Termination Restricted to Right-Forward Closures

René Thiemann[1], Dieter Hofbauer[2], Ulysse Le Huitouze[3], and Johannes Waldmann[4]

[1]University of Innsbruck, Austria
[2]ASW Saarland, Germany
[3]University of Rennes, France
[4]HTWK Leipzig, Germany

July 17, 2026

Abstract

In this AFP entry we formalize Dershowitz' theorem that termination of a term rewrite system (TRS) starting from arbitrary terms is equivalent to termination starting from terms in the right-forward closures of right-hand sides, provided that the TRS is right-linear or orthogonal [1]. Our proof deviates from the original one in that no reorderings of steps in infinite derivations are required, making it more precise in its argumentation. We also integrate a later result that one can weaken orthogonality to locally confluent overlay TRSs [2].

In order to arrive at these statements, we had to formalize two further results: Following Gramlich, we prove that for locally confluent overlay TRSs the notions of termination and innermost termination coincide [3]; and we verify that narrowing with a right-linear TRS preserves linearity of terms.

For further details, we refer to our FSCD 2026 paper [4].

Contents

1 Rewriting with the Innermost Strategy

theory *Innermost-Rewriting*
 imports *First-Order-Rewriting.Trs*
begin

inductive-set *inn-rstep* :: $('f, 'v)$ *trs* $\Rightarrow$ $('f, 'v)$ *term rel* **for** R
 where *inn-rstep*[*intro*]: $(l,r) \in R \Longrightarrow$ *set* (*args* $(l \cdot \sigma)$) $\subseteq$ *NF-trs* $R \Longrightarrow (C\langle l \cdot \sigma\rangle, C\langle r \cdot \sigma\rangle) \in$ *inn-rstep* R

lemma *inn-rstep-rstep*: *inn-rstep* $R \subseteq$ *rstep* R
 ⟨*proof*⟩

lemma *inn-rstep-ctxt*: **assumes** $(s,t) \in$ *inn-rstep* R
 shows $(C\langle s\rangle, C\langle t\rangle) \in$ *inn-rstep* R
 ⟨*proof*⟩

lemma *ctxt-closed-inn-rstep*[*simp*, *intro*]: *ctxt.closed* (*inn-rstep* R)
 ⟨*proof*⟩

inductive-set *inn-nrrstep* :: $('f, 'v)$ *trs* $\Rightarrow$ $('f, 'v)$ *term rel* **for** R
 where *inn-nrrstep*[*intro*]: $(l,r) \in R \Longrightarrow$ *set* (*args* $(l \cdot \sigma)$) $\subseteq$ *NF-trs* $R \Longrightarrow C \neq$ *Hole* $\Longrightarrow (C\langle l \cdot \sigma\rangle, C\langle r \cdot \sigma\rangle) \in$ *inn-nrrstep* R

lemma *inn-nrrstep-nrrstep*: *inn-nrrstep* $R \subseteq$ *nrrstep* R
 ⟨*proof*⟩

lemma *inn-nrrstep-inn-rstep*: *inn-nrrstep* $R \subseteq$ *inn-rstep* R
 ⟨*proof*⟩

lemma *inn-nrrstep-ctxt*: **assumes** $(s,t) \in$ *inn-nrrstep* R
 shows $(C\langle s\rangle, C\langle t\rangle) \in$ *inn-nrrstep* R
 ⟨*proof*⟩

lemma *inn-rstep-ctxt-inn-nrrstep*: **assumes** $(s,t) \in$ *inn-rstep* R
 and $C \neq$ *Hole*
 shows $(C\langle s\rangle, C\langle t\rangle) \in$ *inn-nrrstep* R
 ⟨*proof*⟩

lemma *NF-inn-rstep-rstep*: *NF* (*inn-rstep* R) = *NF* (*rstep* R)
⟨*proof*⟩

lemma *NF-inn-nrrstep-nrrstep*: *NF* (*inn-nrrstep* R) = *NF* (*nrrstep* R)
⟨*proof*⟩

inductive-set *inn-rrstep* :: $('f, 'v)$ *trs* $\Rightarrow$ $('f, 'v)$ *term rel* **for** *R*
where *inn-rrstep*[*intro*]: $(l,r) \in R \Longrightarrow$ *set* (*args* $(l \cdot \sigma)) \subseteq$ *NF-trs* $R \Longrightarrow (l \cdot \sigma, r \cdot \sigma) \in$ *inn-rrstep R*

lemma *inn-rrstep-rrstep*: *inn-rrstep R* $\subseteq$ *rrstep R*
⟨*proof*⟩

lemma *inn-rrstep-inn-rstep*: *inn-rrstep R* $\subseteq$ *inn-rstep R*
⟨*proof*⟩

lemma *inn-rstep-iff-inn-rrstep-or-inn-nrrstep*: *inn-rstep R* = (*inn-rrstep R* $\cup$ *inn-nrrstep R*)
⟨*proof*⟩

lemma *inn-rstep-cases*[*consumes 1*, *case-names root nonroot*]:
$[\![(s,t) \in$ *inn-rstep R*; $(s,t) \in$ *inn-rrstep R* $\Longrightarrow P$; $(s,t) \in$ *inn-nrrstep R* $\Longrightarrow P]\!]$ $\Longrightarrow P$
⟨*proof*⟩

lemma *inn-nrrstep-args*:
assumes $(s, t) \in$ *inn-nrrstep R*
shows $\exists f$ *ss ts*. s = *Fun f ss* $\wedge$ t = *Fun f ts* $\wedge$ *length ss* = *length ts*
$\wedge$ ($\exists j$<*length ss*. (*ss*!*j*, *ts*!*j*) $\in$ *inn-rstep R* $\wedge$ ($\forall i$<*length ss*. $i \neq j \longrightarrow$ *ss*!*i* = *ts*!*i*))
⟨*proof*⟩

lemma *Tinf-imp-SN-inn-nrrstep*: **assumes** $t \in$ *Tinf* (*inn-rstep R*)
shows *SN-on* (*inn-nrrstep R*) $\{t\}$
⟨*proof*⟩

lemma *Tinf-inn-rstep-imp-first-root-step*: **assumes** $s \in$ *Tinf* (*inn-rstep R*)
shows $\exists$ *t u*. $(s,t) \in$ (*inn-nrrstep R*)^* $\wedge$ $(t,u) \in$ *inn-rrstep R* $\wedge$ $\neg$ *SN-on* (*inn-rstep R*) $\{u\}$
⟨*proof*⟩

end

2 Gramlich's Criterion to Prove Termination by Innermost Termination

Gramlich showed that for locally confluent overlay TRSs, innermost termination and termination coincide. We formalize this result in this theory.

theory *Gramlich-Innermost-Switch*
  imports
    First-Order-Rewriting.Critical-Pairs
    Innermost-Rewriting
begin

lemma *WCR-on-rstep-imp-WCR-on-nrrstep*: **assumes** *WCR-on* (*rstep R*) $\{t$. *SN-on* (*rstep R*) $\{t\}\}$
  shows *WCR-on* (*nrrstep R*) $\{t$. *SN-on* (*nrrstep R*) $\{t\}\}$
⟨*proof*⟩

lemma *SN-innermost-switch-locally-confluent-overlay-local*:
  assumes *WCR*: *WCR-on* (*rstep R*) $\{t$. *SN-on* (*rstep R*) $\{t\}\}$
  and *overlay*: $\bigwedge$ *l r*. (*False*,*l*,*r*) $\notin$ *critical-pairs ren R R*
  and *wf*: *wf-trs R*
shows *SN-on* (*inn-rstep R*) = *SN-on* (*rstep R*)
⟨*proof*⟩

lemma *SN-innermost-switch-locally-confluent-overlay*:
  assumes *WCR*: *WCR-on* (*rstep R*) $\{t$. *SN-on* (*rstep R*) $\{t\}\}$
  and *overlay*: $\bigwedge$ *l r*. (*False*,*l*,*r*) $\notin$ *critical-pairs ren R R*
  and *wf*: *wf-trs R*
shows *SN* (*inn-rstep R*) = *SN* (*rstep R*)
  ⟨*proof*⟩

end

3 Linearity Preservation by Unification

theory *Linear-Unification*
  imports
    First-Order-Terms.Unification-More
begin

A sufficient criterion to ensure that $t \cdot \sigma$ is linear.

lemma *linear-term-subst*: *linear-term t*
  $\Longrightarrow$ ($\bigwedge$ *x*. *x* $\in$ *vars-term t* $\Longrightarrow$ *linear-term* (σ *x*))
  $\Longrightarrow$ ($\bigwedge$ *x y*. *x* $\in$ *vars-term t* $\Longrightarrow$ *y* $\in$ *vars-term t* $\Longrightarrow$ $x \neq y$ $\Longrightarrow$ *vars-term* (σ *x*) $\cap$ *vars-term* (σ *y*) = {})
  $\Longrightarrow$ *linear-term* ($t \cdot \sigma$)
⟨*proof*⟩

Unification of two var disjoint terms where one of them is linear results in a partially linear substitution and linear terms

definition *vars-mset-left* :: (($'f$,$'v$)*term* × ($'f$,$'v$)*term*) *multiset* ⇒ $'v$ *multiset* **where**
  vars-mset-left m = *sum-mset* (*image-mset* (*vars-term-ms o fst*) *m*)

definition *vars-mset-right* :: $(('f,'v)term \times ('f,'v)term)$ *multiset* $\Rightarrow$ *'v multiset* **where**
vars-mset-right m = *sum-mset* (*image-mset* (*vars-term-ms o snd*) *m*)

definition *linear-mset* :: *'a multiset* $\Rightarrow$ *bool* **where**
linear-mset m = ($\forall$ *x*. *count m x* $\leq$ *1*)

lemma *count-sum-mset-image-mset*:
count (*sum-mset* (*image-mset f m*)) *x* = *sum-mset* (*image-mset* (λ *a*. *count* (*f a*) *x*) *m*)
⟨*proof*⟩

lemma *linear-vars-term-ms*: *linear-mset* (*vars-term-ms t*) = *linear-term t*
⟨*proof*⟩

lemma *linear-term-count*: **assumes** *linear-term t*
shows *count* (*vars-term-ms t*) *x* $\leq$ *1*
⟨*proof*⟩

lemma *linear-term-Var-subst*: *linear-term* ($t \cdot$ (*Var o r*)) $\Longrightarrow$ *linear-term t*
⟨*proof*⟩

lemma *vars-mset-right-add*[*simp*]: *vars-mset-right* (*add-mset p E*) = *vars-term-ms* (*snd p*) + *vars-mset-right E*
⟨*proof*⟩

lemma *right-linear-var-disjoint-mgu-mset*: **fixes** *E* :: $(('f,'v)term \times ('f,'v)term)$ *multiset*
and *u* :: $('f,'v)term$
assumes *set-mset* (*vars-mset-left E*) $\cap$ *set-mset* (*vars-mset-right E*) = {}
and *linear-mset* (*vars-mset-right E*)
and *is-mgu* σ (*set-mset E*)
and *vars-term u* $\cap$ *set-mset* (*vars-mset-right E*) = {}
and *linear-term u*
shows *linear-term* ($u \cdot \sigma$)
⟨*proof*⟩

lemma *right-linear-var-disjoint-mgu*: **fixes** *s t* :: $('f,'v)term$
assumes *disj*: *vars-term s* $\cap$ *vars-term t* = {}
and *lin*: *linear-term t*
and *mgu*: *is-mgu* σ {(*s*,*t*)}
and *linu*: *linear-term u*
and *disju*: *vars-term u* $\cap$ *vars-term t* = {}
shows *linear-term* ($u \cdot \sigma$)
⟨*proof*⟩

Corollary: Unification of two linear var disjoint terms results in a linear substitution and linear unified terms.

lemma *linear-var-disjoint-is-mgu*: **fixes** $s\ t$ :: $('f,'v)term$
  assumes *disj*: *vars-term* $s \cap$ *vars-term* $t = \{\}$
  and *lin*: *linear-term* s *linear-term* t
  and *mgu*: *is-mgu* σ $\{(s,t)\}$
shows *vars-term* $u \cap$ *vars-term* $t = \{\} \Longrightarrow$ *linear-term* $u \Longrightarrow$ *linear-term* $(u \cdot \sigma)$
  and *vars-term* $u \cap$ *vars-term* $s = \{\} \Longrightarrow$ *linear-term* $u \Longrightarrow$ *linear-term* $(u \cdot \sigma)$
  and *linear-term* $(s \cdot \sigma)$ *linear-term* $(t \cdot \sigma)$
  and *linear-term* $(\sigma\ x)$
⟨*proof*⟩
end

4 Narrowing and Linearity Preservation

In this theory we define narrowing and show that narrowing of a linear term with a right-linear TRS results in a linear term again. Moreover, some further results on narrowing are formalized.

theory *Linear-Narrowing*
  imports
    Automatic-Refinement.Misc
    Linear-Unification
    First-Order-Rewriting.Trs
begin

4.1 Preparations for Narrowing

definition *subst-term-mset* :: $('f,'v)term \Rightarrow ('f,'v)subst \Rightarrow ('f,'v)term\ multiset$
  where *subst-term-mset* $t\ \sigma$ = *image-mset* σ (*vars-term-ms* t)

lemma *vars-term-ms-subst-compose-split*:
  $(\delta$ ‘# $(\sum x{\in}\#xs.$ *vars-term-ms* $(\mu\ x))$, $(\mu \circ_s \delta)$ ‘# $xs) \in$ (*mult* $\{\lhd\})^=$
    $\wedge$ (*Ball* (μ ‘ *set-mset* xs) *is-Var* $\vee$ (δ ‘# $(\sum x{\in}\#xs.$ *vars-term-ms* $(\mu\ x))$, $(\mu \circ_s$
$\delta)$ ‘# $xs) \in$ *mult* $\{\lhd\}$)
    (**is** - $\in$ *?R*$^=$ $\wedge$ -)
⟨*proof*⟩

lemma *linear-term-vars-term-ms*: **assumes** *vars-term* $r \subseteq$ *vars-term* l
    and *linear-term* r
  shows *vars-term-ms* $r \subseteq\#$ *vars-term-ms* l
⟨*proof*⟩

lemma *vars-term-ms-map-vars-term*[*simp*]: *vars-term-ms* (*map-vars-term* $f\ t$) = f
‘# *vars-term-ms* t
⟨*proof*⟩

lemma *linear-term-map-inj-on*: **assumes** *linear-term* (*map-vars-term* $f\ t$)
  shows *inj-on* f (*vars-term* t)
⟨*proof*⟩

4.2 Narrowing Relations

context
fixes *ren* :: *'v* :: *infinite renaming2*
begin

definition
narrows-r-p-s :: ('*f*, '*v*) *trs* ⇒ ('*f*, '*v*) *rule* ⇒ *pos* ⇒ ('*f*,'*v*)*subst* ⇒ ('*f*, '*v*)*trs*
where
narrows-r-p-s R r p μ ≡ {(*s*,*t*). ∃ *μ2*. *p* ∈ *poss s* ∧ *is-Fun* (*s* |- *p*) ∧ *r* ∈ *R* ∧
mgu-vd ren (*s* |- *p*) (*fst r*) = *Some* (*μ*, *μ2*)
∧ *t* = *replace-at* (*s* · *μ*) *p* (*snd r* · *μ2*)}

definition
narrow-step-s :: ('*f*,'*v*) *trs* ⇒ ('*f*,'*v*)*subst* ⇒ ('*f*,'*v*)*trs* **where**
narrow-step-s R μ = {*st* | *st lr p*. *st* ∈ *narrows-r-p-s R lr p μ*}

definition *narrow-step* :: ('*f*,'*v*) *trs* ⇒ ('*f*,'*v*)*trs* **where**
narrow-step R = {*st* | *st lr p μ*. *st* ∈ *narrows-r-p-s R lr p μ*}

lemma *narrow-step-to-s*: *narrow-step R* = ⋃ (*range* (*narrow-step-s R*))
⟨*proof*⟩

theorem *right-linear-rule-narrowing*: **fixes** *R* :: ('*f*,'*v*)*trs*
assumes *linr*: *linear-term r*
and *lins*: *linear-term s*
and *narr*: (*s*,*t*) ∈ *narrows-r-p-s R* (*l*,*r*) *p μ*
shows *linear-term t*
⟨*proof*⟩

theorem *right-linear-rule-narrow-step*:
assumes ⋀ *lr*. *lr* ∈ *R* ⟹ *linear-term* (*snd lr*)
and *lins*: *linear-term s*
and *narr*: (*s*,*t*) ∈ *narrow-step R*
shows *linear-term t*
⟨*proof*⟩

lemma *narrowing-right-linear-one-step-simulation-r-p-s*: **fixes** *R* :: ('*f*,'*v*)*trs*
assumes (*s* · *σ*, *t*) ∈ *rstep-r-p-s R* (*l*,*r*) *p μ*
and *lins*: *linear-term s*
and *linr*: *linear-term r*
and *var-cond*: *vars-term r* ⊆ *vars-term l*
shows *p* ∉ *fun-poss s* ∧
(∃ *δ*. *t* = *s* · *δ* ∧ (*subst-term-mset s δ*, *subst-term-mset s σ*) ∈ *mult1* ((*rstep R*)$\hat{}$−*1*))
∨
p ∈ *fun-poss s* ∧
(∃ *μ1 μ2 δ u*. *t* = *u* · *δ* ∧ *linear-term u* ∧ (*s*,*u*) ∈ *narrows-r-p-s R* (*l*,*r*) *p μ1* ∧ *σ* = *μ1* $\circ_s$ *δ* ∧ *μ* = *μ2* $\circ_s$ *δ*

$\wedge$ (*subst-term-mset u* δ, *subst-term-mset s* σ) $\in$ (*mult* $\{\triangleleft\})^{=}$)
⟨*proof*⟩

lemma *narrowing-right-linear-one-step-simulation*: **fixes** R :: ('f,'v)*trs*
assumes $(s \cdot \sigma,\ t) \in$ *rstep R*
and *lins*: *linear-term s*
and *right-lin*: $\bigwedge$ *lr*. *lr* $\in R \Longrightarrow$ *linear-term* (*snd lr*)
and *var-cond*: $\bigwedge$ *lr*. *lr* $\in R \Longrightarrow$ *vars-term* (*snd lr*) $\subseteq$ *vars-term* (*fst lr*)
shows ($\exists$ δ. $t = s \cdot \delta \wedge$ (*subst-term-mset s* δ, *subst-term-mset s* σ) $\in$ *mult1* ((*rstep R*)$\widehat{\ }{-1}$))
$\vee$ ($\exists$ μ δ *u*. $t = u \cdot \delta \wedge$ *linear-term u* $\wedge$ (*s*,*u*) $\in$ *narrow-step-s R* $\mu \wedge \sigma = \mu \circ_s \delta$
$\wedge$ (*subst-term-mset u* δ, *subst-term-mset s* σ) $\in$ (*mult* $\{\triangleleft\})^{=}$)
⟨*proof*⟩

lemma *rstep-instance-imp-narrows-r-p-s*: **assumes** *step*: $(s \cdot \sigma, t) \in$ *rstep-r-p-s R r p* τ
and *not*σ: $p \in$ *poss s is-Fun* (*s* |- *p*)
shows $\exists$ μ $\mu 2$ δ *u*. (*s*,*u*) $\in$ *narrows-r-p-s R r p* $\mu \wedge s \cdot \sigma = s \cdot \mu \cdot \delta \wedge t = u \cdot \delta \wedge \sigma = \mu \circ_s \delta \wedge \tau = \mu 2 \circ_s \delta$
⟨*proof*⟩

lemma *narrows-r-p-s-imp-rstep-r-p-s*: **assumes** *narr*: (*s*,*t*) $\in$ *narrows-r-p-s R r p* μ
shows $\exists$ δ. $(s \cdot \mu,\ t) \in$ *rstep-r-p-s R r p* δ
⟨*proof*⟩

lemma *narrows-r-p-s-imp-rstep*: **assumes** *narr*: (*s*,*t*) $\in$ *narrows-r-p-s R r p* μ
shows $(s \cdot \mu,\ t) \in$ *rstep R*
⟨*proof*⟩

lemma *narrow-step-s-to-rstep*: (*s*,*t*) $\in$ *narrow-step-s R* $\mu \Longrightarrow (s \cdot \mu,\ t) \in$ *rstep R*
⟨*proof*⟩

lemma *rstep-imp-narrows-r-p-s*: **assumes** *step*: (*s*,*t*) $\in$ *rstep-r-p-s R r p* τ
and *wf*: *wf-trs R*
shows $\exists$ μ δ *u*. (*s*,*u*) $\in$ *narrows-r-p-s R r p* $\mu \wedge s = s \cdot \mu \cdot \delta \wedge t = u \cdot \delta \wedge u = t \cdot \mu \wedge \mu \circ_s \delta = Var$
⟨*proof*⟩

lemma *rstep-imp-narrows-s*: **assumes** *step*: (*s*,*t*) $\in$ *rstep R*
and *wf*: *wf-trs R*
shows $\exists$ μ δ *u*. (*s*,*u*) $\in$ *narrow-step-s R* $\mu \wedge s = s \cdot \mu \cdot \delta \wedge t = u \cdot \delta \wedge u = t \cdot \mu \wedge \mu \circ_s \delta = Var$
⟨*proof*⟩

lemma *rstep-imp-narrows*: **assumes** *step*: (*s*,*t*) $\in$ *rstep R*
and *wf*: *wf-trs R*

shows $\exists\ \mu\ \delta\ u.\ (s,u) \in \textit{narrow-step}\ R \wedge t = u \cdot \delta \wedge u = t \cdot \mu$
⟨*proof*⟩

lemma *narrow-instance-pos-in-subst*: **assumes**
narr: $(t \cdot \gamma,\ s) \in \textit{narrows-r-p-s}\ (R :: ('f,'v)trs)\ (l,r)\ p\ \mu$
and *pos*: $p \notin \textit{fun-poss}\ t$
and *vars*: $\textit{vars-term}\ r \subseteq \textit{vars-term}\ l$
and *lin*: $\textit{linear-term}\ (t :: ('f,'v)term)\ \textit{linear-term}\ r$
shows $\exists\gamma.\ t \cdot \gamma = s$
⟨*proof*⟩

lemma *narrow-instance-pos-in-term*: **fixes** $t :: ('f,'v)term$
assumes *narr*: $(t,\ s) \in \textit{narrows-r-p-s}\ R\ (l,\ r)\ p\ \mu 1$
and *narr-inst*: $(t \cdot \gamma,\ u) \in \textit{narrows-r-p-s}\ R\ (l,\ r)\ p\ \mu 2$
shows $\exists\delta.\ s \cdot \delta = u \wedge \gamma \circ_s \mu 2 = \mu 1 \circ_s \delta$
⟨*proof*⟩

lemma *SN-on-narrows-imp-SN-on-rstep*: **fixes** $R :: ('f,'v)trs$
assumes *SN-on* (*narrow-step* R) $\{s\}$
and *wf*: *wf-trs* R
shows *SN-on* (*rstep* R) $\{s\}$
⟨*proof*⟩

lemma *exists-narrow-steps-to-infinite-rsteps*: **fixes** $R :: ('f,'v)trs$
defines $Q \equiv \lambda\ (n :: nat)\ s\ (\sigma :: ('f,'v)subst)\ u\ (\delta :: ('f,'v)subst).$
$(\exists\ \mu.\ (s,u) \in \textit{narrow-step-s}\ R\ \mu \wedge \sigma = \mu \circ_s \delta)$
assumes *wf*: *wf-trs* R
and *single-steps*: $\bigwedge n\ s\ \sigma.\ P\ n\ s\ \sigma \Longrightarrow \exists u\ \delta.\ P\ (Suc\ n)\ u\ \delta \wedge Q\ n\ s\ \sigma\ u\ \delta$
and *P0*: $P\ 0\ r\ \sigma$
shows $\neg$ *SN-on* (*narrow-step* R) $\{r\}$
$\neg$ *SN-on* (*rstep* R) ((*narrow-step* R)* `` $\{r\}$)
⟨*proof*⟩

end
end

5 Dershowitz' Right-Forward Closures for Proving Termination

Right-Forward Closures (RFC) are defined, and it is shown that termination on all terms is equivalent to termination starting on RFC for TRSs that are right-linear, or for overlay TRSs that are locally confluent.

theory *Right-Forward-Closure*
imports
Linear-Narrowing
First-Order-Rewriting.Critical-Pairs

Gramlich-Innermost-Switch
begin

context
fixes *ren* :: $'v$ :: *infinite renaming2*
begin

definition *right-forw-closure* :: $('f,'v)$*trs* $\Rightarrow$ $('f,'v)$*term set* **where**
right-forw-closure R = (*narrow-step ren R*)* ‘‘ *rhss R*

lemma *right-forw-closure-SN-main*: **fixes** *R* :: $('f,'v)$*trs*
assumes *right-lin*: $\bigwedge$ *lr*. *lr* $\in$ *R* $\Longrightarrow$ *linear-term* (*snd lr*)
and *wf*: *wf-trs R*
and *nSN*: $\neg$ *SN* (*rstep R*)
shows $\neg$ *SN-on* (*rstep R*) (*right-forw-closure R*)
⟨*proof*⟩

theorem *right-linear-SN-rstep-RFC*:
assumes *wf*: *wf-trs R*
and *rlin*: $\bigwedge$*lr*. *lr* $\in$ *R* $\Longrightarrow$ *linear-term* (*snd lr*)
shows *SN* (*rstep R*) $\longleftrightarrow$ *SN-on* (*rstep R*) (*right-forw-closure R*) (**is** *?A* = *?B*)
⟨*proof*⟩

context
fixes *R* :: $('f,'v)$*trs*
assumes *wf*: *wf-trs R*
and *WCR*: *WCR* (*rstep R*)
and *overlay*: $\bigwedge$ *l r*. (*False*,*l*,*r*) $\notin$ *critical-pairs ren R R*
begin

lemma *WCRO-not-SN-imp-non-terminating-innermost-rhs-instance*:
assumes $\neg$ *SN* (*rstep R*)
shows $\exists$ *r* σ. *r* $\in$ *rhss R* $\wedge$ $\neg$ *SN-on* (*inn-rstep R*) $\{r \cdot \sigma\}$ $\wedge$ σ ‘ *vars-term r* $\subseteq$ *NF* (*rstep R*)
⟨*proof*⟩

lemma *WCRO-one-step-simulation-by-narrowing*: **fixes** *s* :: $('f,'v)$*term*
assumes *nSN*: $\neg$ *SN-on* (*inn-rstep R*) $\{s \cdot \sigma\}$
and *NF-sigma*: σ ‘ *vars-term s* $\subseteq$ *NF* (*rstep R*)
shows $\exists$ *u* μ δ. (*s*,*u*) $\in$ *narrow-step-s ren R* μ
$\wedge$ $\neg$ *SN-on* (*inn-rstep R*) $\{u \cdot \delta\}$
$\wedge$ δ ‘ *vars-term u* $\subseteq$ *NF* (*rstep R*)
$\wedge$ $\sigma = \mu \circ_s \delta$
⟨*proof*⟩

lemma *WCRO-not-SN-rstep-imp-not-SN-rstep*: **assumes** $\neg$ *SN* (*rstep R*)
  shows $\neg$ *SN-on* (*rstep R*) (*right-forw-closure R*)
⟨*proof*⟩
end

theorem *WCRO-SN-rstep-RFC*: **assumes** *wf*: *wf-trs R*
  and *WCR*: *WCR* (*rstep R*)
  and *overlay*: $\bigwedge$ *l r*. (*False*,*l*,*r*) $\notin$ *critical-pairs ren R R*
shows *SN* (*rstep R*) $\longleftrightarrow$ *SN-on* (*rstep R*) (*right-forw-closure R*) (**is** *?A* = *?B*)
⟨*proof*⟩
end
end

References

[1] N. Dershowitz. Termination of linear rewriting systems (preliminary version). In S. Even and O. Kariv, editors, *Automata, Languages and Programming, 8th Colloquium, Acre (Akko), Israel, July 13-17, 1981. Proceedings*, volume 115 of *Lecture Notes in Computer Science*, pages 448–458. Springer, 1981.

[2] N. Dershowitz and C. Hoot. Natural termination. *Theor. Comput. Sci.*, 142(2):179–207, 1995.

[3] B. Gramlich. Relating innermost, weak, uniform and modular termination of term rewriting systems. In A. Voronkov, editor, *Logic Programming and Automated Reasoning, International Conference, LPAR '92, St. Petersburg, Russia, July 15-20, 1992. Proceedings*, volume 624 of *Lecture Notes in Computer Science*, pages 285–296. Springer, 1992.

[4] R. Thiemann, D. Hofbauer, U. Le Huitouze, and J. Waldmann. New and Formalized Proofs for Right-Forward Closures and Core Matrix Interpretations. In F. Pfenning, editor, *11th International Conference on Formal Structures for Computation and Deduction (FSCD 2026)*, volume 378 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 32:1–32:19, Dagstuhl, Germany, 2026. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.