

# Termination of Narrowing Revisited

María Alpuente

Departamento de Sistemas Informáticos y Computación (DSIC)  
Technical University of Valencia (UPV)  
Valencia, Spain

Pisa - October 23th, 2009

# Our motivation

(1991) Ph.D. on Equational Constraint Satisfaction via Narrowing



Supervisors: *Giorgio Levi and Isidro Ramos*

## Narrowing

Generalization of rewriting with unification and logic variables

**Problem: Termination of narrowing was an issue**

Naïve solution: Incremental constraint solving approach

## Our motivation

For 20 years a recurrent problem in many narrowing applications

- Operational model of functional–logic languages
- Equational unification
- Analysis and certification of security policies
- Symbolic reachability
- Type-checking of dependently typed languages
- Compact approximations of program semantics
- Automated proofs of termination for rewriting
- Model-checking
- Narrowing-driven partial evaluation
- Declarative debugging

# Our Goals

Despite the great number of narrowing-fueled tools, surprisingly termination of narrowing has received little attention (less than completeness)

## The main goals

- Investigate the (mostly unexplored) problem of narrowing termination
- Develop a practical tool (previously lacking): an automated narrowing termination prover

## Rewriting

$s \rightarrow_{p, \mathcal{R}} t \equiv s[r\sigma]_p$  if there is
 

- a position  $p$  of  $s$
- a rule  $(l \rightarrow r)$  in  $\mathcal{R}$
- a **matcher**  $\sigma$  s.t.  $s|_p = l\sigma$

↓

## Narrowing

$s \rightsquigarrow_{p, \mathcal{R}, \sigma} t \equiv (s[r]_p)\sigma$  if there is
 

- a **nonvariable** position  $p$  of  $s$
- a rule  $(l \rightarrow r)$  in  $\mathcal{R}$
- a **unifier**  $\sigma$  s.t.  $s|_p\sigma = l\sigma$   
 $[\sigma = \text{mgu}(s|_p, l)]$

## An example

Standard definition  
of addition (TRS)

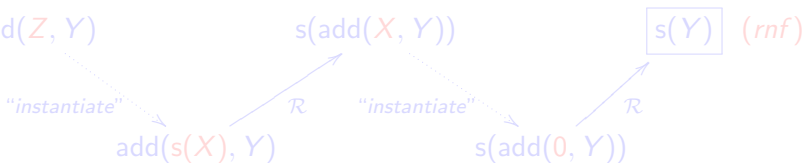
$$\mathcal{R}_{\text{add}} = \left\{ \begin{array}{l} \text{add}(0, Y) \rightarrow Y \\ \text{add}(s(X), Y) \rightarrow s(\text{add}(X, Y)) \end{array} \right.$$

With **rewriting**:  $\text{add}(s(0), 0) \rightarrow_{\mathcal{R}} s(\text{add}(0, 0)) \rightarrow_{\mathcal{R}} s(0)$

With **narrowing**:  $\text{add}(s(0), 0) \rightsquigarrow_{\mathcal{R}} s(\text{add}(0, 0)) \rightsquigarrow_{\mathcal{R}} s(0)$

but

also:  $\text{add}(Z, Y)$



with *computed answer substitution*  $\{Z/s(0)\}$

## An example

Standard definition  
of addition (TRS)

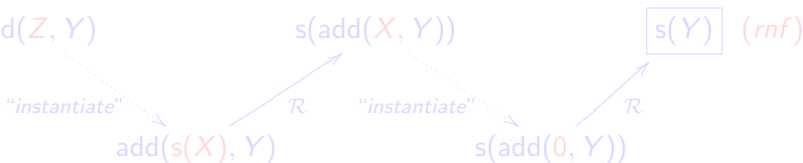
$$\mathcal{R}_{\text{add}} = \left\{ \begin{array}{l} \text{add}(0, Y) \rightarrow Y \\ \text{add}(s(X), Y) \rightarrow s(\text{add}(X, Y)) \end{array} \right.$$

With **rewriting**:  $\text{add}(s(0), 0) \rightarrow_{\mathcal{R}} s(\text{add}(0, 0)) \rightarrow_{\mathcal{R}} s(0)$

With **narrowing**:  $\text{add}(s(0), 0) \rightsquigarrow_{\mathcal{R}} s(\text{add}(0, 0)) \rightsquigarrow_{\mathcal{R}} s(0)$

but

also:  $\text{add}(Z, Y)$



with *computed answer substitution*  $\{Z/s(0)\}$

## An example

Standard definition  
of addition (TRS)

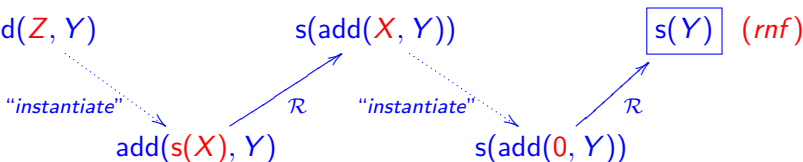
$$\mathcal{R}_{\text{add}} = \left\{ \begin{array}{l} \text{add}(0, Y) \rightarrow Y \\ \text{add}(s(X), Y) \rightarrow s(\text{add}(X, Y)) \end{array} \right.$$

With **rewriting**:  $\text{add}(s(0), 0) \rightarrow_{\mathcal{R}} s(\text{add}(0, 0)) \rightarrow_{\mathcal{R}} s(0)$

With **narrowing**:  $\text{add}(s(0), 0) \rightsquigarrow_{\mathcal{R}} s(\text{add}(0, 0)) \rightsquigarrow_{\mathcal{R}} s(0)$

but

also:  $\text{add}(Z, Y)$



with *computed answer substitution*

$\{Z/s(0)\}$



## An example

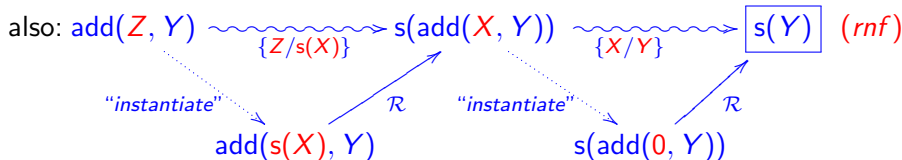
Standard definition  
of addition (TRS)

$$\mathcal{R}_{add} = \left\{ \begin{array}{l} \text{add}(0, Y) \rightarrow Y \\ \text{add}(s(X), Y) \rightarrow s(\text{add}(X, Y)) \end{array} \right.$$

With **rewriting**:  $\text{add}(s(0), 0) \rightarrow_{\mathcal{R}} s(\text{add}(0, 0)) \rightarrow_{\mathcal{R}} s(0)$

With **narrowing**:  $\text{add}(s(0), 0) \rightsquigarrow_{\mathcal{R}} s(\text{add}(0, 0)) \rightsquigarrow_{\mathcal{R}} s(0)$

but



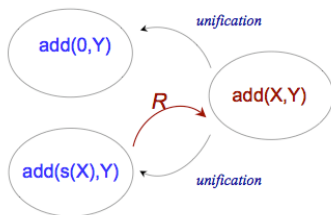
with *computed answer substitution*  $\{Z/s(0)\}$

## An example

Infinitely many narrowing derivations exist for the input call

$$\begin{aligned} \text{add}(Z, Y) &\rightsquigarrow_{\{Z/0\}} Y \\ &\vdots \\ \text{add}(Z, Y) &\rightsquigarrow_{\{Z/s^n(0)\}}^* s^n(Y) \end{aligned}$$

since there is a loop in the call graph (via unification)



## Narrowing termination matters!

Narrowing is extremely non-terminating, but there are many practical situations where narrowing termination matters, and besides it can be proven!

### Example

Narrowing **does not terminate** in  $\mathcal{R}_{id}$  due to uncontrolled recursion

$$\mathcal{R}_{id} = \left\{ \begin{array}{ll} id(X) & \rightarrow X \\ id(0) & \rightarrow 0 \\ id(s(X)) & \rightarrow s(id(X)) \end{array} \right.$$

$$id(X) = 0 \rightsquigarrow_{id} X = 0 \rightsquigarrow_{\{X/0\}} true$$

$$id(X) = 0 \rightsquigarrow_{\{X/s(X')\}} s(id(X')) = 0 \rightsquigarrow_{\{X'/s(X'')\}} s(s(id(X''))) = 0 \dots$$

## Narrowing termination matters!

In bottom-up program analysis, often we consider semantics also expressed by rules

it can be important that narrowing terminates in the denotations

E.g., in order to apply an immediate consequences operator

We need to **narrow**  
**expressions in the**  
**interpretation  $\mathcal{I}$**

$$\mathcal{I}_{\mathcal{R}}(\mathcal{I}) := \text{zip}\{l\theta \mapsto u \mid l \mapsto r \in \mathcal{R}, r \rightsquigarrow_{\mathcal{I}, \theta}^* u\}$$

The good news are that the rhs's of semantic rules have more limited recursion, making narrowing terminate!

Example

$$S(\mathcal{R}_{id}) = \{\text{id}(X) \mapsto X\}$$

# *...running calls in the semantics*



## **Which Semantics for Logic Languages?**

Giorgio Levi  
Dipartimento di Informatica  
Università di Pisa

Valencia, October 1989

## Completeness and Termination

Previous termination results were obtained as a *by-product of addressing the completeness* of narrowing-based procedures for equational unification

Hullot (1980) (*unification-completeness*)

In **canonical theories** (also called “complete”), narrowing is complete as an equational unification algorithm.

Example.

We have seen that narrowing is able to compute the solutions for

$$\exists X, Y, Z \text{ s.t. } \text{add}(X, Y) =_E Z$$

# Completeness and Termination

General problem of symbolic reachability in **non-confluent** TRSs  
(that is, to find “more general” solutions  $\sigma$  such that  $s\sigma \rightarrow_{\mathcal{R}}^* t\sigma$ )

Meseguer and Thati (2007) (*reachability-completeness*)

**Without assuming canonicity**, narrowing is complete as a procedure to solve reachability problems

Example.

$$\mathcal{R} = \mathcal{R}_{\text{add}} \cup \left\{ \begin{array}{l} \text{key} \rightarrow 0 \\ \text{key} \rightarrow s(0) \end{array} \right.$$

Narrowing is still able to compute the solutions for

$$\exists X, Y, Z \text{ s.t. } \text{add}(X, Y) \rightarrow_{\mathcal{R}}^* Z$$

That is, we can get rid of canonicity if we are not interested in equational unification.

This extends narrowing capabilities to the *analysis of concurrent systems*.

# Completeness and Termination

In our own work, we are interested in classes of TRSs where narrowing **terminates** and is **strongly reachability-complete** (i.e., w.r.t. non-normalized solutions)

Meseguer and Thati (2007) (*strong reachability-completeness*)

**Without assuming canonicity**, narrowing is strongly complete as a procedure to solve reachability problems for two classes of TRSs:

- 1 topmost
- 2 right-linear (for linear goals)

*unification-completeness*  $\Rightarrow$  *reachability-completeness*



1 Classical termination of narrowing: Christian and Hullot

2 Extending Hullot's termination criterion

3 Getting rid of canonicity and linearity

4 The Narradar Termination Tool

# Classical termination of narrowing: Christian and Hullot

The only positive termination result was proved by J. Christian

Theorem [Christian, 1992]

Narrowing terminates in *left-flat* TRS (lhs's arguments are *variables* or *ground terms*) that are compatible with a termination ordering  $<$

Example (Non-flat rules cause **echoing**)

$$\mathcal{R} = \{f(f(X)) \rightarrow X\}$$

$$\bullet (f(X), X) \rightsquigarrow_{\{X/f(X')\}} \bullet (X', f(X')) \rightsquigarrow_{\{X'/f(X'')\}} \bullet (f(X''), X'') \rightsquigarrow \dots$$

# Classical termination of narrowing: Christian and Hullot

The starting point for our work, a popular (**but faulty!**) termination result

Theorem [Hullot, 1980]

**Narrowing** terminates in **canonical** TRSs if all **basic narrowing derivations** issuing from the rhs's of the rules **terminate** (i.e., derivations which do not reduce some *blocked* positions).

**Example**

Remind that Christian's TRS  $\{f(f(X)) \rightarrow X\}$  is canonical and trivially satisfies the condition on the rhs  $X$

# Classical termination of narrowing: Christian and Hullot

(Faulty) Theorem [Hullot, 1980]

Narrowing terminates in **canonical** TRSs if all *basic narrowing derivations* issuing from the rhs's of the rules **terminate**.

(Downgraded) **b.n.** Theorem [Hullot's PhD. Thesis, 1981]

**Basic narrowing** terminates in **canonical** TRSs if all *basic narrowing derivations* issuing from the rhs's of the rules **terminate**.

# Classical termination of narrowing: Christian and Hullot

A naïve generalization of Hullot's criterion **does not hold**:

**Narrowing** terminates in canonical TRSs if all **basic narrowing** derivations issuing from the rhs's of the rules terminate

## Counter-example

Christian's TRS  $\mathcal{R} = \{f(f(X)) \rightarrow X\}$  does satisfy the stronger condition as well.

- 1 Classical termination of narrowing: Christian and Hullot
- 2 Extending Hullot's termination criterion**
- 3 Getting rid of canonicity and linearity
- 4 The Narradar Termination Tool

## Narrowing

$$s \rightsquigarrow_{p, \mathcal{R}, \sigma} (s[r]_p)\sigma \text{ if } \begin{cases} \bullet \text{ a nonvariable position } p \text{ of } s \\ \bullet \text{ a rule } (l \rightarrow r) \text{ in } \mathcal{R} \\ \bullet \text{ a unifier } \sigma. s|_p\sigma = l\sigma \\ \quad [\sigma = \text{mgu}(s|_p, l)] \end{cases}$$

⇓

## Basic narrowing

The idea is to avoid narrowing steps on subterms introduced by instantiation

$$\langle s, \rho \rangle \rightsquigarrow_{\sigma, \mathcal{R}}^p \langle s[r]_p, \rho\sigma \rangle \text{ if } \begin{cases} \bullet \text{ a nonvariable position } p \text{ of } s \\ \bullet \text{ a rule } (l \rightarrow r) \text{ in } \mathcal{R} \\ \bullet \text{ a unifier } \sigma. s|_p\rho\sigma = l\sigma \\ \quad [\sigma = \text{mgu}(s|_p\rho, l)] \end{cases}$$

## Basic narrowing terminates more often

Example.

$$f(f(X)) \rightarrow X$$

Basic narrowing blocks the  $\infty$  derivation of Christian's example at the first step (because echoing is not produced any more):

$$\langle \bullet(f(X), X), id \rangle \rightsquigarrow_{\{X/f(X')\}} \langle \bullet(X', X), \{X/f(X')\} \rangle \not\rightsquigarrow$$

whereas ordinary narrowing ran in a loop:

$$\bullet(\underline{f(X)}, X) \rightsquigarrow_{\{X/f(X')\}} \bullet(X', \underline{f(X')}) \rightsquigarrow_{\{X'/f(X'')\}} \bullet(\underline{f(X'')}, X'') \rightsquigarrow \dots$$



# How to get the extension?

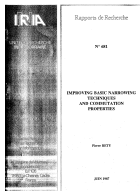
By just considering a class of TRSs identified by Réty where, for each narrowing derivation there is an commuted **b. n.** derivation.

## (Generalized) Hullot's Theorem

**Narrowing** terminates in TRSs that satisfy Réty's maximal commutation property if *basic* narrowing terminates.

(Réty, 1987) The TRS  $\mathcal{R}$  satisfies the maximal commutation property if

- 1) it is right-linear, and either left-linear or conservative ( $\text{Var}(l) = \text{Var}(r)$ ), and
- 2) *narrowing only produces normalized substitutions* in  $\mathcal{R}$  (this holds in *left-linear CSs*)



- 1 Classical termination of narrowing: Christian and Hullot
- 2 Extending Hullot's termination criterion
- 3 Getting rid of canonicity and linearity
- 4 The Narradar Termination Tool

## Getting rid of canonicity

Canonicity is required but superfluous in Hullot's b.n. termination result (required for deriving *termination & unification-completeness* in one go).

(Downgraded and purged) Hullot's b.n. Theorem

Basic narrowing terminates (~~in-canonical TRSs~~) if all basic narrowing derivations issuing from the rhs's of the rules terminate.

Termination Corollary

Narrowing terminates if  $\mathcal{R}$  satisfies Réty's maximal commutation property and all basic narrowing derivations issuing from the rhs's of the rules terminate.

## Getting rid of left-linearity

From the above corollary, we distill a practical termination criterion in two steps

First, we get rid of left-linearity by generalizing the class of TRSs where *Réty's normalization condition* holds

### Definition

A TRS is *rnf-based* if the arguments of the lhs's of all rules are *rigid normal forms* (i.e., unnarrowable)

The class of rnf-based TRSs subsume (typical functional programs):

- left-linear CSs
- almost orthogonal TRSs

## Getting rid of right-linearity

Second, we get rid of right-linearity by generalizing together the *rnf-based TRSs* and *left-flat TRSs*

### Definition

- *left-plain TRSs*: every non-ground strict subterm of the lhs's is a rnf
- *right-rnf TRSs*: all rhs's are rigid normal forms (i.e., unarrowable)
- *reachability-complete TRSs*: those where narrowing is strongly reachability-complete

(the inspiration for naming this class comes from the “complete TRSs”, a synonymous for canonical TRSs because narrowing is unification-complete in them)

## Corollary [Termination of narrowing for right-rnf TRSs]

**Narrowing** terminates in any **right-rnf** TRS which is either

- 1 **right-linear**,
- 2 **confluent and left-plain**, or
- 3 **topmost**

\*\* in case (1), for linear input terms

This result is very handy since

- it is (almost) syntactical, and does not resort to termination orderings
- it dispenses with linearity in some cases
- it applies to TRSs that are not purely left-flat or rnf-based
- it ensures reachability completeness as well

## Some final examples

### Example

- Note that  $\mathcal{R}_{\text{add}}$  satisfies all the criteria, except for right-rnf
- Also note that [Christian's example](#) is right-rnf and right linear (but the input call was not linear)

### Example

Exponentiation function (used as a primitive operation for key exchange in the Diffie-Hellman key agreement protocol) where  $*$  and  $g$  are constructors

$$\text{exp}(\text{exp}(g, y), z) \rightarrow \text{exp}(g, y * z)$$

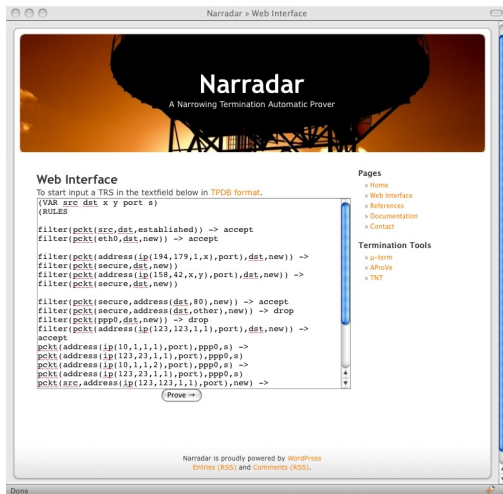
This rule satisfies both criteria (1) and (2), hence narrowing terminates

- 1 Classical termination of narrowing: Christian and Hullot
- 2 Extending Hullot's termination criterion
- 3 Getting rid of canonicity and linearity
- 4 The Narradar Termination Tool**



# The Narradar Termination Tool. Enjoy!

<http://safe-tools.dsic.upv.es/narradar>



Narradar » Web Interface

## Narradar

A Narrowing Termination Automatic Prover

### Web Interface

To start input a TRS in the textfield below in **TPDB format**.

```
(VAR src dst x y port s)
(RULES
  filter(pkt(src,dst,established)) -> accept
  filter(pkt(eth0,dst,new)) -> accept

  filter(pkt(address(ip(194,179,1,x),port),dst,new)) ->
  filter(pkt(address(ip(158,42,x,y),port),dst,new)) ->
  filter(pkt(secure,dst,new))

  filter(pkt(secure,address(dst,80),new)) -> accept
  filter(pkt(secure,address(dst,other),new)) -> drop
  filter(pkt(ppp0,dst,new)) -> drop
  filter(pkt(address(ip(123,123,1,1),port),dst,new)) ->
  accept
  pkt(address(ip(10,1,1,1),port),ppp0,s) ->
  pkt(address(ip(123,23,1,1),port),ppp0,s)
  pkt(address(ip(10,1,1,2),port),ppp0,s) ->
  pkt(address(ip(123,23,1,1),port),ppp0,s)
  pkt(src,address(ip(123,123,1,1),port),new) ->
```

Narradar is proudly powered by [WordPress](#)  
[Entries \(RSS\)](#) and [Comments \(RSS\)](#).

Done

**Pages**

- » Home
- » Web Interface
- » References
- » Documentation
- » Contact

**Termination Tools**

- »  $\mu$ -term
- » AProVe
- » TNT

## Additional results

Narradar is also based on additional results regarding modularity and mechanization of termination proofs by using dependency pairs.



Alpuente, M., Escobar, S., and Iborra, J. (2008a).  
Modular Termination of Basic Narrowing.  
In Proc. RTA 2008, Springer LNCS 5117:1–16.



Alpuente, M., Escobar, S., and Iborra, J. (2008b).  
Termination of Narrowing using Dependency Pairs.  
In Proc. ICLP 2008, Springer LNCS 5366: 317-33.

# Wrap up

Restrictions on $\mathcal{R}$	Reference
<b>LF + cT</b>	<b>(Christian, 1992)</b>
<b>RL + (LL or Co) + NC + R-bnT</b> R-rnf + L + rnf-B	<b>(Hullot's result generalized)</b> e.g. <u>R-rnf + L + CS</u>
<b>RL + R-rnf (+linear term)</b>	
<b>LP + RC + R-rnf</b> R-rnf + LP + C R-rnf + Tp	e.g. <u>R-rnf + (either aO or CS + C)</u>
<b>RL + (LL or Co) + NC + St</b>	<b>using (Nieuwenhuis, 1996)</b>

## Legend

C	confluent	LL	left-linear	RL	right-linear
Tp	topmost	Co	conservative	CS	constructor system
R-rnf	right-rnf	rnf-B	rnf-based	LP	left-plain
LF	left-flat	L	linear	aO	almost Orthogonal
bnT	basic narrowing terminates	NC	Rety's normalization condition		
R-bnT	all basic narrowing derivations starting from rule rhs's terminate				
St	standard theories saturated by basic paramodulation				
cT	compatible with a termination ordering				

Figure: Summary of criteria for narrowing termination

# Conclusion

## Main results

- i We extend to narrowing Hullot's **b.n.** theorem and drop canonicity
- ii By considering the class of *strongly reachability-complete* TRSs, we prove termination in a number of TRSs where neither canonicity nor linearity are assumed.
- iii We developed an automated termination prover that is publicly available

One plus: some of our results are based (and hence preserve) reachability completeness, besides ensuring termination

Thank you!



The Valencian School of Giorgio Levi