

Context-Sensitive Rewriting in Programming

Salvador Lucas

Departamento de Sistemas Informáticos y Computación (DSIC)
Universidad Politécnica de Valencia (UPV)

<http://www.dsic.upv.es/users/elp/slucas.html>

Introduction

Consider a function call $f(t_1, \dots, t_k)$

- A *lazy* strategy evaluates a given t_i , $1 \leq i \leq k$ if *necessary*.
 - (+) Improves termination. Unwasteful.
 - (−) Implementation is complex.
- An *eager* strategy first evaluates *each* t_i , $1 \leq i \leq k$.
 - (+) Easy to implement (and understand).
 - (−) Non-termination.

Introducing context-sensitive rewriting (*CSR*)

Given a function call $f(t_1, \dots, t_k)$ we (only) evaluate the arguments indicated by $\mu(f) \subseteq \{1, \dots, k\}$.

Example :

`if(true,x,y) → x`

`if(false,x,y) → y`

Given a call

`if(cond,exp,exp')`

we avoid reductions on both *exp* and *exp'* if $\mu(\text{if}) = \{1\}$.

Introducing context-sensitive rewriting (*CSR*)

The following TRS can be used to arbitrarily approximate (by evaluating $\text{first}(n, \text{terms}(1))$) the value of $\pi^2/6$ (Lucas [IC'02]):

$\text{sqr}(0) \rightarrow 0$	$0 + x \rightarrow x$
$\text{sqr}(s(x)) \rightarrow s(\text{sqr}(x) + \text{dbl}(x))$	$s(x) + y \rightarrow s(x+y)$
$\text{dbl}(0) \rightarrow 0$	$\text{first}(0, x) \rightarrow []$
$\text{dbl}(s(x)) \rightarrow s(s(\text{dbl}(x)))$	$\text{first}(s(x), y:z) \rightarrow y:\text{first}(x, z)$
$\text{terms}(n) \rightarrow \text{recip}(\text{sqr}(n)):\text{terms}(s(n))$	

with $\mu(:) = \{1\}$ and $\mu(f) = \{1, \dots, k\}$ for any other k -ary symbol f .

Is it ‘terminating’? Can we actually obtain the first n terms of the series that approximates $\pi^2/6$?

Challenges

- ① How to lift the idea to positions within terms?
- ② Can we actually improve termination in this way?
- ③ Do we keep computational power?
- ④ Are we introducing wasteful computations?
- ⑤ How can be found a convenient μ ?

Summary

- ① Basic description of *CSR*
- ② Computational properties of *CSR*: Confluence and termination
- ③ Completeness of computations (to head-normal forms, values, (infinite) normal forms)
- ④ Context-sensitive strategies
- ⑤ *CSR* in programming: OBJ languages
- ⑥ Extensions

Replacement maps

Definition A mapping $\mu : \mathcal{F} \rightarrow \mathcal{P}(\mathbb{N})$ such that, for every k -ary $f \in \mathcal{F}$, $\mu(f) \subseteq \{1, \dots, k\}$ is called a **replacement map** or \mathcal{F} -map (Lucas [JFLP'98]).

Ordering \sqsubseteq on $M_{\mathcal{F}}$, the set of all \mathcal{F} -maps.

$$\mu \sqsubseteq \mu' \quad \text{iff} \quad \forall f \in \mathcal{F}, \mu(f) \subseteq \mu'(f)$$

Replacing positions

The set of replacing positions is given by:

$$\mathcal{P}os^\mu(x) = \{\epsilon\} \quad \text{if } x \in \mathcal{X}$$

$$\mathcal{P}os^\mu(f(\tilde{t})) = \{\epsilon\} \cup (\cup_{i \in \mu(f)} i. \mathcal{P}os^\mu(t_i))$$

Example Consider the TRS:

$$\begin{array}{llll} \text{if}(\text{true}, x, y) \rightarrow x & 0+x \rightarrow x & \text{and}(\text{true}, x) \rightarrow x & \\ \text{if}(\text{false}, x, y) \rightarrow y & s(x)+y \rightarrow s(x+y) & \text{and}(\text{false}, y) \rightarrow \text{false} & \end{array}$$

with $\mu(\text{if}) = \mu(s) = \mu(\text{and}) = \mu(+) = \{1\}$.

For $t = \text{if}(\text{and}(\text{true}, \text{false}), s(0)+s(0), 0)$, we have

$$\mathcal{P}os^\mu(t) = \{\epsilon, 1, 1.1\} \quad \text{and} \quad \mathit{MRC}^\mu(t) = \text{if}(\text{and}(\text{true}, \square), \square, \square)$$

Context-sensitive rewriting

Let $\mathcal{R} = (\mathcal{F}, R)$ be a TRS, and μ be a \mathcal{F} -map. In *CSR*, we only rewrite replacing redexes: t μ -rewrites to s , written

$$t \hookrightarrow_{\mathcal{R}(\mu)} s,$$

if $t \xrightarrow{p}_{\mathcal{R}} s$ and $p \in \mathcal{P}os^{\mu}(t)$.

Example We can perform the following μ -rewriting step

$$\text{if}(\underline{\text{and}(\text{true}, \text{false})}, s(0)+s(0), 0) \hookrightarrow \text{if}(\text{false}, s(0)+s(0), 0)$$

but the following one is disallowed

$$\text{if}(\text{and}(\text{true}, \text{false}), \underline{s(0)+s(0)}, 0) \not\hookrightarrow \text{if}(\text{and}(\text{true}, \text{false}), s(0+s(0)), 0)$$

Termination of *CSR*

A TRS is μ -terminating if there is no infinite μ -reduction sequence

$$t_1 \hookrightarrow t_2 \hookrightarrow \dots \hookrightarrow t_n \hookrightarrow \dots$$

Trivially, termination implies μ -termination.

Example Consider the nonterminating TRS \mathcal{R} :

$$\begin{array}{ll} \text{first}(0, x) & \rightarrow [] & \text{from}(x) \rightarrow x:\text{from}(s(x)) \\ \text{first}(s(x), y:z) & \rightarrow y:\text{first}(x, z) \end{array}$$

and

$$\mu(\cdot) = \mu(\text{from}) = \mu(s) = \{1\}, \quad \mu(\text{first}) = \{1, 2\}$$

Every μ -rewriting sequence terminates (must be proved!).

Termination of *CSR*: proof methods

- ① μ -reduction orderings
- ② Transformations
- ③ Direct methods
- ④ Combined methods

Termination of *CSR*: orderings (Zantema [RTA97])

A TRS $\mathcal{R} = (\mathcal{F}, R)$ is μ -terminating if and only if there exists a well-founded, stable and μ -monotonic ordering $>$ (i.e., a μ -reduction ordering) such that

$$l > r \quad \text{for all rule } l \rightarrow r \in \mathcal{R}$$

where μ -monotonic means that, for all $f \in \mathcal{F}$, $i \in \mu(f)$, and $t, s, t_1, \dots, t_k \in \mathcal{T}(\mathcal{F}, \mathcal{X})$,

$$t > s \quad \text{implies} \quad f(t_1, \dots, t_{i-1}, t, \dots, t_k) > f(t_1, \dots, t_{i-1}, s, \dots, t_k)$$

Termination of *CSR*: transformations

The μ -termination of a TRS \mathcal{R} is demonstrated by proving termination of a TRS $\mathcal{R}_{\Theta}^{\mu}$ for a given transformation Θ :

- ① Lucas [ICALP'96]
- ② Zantema [RTA'97]
- ③ Ferreira and Ribeiro [RTA'99]
- ④ Giesl and Middeldorp (gave a complete transformation) [RTA'99]

Termination of *CSR*: transformations

Lucas' transformation [ICALP'96]: remove all non-replacing subterms from the rules of the TRS \mathcal{R} .

Example From

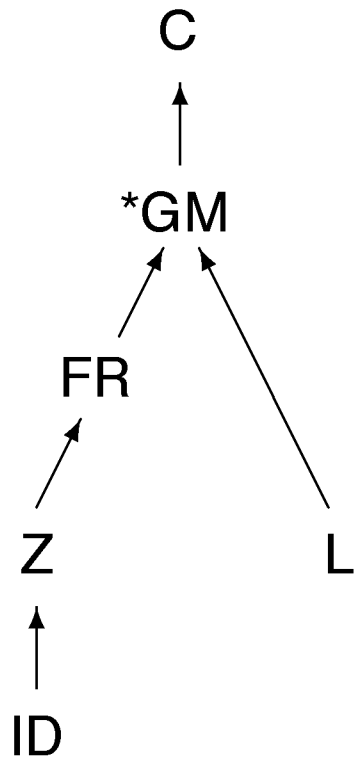
$$\begin{array}{ll} \text{first}(0, x) & \rightarrow [] & \text{from}(x) \rightarrow x:\text{from}(s(x)) \\ \text{first}(s(x), y:z) & \rightarrow y:\text{first}(x, z) \end{array}$$

and $\mu(:) = \mu(\text{from}) = \mu(s) = \{1\}$, $\mu(\text{first}) = \{1, 2\}$, we get

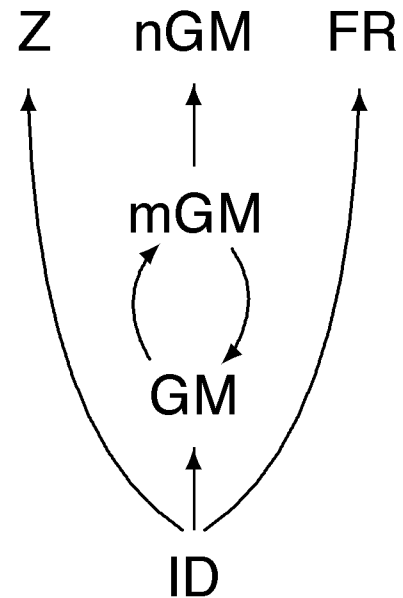
$$\begin{array}{ll} \text{first}(0, x) & \rightarrow [] & \text{from}(x) \rightarrow :(x) \\ \text{first}(s(x), :(y)) & \rightarrow :(y) \end{array}$$

which is terminating (use an *rpo* with $\text{first} > []$ and $\text{from} > :$).

Termination of *CSR*: transformations



Termination [RTA'99]



Simple termination [RTA'02]

Termination of *CSR*: direct methods

- ① CS recursive path ordering (Borralleras, Lucas, and Rubio [CADE'02])
- ② Polynomial orderings (Gramlich and Lucas [Draft'02])
- ③ CS Knuth-Bendix ordering (Borralleras [PhD'02])
- ④ Modular approach (Gramlich and Lucas [PPDP'02])

Termination of *CSR*: direct methods

Consider the TRS \mathcal{R} :

$$f(a, b, x) \rightarrow f(x, x, x) \quad c \rightarrow a \quad c \rightarrow b$$

with $\mu(f) = \{3\}$. The μ -termination of \mathcal{R} cannot be proved by using transformations (or CSRPO)!

A (μ -compatible) polynomial interpretation can be used: Let $f_{\mathbb{N}}(x, y, z) = 1 + (x - y)^2 + z$, $c_{\mathbb{N}} = 3$, $a_{\mathbb{N}} = 2$, and $b_{\mathbb{N}} = 1$. Then,

$$\begin{aligned} [f(a, b, x)] &= 2 + x > 1 + x = [f(x, x, x)] \\ [c] &= 3 > 2 = [a] \\ [c] &= 3 > 1 = [b] \end{aligned}$$

The modular approach also works! (see Gramlich and Lucas [PPDP'02])

Confluence of *CSR*

Nontrivial. Sharp differences with unrestricted rewriting.

Fortunately, standard results:

- ① *'Terminating TRSs having joinable critical pairs are confluent'* [Hue80].
- ② *'Orthogonal TRSs are confluent'* [HL91].

easily generalize to *CSR* after imposing some additional conditions and considering the corresponding notions (e.g., μ -termination, etc.).

Not really necessary for achieving semantically meaningful computations (standard confluence suffices, see below), . . .

Completeness of *CSR*

Consider the following TRS \mathcal{R} :

$$f(x, y) \rightarrow g(x, y)$$

$$g(b, b) \rightarrow b$$

with $\mu(f) = \mu(g) = \{1\}$. Consider the term $f(b, a)$, and derivations

$$\underline{f(b, a)} \hookrightarrow g(b, a)$$

$$\underline{f(b, a)} \rightarrow g(b, \underline{a}) \rightarrow \underline{g(b, b)} \rightarrow b$$

CSR is not able to obtain a *head-normal form*!

Completeness of *CSR*

If we want to compute ‘interesting’ information:

- ① head-normal forms,
- ② (infinite) values,
- ③ (infinite) normal forms

we need to make the *cs*-restrictions compatible with unrestricted computations:

$$l = g(b, b) \quad \text{with } \mu(g) = \{1\}$$

$$t = g(b, \boxed{a}) \quad \text{Matching impossible!}$$

We should let $\mu(g) = \{1, 2\}$.

Canonical replacement map

The *canonical replacement map* $\mu_{\mathcal{R}}^{can}$ for a TRS \mathcal{R} is [JFLP'98]:

the most restrictive replacement map ensuring that the non-variable subterms of the left-hand sides of the rules of \mathcal{R} are replacing.

Formally, $\forall f \in \mathcal{F}, i \in \{1, \dots, ar(f)\}$,

$$i \in \mu_{\mathcal{R}}^{can}(f) \quad \text{iff} \quad \exists l \in L(\mathcal{R}), p \in Pos_{\mathcal{F}}(l), (root(l|_p) = f \wedge p.i \in Pos_{\mathcal{F}}(l))$$

Let $CM_{\mathcal{R}} = \{\mu \in M_{\mathcal{R}} \mid \mu_{\mathcal{R}}^{can} \sqsubseteq \mu\}$ be the set of replacement maps which are less or equally restrictive than $\mu_{\mathcal{R}}^{can}$.

Canonical replacement map

Consider the TRS \mathcal{R} :

$$\begin{aligned} \text{first}(0, x) &\rightarrow [] & \text{from}(x) &\rightarrow x:\text{from}(s(x)) \\ \text{first}(s(x), y:z) &\rightarrow y:\text{first}(x, z) \end{aligned}$$

we have

- $1 \in \mu_{\mathcal{R}}^{\text{can}}(\text{first})$ because, e.g., $\text{first}(0, x)|_1 = 0 \notin \mathcal{X}$; and
- $2 \in \mu_{\mathcal{R}}^{\text{can}}(\text{first})$ because $\text{first}(s(x), y:z)|_2 = y:z \notin \mathcal{X}$.

Therefore,

$$\mu_{\mathcal{R}}^{\text{can}}(\text{first}) = \{1, 2\} \quad \text{and} \quad \mu_{\mathcal{R}}^{\text{can}}(s) = \mu_{\mathcal{R}}^{\text{can}}(:) = \mu_{\mathcal{R}}^{\text{can}}(\text{from}) = \emptyset$$

Left-linearity matters!

Consider the TRS \mathcal{R} :

$$f(x, x) \rightarrow c(x)$$

$$a \rightarrow b$$

with $\mu(f) = \{1, 2\}$ and $\mu(c) = \emptyset$. Note that

$$f(c(a), c(b))$$

is a μ -normal form, but it is **not** a head-normal form!

$$f(c(\underline{a}), c(b)) \rightarrow \underline{f(c(b), c(b))} \rightarrow c(b)$$

Computing (head-)normal forms

Theorem [JFLP'98] Let \mathcal{R} be a left-linear TRS and $\mu \in CM_{\mathcal{R}}$. Every μ -normal form is a head-normal form.

Corollary [JFLP'98, IC'02] Let \mathcal{R} be a left-linear TRS and $\mu \in CM_{\mathcal{R}}$. Every μ -normalizing term is head-normalizing.

Theorem [IC'02] Let \mathcal{R} be a left-linear TRS and $\mu \in CM_{\mathcal{R}}$. If $t \rightarrow^! s$, then $t \hookrightarrow_{\mu}^! t' \rightarrow^! s$ for some term t' .

Corollary [IC'02] Let \mathcal{R} be a left-linear TRS and $\mu \in CM_{\mathcal{R}}$. Every normalizing term is μ -normalizing.

Computing infinite normal forms

A TRS is **infinitary normalizing** if every term t admits a *strongly convergent sequence* (i.e., a sequence that, ultimately, reduces deeper and deeper redexes) starting from t and ending into a (possibly infinite) normal form.

A TRS is **top-terminating** if no infinitary reduction sequence performs infinitely many rewrites at topmost position (Dershowitz et al. [TCS'91]).

The following TRS \mathcal{R} :

$$f(a) \rightarrow f(f(a)) \qquad f(a) \rightarrow a$$

is infinitary normalizing but not top-terminating:

$$\underline{f(a)} \rightarrow \underline{f(f(a))} \rightarrow \underline{f(a)} \rightarrow \dots$$

Computing infinite normal forms

Top-terminating TRSs *only* admit strongly convergent sequences!

		SEQUENCES	
		Finite	Infinite
TRSs	Normalizing		Inf. normalizing
	Terminating		Top-terminating

Theorem [RTA'02] Let \mathcal{R} be a left-linear TRS and $\mu \in CM_{\mathcal{R}}$. If \mathcal{R} is μ -terminating, then \mathcal{R} is top-terminating.

CS-strategies

A *one-step* μ -strategy is a function H that assigns a **non empty** set $H(t)$ of **replacing** redex positions to **every** μ -reducible term t [IC'02].

Every one-step μ -strategy H extends to a one-step strategy S_H as follows:

$$S_H(t) = \begin{cases} H(t) & \text{if } t \text{ is not a } \mu\text{-normal form} \\ \cup_{1 \leq i \leq n} p_i \cdot S_H(t_i) & \text{otherwise, where:} \\ & C[] = \mathit{MRC}^\mu(t), t = C[t_1, \dots, t_n], \\ & \text{and } t_i = t|_{p_i} \text{ for } 1 \leq i \leq n \end{cases}$$

Normalization via μ -normalization

A μ -strategy H is μ -normalizing if, for all μ -normalizing term t , there is no infinite H -sequence starting from t .

Theorem (Normalization via μ -normalization [IC'02]) Let \mathcal{R} be a left-linear, confluent TRS and $\mu \in CM_{\mathcal{R}}$. If H is μ -normalizing, then S_H is normalizing.

Every μ -strategy is μ -normalizing for μ -terminating TRSs. Alternative methods for defining μ -normalizing μ -strategies for non- μ -terminating TRSs are discussed in [IC'02].

Normalization via μ -normalization

We can obtain the first two terms of the infinite series `terms(1)` by evaluating the expression `first(dbl(1), terms(1))` using a restricted leftmost-outermost $\mu_{\mathcal{R}}^{can}$ -strategy H_{lo} :

$$\begin{aligned} \text{first}(\underline{\text{dbl}(1)}, \text{terms}(1)) &\hookrightarrow_{H_{lo}} \text{first}(\text{s}(\text{s}(\text{dbl}(0))), \underline{\text{terms}(1)}) \\ &\hookrightarrow_{H_{lo}} \underline{\text{first}(\text{s}(\text{s}(\text{dbl}(0))), \text{recip}(\text{sqr}(1)) : \text{terms}(2))} \\ &\hookrightarrow_{H_{lo}} \text{recip}(\text{sqr}(1)) : \text{first}(\text{s}(\text{dbl}(0)), \text{terms}(2)) \end{aligned}$$

At this point, the $\mu_{\mathcal{R}}^{can}$ -strategy H_{lo} stops yielding a $\mu_{\mathcal{R}}^{can}$ -normal form ...

CS-strategies

... but we can continue with $S_{H_{lo}}$:

$\text{recip}(\underline{\text{sqr}(1)}): \text{first}(\text{s}(\text{dbl}(0)), \text{terms}(2))$

$\rightarrow_{S_{H_{lo}}} \text{recip}(\text{s}(\underline{\text{sqr}(0)+\text{dbl}(0)})): \text{first}(\text{s}(\text{dbl}(0)), \text{terms}(2))$

$\rightarrow_{S_{H_{lo}}} \text{recip}(\text{s}(\underline{0+\text{dbl}(0)})): \text{first}(\text{s}(\text{dbl}(0)), \text{terms}(2))$

$\rightarrow_{S_{H_{lo}}} \text{recip}(\text{s}(\underline{\text{dbl}(0)})): \text{first}(\text{s}(\text{dbl}(0)), \text{terms}(2))$

$\rightarrow_{S_{H_{lo}}} \text{recip}(1): \text{first}(\text{s}(\text{dbl}(0)), \underline{\text{terms}(2)})$

$\rightarrow_{S_{H_{lo}}} \text{recip}(1): \underline{\text{first}(\text{s}(\text{dbl}(0)), \text{recip}(\text{sqr}(2))): \text{terms}(3)}$

$\rightarrow_{S_{H_{lo}}} \text{recip}(1): \text{recip}(\underline{\text{sqr}(2)}): \text{first}(\text{dbl}(0), \text{terms}(3))$

$\rightarrow_{S_{H_{lo}}} \text{recip}(1): \text{recip}(\text{s}(\underline{\text{sqr}(1)+\text{dbl}(1)})): \text{first}(\text{dbl}(0), \text{terms}(3))$

$\rightarrow_{S_{H_{lo}}} \text{recip}(1): \text{recip}(\text{s}(\underline{\text{s}(\text{sqr}(0)+\text{dbl}(0))+\text{dbl}(1)})): \text{first}(\text{dbl}(0), \text{terms}(3))$

$\rightarrow_{S_{Hlo}}$ recip(1):recip(s(s(sqr(0)+dbl(0)+dbl(1)))):first(dbl(0),terms(3))
 $\rightarrow_{S_{Hlo}}$ recip(1):recip(s(s(0+dbl(0)+dbl(1)))):first(dbl(0),terms(3))
 $\rightarrow_{S_{Hlo}}$ recip(1):recip(s(s(dbl(0)+dbl(1)))):first(dbl(0),terms(3))
 $\rightarrow_{S_{Hlo}}$ recip(1):recip(s(s(0+dbl(1)))):first(dbl(0),terms(3))
 $\rightarrow_{S_{Hlo}}$ recip(1):recip(s(s(dbl(1)))):first(dbl(0),terms(3))
 $\rightarrow_{S_{Hlo}}$ recip(1):recip(s(s(s(s(dbl(0)))))):first(dbl(0),terms(3))
 $\rightarrow_{S_{Hlo}}$ recip(1):recip(4):first(dbl(0),terms(3))
 $\rightarrow_{S_{Hlo}}$ recip(1):recip(4):first(0,terms(3))
 $\rightarrow_{S_{Hlo}}$ recip(1):recip(4): []

The expected result $[1, \frac{1}{4}]$ is obtained without any risk of nontermination.

Normalizing strategies for overlapping TRSs

Consider the TRS \mathcal{R} [IC'02]:

$$\text{sqr}(0) \rightarrow 0$$

$$0 + x \rightarrow x$$

$$\text{sqr}(s(x)) \rightarrow s(\text{sqr}(x) + \text{dbl}(x))$$

$$s(x) + y \rightarrow s(x+y)$$

$$\text{dbl}(0) \rightarrow 0$$

$$\text{first}(0, x) \rightarrow []$$

$$\text{dbl}(s(x)) \rightarrow s(s(\text{dbl}(x)))$$

$$\text{first}(s(x), y:z) \rightarrow y:\text{first}(x, z)$$

$$\text{half}(0) \rightarrow 0$$

$$\text{half}(s(s(x))) \rightarrow s(\text{half}(x))$$

$$\text{half}(s(0)) \rightarrow 0$$

$$\text{half}(\text{dbl}(x)) \rightarrow x$$

$$\text{terms}(n) \rightarrow \text{recip}(\text{sqr}(n)):\text{terms}(s(n))$$

No existing results describing normalizing strategies for left-linear (possibly overlapping) TRSs apply to \mathcal{R} (!).

Normalizing strategies for overlapping TRSs

Even though \mathcal{R} contains non-trivial critical pairs, it can be proved confluent. Moreover, \mathcal{R} can be proved $\mu_{\mathcal{R}}^{can}$ -terminating.

Hence, according to our results,

every $\mu_{\mathcal{R}}^{can}$ -strategy H can be used to obtain a normalizing strategy S_H for \mathcal{R} as given above.

Strategy annotations in OBJ languages

Strategy annotations are used in the OBJ family of programming languages to improve efficiency and achieve termination.

obj EXAMPLE is

...

op cons : Nat LNat -> LNat [strat (1 0)] .

...

eq sel(s(X),cons(Y,Z)) = sel(X,Z) .

eq sel(0,cons(X,Z)) = X .

eq from(X) = cons(X,from(s(X))) .

endo

Strategy annotations in OBJ languages

The *specified* strategy annotation for the list constructor `cons` disables replacements on the second argument.

Strategy annotations of an OBJ program P can be modeled using a replacement map μ_P (the information about the order of evaluation for the arguments gets lost).

Theorem (Termination of OBJ programs [PPDP'01,LPAR'01]) An OBJ program P is terminating if the underlying TRS \mathcal{R}_P is μ_P -terminating.

In general, *CSR* can be thought as a suitable framework for modeling and analyzing OBJ computations (see Lucas [PPDP'01,LPAR'01]).

Extensions of *CSR*

Some extensions of *CSR* have been investigated:

- ① Narrowing (Lucas [JFLP'98])
- ② AC-*CSR* (Ferreira and Ribeiro [RTA'99], Giesl and Middeldorp [TR'02])
- ③ Innermost *CSR* (Lucas [LPAR'01], Giesl and Middeldorp [TR'02b], Gramlich and Lucas [PPDP'02])
- ④ On-demand rewriting (Lucas [PPDP'01, WFLP'01])
- ⑤ On-demand *CSR* (Nakamura and Ogata [TR00])
- ⑥ Lazy Rewriting (Fokkink et al. [TOPLAS'00], Lucas [ENTCS vol. 64])