = t \cdot ?\sigma$  **using** *unf*  
**using** *srsteps-eq-subst-relcomp-closed*[*OF m*, *THEN srsteps-eq-relcomp-gsrsteps-relcomp*,  
*of*  $?\sigma]$   
**by** (*auto simp: UNF-on-def rew-converse-outwards normalizability-def*)  
**from** *funas NF-to-fresh-const-subst-NF*[*OF llrg-linear-sys*[*OF llrg*] *fresh-sys*  
*sig(1)*]  
**have**  $s \in \text{NF} (\text{srstep } \mathcal{F} \mathcal{R}) \implies s \cdot ?\sigma \in \text{NF} (\text{gsrstep } ?\mathcal{H} \mathcal{R})$   $t \in \text{NF} (\text{srstep } \mathcal{F}$   
 $\mathcal{R}) \implies t \cdot ?\sigma \in \text{NF} (\text{gsrstep } ?\mathcal{H} \mathcal{R})$

by *auto*  
 moreover have  $s \cdot ?\sigma = t \cdot ?\sigma \implies s = t$  using *gr funas fresh*  
 by (*cases ground s*) (*auto simp: ground-subst-apply funas-term-subst vars-term-empty-ground split: if-splits*)  
 ultimately have  $UN\text{-redp } \mathcal{F} \ \mathcal{R} \ s \ t$  using *wit unfolding UN-redp-def*  
 by *auto*  
 then show *?thesis* by (*intro UNF-rrstep-intro simp*)  
 qed

lemma *llrg-NFP*:

assumes *sig: funas-rel*  $\mathcal{R} \subseteq \mathcal{F}$  and *fresh: (c, 0) \notin \mathcal{F}*  
 and *llrg: llrg*  $\mathcal{R}$   
 and *nfp: NFP* (*gsrstep* (*insert* (*c, 0*)  $\mathcal{F}$ )  $\mathcal{R}$ )  
 shows *NFP* (*srstep*  $\mathcal{F} \ \mathcal{R}$ )  
 proof –  
 let  $?\sigma = \text{const-subst } c$  let  $?\mathcal{H} = \text{insert } (c, 0) \ \mathcal{F}$   
 from *fresh* have *fresh-sys: (c, 0) \notin funas-rel*  $\mathcal{R}$   
 using *sig* by (*auto simp: funas-rel-def*)  
 have *mono: \mathcal{F} \subseteq ?\mathcal{H}* by *auto*  
 {fix *s t* assume *ass: (s, t) \in comp-rrstep-rel' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R}*  
 from *ass* have *funas: funas-term s \subseteq \mathcal{F} funas-term t \subseteq \mathcal{F}*  
 by (*metis Un-iff relcomp.cases srstepsD srsteps-with-root-step-srstepsD*) +  
 from *ass* have *m: (s, t) \in (srstep ?\mathcal{H} (\mathcal{R}^{-1}))^\* O (srstep ?\mathcal{H} \mathcal{R})^\**  
 using *rtrancl-mono[OF sig-step-mono[OF mono]]*  
 by (*auto dest!: srsteps-with-root-step-srsteps-eqD trancl-into-rtrancl*)  
 have *gr: ground s \vee ground t* using *ass llrg*  
 unfolding *srstep-converse-dist trancl-converse*  
 by (*auto simp: llrg-srsteps-with-root-step-inv-ground llrg-srsteps-with-root-step-ground*)  
 have *wit: t \cdot ?\sigma \in NF (gsrstep ?\mathcal{H} \mathcal{R}) \implies (s \cdot ?\sigma, t \cdot ?\sigma) \in (gsrstep (insert (c, 0) \mathcal{F}) \mathcal{R})^\**  
 using *NFP-stepD[OF nfp]*  
 using *srsteps-eq-subst-relcomp-closed[OF m, THEN srsteps-eq-relcomp-gsrsteps-relcomp, of ?\sigma]*  
 by (*auto simp: rew-converse-outwards*)  
 from *funas NF-to-fresh-const-subst-NF[OF llrg-linear-sys[OF llrg] fresh-sys(1) sig(1)]*  
 have  $t \in NF (srstep \ \mathcal{F} \ \mathcal{R}) \implies t \cdot ?\sigma \in NF (gsrstep \ ?\mathcal{H} \ \mathcal{R})$  by *auto*  
 moreover have  $(s \cdot ?\sigma, t \cdot ?\sigma) \in (rstep \ \mathcal{R})^* \implies (s, t) \in (srstep \ \mathcal{F} \ \mathcal{R})^*$  using  
*gr funas fresh*  
 using *remove-const-subst-steps-eq-lhs[OF llrg-linear-sys[OF llrg] fresh-sys, THEN rsteps-eq-srsteps-eqI[OF sig funas]]*  
 using *remove-const-subst-steps-eq-rhs[OF llrg-linear-sys[OF llrg] fresh-sys, THEN rsteps-eq-srsteps-eqI[OF sig funas]]*  
 by (*cases ground s*) (*auto simp: ground-subst-apply*)  
 ultimately have  $NFP\text{-redp } \mathcal{F} \ \mathcal{R} \ s \ t$  using *wit unfolding NFP-redp-def*  
 by (*auto dest: gsrsteps-eq-to-rsteps-eq*)  
 then show *?thesis* by (*intro NFP-rrstep-intro simp*)  
 qed

**lemma** *llrg-NE-aux*:

**assumes**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{R}$   
**and**  $\text{sig}: \text{funas-rel } \mathcal{R} \subseteq \mathcal{F} \ \text{funas-rel } \mathcal{S} \subseteq \mathcal{F}$  **and**  $\text{fresh}: (c, 0) \notin \mathcal{F}$   
**and**  $\text{llrg}: \text{llrg } \mathcal{R} \ \text{llrg } \mathcal{S}$   
**and**  $\text{ne}: \text{NE } (\text{gsrstep } (\text{insert } (c, 0) \ \mathcal{F}) \ \mathcal{R}) \ (\text{gsrstep } (\text{insert } (c, 0) \ \mathcal{F}) \ \mathcal{S})$   
**shows**  $\text{NE-redp } \mathcal{F} \ \mathcal{R} \ \mathcal{S} \ s \ t$

**proof** –

**from**  $\text{fresh}$  **have**  $\text{fresh-sys}: (c, 0) \notin \text{funas-rel } \mathcal{R} \ (c, 0) \notin \text{funas-rel } \mathcal{S}$   
**using**  $\text{sig}$  **by**  $(\text{auto simp: funas-rel-def})$   
**let**  $?\sigma = \text{const-subst } c$  **let**  $?\mathcal{H} = \text{insert } (c, 0) \ \mathcal{F}$   
**let**  $?\mathcal{H} = \text{insert } (c, 0) \ \mathcal{F}$  **have**  $\text{mono}: \mathcal{F} \subseteq ?\mathcal{H}$  **by**  $\text{auto}$   
**from**  $\text{assms}(1)$  **have**  $\text{gr}: \text{ground } t$  **and**  $\text{funas}: \text{funas-term } s \subseteq \mathcal{F} \ \text{funas-term } t \subseteq$

$\mathcal{F}$

**using**  $\text{llrg-srsteps-with-root-step-ground}[OF \ \text{llrg}(1)]$   
**by**  $(\text{meson srstepsD srsteps-with-root-step-srstepsD})+$

**from**  $\text{funas}$  **have**  $\text{fresh-t}: (c, 0) \notin \text{funas-term } t$  **using**  $\text{fresh}$  **by**  $\text{auto}$

**from**  $\text{srsteps-subst-closed}[OF \ \text{srsteps-monp}[OF \ \text{mono}, \ \text{THEN } \text{subsetD}, \ \text{OF } \text{srsteps-with-root-step-srstepsD}[OF \ \text{assms}(1)]]]$ ,  $\text{of } ?\sigma$

**have**  $(s \cdot ?\sigma, t) \in (\text{gsrstep } ?\mathcal{H} \ \mathcal{R})^+$  **using**  $\text{gr}$

**by**  $(\text{auto simp: ground-subst-apply intro!: ground-srsteps-gsrsteps})$

**then**  $\text{have } t \in \text{NF } (\text{srstep } \mathcal{F} \ \mathcal{R}) \implies (s \cdot ?\sigma, t) \in (\text{gsrstep } (\text{insert } (c, 0) \ \mathcal{F}) \ \mathcal{S})^*$

**using**  $\text{NF-to-fresh-const-subst-NF}[OF \ \text{llrg-linear-sys}[OF \ \text{llrg}(1)] \ \text{fresh-sys}(1)]$   
 $\text{sig}(1) \ \text{funas}(2)$ ,  $\text{of } ?\mathcal{H}$

**using**  $\text{gr}$   $\text{NE-NF-eq}[OF \ \text{ne}, \ \text{symmetric}]$   $\text{ne}$  **unfolding**  $\text{NE-on-def}$

**by**  $(\text{auto simp: normalizability-def ground-subst-apply dest!: trancl-into-rtrancl})$

**then** **show**  $\text{NE-redp } \mathcal{F} \ \mathcal{R} \ \mathcal{S} \ s \ t$  **unfolding**  $\text{NE-redp-def}$

**using**  $\text{remove-const-subst-steps-eq-lhs}[OF \ \text{llrg-linear-sys}[OF \ \text{llrg}(2)] \ \text{fresh-sys}(2)]$   
 $\text{fresh-t}$ ,

$\text{THEN } \text{rsteps-eq-srsteps-eqI}[OF \ \text{sig}(2) \ \text{funas}]$

**by**  $(\text{auto dest: gsrsteps-eq-to-rsteps-eq})$

**qed**

**lemma** *llrg-NE*:

**assumes**  $\text{sig}: \text{funas-rel } \mathcal{R} \subseteq \mathcal{F} \ \text{funas-rel } \mathcal{S} \subseteq \mathcal{F}$  **and**  $\text{fresh}: (c, 0) \notin \mathcal{F}$

**and**  $\text{llrg}: \text{llrg } \mathcal{R} \ \text{llrg } \mathcal{S}$

**and**  $\text{ne}: \text{NE } (\text{gsrstep } (\text{insert } (c, 0) \ \mathcal{F}) \ \mathcal{R}) \ (\text{gsrstep } (\text{insert } (c, 0) \ \mathcal{F}) \ \mathcal{S})$

**shows**  $\text{NE } (\text{srstep } \mathcal{F} \ \mathcal{R}) \ (\text{srstep } \mathcal{F} \ \mathcal{S})$

**proof** –

**have**  $f: (c, 0) \notin \text{funas-rel } \mathcal{R} \ (c, 0) \notin \text{funas-rel } \mathcal{S}$  **using**  $\text{fresh}$   $\text{sig}$

**by**  $(\text{auto simp: funas-rel-def})$

**{fix**  $s \ t$  **assume**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{R}$

**from**  $\text{llrg-NE-aux}[OF \ \text{this } \text{sig } \text{fresh } \text{llrg } \text{ne}]$  **have**  $\text{NE-redp } \mathcal{F} \ \mathcal{R} \ \mathcal{S} \ s \ t$

**by**  $\text{simp}$ }

**moreover**

**{fix**  $s \ t$  **assume**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{S}$

**from**  $\text{llrg-NE-aux}[OF \ \text{this } \text{sig}(2), 1] \ \text{fresh } \text{llrg}(2), 1) \ \text{NE-symmetric}[OF \ \text{ne}]$

```

  have NE-redp  $\mathcal{F} \mathcal{S} \mathcal{R} s t$  by simp}
  ultimately show ?thesis using linear-sys-gNF-eq-NF-eq[OF llrg-linear-sys[OF llrg(1)]]
  llrg-linear-sys[OF llrg(2)] sig f - - NE-NF-eq[OF ne]
  by (intro NE-rrstep-intro) auto
qed

```

## 6.1 Specialized for monadic signature

```

lemma monadic-commute:
  assumes llrg  $\mathcal{R}$  llrg  $\mathcal{S}$  monadic  $\mathcal{F}$ 
  and com: commute (gsrstep  $\mathcal{F} \mathcal{R}$ ) (gsrstep  $\mathcal{F} \mathcal{S}$ )
  shows commute (srstep  $\mathcal{F} \mathcal{R}$ ) (srstep  $\mathcal{F} \mathcal{S}$ )
proof -
  {fix s t assume ass:  $(s, t) \in \text{comp-rrstep-rel}' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{S}$ 
  then obtain u where steps:  $(s, u) \in (\text{srstep } \mathcal{F} (\mathcal{R}^{-1}))^+ (u, t) \in (\text{srstep } \mathcal{F} \mathcal{S})^+$ 
  by (auto simp: sig-step-converse-rstep dest: srsteps-with-root-step-srstepsD)
  have gr: ground s ground t using steps assms(1 - 3)
  unfolding rew-converse-outwards
  by (auto simp: llrg-srsteps-with-root-step-ground llrg-monadic-rsteps-groundness)
  from steps(1) have f: funas-term s  $\subseteq \mathcal{F}$  using srstepsD by blast
  let ? $\sigma = \lambda \cdot . s$ 
  from steps gr have  $(s, u \cdot ?\sigma) \in (\text{gsrstep } \mathcal{F} (\mathcal{R}^{-1}))^+ (u \cdot ?\sigma, t) \in (\text{gsrstep } \mathcal{F} \mathcal{S})^+$ 
  unfolding rew-converse-outwards
  using srsteps-subst-closed[where ? $\sigma = ?\sigma$  and ?s = u, of -  $\mathcal{F}$ ] f
  by (force simp: ground-subst-apply intro: ground-srsteps-gsrsteps)
  then have  $(s, t) \in (\text{gsrstep } \mathcal{F} \mathcal{S})^* O (\text{gsrstep } \mathcal{F} (\mathcal{R}^{-1}))^*$  using com
  unfolding commute-def rew-converse-inwards
  by (meson relcomp.relcompI subsetD trancl-into-rtrancl)
  from gsrsteps-eq-relcomp-srsteps-relcompD[OF this]
  have commute-redp  $\mathcal{F} \mathcal{R} \mathcal{S} s t$  unfolding commute-redp-def
  by (simp add: rew-converse-inwards)}
  then show ?thesis by (intro commute-rrstep-intro simp)
qed

```

```

lemma monadic-CR:
  assumes llrg  $\mathcal{R}$  monadic  $\mathcal{F}$ 
  and CR (gsrstep  $\mathcal{F} \mathcal{R}$ )
  shows CR (srstep  $\mathcal{F} \mathcal{R}$ ) using monadic-commute assms
  unfolding CR-iff-self-commute
  by blast

```

```

lemma monadic-SCR:
  assumes sig: funas-rel  $\mathcal{R} \subseteq \mathcal{F}$  monadic  $\mathcal{F}$ 
  and llrg: llrg  $\mathcal{R}$ 
  and scr: SCR (gsrstep  $\mathcal{F} \mathcal{R}$ )

```

shows  $SCR (srstep \mathcal{F} \mathcal{R})$   
**proof** –  
 {**fix**  $s t u$  **assume**  $ass: (s, t) \in srstep \mathcal{F} \mathcal{R} (s, u) \in srstep \mathcal{F} \mathcal{R}$   
**and**  $root: (s, t) \in sig\text{-step} \mathcal{F} (rrstep \mathcal{R}) \vee (s, u) \in sig\text{-step} \mathcal{F} (rrstep \mathcal{R})$   
**from**  $ass$  **have**  $funas: funas\text{-term } s \subseteq \mathcal{F} \text{ funas-term } t \subseteq \mathcal{F} \text{ funas-term } u \subseteq \mathcal{F}$   
**by**  $blast+$   
**from**  $root$  **have**  $gr: ground\ t\ ground\ u$  **using**  $ass\ sig(2)\ llrg$   
**by**  $(metis\ llrg\ monadic\ rstep\ pres\ groundness)+$   
**let**  $?\sigma = \lambda \cdot. t$  **have**  $grs: ground\ (s \cdot ?\sigma)$  **using**  $gr$  **by**  $auto$   
**from**  $this\ scr$  **obtain**  $v$  **where**  $v: (t, v) \in (gsrstep \mathcal{F} \mathcal{R})^= \wedge (u, v) \in (gsrstep \mathcal{F} \mathcal{R})^*$   
**using**  $srstep\ subst\ closed[OF\ ass(1),\ of\ ?\sigma]$   
**using**  $srstep\ subst\ closed[OF\ ass(2),\ of\ ?\sigma]$   
**using**  $gr$  **unfolding**  $SCR\ on\ def$   
**by**  $(metis\ Int\ iff\ UNIV\ I\ funas(2)\ ground\ subst\ apply\ mem\ Collect\ eq\ mem\ Sigma\ iff)$   
**then** **have**  $SCR_p \mathcal{F} \mathcal{R} t u$   
**by**  $(metis\ Int\ iff\ Un\ iff\ gsrsteps\ eq\ to\ srsteps\ eq)}$   
**then** **show**  $?thesis$  **by**  $(intro\ SCR\ rrstep\ intro)\ (metis\ srrstep\ to\ srestep)+$   
**qed**

**lemma**  $monadic\ WCR:$

**assumes**  $sig: funas\ rel \mathcal{R} \subseteq \mathcal{F} \text{ monadic } \mathcal{F}$   
**and**  $llrg: llrg \mathcal{R}$   
**and**  $wcr: WCR (gsrstep \mathcal{F} \mathcal{R})$   
**shows**  $WCR (srstep \mathcal{F} \mathcal{R})$   
**proof** –  
 {**fix**  $s t u$  **assume**  $ass: (s, t) \in sig\text{-step} \mathcal{F} (rrstep \mathcal{R}) (s, u) \in srstep \mathcal{F} \mathcal{R}$   
**from**  $ass$  **have**  $funas: funas\text{-term } s \subseteq \mathcal{F} \text{ funas-term } t \subseteq \mathcal{F} \text{ funas-term } u \subseteq \mathcal{F}$   
**by**  $blast+$   
**from**  $srrstep\ to\ srestep\ ass$  **have**  $gr: ground\ t\ ground\ u$  **using**  $ass\ sig(2)\ llrg$   
**by**  $(metis\ llrg\ monadic\ rstep\ pres\ groundness)+$   
**let**  $?\sigma = \lambda \cdot. t$  **have**  $grs: ground\ (s \cdot ?\sigma)$  **using**  $gr$  **by**  $auto$   
**from**  $this\ wcr$  **have**  $w: (t, u) \in (gsrstep \mathcal{F} \mathcal{R})^\downarrow$  **using**  $gr\ ass(2)\ funas$   
**using**  $srstep\ subst\ closed[OF\ srrstep\ to\ srestep[OF\ ass(1)],\ of\ ?\sigma]$   
**using**  $srstep\ subst\ closed[OF\ ass(2),\ of\ ?\sigma]$   
**unfolding**  $WCR\ on\ def$   
**by**  $(metis\ IntI\ SigmaI\ UNIV\ I\ ground\ subst\ apply\ mem\ Collect\ eq)$   
**then** **have**  $(t, u) \in (srstep \mathcal{F} \mathcal{R})^\downarrow$  **unfolding**  $join\ def$   
**by**  $(metis\ gsrsteps\ eq\ to\ srsteps\ eq\ joinD\ joinI\ join\ def)}$   
**then** **show**  $?thesis$  **by**  $(intro\ WCR\ rrstep\ intro)\ simp$   
**qed**

**lemma**  $monadic\ UNF:$

**assumes**  $llrg \mathcal{R} \text{ monadic } \mathcal{F}$   
**and**  $unf: UNF (gsrstep \mathcal{F} \mathcal{R})$   
**shows**  $UNF (srstep \mathcal{F} \mathcal{R})$   
**proof** –  
 {**fix**  $s t$  **assume**  $ass: (s, t) \in comp\ rrstep\ rel' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R}$   
**then** **obtain**  $u$  **where**  $steps: (s, u) \in (srstep \mathcal{F} (\mathcal{R}^{-1}))^+ (u, t) \in (srstep \mathcal{F}$

$\mathcal{R})^+$   
 by (auto simp: sig-step-converse-rstep dest: srsteps-with-root-step-srstepsD)  
 have gr: ground s ground t using steps assms(1, 2)  
 unfolding rew-converse-outwards  
 by (auto simp: llrg-srsteps-with-root-step-ground llrg-monic-rsteps-groundness)  
 from steps(1) have f: funas-term s  $\subseteq$   $\mathcal{F}$  using srstepsD by blast  
 let ? $\sigma$  =  $\lambda$  -. s  
 from steps gr have (s, u · ? $\sigma$ )  $\in$  (gsrstep  $\mathcal{F}$  ( $\mathcal{R}^{-1}$ ))<sup>+</sup> (u · ? $\sigma$ , t)  $\in$  (gsrstep  $\mathcal{F}$   $\mathcal{R}$ )<sup>+</sup>  
 unfolding rew-converse-outwards  
 using srsteps-subst-closed[where ? $\sigma$  = ? $\sigma$  and ?s = u, of -  $\mathcal{F}$ ] f  
 by (force simp: ground-subst-apply intro: ground-srsteps-gsrsteps)+  
 then have UN-redp  $\mathcal{F}$   $\mathcal{R}$  s t using unf ground-NF-srstep-gsrstep[OF gr(1), of  $\mathcal{F}$   $\mathcal{R}$ ]  
 using ground-NF-srstep-gsrstep[OF gr(2), of  $\mathcal{F}$   $\mathcal{R}$ ]  
 by (auto simp: UNF-on-def UN-redp-def normalizability-def rew-converse-outwards  
 (meson trancl-into-rtrancl})  
 then show ?thesis by (intro UNF-rrstep-intro) simp  
 qed

lemma monadic-UNC:

assumes llrg  $\mathcal{R}$  monadic  $\mathcal{F}$   
 and well: funas-rel  $\mathcal{R} \subseteq \mathcal{F}$   
 and unc: UNC (gsrstep  $\mathcal{F}$   $\mathcal{R}$ )  
 shows UNC (srstep  $\mathcal{F}$   $\mathcal{R}$ )  
 proof –  
 {fix s t assume ass: (s, t)  $\in$  (srstep  $\mathcal{F}$   $\mathcal{R}$ )<sup>++\*</sup> s  $\in$  NF (srstep  $\mathcal{F}$   $\mathcal{R}$ ) t  $\in$  NF  
 (srstep  $\mathcal{F}$   $\mathcal{R}$ )  
 then have s = t  
 proof (cases s = t)  
 case False  
 then have  $\exists$  s' t'. (s, t)  $\in$  (srstep  $\mathcal{F}$   $\mathcal{R}$ )<sup>++\*</sup>  $\wedge$  (s', s)  $\in$  (srstep  $\mathcal{F}$   $\mathcal{R}$ )  $\wedge$  (t',  
 t)  $\in$  (srstep  $\mathcal{F}$   $\mathcal{R}$ )  
 using ass by (auto simp: conversion-def rtrancl-eq-or-trancl dest: tranclD  
 tranclD2)  
 then obtain s' t' where split: (s, t)  $\in$  (srstep  $\mathcal{F}$   $\mathcal{R}$ )<sup>++\*</sup> (s', s)  $\in$  (srstep  $\mathcal{F}$   
 $\mathcal{R}$ ) (t', t)  $\in$  (srstep  $\mathcal{F}$   $\mathcal{R}$ ) by blast  
 from split(2, 3) have gr: ground s ground t using llrg-monic-rstep-pres-groundness[OF  
 assms(1, 2)]  
 by blast+  
 from ground-srsteps-eq-gsrsteps-eq[OF this ass(1)[unfolded sig-step-conversion-dist]]  
 have (s, t)  $\in$  (gsrstep  $\mathcal{F}$   $\mathcal{R}$ )<sup>++\*</sup>  
 unfolding conversion-def Restr-smycl-dist  
 by (simp add: rstep-converse-dist srstep-symcl-dist)  
 then show ?thesis using ass(2-) unc gr  
 by (auto simp: UNC-def ground-NF-srstep-gsrstep)  
 qed auto}  
 then show ?thesis by (auto simp: UNC-def UN-redp-def)

qed

**lemma** *monadic-NFP*:

**assumes** *llrg*  $\mathcal{R}$  *monadic*  $\mathcal{F}$   
**and** *nfp*: *NFP* (*gsrstep*  $\mathcal{F}$   $\mathcal{R}$ )  
**shows** *NFP* (*srstep*  $\mathcal{F}$   $\mathcal{R}$ )

**proof** –

{**fix** *s t u* **assume** *ass*:  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   $(s, u) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   $u \in \text{NF}$   
(*srstep*  $\mathcal{F}$   $\mathcal{R}$ )

**have**  $(t, u) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$

**proof** (*cases*  $s = u \vee s = t$ )

**case** [*simp*]: *True* **show** *?thesis* **using** *ass*

**by** (*metis* *NF-not-suc* *True* *rtrancl.rtrancl-refl*)

**next**

**case** *False*

**then have** *steps*:  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$   $(s, u) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$  **using** *ass*(1,

2)

**by** (*auto* *simp* *add*: *rtrancl-eq-or-trancl*)

**then have** *gr*: *ground* *t* *ground* *u* **using** *assms*(1, 2) *llrg-monadic-rsteps-groundness*

**by** *blast+*

**let**  $?\sigma = \lambda -. t$  **from** *steps* *gr* **have**  $(s \cdot ?\sigma, t) \in (\text{gsrstep } \mathcal{F} \mathcal{R})^+$   $(s \cdot ?\sigma, u)$   
 $\in (\text{gsrstep } \mathcal{F} \mathcal{R})^+$

**by** (*auto* *intro!*: *ground-srsteps-gsrsteps*)

(*metis* *ground-subst-apply* *srstepsD* *srsteps-subst-closed*)+

**then have**  $(t, u) \in (\text{gsrstep } \mathcal{F} \mathcal{R})^*$  **using** *nfp* *ass*(3) *gr*

**by** (*auto* *simp*: *NFP-on-def*) (*metis* *ground-NF-srstep-gsrstep* *normalizability-I* *trancl-into-rtrancl*)

**then show** *?thesis* **by** (*auto* *dest*: *gsrsteps-eq-to-srsteps-eq*)

**qed**}

**then show** *?thesis* **unfolding** *NFP-on-def*

**by** *auto*

qed

end

## 7 Reducing Rewrite Properties to Properties on Ground Terms over Ground Systems

**theory** *Ground-Reduction-on-GTRS*

**imports**

*Rewriting-Properties*

*Rewriting-GTRS*

*Rewriting-LLRG-LV-Mondaic*

**begin**

**lemma** *ground-sys-nf-eq-lift*:

**fixes**  $\mathcal{R} :: ('f, 'v) \text{ term rel}$   
**assumes**  $\text{gtrs: ground-sys } \mathcal{R} \text{ ground-sys } \mathcal{S}$   
**and**  $\text{nf: NF (gsrstep } \mathcal{F} \mathcal{R}) = \text{NF (gsrstep } \mathcal{F} \mathcal{S})$   
**shows**  $\text{NF (srstep } \mathcal{F} \mathcal{R}) = \text{NF (srstep } \mathcal{F} \mathcal{S})$   
**proof** –  
**{fix**  $s \mathcal{U} \mathcal{V}$  **assume**  $\text{ass: ground-sys } (\mathcal{U} :: ('f, 'v) \text{ term rel}) \text{ ground-sys } \mathcal{V}$   
 $\text{NF (gsrstep } \mathcal{F} \mathcal{U}) = \text{NF (gsrstep } \mathcal{F} \mathcal{V}) \ s \in \text{NF (srstep } \mathcal{F} \mathcal{U})$   
**have**  $s \in \text{NF (srstep } \mathcal{F} \mathcal{V})$   
**proof** (*rule ccontr*)  
**assume**  $s \notin \text{NF (srstep } \mathcal{F} \mathcal{V})$   
**then obtain**  $C \ l \ r \ \sigma$  **where**  $\text{step: } (l, r) \in \mathcal{V}$  **and**  $\text{rep: } s = C \langle l \cdot \sigma \rangle$   
**and**  $\text{funas: funas-ctxt } C \subseteq \mathcal{F} \ \text{funas-term } l \subseteq \mathcal{F} \ \text{funas-term } r \subseteq \mathcal{F}$  **using**  
 $\text{ass}(2)$   
**by** (*auto simp: funas-term-subst NF-def sig-step-def dest!: rstep-imp-C-s-r*)  
*blast*  
**from**  $\text{step ass}(2)$  **rep** **have**  $\text{rep: } s = C \langle l \rangle \ \text{ground } l \ \text{ground } r$   
**by** (*auto intro: ground-subst-apply*)  
**from**  $\text{step rep}(2-)$  **funas** **have**  $l \notin \text{NF (gsrstep } \mathcal{F} \mathcal{V})$   
**by** (*auto simp: NF-def sig-step-def Image-def*)  
**from**  $\text{this ass}(3)$  **have**  $l \notin \text{NF (srstep } \mathcal{F} \mathcal{U})$  **by** *auto*  
**then obtain**  $t$  **where**  $(l, t) \in \text{srstep } \mathcal{F} \mathcal{U}$  **by** *auto*  
**from**  $\text{srstep-ctxt-closed}[OF \ \text{funas}(1) \ \text{this}, \ \text{unfolded rep}(1)[\text{symmetric}]]$   
**show** *False* **using**  $\text{ass}(4)$   
**by** *auto*  
**qed}**  
**then show** *?thesis* **using**  $\text{assms}$   
**by** (*smt (verit, best) equalityI subsetI*)  
**qed**

**lemma** *ground-sys-inv:*  
 $\text{ground-sys } \mathcal{R} \implies \text{ground-sys } (\mathcal{R}^{-1})$  **by** *auto*

**lemma** *ground-sys-symcl:*  
 $\text{ground-sys } \mathcal{R} \implies \text{ground-sys } (\mathcal{R}^{\leftrightarrow})$  **by** *auto*

**lemma** *ground-sys-comp-rrstep-rel'-ground:*  
**assumes**  $\text{ground-sys } \mathcal{R} \ \text{ground-sys } \mathcal{S}$   
**and**  $(s, t) \in \text{comp-rrstep-rel}' \ \mathcal{F} \ \mathcal{R} \ \mathcal{S}$   
**shows**  $\text{ground } s \ \text{ground } t$   
**proof** –  
**from**  $\text{assms}(3)$  **consider** (a)  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{R} \ O \ (\text{srstep } \mathcal{F} \ \mathcal{S})^+ |$   
(b)  $(s, t) \in (\text{srstep } \mathcal{F} \ \mathcal{R})^+ \ O \ \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{S}$   
**by** *auto*  
**then have**  $\text{ground } s \ \wedge \ \text{ground } t$   
**proof** *cases*  
**case** *a*  
**then show** *?thesis* **using**  $\text{srsteps-with-root-step-ground}(1)[OF \ \text{assms}(1)]$   
**using**  $\text{srsteps-with-root-step-ground}(2)[OF \ \text{assms}(1), \ \text{THEN srsteps-pres-ground-l}[OF$



```

assms(2)]
  by blast
next
case b
then show ?thesis using srsteps-with-root-step-ground(2)[OF assms(2)]
using srsteps-with-root-step-ground(1)[OF assms(2), THEN srsteps-pres-ground-r[OF
assms(1)]]
  by blast
qed
then show ground s ground t by simp-all
qed

```

lemma *GTRS-commute*:

```

assumes ground-sys  $\mathcal{R}$  ground-sys  $\mathcal{S}$ 
and com: commute (gsrstep  $\mathcal{F}$   $\mathcal{R}$ ) (gsrstep  $\mathcal{F}$   $\mathcal{S}$ )
shows commute (srstep  $\mathcal{F}$   $\mathcal{R}$ ) (srstep  $\mathcal{F}$   $\mathcal{S}$ )
proof -
  {fix s t assume ass:  $(s, t) \in \text{comp-rrstep-rel}' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{S}$ 
  then obtain u where steps:  $(s, u) \in (\text{srstep } \mathcal{F} (\mathcal{R}^{-1}))^+ (u, t) \in (\text{srstep } \mathcal{F} \mathcal{S})^+$ 
  by (auto simp: sig-step-converse-rstep dest: srsteps-with-root-step-srstepsD)
  have gr: ground s ground t ground u
  using ground-sys-comp-rrstep-rel'-ground[OF ground-sys-inv[OF assms(1)]]
  assms(2) ass]
  using srsteps-pres-ground-r[OF assms(2) - steps(2)] by auto
  then have  $(s, u) \in (\text{gsrstep } \mathcal{F} (\mathcal{R}^{-1}))^* (u, t) \in (\text{gsrstep } \mathcal{F} \mathcal{S})^*$  using steps
  by (auto dest!: trancl-into-rtrancl intro: ground-srsteps-eq-gsrsteps-eq)
  then have  $(s, t) \in (\text{gsrstep } \mathcal{F} \mathcal{S})^* O (\text{gsrstep } \mathcal{F} (\mathcal{R}^{-1}))^*$  using com steps
  by (auto simp: commute-def rew-converse-inwards)
  from gsrsteps-eq-relcomp-srsteps-relcompD[OF this]
  have commute-redp  $\mathcal{F} \mathcal{R} \mathcal{S} s t$  unfolding commute-redp-def
  by (simp add: rew-converse-inwards)}
  then show ?thesis by (intro commute-rrstep-intro) simp
qed

```

lemma *GTRS-CR*:

```

assumes ground-sys  $\mathcal{R}$ 
and CR (gsrstep  $\mathcal{F}$   $\mathcal{R}$ )
shows CR (srstep  $\mathcal{F}$   $\mathcal{R}$ ) using GTRS-commute assms
unfolding CR-iff-self-commute
by blast

```

lemma *GTRS-SCR*:

```

assumes gtrs: ground-sys  $\mathcal{R}$ 
and scr: SCR (gsrstep  $\mathcal{F}$   $\mathcal{R}$ )
shows SCR (srstep  $\mathcal{F}$   $\mathcal{R}$ )
proof -
  {fix s t u assume ass:  $(s, t) \in \text{srstep } \mathcal{F} \mathcal{R} (s, u) \in \text{srstep } \mathcal{F} \mathcal{R}$ 

```

**and root:**  $(s, t) \in \text{sig-step } \mathcal{F} (\text{rrstep } \mathcal{R}) \vee (s, u) \in \text{sig-step } \mathcal{F} (\text{rrstep } \mathcal{R})$   
**from ass have funas:**  $\text{funas-term } s \subseteq \mathcal{F} \text{ funas-term } t \subseteq \mathcal{F} \text{ funas-term } u \subseteq \mathcal{F}$   
**by** *blast+*  
**from root have gr:**  $\text{ground } s \text{ ground } t \text{ ground } u$  **using** *ass gtrs*  
**using** *srrstep-ground srstep-pres-ground-l srstep-pres-ground-r*  
**by** *metis+*  
**from scr obtain v where**  $v: (t, v) \in (\text{gsrstep } \mathcal{F} \mathcal{R})^= \wedge (u, v) \in (\text{gsrstep } \mathcal{F} \mathcal{R})^*$   
**using gr unfolding** *SCR-on-def*  
**by** *(metis Int-iff UNIV-I ass mem-Collect-eq mem-Sigma-iff)*  
**then have** *SCRp  $\mathcal{F} \mathcal{R} t u$*   
**by** *(metis (full-types) Int-iff Un-iff gsrsteps-eq-to-srsteps-eq)*  
**then show** *?thesis* **by** *(intro SCR-rrstep-intro) (metis srrstep-to-srstep)+*  
**qed**

**lemma** *GTRS-WCR:*

**assumes** *gtrs: ground-sys  $\mathcal{R}$*   
**and** *wcr: WCR (gsrstep  $\mathcal{F} \mathcal{R}$ )*  
**shows** *WCR (srstep  $\mathcal{F} \mathcal{R}$ )*  
**proof** –  
**{fix s t u assume ass:**  $(s, t) \in \text{sig-step } \mathcal{F} (\text{rrstep } \mathcal{R}) (s, u) \in \text{srstep } \mathcal{F} \mathcal{R}$   
**from ass have funas:**  $\text{funas-term } s \subseteq \mathcal{F} \text{ funas-term } t \subseteq \mathcal{F} \text{ funas-term } u \subseteq \mathcal{F}$   
**by** *blast+*  
**from srrstep-ground[OF gtrs ass(1)] have gr:**  $\text{ground } s \text{ ground } t \text{ ground } u$   
**using** *srstep-pres-ground-l[OF gtrs - ass(2)]*  
**by** *simp-all*  
**from this wcr have w:**  $(t, u) \in (\text{gsrstep } \mathcal{F} \mathcal{R})^\downarrow$  **using** *ass funas*  
**unfolding** *WCR-on-def*  
**by** *(metis IntI SigmaI UNIV-I mem-Collect-eq srrstep-to-srstep)*  
**then have**  $(t, u) \in (\text{srstep } \mathcal{F} \mathcal{R})^\downarrow$  **unfolding** *join-def*  
**by** *(metis (full-types) gsrsteps-eq-to-srsteps-eq joinD joinI join-def)*  
**then show** *?thesis* **by** *(intro WCR-rrstep-intro) simp*  
**qed**

**lemma** *GTRS-UNF:*

**assumes** *gtrs: ground-sys  $\mathcal{R}$*   
**and** *unf: UNF (gsrstep  $\mathcal{F} \mathcal{R}$ )*  
**shows** *UNF (srstep  $\mathcal{F} \mathcal{R}$ )*  
**proof** –  
**{fix s t assume ass:**  $(s, t) \in \text{comp-rrstep-rel}' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R}$   
**then obtain u where steps:**  $(s, u) \in (\text{srstep } \mathcal{F} (\mathcal{R}^{-1}))^+ (u, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^+$   
**by** *(auto simp: sig-step-converse-rstep dest: srsteps-with-root-step-srstepsD)*  
**have gr:**  $\text{ground } s \text{ ground } t \text{ ground } u$   
**using** *ground-sys-comp-rrstep-rel'-ground[OF ground-sys-inv[OF gtrs] gtrs ass]*  
**using** *srsteps-pres-ground-r[OF gtrs - steps(2)]* **by** *auto*  
**from steps(1) have f:**  $\text{funas-term } s \subseteq \mathcal{F}$  **by** *(simp add: srstepsD)*  
**let**  $?\sigma = \lambda \cdot s$   
**from steps gr have**  $(s, u \cdot ?\sigma) \in (\text{gsrstep } \mathcal{F} (\mathcal{R}^{-1}))^+ (u \cdot ?\sigma, t) \in (\text{gsrstep } \mathcal{F} \mathcal{R})^+$

$\mathcal{R})^+$   
**unfolding** *srstep-converse-dist Restr-converse trancl-converse*  
**using** *srsteps-subst-closed*[**where**  $?σ = ?σ$  **and**  $?s = u$ , of -  $\mathcal{F}$ ] *f*  
**by** (*force simp: ground-subst-apply intro: ground-srsteps-gsrsteps*)  
**then have** *UN-redp  $\mathcal{F} \mathcal{R} s t$*  **using** *unf ground-NF-srstep-gsrstep*[*OF gr(1)*, of  
 $\mathcal{F} \mathcal{R}$ ]  
**using** *ground-NF-srstep-gsrstep*[*OF gr(2)*, of  $\mathcal{F} \mathcal{R}$ ]  
**by** (*auto simp: UNF-on-def UN-redp-def normalizability-def rew-converse-outwards*)  
*(meson trancl-into-rtrancl)*  
**then show** *?thesis* **by** (*intro UNF-rrstep-intro simp*)  
**qed**

**lemma** *GTRS-UNC:*

**assumes** *gtrs: ground-sys  $\mathcal{R}$*   
**and** *unc: UNC (gsrstep  $\mathcal{F} \mathcal{R}$ )*  
**shows** *UNC (srstep  $\mathcal{F} \mathcal{R}$ )*

**proof** –

**{fix  $s t$  assume** *ass:  $(s, t) \in srsteps-with-root-step \mathcal{F} (\mathcal{R}^{\leftrightarrow})$*   
**from** *ass* **have** *funas: funas-term  $s \subseteq \mathcal{F}$  funas-term  $t \subseteq \mathcal{F}$*   
**by** (*meson srstepsD srsteps-with-root-step-srstepsD*)  
**from** *ass* **have** *ground  $s$  ground  $t$*  **using** *srsteps-with-root-step-ground*[*OF*  
*ground-sys-symcl*[*OF gtrs*]]  
**by** *auto*  
**then have** *UN-redp  $\mathcal{F} \mathcal{R} s t$*  **unfolding** *UN-redp-def* **using** *ass unc* **unfolding**  
*UNC-def*  
**by** (*simp add: ground-NF-srstep-gsrstep ground-srsteps-eq-gsrsteps-eq gsrstep-conversion-dist*  
*srsteps-with-root-step-srsteps-eqD*)  
**then show** *?thesis* **by** (*intro UNC-rrstep-intro simp*)  
**qed**

**lemma** *GTRS-NFP:*

**assumes** *ground-sys  $\mathcal{R}$*   
**and** *nfp: NFP (gsrstep  $\mathcal{F} \mathcal{R}$ )*  
**shows** *NFP (srstep  $\mathcal{F} \mathcal{R}$ )*

**proof** –

**{fix  $s t$  assume** *ass:  $(s, t) \in comp-rrstep-rel' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R}$*   
**from** *ass* **have** *funas: funas-term  $s \subseteq \mathcal{F}$  funas-term  $t \subseteq \mathcal{F}$*   
**by** (*meson Un-iff relcompEpair srstepsD srsteps-with-root-step-srstepsD*)  
**from** *ass* **have** *ground  $s$  ground  $t$*   
**by** (*metis assms(1) ground-sys-comp-rrstep-rel'-ground ground-sys-inv*)  
**from** *this ass* **have**  *$(s, t) \in (gsrstep \mathcal{F} (\mathcal{R}^{-1}))^* O (gsrstep \mathcal{F} \mathcal{R})^*$*   
**by** (*intro srsteps-eq-relcomp-gsrsteps-relcomp*) (*auto dest!: srsteps-with-root-step-srsteps-eqD*)  
**then have**  *$t \in NF (gsrstep \mathcal{F} \mathcal{R}) \implies (s, t) \in (gsrstep \mathcal{F} \mathcal{R})^*$*  **using**  
*NFP-stepD*[*OF nfp*]  
**by** (*auto simp: rew-converse-outwards*)  
**then have** *NFP-redp  $\mathcal{F} \mathcal{R} s t$*  **unfolding** *NFP-redp-def*  
**by** (*simp add:  $\langle ground t \rangle$  ground-NF-srstep-gsrstep gsrsteps-eq-to-srsteps-eq*)  
**qed**

**then show** *?thesis* **by** (intro NFP-rrstep-intro) simp  
**qed**

**lemma** *GTRS-NE-aux*:

**assumes**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{R}$   
**and** *gtrs*: ground-sys  $\mathcal{R}$  ground-sys  $\mathcal{S}$   
**and** *ne*: *NE* (gsrstep  $\mathcal{F} \ \mathcal{R}$ ) (gsrstep  $\mathcal{F} \ \mathcal{S}$ )  
**shows** *NE-redp*  $\mathcal{F} \ \mathcal{R} \ \mathcal{S} \ s \ t$

**proof** –

**from** *assms(1)* **have** *gr*: ground *s* ground *t*  
**using** *srsteps-with-root-step-ground*[*OF gtrs(1)*] **by** *simp-all*  
**have**  $(s, t) \in (\text{gsrstep } \mathcal{F} \ \mathcal{R})^+$  **using** *gr assms(1)*  
**by** (*auto dest: srsteps-with-root-step-srstepsD intro: ground-srsteps-gsrsteps*)  
**then have**  $t \in \text{NF } (\text{srstep } \mathcal{F} \ \mathcal{R}) \implies (s, t) \in (\text{gsrstep } \mathcal{F} \ \mathcal{S})^*$   
**using** *gr ne unfolding NE-on-def*  
**by** (*auto simp: normalizability-def ground-subst-apply dest!: trancl-into-rtrancl*)

*blast*

**then show** *NE-redp*  $\mathcal{F} \ \mathcal{R} \ \mathcal{S} \ s \ t$  **unfolding** *NE-redp-def*  
**by** (*simp add: gsrsteps-eq-to-srsteps-eq*)

**qed**

**lemma** *GTRS-NE*:

**assumes** *gtrs*: ground-sys  $\mathcal{R}$  ground-sys  $\mathcal{S}$   
**and** *ne*: *NE* (gsrstep  $\mathcal{F} \ \mathcal{R}$ ) (gsrstep  $\mathcal{F} \ \mathcal{S}$ )  
**shows** *NE* (srstep  $\mathcal{F} \ \mathcal{R}$ ) (srstep  $\mathcal{F} \ \mathcal{S}$ )

**proof** –

{**fix** *s t* **assume**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{R}$   
**from** *GTRS-NE-aux*[*OF this gtrs ne*] **have** *NE-redp*  $\mathcal{F} \ \mathcal{R} \ \mathcal{S} \ s \ t$   
**by** *simp*}

**moreover**

{**fix** *s t* **assume**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{S}$   
**from** *GTRS-NE-aux*[*OF this gtrs(2, 1) NE-symmetric*[*OF ne*]]  
**have** *NE-redp*  $\mathcal{F} \ \mathcal{S} \ \mathcal{R} \ s \ t$  **by** *simp*}

**ultimately show** *?thesis*

**using** *ground-sys-nf-eq-lift*[*OF gtrs NE-NF-eq*[*OF ne*]]  
**by** (*intro NE-rrstep-intro*) *auto*

**qed**

**lemma** *gtrs-CE-aux*:

**assumes** *step*:  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ (\mathcal{R}^{\leftrightarrow})$   
**and** *gtrs*: ground-sys  $\mathcal{R}$  ground-sys  $\mathcal{S}$   
**and** *ce*: *CE* (gsrstep  $\mathcal{F} \ \mathcal{R}$ ) (gsrstep  $\mathcal{F} \ \mathcal{S}$ )  
**shows**  $(s, t) \in (\text{srstep } \mathcal{F} \ \mathcal{S})^{\leftrightarrow*}$

**proof** –

**from** *step gtrs(1)* **have** ground *s* ground *t*  
**by** (*metis ground-sys-symcl srsteps-with-root-step-ground*)  
**then have**  $(s, t) \in (\text{gsrstep } \mathcal{F} \ \mathcal{R})^{\leftrightarrow*}$  **using** *step*

by (*simp add: ground-srsteps-eq-gsrsteps-eq gsrstep-conversion-dist srsteps-with-root-step-srsteps-eqD*)  
 then have  $(s, t) \in (\text{gsrstep } \mathcal{F} \mathcal{S})^{\leftrightarrow*}$  using *ce unfolding CE-on-def*  
 by *blast*  
 then show  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{S})^{\leftrightarrow*}$   
 by (*simp add: gsrstep-conversion-dist gsrsteps-eq-to-srsteps-eq symcl-srsteps-conversion*)  
 qed

**lemma** *gtrs-CE*:

assumes *gtrs: ground-sys*  $\mathcal{R}$  *ground-sys*  $\mathcal{S}$   
 and *ce: CE* (*gsrstep*  $\mathcal{F} \mathcal{R}$ ) (*gsrstep*  $\mathcal{F} \mathcal{S}$ )  
 shows *CE* (*srstep*  $\mathcal{F} \mathcal{R}$ ) (*srstep*  $\mathcal{F} \mathcal{S}$ )

**proof** –

{**fix** *s t* **assume**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} (\mathcal{R}^{\leftrightarrow})$   
 from *gtrs-CE-aux*[*OF this gtrs ce*] **have**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{S})^{\leftrightarrow*}$  **by** *simp*}  
 moreover  
 {**fix** *s t* **assume**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} (\mathcal{S}^{\leftrightarrow})$   
 from *gtrs-CE-aux*[*OF this gtrs(2, 1) CE-symmetric* [*OF ce*]]  
**have**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^{\leftrightarrow*}$  **by** *simp*}  
 ultimately show *?thesis*  
 by (*intro CE-rrstep-intro*) *auto*

qed

end

## 8 Reducing Rewrite Properties to Properties on Ground Terms over Linear Variable-Separated Systems

**theory** *Ground-Reduction-on-LV*

**imports**

*Rewriting-Properties*

*Rewriting-LLRG-LV-Mondaic*

**begin**

**lemma** *lv-linear-sys*: *lv*  $\mathcal{R} \implies$  *linear-sys*  $\mathcal{R}$

by (*auto simp: lv-def*)

**lemma** *comp-rrstep-rel'-sig-mono*:

$\mathcal{F} \subseteq \mathcal{G} \implies \text{comp-rrstep-rel}' \mathcal{F} \mathcal{R} \mathcal{S} \subseteq \text{comp-rrstep-rel}' \mathcal{G} \mathcal{R} \mathcal{S}$

by (*meson Un-mono relcomp-mono srsteps-monp srsteps-with-root-step-sig-mono*)

**lemma** *srsteps-eqD*:  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^* \implies (s, t) \in (\text{rstep } \mathcal{R})^*$

by (*metis rtrancl-eq-or-trancl srstepsD*)

## 9 Linear-variable separated results

**locale** *open-terms-two-const-lv* =  
**fixes**  $\mathcal{R} :: ('f, 'v) \text{ term rel}$  **and**  $\mathcal{F} \ c \ d$   
**assumes**  $lv: lv \ \mathcal{R}$  **and**  $sig: \text{funas-rel } \mathcal{R} \subseteq \mathcal{F}$   
**and**  $fresh: (c, 0) \notin \mathcal{F} \ (d, 0) \notin \mathcal{F}$   
**and**  $diff: c \neq d$   
**begin**

**abbreviation**  $\mathcal{H} \equiv \text{insert } (c, 0) \ (\text{insert } (d, 0) \ \mathcal{F})$

**abbreviation**  $\sigma_c \equiv \text{const-subst } c$

**abbreviation**  $\sigma_d \equiv \text{const-subst } d$

**lemma** *sig-mono*:  $\mathcal{F} \subseteq \mathcal{H}$  **by** *auto*

**lemma** *fresh-sym-c*:  $(c, 0) \notin \text{funas-rel } \mathcal{R}$  **using** *sig fresh*

**by** (*auto simp: funas-rel-def*)

**lemma** *fresh-sym-d*:  $(d, 0) \notin \text{funas-rel } \mathcal{R}$  **using** *sig fresh*

**by** (*auto simp: funas-rel-def*)

**lemma** *fresh-sym-c-inv*:  $(c, 0) \notin \text{funas-rel } (\mathcal{R}^{-1})$  **using** *sig fresh*

**by** (*auto simp: funas-rel-def*)

**lemma** *fresh-sym-d-inv*:  $(d, 0) \notin \text{funas-rel } (\mathcal{R}^{-1})$  **using** *sig fresh*

**by** (*auto simp: funas-rel-def*)

**lemmas** *all-fresh* = *fresh-sym-c fresh-sym-d fresh-sym-c-inv fresh-sym-d-inv*

**lemma** *sig-inv*:  $\text{funas-rel } (\mathcal{R}^{-1}) \subseteq \mathcal{F}$  **using** *sig unfolding funas-rel-def by auto*

**lemma** *lv-inv*:  $lv \ (\mathcal{R}^{-1})$  **using** *lv unfolding lv-def by auto*

**lemma** *well-subst*:

$\bigwedge x. \text{funas-term } ((\text{const-subst } c) \ x) \subseteq \mathcal{H}$

$\bigwedge x. \text{funas-term } ((\text{const-subst } d) \ x) \subseteq \mathcal{H}$

**by** *auto*

**lemma** *srsteps-with-root-step-to-grsteps*:

**assumes**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{R}$

**shows**  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{gsrstep } \mathcal{H} \ \mathcal{R})^*$

**proof** –

**from** *assms* **have** *lift*:  $(s, t) \in \text{srsteps-with-root-step } \mathcal{H} \ \mathcal{R}$

**using** *srsteps-with-root-step-sig-mono[OF sig-mono]*

**by** *blast*

**note** [*dest!*] = *lv-srsteps-with-root-step-idep-subst[OF lv - well-subst, THEN srsteps-with-root-step-srsteps-eq]*

**have**  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{srstep } \mathcal{H} \ \mathcal{R})^*$  **using** *lift*

**using** *srsteps-eq-subst-closed[OF - well-subst(1)]*

**using** *srsteps-eq-subst-closed[OF - well-subst(2)]*

**by** (*auto dest: trancl-into-rtrancl*)

**then show** *?thesis*

**by** (*intro ground-srsteps-eq-gsrsteps-eq*) *auto*

**qed**

**lemma** *comp-rrstep-rel'-to-grsteps*:  
**assumes**  $(s, t) \in \text{comp-rrstep-rel}' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{R}$   
**shows**  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{gsrstep } \mathcal{H} (\mathcal{R}^{-1}))^* O (\text{gsrstep } \mathcal{H} \mathcal{R})^*$   
**proof** –  
**from** *assms* **have** *lift*:  $(s, t) \in \text{comp-rrstep-rel}' \mathcal{H} (\mathcal{R}^{-1}) \mathcal{R}$  **using** *sig-mono*  
**by** (*meson in-mono relcomp-mono srsteps-monp srsteps-with-root-step-sig-mono sup-mono*)  
**note** [*dest!*] = *lv-srsteps-with-root-step-idep-subst*[*OF lv-well-subst, THEN srsteps-with-root-step-srsteps-eqD*]  
**note** [*dest!*] = *lv-srsteps-with-root-step-idep-subst*[*OF lv-inv-well-subst, THEN srsteps-with-root-step-srsteps-eqD*]  
**have**  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{srstep } \mathcal{H} (\mathcal{R}^{-1}))^* O (\text{srstep } \mathcal{H} \mathcal{R})^*$  **using** *lift*  
**using** *srsteps-eq-subst-closed*[*OF-well-subst(1)*]  
**using** *srsteps-eq-subst-closed*[*OF-well-subst(2)*]  
**by** (*auto dest!: trancl-into-rtrancl*) *blast*  
**then show** *?thesis*  
**by** (*intro srsteps-eq-relcomp-gsrsteps-relcomp*) *auto*  
**qed**

**lemma** *gsrsteps-eq-to-srsteps*:  
**assumes**  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{gsrstep } \mathcal{H} \mathcal{R})^*$   
**and** *funas*: *funas-term*  $s \subseteq \mathcal{F}$  *funas-term*  $t \subseteq \mathcal{F}$   
**shows**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{R})^*$   
**proof** –  
**from** *funas* **have**  $d: (d, 0) \notin \text{funas-term } s$  **and**  $c: (c, 0) \notin \text{funas-term } (t \cdot \sigma_d)$   
**using** *fresh diff* **by** (*auto simp: funas-term-subst*)  
**have**  $(s, t) \in (\text{rstep } \mathcal{R})^*$  **using** *c gsrsteps-eq-to-rsteps-eq*[*OF assms(1)*]  
**using** *remove-const-subst-steps-eq-lhs*[*OF lv-linear-sys*[*OF lv*] *fresh-sym-c*,  
*THEN remove-const-subst-steps-eq-rhs*[*OF lv-linear-sys*[*OF lv*] *fresh-sym-d*  
*d*]]  
**by** *auto*  
**then show** *?thesis* **using** *funas sig* **by** *blast*  
**qed**

**lemma** *convert-NF-to-GNF*:  
*funas-term*  $t \subseteq \mathcal{F} \implies t \in \text{NF} (\text{srstep } \mathcal{F} \mathcal{R}) \implies t \cdot \sigma_c \in \text{NF} (\text{gsrstep } \mathcal{H} \mathcal{R})$   
*funas-term*  $t \subseteq \mathcal{F} \implies t \in \text{NF} (\text{srstep } \mathcal{F} \mathcal{R}) \implies t \cdot \sigma_d \in \text{NF} (\text{gsrstep } \mathcal{H} \mathcal{R})$   
**using** *NF-to-fresh-const-subst-NF*[*OF lv-linear-sys*[*OF lv*] *fresh-sym-c sig*]  
**using** *NF-to-fresh-const-subst-NF*[*OF lv-linear-sys*[*OF lv*] *fresh-sym-d sig*]  
**by** *blast+*

**lemma** *convert-GNF-to-NF*:  
*funas-term*  $t \subseteq \mathcal{F} \implies t \cdot \sigma_c \in \text{NF} (\text{gsrstep } \mathcal{H} \mathcal{R}) \implies t \in \text{NF} (\text{srstep } \mathcal{F} \mathcal{R})$   
*funas-term*  $t \subseteq \mathcal{F} \implies t \cdot \sigma_d \in \text{NF} (\text{gsrstep } \mathcal{H} \mathcal{R}) \implies t \in \text{NF} (\text{srstep } \mathcal{F} \mathcal{R})$   
**using** *fresh-const-subst-NF-pres*[*OF fresh-sym-c sig*]  
**using** *fresh-const-subst-NF-pres*[*OF fresh-sym-d sig*]  
**using** *sig-mono* **by** *blast+*

**lemma** *lv-CR*:

**assumes** *cr*:  $CR (gsrstep \mathcal{H} \mathcal{R})$

**shows**  $CR (srstep \mathcal{F} \mathcal{R})$

**proof** –

{**fix** *s t* **assume** *ass*:  $(s, t) \in (srstep \mathcal{F} (\mathcal{R}^{-1}))^+ O srsteps\text{-with-root-step} \mathcal{F} \mathcal{R}$

**from** *ass* **have** *funas*: *funas-term*  $s \subseteq \mathcal{F}$  *funas-term*  $t \subseteq \mathcal{F}$

**by** (*metis Pair-inject ass relcompE srstepsD srsteps-with-root-step-srstepsD*)+

**have**  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (gsrstep \mathcal{H} (\mathcal{R}^{-1}))^* O (gsrstep \mathcal{H} \mathcal{R})^*$

**using** *ass comp-rrstep-rel'-to-grsteps* **by** *auto*

**then have**  $s: (s \cdot \sigma_c, t \cdot \sigma_d) \in (gsrstep \mathcal{H} \mathcal{R})^* O (gsrstep \mathcal{H} (\mathcal{R}^{-1}))^*$

**using** *cr unfolding CR-on-def*

**by** (*auto simp: join-def rew-converse-outwards*)

**from** *gsrsteps-eq-relcomp-to-rsteps-relcomp*[*OF this*]

**have** *commute-redp*  $\mathcal{F} \mathcal{R} \mathcal{R} s t$  **unfolding** *commute-redp-def* **using** *funas fresh*

**using** *remove-const-subst-relcomp*[*OF lv-linear-sys*[*OF lv*]] *lv-linear-sys*[*OF*

*lv-inv*]

*all-fresh diff, THEN rsteps-eq-relcomp-srsteps-eq-relcompI*[*OF sig sig-inv*

*funas*]

**by** (*metis srstep-converse-dist subsetD*)}

**then show** *?thesis* **by** (*intro CR-rrstep-intro simp*

**qed**

**lemma** *lv-WCR*:

**assumes** *wcr*:  $WCR (gsrstep \mathcal{H} \mathcal{R})$

**shows**  $WCR (srstep \mathcal{F} \mathcal{R})$

**proof** –

**note** *sig-trans-root* = *subsetD*[*OF srrstep-monp*[*OF sig-mono*]]

**note** *sig-trans* = *subsetD*[*OF srstep-monp*[*OF sig-mono*]]

{**fix** *s t u* **assume** *ass*:  $(s, t) \in sig\text{-step} \mathcal{F} (rrstep \mathcal{R})$   $(s, u) \in srstep \mathcal{F} \mathcal{R}$

**from** *ass* **have** *funas*: *funas-term*  $s \subseteq \mathcal{F}$  *funas-term*  $t \subseteq \mathcal{F}$  *funas-term*  $u \subseteq \mathcal{F}$

**by** *blast+*

**then have** *free*:  $(d, 0) \notin funas\text{-term } u$   $(c, 0) \notin funas\text{-term } t$  **using** *fresh* **by**

*blast+*

**have** *\**: *ground*  $(s \cdot \sigma_c)$  *ground*  $(t \cdot \sigma_d)$  *ground*  $(u \cdot \sigma_c)$  **by** *auto*

**moreover have**  $(s \cdot \sigma_c, t \cdot \sigma_d) \in srstep \mathcal{H} \mathcal{R}$  **using** *lv sig-trans-root*[*OF ass(1)*]

**by** (*intro lv-root-step-idep-subst*[*THEN srrstep-to-srstep*]) *auto*

**moreover have**  $(s \cdot \sigma_c, u \cdot \sigma_c) \in srstep \mathcal{H} \mathcal{R}$

**using** *srstep-subst-closed*[*OF sig-trans*[*OF ass(2)*], *of*  $\sigma_c$ ]

**by** *auto*

**ultimately have**  $w: (t \cdot \sigma_d, u \cdot \sigma_c) \in (gsrstep \mathcal{H} \mathcal{R})^\downarrow$  **using** *wcr unfolding*

*WCR-on-def*

**by** *auto* (*metis* (*no-types, lifting*) *\**)

**note** *join-unfolded* = *w*[*unfolded join-def rew-converse-inwards*]

**have**  $(t, u) \in (srstep \mathcal{F} \mathcal{R})^\downarrow$  **unfolding** *join-def*

**using** *remove-const-subst-relcomp*[*OF lv-linear-sys*[*OF lv*]] *lv-linear-sys*[*OF*

*lv-inv*]

*fresh-sym-d fresh-sym-c fresh-sym-d-inv fresh-sym-c-inv diff*[*symmetric*]

*free*

*gsrsteps-eq-relcomp-to-rsteps-relcomp*[*OF join-unfolded*],



THEN  $rsteps\text{-}eq\text{-}relcomp\text{-}srsteps\text{-}eq\text{-}relcompI[OF\ sig\ sig\text{-}inv\ funas(2-)]$   
 by  $(metis\ (no\text{-}types,\ lifting)\ srstep\text{-}converse\text{-}dist)$   
 then show  $?thesis$  by  $(intro\ WCR\text{-}rrstep\text{-}intro)\ simp$   
 qed

lemma  $lv\text{-}NFP$ :

assumes  $nfp: NFP\ (gsrstep\ \mathcal{H}\ \mathcal{R})$

shows  $NFP\ (srstep\ \mathcal{F}\ \mathcal{R})$

proof –

{fix  $s\ t$  assume  $ass: (s, t) \in comp\text{-}rrstep\text{-}rel'\ \mathcal{F}\ (\mathcal{R}^{-1})\ \mathcal{R}$

from  $ass$  have  $funas: funas\text{-}term\ s \subseteq \mathcal{F}\ funas\text{-}term\ t \subseteq \mathcal{F}$

by  $(metis\ Un\text{-}iff\ ass\ relcompEpair\ srstepsD\ srsteps\text{-}with\text{-}root\text{-}step\text{-}srstepsD)+$

from  $comp\text{-}rrstep\text{-}rel'\text{-}to\text{-}grsteps[OF\ ass]$

have  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (gsrstep\ \mathcal{H}\ (\mathcal{R}^{-1}))^* O\ (gsrstep\ \mathcal{H}\ \mathcal{R})^*$  by  $simp$

then have  $t \cdot \sigma_d \in NF\ (gsrstep\ \mathcal{H}\ \mathcal{R}) \implies (s \cdot \sigma_c, t \cdot \sigma_d) \in (gsrstep\ \mathcal{H}\ \mathcal{R})^*$

using  $NFP\text{-}stepD[OF\ nfp]$

by  $(auto\ simp: rew\text{-}converse\text{-}outwards)$

from  $gsrsteps\text{-}eq\text{-}to\text{-}srsteps[OF\ this\ funas]$

have  $NFP\text{-}redp\ \mathcal{F}\ \mathcal{R}\ s\ t$  unfolding  $NFP\text{-}redp\text{-}def$

using  $convert\text{-}NF\text{-}to\text{-}GNF(2)[OF\ funas(2)]$

by  $simp$ }

then show  $?thesis$  by  $(intro\ NFP\text{-}rrstep\text{-}intro)\ simp$

qed

lemma  $lv\text{-}UNF$ :

assumes  $unf: UNF\ (gsrstep\ \mathcal{H}\ \mathcal{R})$

shows  $UNF\ (srstep\ \mathcal{F}\ \mathcal{R})$

proof –

{fix  $s\ t$  assume  $ass: (s, t) \in comp\text{-}rrstep\text{-}rel'\ \mathcal{F}\ (\mathcal{R}^{-1})\ \mathcal{R}$

from  $ass$  have  $funas: funas\text{-}term\ s \subseteq \mathcal{F}\ funas\text{-}term\ t \subseteq \mathcal{F}$

by  $(metis\ Un\text{-}iff\ ass\ relcompEpair\ srstepsD\ srsteps\text{-}with\text{-}root\text{-}step\text{-}srstepsD)+$

from  $comp\text{-}rrstep\text{-}rel'\text{-}to\text{-}grsteps[OF\ ass]$

have  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (gsrstep\ \mathcal{H}\ (\mathcal{R}^{-1}))^* O\ (gsrstep\ \mathcal{H}\ \mathcal{R})^*$  by  $simp$

then have  $s \cdot \sigma_c \in NF\ (gsrstep\ \mathcal{H}\ \mathcal{R}) \implies t \cdot \sigma_d \in NF\ (gsrstep\ \mathcal{H}\ \mathcal{R}) \implies s \cdot$

$\sigma_c = t \cdot \sigma_d$

using  $unf$  unfolding  $UNF\text{-}on\text{-}def$

by  $(auto\ simp: normalizability\text{-}def\ rew\text{-}converse\text{-}outwards)$

then have  $UN\text{-}redp\ \mathcal{F}\ \mathcal{R}\ s\ t$  unfolding  $UN\text{-}redp\text{-}def$

using  $convert\text{-}NF\text{-}to\text{-}GNF(1)[OF\ funas(1)]$

using  $convert\text{-}NF\text{-}to\text{-}GNF(2)[OF\ funas(2)]$

by  $(metis\ NF\text{-}not\text{-}suc\ funas\ gsrsteps\text{-}eq\text{-}to\text{-}srsteps\ rtrancl\text{-}eq\text{-}or\text{-}trancl)$ }

then show  $?thesis$  by  $(intro\ UNF\text{-}rrstep\text{-}intro)\ simp$

qed

lemma  $lv\text{-}UNC$ :

assumes  $unc: UNC\ (gsrstep\ \mathcal{H}\ \mathcal{R})$

shows  $UNC\ (srstep\ \mathcal{F}\ \mathcal{R})$

proof –

have  $lv\text{-}conv: lv\ (\mathcal{R}^{\leftrightarrow})$  using  $lv$  by  $(auto\ simp: lv\text{-}def)$

**{fix s t assume** *ass*:  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} (\mathcal{R}^{\leftrightarrow})$   
**from** *ass* **have** *funas*: *funas-term*  $s \subseteq \mathcal{F}$  *funas-term*  $t \subseteq \mathcal{F}$   
**by** (*metis Un-iff ass relcompEpair srstepsD srsteps-with-root-step-srstepsD*)  
**have**  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{gsrstep } \mathcal{H} \mathcal{R})^{\leftrightarrow*}$   
**using** *lv-srsteps-with-root-step-idep-subst*[*OF lv-conv srsteps-with-root-step-sig-mono*][*OF sig-mono*, *THEN subsetD*, *OF ass*]  
*well-subst*,*THEN srsteps-with-root-step-srsteps-eqD*  
**unfolding** *conversion-def Restr-smycl-dist srstep-symcl-dist*  
**by** (*intro ground-srsteps-eq-gsrsteps-eq*) *auto*  
**then have**  $s \cdot \sigma_c \in \text{NF} (\text{gsrstep } \mathcal{H} \mathcal{R}) \implies t \cdot \sigma_d \in \text{NF} (\text{gsrstep } \mathcal{H} \mathcal{R}) \implies s \cdot \sigma_c = t \cdot \sigma_d$   
**using** *unc unfolding UNC-def*  
**by** (*auto simp: normalizability-def sig-step-converse-rstep rtrancl-converse Restr-converse*)  
**then have** *UN-redp*  $\mathcal{F} \mathcal{R} s t$  **unfolding** *UN-redp-def*  
**using** *convert-NF-to-GNF(1)*[*OF funas(1)*]  
**using** *convert-NF-to-GNF(2)*[*OF funas(2)*]  
**by** (*metis NF-not-suc funas gsrsteps-eq-to-srsteps rtrancl-eq-or-trancl*)  
**then show** *?thesis* **by** (*intro UNC-rrstep-intro simp*)  
**qed**

**lemma** *lv-SCR*:

**assumes** *scr*: *SCR* (*gsrstep*  $\mathcal{H} \mathcal{R}$ )

**shows** *SCR* (*srstep*  $\mathcal{F} \mathcal{R}$ )

**proof** –

**note** *sig-trans-root* = *subsetD*[*OF srrstep-monp*][*OF sig-mono*]

**note** *sig-trans* = *subsetD*[*OF srstep-monp*][*OF sig-mono*]

**note** *cl-on-c* = *lin-fresh-rstep-const-replace-closed*[*OF lv-linear-sys*][*OF lv*] *fresh-sym-c*

*lin-fresh-rstep-const-replace-closed*[*OF lv-linear-sys*][*OF lv-inv*] *fresh-sym-c-inv*

**note** *cl-on-d* = *lin-fresh-rstep-const-replace-closed*[*OF lv-linear-sys*][*OF lv*] *fresh-sym-d*

*lin-fresh-rstep-const-replace-closed*[*OF lv-linear-sys*][*OF lv-inv*] *fresh-sym-d-inv*

**{fix s t u assume** *ass*:  $(s, t) \in \text{srstep } \mathcal{F} \mathcal{R}$   $(s, u) \in \text{srstep } \mathcal{F} \mathcal{R}$

**and** *root*:  $(s, t) \in \text{sig-step } \mathcal{F} (\text{rrstep } \mathcal{R}) \vee (s, u) \in \text{sig-step } \mathcal{F} (\text{rrstep } \mathcal{R})$

**from** *ass* **have** *funas*: *funas-term*  $s \subseteq \mathcal{F}$  *funas-term*  $t \subseteq \mathcal{F}$  *funas-term*  $u \subseteq \mathcal{F}$

**by** (*force dest: sig-stepE*)**+**

**then have** *fresh-c-t*:  $(c, 0) \notin \text{funas-term } (t \cdot \sigma_d)$  **and** *fresh-d-u*:  $(d, 0) \notin \text{funas-term } u$  **using** *fresh diff*

**by** (*auto simp: funas-term-subst*)

**have**  $*$ :  $(s, t) \in \text{sig-step } \mathcal{F} (\text{rrstep } \mathcal{R}) \implies (s \cdot \sigma_c, t \cdot \sigma_d) \in \text{gsrstep } \mathcal{H} \mathcal{R} \wedge (s \cdot \sigma_c, u \cdot \sigma_c) \in \text{gsrstep } \mathcal{H} \mathcal{R}$

$(s, u) \in \text{sig-step } \mathcal{F} (\text{rrstep } \mathcal{R}) \implies (s \cdot \sigma_d, t \cdot \sigma_d) \in \text{gsrstep } \mathcal{H} \mathcal{R} \wedge (s \cdot \sigma_d, u \cdot \sigma_c) \in \text{gsrstep } \mathcal{H} \mathcal{R}$

**using** *srstep-subst-closed*[*OF sig-trans*][*OF ass(2)*] *well-subst(1)*

**using** *srstep-subst-closed*[*OF sig-trans*][*OF ass(2)*] *well-subst(2)*

**using** *lv-root-step-idep-subst*[*OF lv*] *sig-trans-root*[*of* ( $s, t$ )  $\mathcal{R}$ ] *sig-trans-root*[*of* ( $s, u$ )  $\mathcal{R}$ ]

**by** (*simp-all add: ass(1) sig-trans srstep-subst-closed srrstep-to-srstep*)

**from** *this scr* **have**  $(u \cdot \sigma_c, t \cdot \sigma_d) \in (\text{gsrstep } \mathcal{H} \mathcal{R})^* O ((\text{gsrstep } \mathcal{H} \mathcal{R})^=)^{-1}$

```

    using root unfolding SCR-on-def by (meson UNIV-I converse-iff rel-
comp.simps)
  then have  $v: (u \cdot \sigma_c, t \cdot \sigma_d) \in (srstep \mathcal{H} \mathcal{R})^* O ((srstep \mathcal{H} \mathcal{R})^{-1})^{-1}$ 
    by (auto dest: gsrsteps-eq-to-srsteps-eq)
  have [simp]:  $funas-term v \subseteq \mathcal{F} \implies term-to-sig \mathcal{F} x v = v$  for  $x v$ 
    by (simp add: subset-insertI2)
  have  $(u, t) \in (srstep \mathcal{F} \mathcal{R})^* O ((srstep \mathcal{F} \mathcal{R})^{-1})^{-1}$ 
  proof -
    from  $v$  have  $(u, t \cdot \sigma_d) \in (rstep \mathcal{R})^* O (rstep (\mathcal{R}^{-1}))^{-1}$ 
    using const-replace-closed-relcomp[THEN const-replace-closed-remove-subst-lhs,
OF
      const-replace-closed-rtrancl[OF cl-on-c(1)]
      const-replace-closed-symcl[OF cl-on-c(2)] fresh-c-t, of  $u$ ]
    by (auto simp: relcomp.relcompI srstepD simp flip: rstep-converse-dist dest:
srsteps-eqD)
    then have  $(u, t) \in (rstep \mathcal{R})^* O (rstep (\mathcal{R}^{-1}))^{-1}$ 
    using const-replace-closed-relcomp[THEN const-replace-closed-remove-subst-lhs,
OF
      const-replace-closed-symcl[OF cl-on-d(1)]
      const-replace-closed-rtrancl[OF cl-on-d(2)] fresh-d-u, of  $t$ ]
    using converse-relcomp[where  $?s = (rstep (\mathcal{R}^{-1}))^{-1}$  and  $?r = (rstep \mathcal{R})^*$ ]
    by (metis (no-types, lifting) converseD converse-Id converse-Un con-
verse-converse rstep-converse-dist rtrancl-converse)
    then obtain  $v$  where  $(u, v) \in (rstep \mathcal{R})^* (v, t) \in (rstep (\mathcal{R}^{-1}))^{-1}$  by auto
    then have  $(u, term-to-sig \mathcal{F} x v) \in (srstep \mathcal{F} \mathcal{R})^* (term-to-sig \mathcal{F} x v, t) \in$ 
 $(srstep \mathcal{F} (\mathcal{R}^{-1}))^{-1}$ 
      using funas(2, 3) sig fresh
    by (auto simp: rtrancl-eq-or-trancl rstep-trancl-sig-step-r subset-insertI2
rew-converse-outwards dest: fuans-term-term-to-sig[THEN subsetD]
intro!: rstep-term-to-sig-r rstep-srstepI rsteps-eq-srsteps-eqI)
    then show ?thesis
    by (metis converse-Id converse-Un relcomp.relcompI srstep-converse-dist)
  qed
  then have SCRp  $\mathcal{F} \mathcal{R} t u$ 
    by auto}
  then show ?thesis by (intro SCR-rrstep-intro) (metis srrstep-to-srstep)+
qed

end

locale open-terms-two-const-lv-two-sys =
  open-terms-two-const-lv  $\mathcal{R}$ 
  for  $\mathcal{R} :: ('f, 'v) term rel +$ 
  fixes  $S :: ('f, 'v) term rel$ 
  assumes  $lv-S: lv S$  and  $sig-S: funas-rel S \subseteq \mathcal{F}$ 
begin

```

**lemma** *fresh-sym-c-S*:  $(c, 0) \notin \text{funas-rel } \mathcal{S}$  **using** *sig-S fresh*  
**by** (*auto simp: funas-rel-def*)

**lemma** *fresh-sym-d-S*:  $(d, 0) \notin \text{funas-rel } \mathcal{S}$  **using** *sig-S fresh*  
**by** (*auto simp: funas-rel-def*)

**lemma** *lv-commute*:

**assumes** *com*: *commute* (*gsrstep*  $\mathcal{H}$   $\mathcal{R}$ ) (*gsrstep*  $\mathcal{H}$   $\mathcal{S}$ )

**shows** *commute* (*srstep*  $\mathcal{F}$   $\mathcal{R}$ ) (*srstep*  $\mathcal{F}$   $\mathcal{S}$ )

**proof** –

**have** *linear*: *linear-sys*  $\mathcal{S}$  *linear-sys*  $(\mathcal{R}^{-1})$  **using** *lv lv-S unfolding lv-def* **by**  
*auto*

**{fix** *s t* **assume** *ass*:  $(s, t) \in \text{comp-rrstep-rel}' \mathcal{F} (\mathcal{R}^{-1}) \mathcal{S}$

**from** *ass* **have** *funas*: *funas-term*  $s \subseteq \mathcal{F}$  *funas-term*  $t \subseteq \mathcal{F}$

**by** (*metis Un-iff ass relcompEpair srstepsD srsteps-with-root-step-srstepsD*)**+**

**have**  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{gsrstep } \mathcal{H} (\mathcal{R}^{-1}))^* O (\text{gsrstep } \mathcal{H} \mathcal{S})^*$

**using** *comp-rrstep-rel'-sig-mono*[*OF sig-mono, THEN subsetD, OF ass*]

**using** *srsteps-eq-subst-closed*[*OF - well-subst(1)*] *srsteps-eq-subst-closed*[*OF -*  
*well-subst(2)*]

**by** (*auto simp: rew-converse-inwards dest!: trancl-into-rtrancl*

*lv-srsteps-with-root-step-idep-subst*[*OF lv-inv - well-subst, THEN srsteps-with-root-step-srsteps-eqD*]

*lv-srsteps-with-root-step-idep-subst*[*OF lv-S - well-subst, THEN srsteps-with-root-step-srsteps-eqD*]

*intro!*: *srsteps-eq-relcomp-gsrsteps-relcomp*) **blast**

**then** **have**  $w: (s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{gsrstep } \mathcal{H} \mathcal{S})^* O (\text{gsrstep } \mathcal{H} (\mathcal{R}^{-1}))^*$  **using**

*com*

**unfolding** *commute-def Restr-converse srstep-converse-dist*

**by** *blast*

**have**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{S})^* O (\text{srstep } \mathcal{F} (\mathcal{R}^{-1}))^*$  **using** *funas sig-S sig-inv*

*fresh*

**using** *remove-const-subst-relcomp*[*OF linear fresh-sym-c-S fresh-sym-d-S fresh-sym-c-inv*  
*fresh-sym-d-inv diff - -*

*gsrsteps-eq-relcomp-to-rsteps-relcomp*[*OF w*]

**by** (*intro rsteps-eq-relcomp-srsteps-eq-relcompI*) *auto*

**then** **have** *commute-redp*  $\mathcal{F}$   $\mathcal{R}$   $\mathcal{S}$  *s t* **unfolding** *commute-redp-def*

**by** (*simp add: rew-converse-inwards*)**}**

**then** **show** *?thesis* **by** (*intro commute-rrstep-intro simp*)

**qed**

**lemma** *lv-NE*:

**assumes** *ne*: *NE* (*gsrstep*  $\mathcal{H}$   $\mathcal{R}$ ) (*gsrstep*  $\mathcal{H}$   $\mathcal{S}$ )

**shows** *NE* (*srstep*  $\mathcal{F}$   $\mathcal{R}$ ) (*srstep*  $\mathcal{F}$   $\mathcal{S}$ )

**proof** –

**from** *NE-NF-eq*[*OF ne*] **have** *ne-eq*: *NF* (*srstep*  $\mathcal{F}$   $\mathcal{R}$ ) = *NF* (*srstep*  $\mathcal{F}$   $\mathcal{S}$ )

**using** *lv lv-S sig sig-S fresh-sym-c fresh-sym-c-S*

**by** (*intro linear-sys-gNF-eq-NF-eq*) (*auto dest: lv-linear-sys*)

**{fix** *s t* **assume** *step*:  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R}$

**then** **have** *funas*: *funas-term*  $s \subseteq \mathcal{F}$  *funas-term*  $t \subseteq \mathcal{F}$

**by** (*metis Un-iff step relcompEpair srstepsD srsteps-with-root-step-srstepsD*)**+**

**then have** *fresh*:  $(c, 0) \notin \text{funas-term } t$   $(d, 0) \notin \text{funas-term } s$  **using** *fresh* **by**  
*auto*  
**from** *srsteps-with-root-step-sig-mono*[*OF sig-mono*, *THEN subsetD*, *OF step*]  
**have**  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{gsrstep } \mathcal{H} \mathcal{R})^*$   
**using** *lv-srsteps-with-root-step-idep-subst*[*OF lv - well-subst*, *THEN srsteps-with-root-step-srsteps-eqD*]  
**by** (*intro ground-srsteps-eq-gsrsteps-eq*) *auto*  
**then have**  $t \cdot \sigma_d \in \text{NF} (\text{gsrstep } \mathcal{H} \mathcal{S}) \implies (s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{gsrstep } \mathcal{H} \mathcal{S})^*$   
**using** *ne NE-NF-eq*[*OF ne*] **unfolding** *NE-on-def*  
**by** (*auto simp: normalizability-def*)  
**from** *this*[*THEN gsrsteps-eq-to-rsteps-eq*, *THEN remove-const-subst-steps*[*OF*  
*lv-linear-sys*[*OF lv-S*] *fresh-sym-c-S* *fresh-sym-d-S* *diff fresh*]]  
**have** *NE-redp*  $\mathcal{F} \mathcal{R} \mathcal{S} s t$  **using** *NF-to-fresh-const-subst-NF*[*OF lv-linear-sys*[*OF*  
*lv-S*] *fresh-sym-d-S* *sig-S* *funas(2)*]  
**using** *funas sig-S* **unfolding** *NE-redp-def ne-eq*  
**by** (*auto intro: rsteps-eq-srsteps-eqI*)}  
**moreover**  
**{fix** *s t* **assume** *step*:  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{S}$   
**then have** *funas*: *funas-term*  $s \subseteq \mathcal{F}$  *funas-term*  $t \subseteq \mathcal{F}$   
**by** (*metis sig-S sig-stepE sig-step-rsteps-dist srsteps-with-root-step-srstepsD*) +  
**then have** *fresh*:  $(c, 0) \notin \text{funas-term } t$   $(d, 0) \notin \text{funas-term } s$  **using** *fresh* **by**  
*auto*  
**from** *srsteps-with-root-step-sig-mono*[*OF sig-mono*, *THEN subsetD*, *OF step*]  
**have**  $(s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{gsrstep } \mathcal{H} \mathcal{S})^*$   
**using** *lv-srsteps-with-root-step-idep-subst*[*OF lv-S - well-subst*, *THEN srsteps-with-root-step-srsteps-eqD*]  
**by** (*intro ground-srsteps-eq-gsrsteps-eq*) *auto*  
**then have**  $t \cdot \sigma_d \in \text{NF} (\text{gsrstep } \mathcal{H} \mathcal{R}) \implies (s \cdot \sigma_c, t \cdot \sigma_d) \in (\text{gsrstep } \mathcal{H} \mathcal{R})^*$   
**using** *ne NE-NF-eq*[*OF ne*] **unfolding** *NE-on-def*  
**by** (*auto simp: normalizability-def*)  
**from** *this*[*THEN gsrsteps-eq-to-rsteps-eq*, *THEN remove-const-subst-steps*[*OF*  
*lv-linear-sys*[*OF lv*] *fresh-sym-c* *fresh-sym-d* *diff fresh*]]  
**have** *NE-redp*  $\mathcal{F} \mathcal{S} \mathcal{R} s t$  **using** *NF-to-fresh-const-subst-NF*[*OF lv-linear-sys*[*OF*  
*lv*] *fresh-sym-d* *sig* *funas(2)*, *of*  $\mathcal{H}$ ]  
**using** *funas sig* **unfolding** *NE-redp-def ne-eq*  
**by** (*auto intro: rsteps-eq-srsteps-eqI*)}  
**ultimately show** *?thesis* **using** *ne-eq*  
**by** (*intro NE-rrstep-intro*) *auto*  
**qed**  
**end**

— CE is special as it only needs one additional constant therefore not included in the locale

**lemma** *lv-CE-aux*:

**assumes**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} (\mathcal{R}^{\leftrightarrow})$   
**and** *sig*: *funas-rel*  $\mathcal{R} \subseteq \mathcal{F}$  *funas-rel*  $\mathcal{S} \subseteq \mathcal{F}$   
**and** *fresh*:  $(c, 0) \notin \mathcal{F}$  **and** *const*:  $(a, 0) \in \mathcal{F}$   
**and** *lv*: *lv*  $\mathcal{R}$  *lv*  $\mathcal{S}$   
**and** *ce*: *CE*  $(\text{gsrstep } (\text{insert } (c, 0) \mathcal{F}) \mathcal{R}) (\text{gsrstep } (\text{insert } (c, 0) \mathcal{F}) \mathcal{S})$   
**shows**  $(s, t) \in (\text{srstep } \mathcal{F} \mathcal{S})^{\leftrightarrow*}$

**proof** –

**let**  $\mathcal{H} = \text{insert } (c, 0) \mathcal{F}$  **have**  $\text{mono}: \mathcal{F} \subseteq \mathcal{H}$  **by** *auto*

**have**  $\text{lv-conv}: \text{lv } (\mathcal{R}^{\leftrightarrow}) \text{ lv } (\mathcal{S}^{\leftrightarrow})$  **using**  $\text{lv}$  **by** *(auto simp: lv-def)*

**have**  $\text{sig-conv}: \text{funas-rel } (\mathcal{S}^{\leftrightarrow}) \subseteq \mathcal{F}$  **using**  $\text{sig}(2)$  **by** *(auto simp: funas-rel-def)*

**from** *fresh* **have**  $\text{fresh-sys}: (c, 0) \notin \text{funas-rel } \mathcal{R} \ (c, 0) \notin \text{funas-rel } \mathcal{S} \ (c, 0) \notin \text{funas-rel } (\mathcal{S}^{\leftrightarrow})$

**using**  $\text{sig}$  **by** *(auto simp: funas-rel-def)*

**from**  $\text{assms}(1)$  **have**  $\text{lift}: (s, t) \in \text{srsteps-with-root-step } \mathcal{H} \ (\mathcal{R}^{\leftrightarrow})$

**unfolding**  $\text{srsteps-with-root-step-def}$

**by** *(meson in-mono mono relcomp-mono rtrancl-mono srstep-monp srstep-monp)*

**from**  $\text{assms}(1)$  **have**  $\text{funas}: \text{funas-term } s \subseteq \mathcal{F} \ \text{funas-term } t \subseteq \mathcal{F}$

**by** *(meson srstepsD srsteps-with-root-step-srstepsD)+*

**from**  $\text{srsteps-with-root-step-srsteps-eqD}[OF \ \text{assms}(1), \ \text{THEN } \text{subsetD}[OF \ \text{srsteps-eq-monp}[OF \ \text{mono}]]]$

**have**  $(s \cdot \text{const-subst } c, t \cdot \text{const-subst } c) \in (\text{srstep } \mathcal{H} \ \mathcal{R})^{\leftrightarrow*}$

**by** *(auto simp: sig-step-conversion-dist intro: srsteps-eq-subst-closed)*

**moreover** **have**  $(s \cdot \text{const-subst } c, t \cdot \text{const-subst } a) \in (\text{srstep } \mathcal{H} \ \mathcal{R})^{\leftrightarrow*}$

**unfolding**  $\text{sig-step-conversion-dist}$  **using**  $\text{const}$

**by** *(intro lv-srsteps-with-root-step-idep-subst[OF lv-conv(1) lift, THEN srsteps-with-root-step-srsteps-eqD])*

*auto*

**moreover** **have**  $\text{ground } (s \cdot \text{const-subst } c) \ \text{ground } (t \cdot \text{const-subst } a) \ \text{ground } (t \cdot \text{const-subst } a)$

**by** *auto*

**ultimately** **have**  $\text{toS}: (s \cdot \text{const-subst } c, t \cdot \text{const-subst } c) \in (\text{gsrstep } \mathcal{H} \ \mathcal{S})^{\leftrightarrow*}$

$(s \cdot \text{const-subst } c, t \cdot \text{const-subst } a) \in (\text{gsrstep } \mathcal{H} \ \mathcal{S})^{\leftrightarrow*}$

**using**  $\text{ground-srsteps-eq-gsrsteps-eq}[\text{where } \mathcal{F} = \mathcal{H} \ \text{and } \mathcal{R} = \mathcal{R}^{\leftrightarrow}]$

**using**  $\text{ce}$  **unfolding**  $\text{CE-on-def}$

**by** *(auto simp: Restr-smycl-dist conversion-def srstep-symcl-dist)*

**then** **have**  $*$ :  $(t \cdot \text{const-subst } a, t \cdot \text{const-subst } c) \in (\text{gsrstep } \mathcal{H} \ \mathcal{S})^{\leftrightarrow*}$

**by** *(metis (no-types, lifting) conversion-inv conversion-rtrancl rtrancl.rtrancl-into-rtrancl)*

**have**  $(t \cdot \text{const-subst } a, t) \in (\text{srstep } \mathcal{F} \ \mathcal{S})^{\leftrightarrow*}$  **using**  $\text{const}$

**using**  $\text{funas}(2)$  *fresh*

**using**  $\text{remove-const-subst-steps-eq-rhs}[OF \ \text{lv-linear-sys}[OF \ \text{lv-conv}(2)] \ \text{fresh-sys}(3)]$

–

$\text{gsrsteps-eq-to-rsteps-eq}[OF \ *[\text{unfolded } \text{gsrstep-conversion-dist}]]]$

**by** *(cases vars-term t = {})*

*(auto simp: funas-term-subst sig-step-conversion-dist split: if-splits intro!:*

$\text{rsteps-eq-srsteps-eqI}[OF \ \text{sig-conv}]$

**moreover** **have**  $(s, t \cdot \text{const-subst } a) \in (\text{srstep } \mathcal{F} \ \mathcal{S})^{\leftrightarrow*}$  **using**  $\text{const}$

**using**  $\text{funas}$  *fresh*

**using**  $\text{remove-const-subst-steps-eq-lhs}[OF \ \text{lv-linear-sys}[OF \ \text{lv-conv}(2)] \ \text{fresh-sys}(3)]$

–

$\text{gsrsteps-eq-to-rsteps-eq}[OF \ \text{toS}(2)[\text{unfolded } \text{gsrstep-conversion-dist}]]]$

**by** *(cases vars-term t = {})*

*(auto simp: sig-step-conversion-dist funas-term-subst split: if-splits intro!:*

$\text{rsteps-eq-srsteps-eqI}[OF \ \text{sig-conv}]$

**ultimately** **show**  $(s, t) \in (\text{srstep } \mathcal{F} \ \mathcal{S})^{\leftrightarrow*}$

**by** *(meson conversionE conversionI rtrancl-trans)*

**qed**

**lemma** *lv-CE*:

**assumes** *sig*:  $\text{funas-rel } \mathcal{R} \subseteq \mathcal{F} \text{ funas-rel } \mathcal{S} \subseteq \mathcal{F}$   
**and** *fresh*:  $(c, 0) \notin \mathcal{F}$  **and** *const*:  $(a, 0) \in \mathcal{F}$   
**and** *lv*:  $lv \ \mathcal{R} \ lv \ \mathcal{S}$   
**and** *ce*:  $CE \ (gsrstep \ (insert \ (c, 0) \ \mathcal{F}) \ \mathcal{R}) \ (gsrstep \ (insert \ (c, 0) \ \mathcal{F}) \ \mathcal{S})$   
**shows**  $CE \ (srstep \ \mathcal{F} \ \mathcal{R}) \ (srstep \ \mathcal{F} \ \mathcal{S})$

**proof** –

**{fix** *s t* **assume**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ (\mathcal{R}^{\leftrightarrow})$   
**from** *lv-CE-aux*[*OF this assms*] **have**  $(s, t) \in (srstep \ \mathcal{F} \ \mathcal{S})^{\leftrightarrow*}$  **by** *simp*}  
**moreover**  
**{fix** *s t* **assume**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ (\mathcal{S}^{\leftrightarrow})$   
**from** *lv-CE-aux*[*OF this sig*(2, 1) *fresh const lv*(2, 1) *CE-symmetric*[*OF ce*]]  
**have**  $(s, t) \in (srstep \ \mathcal{F} \ \mathcal{R})^{\leftrightarrow*}$  **by** *simp*}  
**ultimately show** *?thesis*  
**by** (*intro CE-rrstep-intro*) *auto*

**qed**

## 9.1 Specialized for monadic signature

**lemma** *lv-NE-aux*:

**assumes**  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \ \mathcal{R}$  **and** *fresh*:  $(c, 0) \notin \mathcal{F}$   
**and** *sig*:  $\text{funas-rel } \mathcal{R} \subseteq \mathcal{F} \text{ funas-rel } \mathcal{S} \subseteq \mathcal{F}$   
**and** *lv*:  $lv \ \mathcal{R} \ lv \ \mathcal{S}$   
**and** *mon*: *monadic*  $\mathcal{F}$   
**and** *ne*:  $NE \ (gsrstep \ (insert \ (c, 0) \ \mathcal{F}) \ \mathcal{R}) \ (gsrstep \ (insert \ (c, 0) \ \mathcal{F}) \ \mathcal{S})$   
**shows**  $NE\text{-redp } \mathcal{F} \ \mathcal{R} \ \mathcal{S} \ s \ t$

**proof** –

**let**  $?s = \text{const-subst } c$  **let**  $?H = \text{insert } (c, 0) \ \mathcal{F}$   
**have** *mono*:  $\mathcal{F} \subseteq ?H$  **by** *auto*  
**from** *mon* **have** *mh*: *monadic*  $?H$  **by** (*auto simp: monadic-def*)  
**from** *fresh* **have** *fresh-sys*:  $(c, 0) \notin \text{funas-rel } \mathcal{R} \ (c, 0) \notin \text{funas-rel } \mathcal{S}$  **using** *sig*  
**by** (*auto simp: funas-rel-def*)  
**from** *assms* **have** *funas*:  $\text{funas-term } s \subseteq \mathcal{F} \ \text{funas-term } t \subseteq \mathcal{F}$   
**by** (*meson srstepsD srsteps-with-root-step-srstepsD*)  
**from** *funas* **have** *fresh-t*:  $(c, 0) \notin \text{funas-term } t$  **using** *fresh* **by** *auto*  
**from** *srsteps-subst-closed*[*OF srsteps-monp*[*OF mono*, *THEN subsetD*, *OF srsteps-with-root-step-srstepsD*[*OF assms*(1)]]], *of ?s*  
**have** *gstep*:  $(s \cdot ?s, t \cdot ?s) \in (gsrstep \ (insert \ (c, 0) \ \mathcal{F}) \ \mathcal{R})^+$   
**by** (*auto simp: ground-subst-apply intro!: ground-srsteps-gsrsteps*)  
**then** **have** *neg*:  $t \in NF \ (srstep \ \mathcal{F} \ \mathcal{R}) \implies s \cdot ?s \neq t \cdot ?s$   
**using** *NF-to-fresh-const-subst-NF*[*OF lv-linear-sys*[*OF lv*(1)]] *fresh-sys*(1) *sig*(1) *funas*(2), *of ?H*  
**by** (*metis NF-no-trancl-step*)  
**then** **have**  $t \in NF \ (srstep \ \mathcal{F} \ \mathcal{R}) \implies (s \cdot ?s, t \cdot ?s) \in (gsrstep \ (insert \ (c, 0) \ \mathcal{F}) \ \mathcal{S})^+$  **using** *gstep*  
**using** *NF-to-fresh-const-subst-NF*[*OF lv-linear-sys*[*OF lv*(1)]] *fresh-sys*(1) *sig*(1) *funas*(2), *of ?H*

```

    using NE-NF-eq[OF ne, symmetric] ne unfolding NE-on-def
  by (auto simp: normalizability-def ground-subst-apply) (meson rtrancl-eq-or-trancl)
  then show ?thesis unfolding NE-redp-def
    using remove-const-lv-mondaic-steps[OF lv(2) fresh-sys(2) mh, THEN srstepsD[THEN
conjunct1],
      THEN rsteps-srstepsI[OF sig(2) funas]]
    by (auto dest!: gsrsteps-to-srsteps)
qed

```

**lemma** *lv-NE*:

```

  assumes sig: funas-rel  $\mathcal{R} \subseteq \mathcal{F}$  funas-rel  $\mathcal{S} \subseteq \mathcal{F}$ 
    and mon: monadic  $\mathcal{F}$  and fresh:  $(c, 0) \notin \mathcal{F}$ 
    and lv: lv  $\mathcal{R}$  lv  $\mathcal{S}$ 
    and ne: NE (gsrstep (insert (c, 0)  $\mathcal{F}$ )  $\mathcal{R}$ ) (gsrstep (insert (c, 0)  $\mathcal{F}$ )  $\mathcal{S}$ )
  shows NE (srstep  $\mathcal{F}$   $\mathcal{R}$ ) (srstep  $\mathcal{F}$   $\mathcal{S}$ )
proof -
  from fresh have fresh-sys:  $(c, 0) \notin \text{funas-rel } \mathcal{R}$   $(c, 0) \notin \text{funas-rel } \mathcal{S}$ 
    using sig by (auto simp: funas-rel-def)
  from NE-NF-eq[OF ne] have ne-eq:  $NF (srstep \mathcal{F} \mathcal{R}) = NF (srstep \mathcal{F} \mathcal{S})$  using
lv sig fresh-sys
    by (intro linear-sys-gNF-eq-NF-eq) (auto dest: lv-linear-sys)
  {fix s t assume  $(s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{R}$ 
    from lv-NE-aux[OF this fresh sig lv mon ne] have NE-redp  $\mathcal{F} \mathcal{R} \mathcal{S} s t$  by
simp}
  moreover
  {fix s t assume  $\text{ass}: (s, t) \in \text{srsteps-with-root-step } \mathcal{F} \mathcal{S}$ 
    from lv-NE-aux[OF this fresh sig(2, 1) lv(2, 1) mon NE-symmetric[OF ne]]
  have NE-redp  $\mathcal{F} \mathcal{S} \mathcal{R} s t$  by simp}
  ultimately show ?thesis using ne-eq
    by (intro NE-rrstep-intro) auto
qed
end

```

## References

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