Technical Report: TR-95-20, December 01, 1995

Rewriting modulo a rewrite system

Viry, Patrick

Dipartimento di Informatica, Università di Pisa
corso Italia 40, 56125 Pisa, Italy

Abstract

We introduce rewriting with two sets of rules, the first interpreted equationally and the second not. A semantic view considers equational rules as defining an equational theory and reduction rules as defining a rewrite relation modulo this theory. An operational view considers both sets of rules as similar. We introduce sufficient properties for these two views to be equivalent (up to different notions of equivalence). The paper ends with a collection of example showing the effectiveness of this approach.

ACM–CR Subject Classification:

Keyword and phrases:

UNIVERSITÀ DEGLI STUDI DI PISA
DIPARTIMENTO DI INFORMATICA

Technical Report TR-20/95

Rewriting modulo a rewrite system

Patrick Viry*

Abstract. We introduce rewriting with two sets of rules, the first interpreted equationally and the second not. A semantic view considers equational rules as defining an equational theory and reduction rules as defining a rewrite relation modulo this theory. An operational view considers both sets of rules as similar. We introduce sufficient properties for these two views to be equivalent (up to different notions of equivalence). The paper ends with a collection of example showing the effectiveness of this approach.

December 1995

* Stay in Pisa supported by an HCM fellowship, EuroFOCS network.

Dipartimento di Informatica – Università di Pisa – Corso Italia, 40 – 56100 Pisa – Italy
Tel. +39 50 887001 – Fax. +39 50 887226 – Tlx. 590291 DIPISAI – Email. di.unipi.it

Rewriting modulo a rewrite system

Patrick Viry

Dipartimento di Informatica, Università di Pisa
Corso Italia 40, 56100 Pisa, Italy
Email: viry@di.unipi.it

Rewriting can be viewed simultaneously as the most basic symbol-manipulating method, and as a very expressive specification framework, given the expressive power of rewriting modulo equations. It is a primary candidate to the role of a general logical framework [Mes92, MOM93].

Historically, rewriting has been given an equational semantics, saying that a rewrite rule $u \longrightarrow v$ is interpreted as u is equal to v. This is the case for instance when defining functions or solving the word problem in a equational theory. But rewriting is also useful for specifying non equational relations, such as transitions between states or deduction steps, in which case it is solely interpreted as reduction. However, a naive combination of these two kinds of semantics does not work, as shown by the non-deterministic choice example:

$$\begin{array}{ll} 0 + x \longrightarrow x & ? \longrightarrow 0 \\ s(x) + y \longrightarrow s(x + y) & ? \longrightarrow 1 \end{array}$$

The set of rules on the left specify addition in Peano arithmetics, and one would like to give it an equational interpretation ($1 + 2$ is *equal* to 3). However, extending the equational interpretation to the two rules on the right, which specify a non-deterministic choice operator, would allow to deduce $0 = 1$, and make the whole algebra collapse.

The solution suggested by the work on rewriting logic is to keep all rules with an equational interpretation as a set E of non-oriented equations, and consider the remaining rules as defining rewrite steps over classes modulo E (E can be thought of as defining a structure and R as computation over that structure). This is satisfactory on a semantic point of view, but disastrous on an operational one, since rewriting was precisely designed originally as a mean to handle such arbitrary big equational theories!

We propose here an extension to rewriting logic allowing for two kinds of rules, *equational rules* interpreted equationally and *reduction rules* not necessarily. The equational rules are given an equational semantics, but still have to be used only according to their orientation.

The central notion appearing in this framework is the property of *coherence*, which ensures a correspondance between operational behaviour and intended semantics. We show how to check coherence, and possibly obtain it through coherence completion. An important subcase is when the reduction rules also have an equational semantics. In this case, we show how coherence completion corresponds to completion modulo a rewrite system.

The paper ends with a collection of examples showing the relevance of this approach.

The ideas presented here originate in [Vir94], where the simplest case of coherence (called here strong coherence) was considered. Strong coherence is a very restrictive property, and the other notions introduced here make similar techniques applicable to a much broader set of applications, as examplified by new examples. Definitions and proofs have been made simpler and more general through the use of a generic permutation lemma (for instance, the usual technique of building in associativity by rewriting flat terms now fits in the framework). Finally, the notion of oriented matching has been introduced in order to implement efficiently some subsets of theories.

1 Basic concepts

Relations. Given a binary relation $\longrightarrow$, we denote

$\longleftarrow$ its inverse
$\longrightarrow\!\!\!\!\!\rightarrow$ its transitive closure
$\longleftrightarrow$ its symmetric closure
$\leftrightarrow\!\!\!\!\!\!\rightarrow$ its transitive symmetric closure

The notation $\longrightarrow\!\!\!\!\!\rightarrow$ is equivalent to the usual $\longrightarrow^*$, but makes diagrams much more readable. An element u is a $\longrightarrow$-normal form if there is no v such that $u \longrightarrow v$. The normalising relation $\xrightarrow{!}$ is defined as $u \xrightarrow{!} v$ if $u \longrightarrow\!\!\!\!\!\rightarrow v$ and v is a $\longrightarrow$-normal form (note that $\xrightarrow{!}\!\!\!\!\!\rightarrow = \xrightarrow{!}$).

Given a rewrite system (a set of rules) $R = \{l_i \to r_i\}$, the rewrite relation $\xrightarrow{R}$ is the smallest relation stable by context and instanciation containing $l_i \xrightarrow{R} r_i$ for each $l_i \to r_i \in R$. For a set of equations $E = \{u_i = v_i\}$, the replacement relation $\xleftrightarrow{E}$ is defined similarly, and closed by symmetry. $[u]_E$ denotes the equivalence class of u modulo E, ie. $[u]_E$ is the set $\{v \mid v \xleftrightarrow{E}\!\!\!\!\!\rightarrow u\}$. Similarly, a set of rules R, when considered as equations, also defines equivalences classes $[u]_R$. For any relations $\xrightarrow{R}$ and $\xrightarrow{S}$, define

$$\xrightarrow{R/S} \;=\; \xrightarrow{S}\!\!\!\!\!\rightarrow \xrightarrow{R} \xrightarrow{S}\!\!\!\!\!\rightarrow$$

In particular, when $\xrightarrow{S}$ is the replacement relation $\xleftrightarrow{E}$ generated by a set E of equations, $\xrightarrow{R/E}$ is the basis for defining rewriting over classes modulo E :

$$[u]_E \xrightarrow{[R]_E} [v]_E \qquad \text{iff.} \qquad u \xrightarrow{R/E} v$$

We use the abbreviations $\xrightarrow{[R/S]_E}$ for $\xrightarrow{[R]_E/[S]_E} = \xrightarrow{[S]_E}\!\!\!\!\!\rightarrow \xrightarrow{[R]_E} \xrightarrow{[S]_E}\!\!\!\!\!\rightarrow$, and $\xrightarrow{[R\cup S]_E}$ for $\xrightarrow{[R]_E} \cup \xrightarrow{[S]_E}$.

Finally, because it makes definitions and diagrams simpler, instead of $\xrightarrow{[R/S]_E}$ we will often use the relation $\xrightarrow{[R(S)]_E}$ defined as

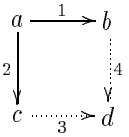
$$\xrightarrow{[R(S)]_E} \;=\; \xrightarrow{[S]_E}\!\!\!\!\!\rightarrow \xrightarrow{[R]_E}$$

$\xrightarrow{[S]_E}\!\!\!\!\!\rightarrow$ steps are then allowed only before the $\xrightarrow{[R]_E}$ steps, but we have the following correspondance :

$$\xrightarrow{[R/S]_E} = \xrightarrow{[R(S)]_E} \xrightarrow{[S]_E}\!\!\!\!\!\rightarrow$$

$$\text{and for derivations} \quad \xrightarrow{[R/S]_E}\!\!\!\!\!\rightarrow = \xrightarrow{[R(S)]_E}\!\!\!\!\!\rightarrow \xrightarrow{[S]_E}\!\!\!\!\!\rightarrow$$

Diagrams. A diagram is a way to express complex formulas about relations. Plain arrows are quantified universally and dotted arrows existentially. In a complex diagram, the numbers inside the smaller diagrams state in what order we prove them. For instance :



reads : for all a, b and c such that $a \xrightarrow{1} b$ and $a \xrightarrow{2} c$, there exists d such that $c \xrightarrow{3} d$ and $b \xrightarrow{4} d$.

2 Oriented rewrite theory

Extending the notion of a rewrite theory of rewriting logic [Mes92] , define an *oriented rewrite theory* as a tuple $\mathcal{O} = (\Sigma, A, E, ER, R)$, where

- Σ is the signature upon which terms are built
- A is a set of rule labels. They are not actually used here and will be omitted in the following.
- E is a set of *equations* $(u = v) \in T^2_{\Sigma,X}$

- ER is a set of *equational rules* $(d : u \xrightarrow{=} v) \in A \times T^2_{\Sigma,X}$
- R is a set of *reduction rules* $(d : u \longrightarrow v) \in A \times T^2_{\Sigma,X}$

Compared to a "classical" rewrite theory, this definition makes room for two kind of rules :

- Rules of ER are always given an equational interpretation : the oriented rewrite theory defines equivalence classes modulo $E \cup ER$. We will always suppose that the rules of ER form a rewrite system confluent modulo E (ie. $\overset{[ER]_E}{\longrightarrow}$ is confluent), but not necessarily terminating.
- Rules of R are not necessarily interpreted equationally. They may be interpreted for instance as transitions between states or deduction steps. However, nothing prevents giving an equational interpretation to the reduction rules as well, and this important case will be treated in section 2.6.
  When both R and ER are interpreted equationally, the choice of putting a rule in one or the other system will influence the *view* one can have of a derivation. We can think of the $\overset{[ER]_E}{\longrightarrow}$ steps as being *hidden*, as are hidden the $\overset{E}{\longleftrightarrow}$ steps in a derivation $\overset{R/E}{\longrightarrow\!\!\!\!\!\rightarrow}$.

In "classical" rewriting logic, the same *semantics* would be obtained by considering ER rules as non oriented equations. The idea behind the above definition is that even though rules of ER have an equational interpretation, they must be used only according to their orientation. Each of these considerations leads to a different view :

Operational view : "the rules of ER must be used according to their orientation". This leads to consider sequents for the rewrite theory $(E, ER \cup R)$:

$$[s]_E \overset{[R(ER)]_E}{\longrightarrow} [t]_E$$

Semantic view : "the rules of ER have an equational semantics". This leads to consider sequents for the rewrite theory $(E \cup ER, R)$:

$$[s]_{E \cup ER} \overset{[R]_{E \cup ER}}{\longrightarrow} [t]_{E \cup ER}$$

These two aspects are not equivalent. We have

$$[s]_E \overset{[R(ER)]_E}{\longrightarrow} [t]_E \quad \Longrightarrow \quad [s]_{E \cup ER} \overset{[R]_{E \cup ER}}{\longrightarrow} [t]_{E \cup ER}$$

but the opposite is not true in general.

In the following, we introduce different notions of *coherence*. Strong coherence is a sufficient condition for this implication to be an equivalence, but may be difficult to verify. Weaker notions of coherence correspond to weaker correspondances between $\overset{[R(ER)]_E}{\longrightarrow}$ and $\overset{[R]_{E \cup ER}}{\longrightarrow}$.

2.1 Coherence

Depending on how close a correspondance we seek between the operational and semantic views of oriented rewriting logic, we introduce three notions of coherence.

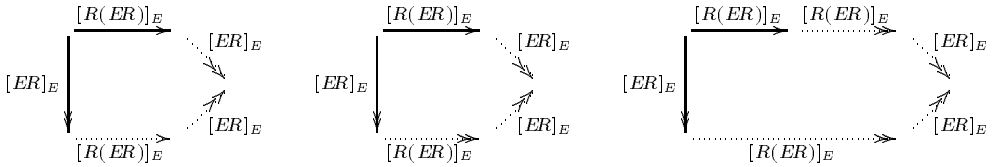
Strong coherence states that the exact number of steps of a derivation is preserved (one may wish also to impose that the same rule is used – or a rule generated by completion from the original one, see section 2.5 – the results extend straightforwardly to this case).

Weak coherence states that derivability is preserved, but not necessarily the intermediate steps.

Equational coherence states that the set of normal forms is preserved, except possibly for some cycles.

Equational coherence is the most general property, it can be checked by looking at critical pairs. Weak coherence may be more difficult to establish in the presence of non left-linear rules in R, and strong coherence in the presence of non right-linear rules as well.

Definition. $\overset{[R(ER)]_E}{\longrightarrow}$ is strong (1), weak (2) or equational (3) coherent if



(1) strong coherence (2) weak coherence (3) equational coherence

Properties (1) and (2) differ only in the number of steps allowed to close the diagram. It is also possible to imagine version of weak and equational coherence requiring at least one $[R(ER)]_E$ step, in order to preserve termination. The following results apply to this case as well.

Theorem 1. – *If* $\overset{[R(ER)]_E}{\longrightarrow}$ *is strong coherent, then*

$$[s]_{E\cup ER} \overset{[R]_{E\cup ER}}{\longrightarrow} [t]_{E\cup ER} \quad \Longrightarrow \quad [s]_E \overset{[R(ER)]_E}{\longrightarrow} [t']_E \overset{[ER]_E}{\longleftarrow\!\!\!\!\!\!\rightarrow} [t]_E$$

– *If* $\overset{[R(ER)]_E}{\longrightarrow}$ *is weak coherent, then*

$$[s]_{E\cup ER} \overset{[R]_{E\cup ER}}{\longrightarrow\!\!\!\!\!\!\rightarrow} [t]_{E\cup ER} \quad \Longrightarrow \quad [s]_E \overset{[R(ER)]_E}{\longrightarrow\!\!\!\!\!\!\rightarrow} [t']_E \overset{[ER]_E}{\longleftarrow\!\!\!\!\!\!\rightarrow} [t]_E$$

– *If* $\overset{[R(ER)]_E}{\longrightarrow}$ *is equational coherent, then*
if $[u]_E$ *is a normal form for* $\overset{[R(ER)]_E}{\longrightarrow}$*, then either* $[u]_{E\cup ER}$ *is a normal form for* $\overset{[R]_{E\cup ER}}{\longrightarrow}$*, either there is a non-empty cycle* $[u]_{E\cup ER} \overset{[R]_{E\cup ER}}{\longrightarrow\!\!\!\!\!\!\rightarrow} [u]_{E\cup ER}$

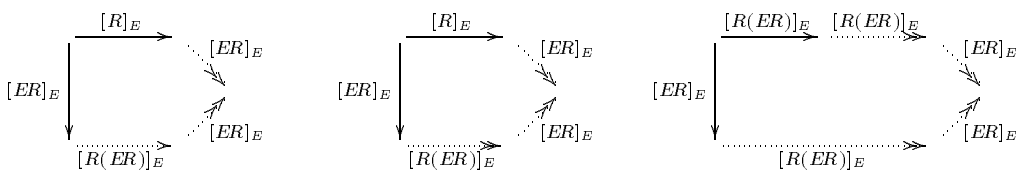
Proof: See appendix.

In the case of equational coherence, the set of normal forms is not exactly preserved, since a cycle in $\overset{[R]_{E\cup ER}}{\longrightarrow}$ may be matched by an empty derivation of $\overset{[R(ER)]_E}{\longrightarrow}$. This is the reason why in the identity rewriting example (section 4.1), a non-terminating relation modulo AC1 can be simulated by a terminating relation. If an exact correspondance between the sets of normal forms is required, the definition of equational coherence has to be adapted in order to close the diagram with at least one $\overset{[R(ER)]_E}{\longrightarrow}$ step.

2.2 Checking coherence

In order to be able to check coherence syntactically (looking only at the rewrite systems R and ER), we first define the local versions of strong and weak coherence :

Definition. $\overset{[R]_E}{\longrightarrow}$ is local strong (1), local weak (2) or local equational (3) coherent with $\overset{[ER]_E}{\longrightarrow}$ if



(1) local strong coherence (2) local weak coherence (3) local equational coherence

Then we boil down coherence to local coherence :

Theorem 2. *If* $\overset{[ER]_E}{\longrightarrow}$ *is terminating, then*

– $\xrightarrow{[R(ER)]_E}$ *is strong coherent if and only if* $\xrightarrow{[R]_E}$ *is local strong coherent with* $\xrightarrow{[ER]_E}$.
– $\xrightarrow{[R(ER)]_E}$ *is weak coherent if and only if* $\xrightarrow{[R]_E}$ *is local weak coherent with* $\xrightarrow{[ER]_E}$.

If $\xrightarrow{[R \cup ER]_E}$ *is terminating, then*

– $\xrightarrow{[R(ER)]_E}$ *is equational coherent if and only if* $\xrightarrow{[R]_E}$ *is local equational coherent with* $\xrightarrow{[ER]_E}$.

Proof: See appendix.

It remains an open question whether it is possible to drop the hypothesis of termination of $\xrightarrow{[R \cup ER]_E}$ is the case of equational coherence.

2.3 Checking local coherence

Now that equivalence between coherence and local coherence has been established, checking coherence reduces to ensuring that for all *critical peaks* $\xleftarrow{[ER]_E} \xrightarrow{[R]_E}$, the definition of local equational, local weak or local strong coherence is verified.

Checking all the possible critical peaks can be reduced to checking a finite number of *critical pairs* modulo E, for all theories E for which an instance of the generic permutation lemma is valid (see appendix, section A.1), together with the appropriate notion of critical pairs modulo E. Instances of the permutation lemma are known for the theories $E = \emptyset$, $E = \mathsf{A}$ (associativity) and $E = \mathsf{AC}$ (associativity and commutativity).

The permutation lemma distinguishes two cases. In the non superposition case, any critical peak can be closed according to the commutation properties of the underlying theory E. Checking local coherence in this case consist of verifying that the way the critical peak can be closed verifies the appropriate definition of local coherence.

In the superposition case, the permutation lemma ensures that there exists a critical pair modulo E, and local coherence is verified if it is true for all critical pairs.

– **Equational coherence.** The definition of equational coherence is verified in the non-superposition case.
– **Weak coherence.** The definition of weak coherence is not necessarily verified in the non-superposition case : no $\xrightarrow{[R(ER)]_E}$ derivation is allowed after the first $\xrightarrow{[R(ER)]_E}$ step.
  This is guaranteed by the generic permutation lemma if a non left-linear rule of ER cannot be applied above a rule of R. See some sufficient conditions below.
– **Strong coherence.** Similarly, the definition of strong coherence is not necessarily verified in the non-superposition case : the diagram has to be closed by exactly one $\xrightarrow{[R(ER)]_E}$ step.
  This is guaranteed by the permutation lemma if any rule of ER that can be applied above a rule of R is left-linear, right-linear and regular (the same set of variables appear in the left and right-hand sides).

There is no general way of ensuring the conditions about non linear rules, but we can however give sufficient conditions ensuring that such critical peaks will never happen.

The idea is to restrict the set of possible terms such that a rule of ER can never apply above a rule of R. For instance, separate "deterministic" and "non deterministic" function symbols, and impose sort conditions so that the former cannot appear above the latter.

This reduces somewhat the expressiveness, but nevertheless allows to handle the case of abstract data types : terms are built upon contructors and defined symbols, the constructors having to be "deterministic".

It also includes the example of LOTOS (see section 4.3) : the reduction rule can only be applied at the top of a process.

It is not possible to impose conditions on rules, except the very brutal one that forbids non linear rules in ER, since the kind of critical peaks that we have to avoid does not correspond to a superposition of left hand sides of rules.

Note that the notion of applying a rule "above" another has to be understood modulo E. Such a notion does not exist for arbitrary theories, but we know appropriate definitions for $E = \mathsf{A}$ and $E = \mathsf{AC}$ [Mar93].

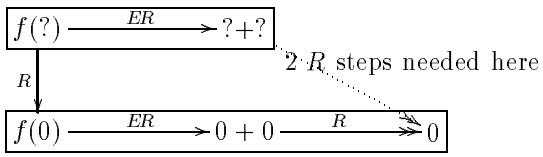
2.4 Examples of non-linearity

Let P be the classical convergent system specifying Peano arithmetics, ? a new constant, and

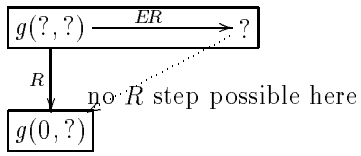
$$ER = P \cup \begin{cases} f(x) & \longrightarrow x + x \\ g(x,x) & \longrightarrow x \end{cases} \qquad R = \begin{cases} ? & \longrightarrow 0 \\ ? & \longrightarrow 1 \end{cases}$$

ER is convergent and there is no critical pair between R and ER.

Non Right-Linearity. There is an equational rewrite step $[f(?)]_{ER} \overset{[R]_{ER}}{\longrightarrow} [f(0)]_{ER}$, that we cannot simulate by a single rewrite step if we start from the representative $?+? \in [f(?)]_{ER}$. Two $\overset{R}{\longrightarrow}$ steps would be needed, hence $\overset{[R]_{ER}}{\longrightarrow}$ is not strong coherent :



Non Left-Linearity. Though there is an equational rewrite step $[g(?,?)]_{ER} \overset{[R]_{ER}}{\longrightarrow} [g(0,?)]_{ER}$, there is no derivation $? \overset{R}{\longrightarrow\!\!\!\!\!\rightarrow} t \in [g(0,?)]_{ER}$, hence $\overset{[R]_{ER}}{\longrightarrow}$ is not even weak coherent :



2.5 Coherence completion

If not all critical pairs are coherent, we can try to achieve coherence by adding new rules to R (Deduce). Note that the orientation of the new rule if fixed by the orientation of the rules already in R. We can also add some simplification rules (SimplL) and (SimplR) :

(Deduce)	$R \vdash R \cup \{u \longrightarrow v\}$	if	$(u,v) \in CP_E(R, ER)$
(SimplR)	$R \cup \{u \longrightarrow v\} \vdash R \cup \{u \longrightarrow v'\}$	if	$v \overset{ER(E)}{\longrightarrow\!\!\!\!\!\rightarrow} v'$
(SimplR)	$R \cup \{u \longrightarrow v\} \vdash R \cup \{u' \longrightarrow v\}$	if	$u \overset{ER(E)}{\longrightarrow\!\!\!\!\!\rightarrow} u'$

Applying these rules as much as possible to an original system R_0 may not terminate, but if it does it provides an equivalent system which is coherent :

Theorem 3. *Suppose that $R_0 \vdash \cdots \vdash R_\infty$ using the above rules, and that no rule applies to R_∞, then $\overset{[R_\infty]_{E \cup ER}}{\longrightarrow} = \overset{[R_0]_{E \cup ER}}{\longrightarrow}$ and $\overset{[R_\infty (ER)]}{\longrightarrow}$ is equational coherent.*

Proof. One can easily check that if $R_i \vdash R_{i+1}$ using the above rules, then $\overset{[R_i]_{E \cup ER}}{\longrightarrow} = \overset{[R_{i+1}]_{E \cup ER}}{\longrightarrow}$. If the rule (Deduce) does not apply to R_∞, then $CP_E(R, ER) = \emptyset$, hence $\overset{[R_\infty (ER)]}{\longrightarrow}$ is equational coherent by theorem 2. □

This completion procedure works for the case of equational coherence. For strong and weak coherence, the conditions about critical peaks involving non linear rules still have to be dealt with separately.

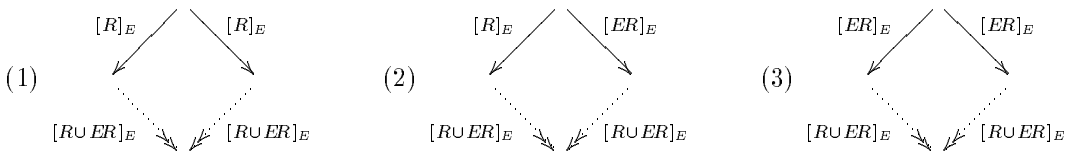
Note that the orientation of the new rules added by the rule (Deduce) is determined by the orientation of the rules already in R.

2.6 Completion modulo a rewrite system

In the case where R has an equational interpretation, there is a strong relationship between coherence completion and Knuth-Bendix completion:

Theorem 4. *Remember that we always suppose* $\xrightarrow{[ER]_E}$ *confluent. Suppose that* $\xrightarrow{[R\cup ER]_E}$ *terminates. If* $\xrightarrow{[R]_E}$ *is confluent and* $\xrightarrow{[R(ER)]_E}$ *is equational coherent, then* $\xrightarrow{[R\cup ER]_E}$ *is confluent.*

Proof. Any of the following local critical peaks can be closed



$$(1)\quad \begin{tikzcd} & \cdot \arrow[dl, "{[R]_E}"'] \arrow[dr, "{[R]_E}"] & \\ \cdot \arrow[dr, dotted, "{[R\cup ER]_E}"'] & & \cdot \arrow[dl, dotted, "{[R\cup ER]_E}"] \\ & \cdot & \end{tikzcd} \qquad (2)\quad \begin{tikzcd} & \cdot \arrow[dl, "{[R]_E}"'] \arrow[dr, "{[ER]_E}"] & \\ \cdot \arrow[dr, dotted, "{[R\cup ER]_E}"'] & & \cdot \arrow[dl, dotted, "{[R\cup ER]_E}"] \\ & \cdot & \end{tikzcd} \qquad (3)\quad \begin{tikzcd} & \cdot \arrow[dl, "{[ER]_E}"'] \arrow[dr, "{[ER]_E}"] & \\ \cdot \arrow[dr, dotted, "{[R\cup ER]_E}"'] & & \cdot \arrow[dl, dotted, "{[R\cup ER]_E}"] \\ & \cdot & \end{tikzcd}$$

(1) by confluence of $\xrightarrow{[R]_E}$, (2) by equational coherence of $\xrightarrow{[R(ER)]_E}$ and (3) by confluence of $\xrightarrow{[ER]_E}$. Since $\xrightarrow{[R\cup ER]_E}$ terminates, applying the above properties to a critical peak $u \xleftarrow{[R\cup ER]_E} \xrightarrow{[R\cup ER]_E} v$ eventually terminates and we get $u \xrightarrow{[R\cup ER]_E} \xleftarrow{[R\cup ER]_E} v$. □

The theorem suggests a straightforward procedure for completion modulo E and ER: first complete a set of equations in order to get a system R convergent modulo E, then do coherence completion between R and ER.

This is obviously not as efficient as a specially designed procedure (see below), since we certainly miss many possible simplifications, but it does the job. The termination proof can be done in various ways:

- Prove directly the termination of $\xrightarrow{[R]_{E\cup ER}}$, eg. with a simplification ordering compatible with $E \cup ER$. This is usually not very practical, as finding such orderings can be very tricky.
- Prove separately the termination of $\xrightarrow{[R]_E}$ and $\xrightarrow{[ER]_E}$ using the *same* simplification ordering compatible with E.
- Prove separately the termination of $\xrightarrow{[R]_E}$ and $\xrightarrow{[ER]_E}$ and apply a modularity theorem [Toy87, Rus87, Gra92]. Unfortunately such results are for the moment restricted to the empty theory.

Completion modulo a rewrite system ER and a set of equations E has many advantages compared respectively to completion modulo the whole theory $E \cup ER$ or completion modulo E of a set of equations and ER.

- Completion modulo $E \cup ER$ is often not feasible, because a unification algorithm modulo this theory maybe unknown or too costly.
- Critical pairs can be computed using unification modulo any theory between E and $E \cup ER$. Typically, using AC1 instead of AC leads to a much smaller set of critical pairs [Dom92].
- For a given theory ER known in advance, it is possible to "precompute" critical pairs and inference steps. A large number of critical pairs can be precomputed syntactically, without even the need of E-unification, and a large number of simplifications can be made at an early stage.

These optimisations have been described in detail in [Mar93] for the case $E = \mathsf{AC}$, formalized by the notion of *normalizing pair* and *symmetrization*.

The completion procedure proposed in [Mar93] is based on the *normalizing rewriting* relation $\xrightarrow{[R(ER!)]_{AC}}$ (first apply ER rules until a normal form is found, then perform one R

step), but the resulting completed system is convergent for the relation $\xrightarrow{[R(ER)]_E}$ as well (the notion of normalizing rewriting is used there to prove the completeness of the procedure).

The power of completion modulo a rewrite system is illustrated by new decidability results given in [Mar93].

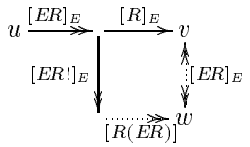
3 Implementation

Although we have always considered the relation $\xrightarrow{[R(ER)]_E}$, which relation has to be implemented depends on what we are looking for. The first question to ask is what do we need to implement:

- If we are interested in computing a (or the) normal form for $\xrightarrow{[R(ER)]_E}$, in which case only equational coherence is needed, we can notice that

$$u \xrightarrow{[R(ER)]_E}\!\!\!\!\!\rightarrow \xrightarrow{[ER]_E}\!\!\!\!\!\rightarrow v \quad \text{iff.} \quad u \xrightarrow{[R \cup ER]_E}\!\!\!\!\!\rightarrow v$$

  It is thus enough to provide implementations for $\xrightarrow{[R]_E}$ and $\xrightarrow{[ER]_E}$, and compute with the union of these relations as much as possible.
- If the goal is to find all normal forms, weak coherence is needed, and we have



  by weak coherence of $\xrightarrow{[R(ER)]_E}$, hence $u \xrightarrow{[R(ER)]_E}\!\!\!\!\!\rightarrow v$ if and only if $u \xrightarrow{[R(ER')]_E}\!\!\!\!\!\rightarrow w$ and $v \xleftarrow{[ER]_E}\!\!\!\!\!\rightarrow w$. We can thus strongly reduce the search space by systematically reducing to normal form by $\xrightarrow{[ER]_E}$, and only then look at all the possible derivations starting from that normal form.
- Then, if we want to compute exactly the whole derivation space, strong coherence is required, and we can again use the same optimisation as above.

3.1 Oriented matching

A well known technique for implementing a relation over equivalence classes $\xrightarrow{[R]_E}$ is to replace it by a weaker one $\xrightarrow{R,E}$ over terms. If an appropriate coherence property is verified [JK86], then the former can be "simulated" by the latter. The major interest of using the relation $\xrightarrow{R,E}$ is that it can be implemented by matching modulo E.

We can extend these ideas to the case of matching modulo oriented rules. Define

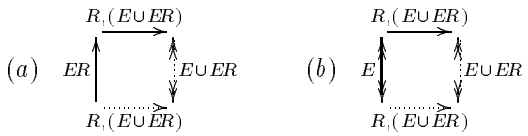
$$\xrightarrow{R,(E \cup ER)} \quad = \quad (\underset{\geq p}{\overset{E}{\longleftrightarrow}} \cup \underset{\geq p}{\overset{ER}{\longrightarrow}})^* \underset{p}{\overset{R}{\longrightarrow}}$$

This relation can be implemented by an *oriented matching algorithm* that, given two terms u and v, returns a minimal set of most general substitutions σ such that σu and v are joinable by applying equations of E and rules of ER *according to their orientation*, ie. $\sigma u \ (\underset{\geq p}{\overset{E}{\longleftrightarrow}} \cup \underset{\geq p}{\overset{ER}{\longrightarrow}})^* \ v$.

An example of such an algorithm is the special treatment of matching modulo identity found in the OBJ interpreter [GWM+92].

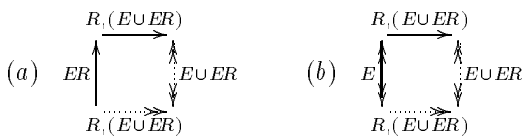
Under some conditions, the relation $\xrightarrow{R,(E \cup ER)}$ can be used in place of $\xrightarrow{[R(ER)]_E}$. As for the definition of coherence, the needed conditions depend on how close a correspondence we seek between these two relations:

– Strong case. If the two following properties are verified



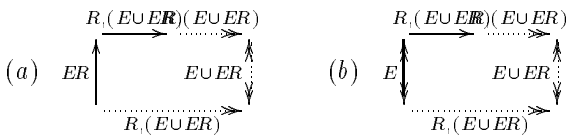
then $[u]_E \xrightarrow{[R(ER)]_E} [v]_E \iff u \xrightarrow{R_!(E \cup ER)} v' \xrightarrow{E \cup ER} v$.
The proof is straightforward from the definition of $\xrightarrow{[R(ER)]_E}$.

– Weak case. If $\xrightarrow{[R \cup ER]_E}$ is terminating and the two following properties are verified



then $[u]_E \xrightarrow{[R(ER)]_E} [v]_E \iff u \xrightarrow{R_!(E \cup ER)} v' \xrightarrow{E \cup ER} v$.
The proof is similar as before, but requires the termination of $\xrightarrow{[R \cup ER]_E}$.

– Equational case. If $\xrightarrow{[R \cup ER]_E}$ is terminating and the two following properties are verified



then $\xrightarrow{R_!(E \cup ER)}$ and $\xrightarrow{R(ER)/E}$ have the same normal forms.
The proof goes as follows: suppose that $u \xrightarrow{R(ER)/E} v$, using the two properties above, one can show that there is a derivation $\xrightarrow{R_!(E \cup ER)}$ starting from u, and that if this derivation is empty, then there is a cycle $u \xrightarrow{[R \cup ER]_E} u$, which contradicts the termination hypothesis.

If $ER = \emptyset$, we are in the situation of rewriting modulo a set of equations, and property (b) for the equational case coincides with the notion of coherence of [JK86], from which we can borrow the techniques to check or ensure property (b) for the equational case, and possibly also for the strong and weak case.

Property (a) is more unusual. Seemingly, checking it involves critical pairs between a left-hand side of R and a right-hand side of ER. We can notice however that it is true in an important subcase, including identity and idempotency rules. Assume that all right-hand sides of the rules in ER are variables, and consider a derivation

$$u \underset{p}{\xrightarrow{ER}} \underset{q}{\xrightarrow{R_!(E \cup ER)}} v \quad \iff \quad u \underset{p}{\xrightarrow{ER}} \left(\underset{\geq q}{\xleftrightarrow{E}} \cup \underset{\geq q}{\xrightarrow{ER}} \right)^* \underset{q}{\xrightarrow{R}} v$$

If $p \geq q$, then $u \xrightarrow{R_!(E \cup ER)} v$, since the first ER step is below q, and property (a) is trivially statisfied. In all other cases, the two steps commute (since by hypothesis the rules in ER have variables as right-hand sides), and property (a) is verified for the weak and equational case. For the strong case, we also need to require that the variable forming the right-hand side of a rule appears exactly once in its left-hand side.

Note that the techniques proposed here can also be applied to a subset of ER, using the fact that if $ER = S \cup T$, then $\xrightarrow{[R(S \cup T)]_E} \iff \xrightarrow{[R(S)(T(S))]_E}$.

Extension to equivalence classes. A limitation of the definition of $\xrightarrow{R_!(E \cup ER)}$ is that it does not extend to rewriting classes modulo another theory E', because the notion of position usually does not make sense in an equivalence class.

We can however extend it to the important case of classes modulo associativity (A), using for instance the notion of position in a flat term defined in [Mar93]. Using the same definition for $\overset{R,(E\cup ER)}{\longrightarrow}$ with that appropriate notion of position, we can define the relation $\overset{[R,(E\cup ER)]_A}{\longrightarrow}$ and all the results above apply as well, since they are based on the permutation lemma which is also valid for classes modulo A.

For instance, the implementation of rewriting modulo AC1 by rewriting over flat terms can be formalised as follows:

The relation $\overset{[R]_{AC1}}{\longrightarrow}$ is equivalent to $\overset{[R(1)]_{AC}}{\longrightarrow}$ (see the discussion in example 4.1 below), and properties (a) and (b) for the strong case are verified, hence

$$[u]_{AC1} \overset{[R]_{AC1}}{\longrightarrow} [v]_{AC1} \quad \Longleftrightarrow \quad [u]_A \overset{[R,(1\cup C)]_A}{\longrightarrow} [v']_A \overset{[1\cup C]_A}{\twoheadrightarrow} [v]_A$$

4 Examples

4.1 Rewriting modulo identity

Rewriting modulo identity is famous for its termination problems. Consider $R = \{f(x+y) \longrightarrow f(x)+f(y)\}$ and $E = \{x+0=x\}$. Then $[R]_E$ does not terminate. We have

$$[f(x)]_E \overset{[R]_E}{\longrightarrow} [f(x)+f(0)]_E \overset{[R]_E}{\longrightarrow} \cdots$$

since $f(x) \overset{E}{\twoheadleftarrow} f(x+0) \overset{R}{\longrightarrow} f(x)+f(0)$.

The problem here is that the equation $x+0=x$ can be used as expansion (from right to left). The theory AC1 should be specified as $E =$ AC plus an oriented rule $x+0 \overset{=}{\longrightarrow} x$ in ER. In this case, $[R]_E$ is not coherent with $[ER]_E$, but coherence completion is able to find an equivalent strong coherent system by adding a rule to R:

$$ER = \left\{ \; x+0 \overset{=}{\longrightarrow} x \; \right\} \qquad R = \left\{ \begin{array}{c} f(x+y) \longrightarrow f(x)+f(y) \\ f(x) \longrightarrow f(x)+f(0) \end{array} \right\}$$

$R(ER)$ is still non terminating, but one cannot blame coherence (or identity) completion: since completion tries to simulate the relation $[R]_E$, which does not terminate, one cannot expect to get a terminating relation. However, there can be a solution if we are only interested in equational coherence. Suppose that R contains additional rules for f, for instance:

$$ER = \left\{ \; x+0 \overset{=}{\longrightarrow} x \; \right\} \qquad R = \left\{ \begin{array}{c} f(x+y) \longrightarrow f(x)+f(y) \\ f(0) \longrightarrow 0 \end{array} \right\}$$

Then R is equational coherent with ER, and $[R(ER)]_{AC}$ is terminating even though $[R]_{AC\cup ER}$ is not, basically because equational coherence does not require that we simulate the identity step

$$[f(x)]_{AC\cup ER} \overset{[R]_{AC\cup ER}}{\longrightarrow} [f(x)]_{AC\cup ER}$$

Of course, it would be impossible to get a terminating relation if we demand strong coherence (weak coherence is possible by adding rules to ER instead of R).

Using oriented rewrite logic rather than the classical one in this example allowed to preserve termination of the rewrite relation, specify normal forms (always remove extra zeroes), and directly provide an implementation through rewriting modulo AC.

We can also remark in this example that coherence completion encompasses the process of *identity completion* used eg. in the OBJ interpreter [GWM+92].

4.2 Calculi with bound variables

Bound variables appear in many different calculi. In λ-calculus, they are higher-order variables, bound by λ. In π-calculus, they are channel names bound by ν and $\overline{x}$. Other examples of variable binders are the logical quantifiers $\forall$ and $\exists$, or the integration operator (note that $\int_m^n f(x)dx$ binds only one of its arguments, x).

In all these calculi, terms are taken up to the renaming of bound variables (α-conversion), and substitution is considered as a meta-theoretic operator.

Trying to describe such calculi in a first order setting, bound variables become just *names*, and substitution has to be explicited as a first-order operator. Bound variables are usually replaced by De Bruijn indices in order to "built-in" α-conversion. We end up with two different notions of variables and substitutions, however usually only ground terms (not containing first-order variables) are considered : in λ-calculus for instance, first-order substitution is not compatible with β-reduction. First-order variables may be introduced for solving equations (eg. higher-order unification [DHK94]), with restrictions over where they may appear.

Besides trying to map a calculus into a first-order setting, explicit substitutions were first introduced in order to get a better understanding of the operational semantics of a calculus. For instance, it is shown in [ACCL90] how a λ-calculus with explicit substitutions leads quite straightforwardly to the definition of machines implementing λ-calculus. Many different substitution calculi can be defined, see [Les94] for a survey.

As an example, we will consider here the $\lambda\upsilon$-calculus introduced in [BBLRD95] ; this choice is motivated by the fact that it is the simplest in the litterature (it has the fewest number of rules and extra operators).

Bound variables are represented by De Bruijn indices $\underline{n}$ (they can be formally defined as Peano integers, but we stick to the notation $\underline{n}$ for readability). A substitution operator $[\_/]$ is introduced, with the intuition that $[a/]$ corresponds to the substitution

$$\underline{1} \mapsto a \quad \underline{2} \mapsto \underline{1} \quad \cdots \quad \underline{n+1} \mapsto \underline{n}$$

β-reduction is simulated by the only reduction rule

$$R = \{ \quad (\lambda a)b \longrightarrow a[b/] \quad \}$$

The equational rules specify how to reduce terms containing substitutions (two auxiliary operator $\uparrow$ and $\Uparrow$ are introduced) :

$$ER = \left\{ \begin{array}{lll} (ab)[s] & \xrightarrow{=} & a[s]b[s] \\ (\lambda a)s & \xrightarrow{=} & \lambda(a[\Uparrow(s)]) \\ \underline{1}[a/] & \xrightarrow{=} & a \\ \underline{n+1}[a/] & \xrightarrow{=} & \underline{n} \end{array} \qquad \begin{array}{lll} \underline{1}[\Uparrow(s)] & \xrightarrow{=} & \underline{1} \\ \underline{n+1}[\Uparrow(s)] & \xrightarrow{=} & \underline{n}[s][\uparrow] \\ \underline{n}[\uparrow] & \xrightarrow{=} & \underline{n+1} \end{array} \right\}$$

It is shown in [BBLRD95] that

- $\lambda\upsilon$-calculus is a conservative extension of λ-calculus : the relation $\xrightarrow{R}\xrightarrow{ER!}$ over pure terms (not containing $[\_/]$, $\uparrow$ or $\Uparrow(\_)$) coincides with β-reduction, hence also $\xrightarrow{R(ER!)}$ (putting aside the last $\xrightarrow{ER!}$ steps of a derivation).
- ER is convergent
- $R \cup ER$ is confluent over ground terms. This cannot be shown by the usual critical pair analysis, because there is a non confluent critical pair between the rule in R and the first rule of ER.
  If confluence over open terms is required, the system $\lambda\sigma_{\Uparrow}$ from [HL89] does the job, it is however much more complex than our example.

$R(ER)$ is equational coherent *over ground terms* (or over any terms in the case of $\lambda\sigma_{\Uparrow}$) : this is a trivial consequence of the confluence of $R \cup ER$. It is not strong coherent, because of the following critical peak :

$$\begin{array}{ccc} (ab)[(\lambda c)d] & \xrightarrow{R} & (ab)[c[d/]] \\ \Big\downarrow{\scriptstyle ER} & & \\ a[(\lambda c)d]b[(\lambda c)d] & & \end{array}$$

Two $\stackrel{R}{\longrightarrow}$ steps are required in order to close the diagram. However the normalising relation $R(ER!)$ is strong coherent since by definition such critical peaks do not happen.

We do not know about weak coherence : seemingly, proving it would require a careful examination of the proof of confluence of $R \cup ER$. Weak coherence for $\lambda\sigma_{\Uparrow}$ can be proved or refuted by examining the critical pairs, we leave this however to an automated tool because of the great number of rules.

Similar ideas may be applied to the case of π-calculus or first-order logic (in the latter case, $\stackrel{R}{\longrightarrow}$ steps would correspond to deduction steps), providing a better insight into the operational aspects of these calculi and a direct implementation through rewriting.

This example would have been possible also in the framework of classical rewriting logic, considering the equational rules as non oriented axioms, but it would be restricted to a semantical description, because rewriting modulo such a complex theory is not practically feasible. The fact of having two sets of rules is important for hiding the ER steps and thus being able to express a simple correspondance between β-reduction and $\stackrel{R(ER!)}{\longrightarrow}$.

4.3 LOTOS

LOTOS [BB89] is a specification language combining processes similar to CCS [Mil89] and abstract data types (ADT), in the sense that the messages exchanged by processes can be any user-defined ADT. It is used in the industry for the specification of communication protocols.

The formal semantics of LOTOS [BB89] is based on two independant levels. The first is similar to CCS extended with value-passing, the second is ACT-ONE, a specification language for abstract data types. This layer structure is not completely satisfactory, as noticed in an introductory book ([BB89], page 70) :

> *"(...) many elements of a system can be specified both as processes and as data types. (...) a deeper understanding of the relation between the two components could be beneficial, and some harmonization between them could be attempted (...), for instance a common semantical model."*

Oriented rewrite logic does provide such a common semantical model. Let us first see how we can specify a CCS-like calculus in oriented rewrite logic. The structural axioms of CCS are specified as equations and equational rules :

$$E = \left\{ \begin{array}{c} P + Q = Q + P \\ P + (Q + R) = (P + Q) + R \\ \\ P|Q = Q|P \\ P|(Q|R) = (P|Q)|R \end{array} \right. \qquad ER_1 = \left\{ \begin{array}{c} \overline{\overline{l}} \stackrel{=}{\longrightarrow} l \\ \\ P + P \stackrel{=}{\longrightarrow} P \\ P + 0 \stackrel{=}{\longrightarrow} P \\ P|0 \stackrel{=}{\longrightarrow} P \end{array} \right.$$

In the spirit of [MS92, HY93], we will define a rewrite system R simulation the *reduction relation*, or τ-steps. In order to recover the usual observation-based semantics, one has to introduce explicitely the observer, ie. there is an α-transition $P \stackrel{\alpha}{\mapsto} Q$ if and only if

$P|\overline{a}.U \overset{\tau}{\mapsto} P|U$. The space of labelled transitions starting from a process P corresponds exactly to the tree of *narrowing* steps [Boc93] starting from $P|x$, where x is a new variable.

The first problem is that the reduction relation is not a congruence, while rewriting is, by definition. A simple solution is to use a special unary symbol, say #, marking the "top" of a CCS agent (with for instance sort conditions insuring that this symbol cannot appear inside an agent). The second problem is that (at least in the case of CCS) a rewrite rule simulating the reduction relation should search for redexes at an arbitrary depth. For instance

$$P_1 \mid (P_2 + (P_3 \mid (P_4 + \tau.Q))) \mapsto^{\tau} P_1 \mid P_3 \mid Q$$

But we are saved by the *expansion theorem* [Mil89], which in the case of *ground* (variable-free) agents, can be expressed as a rewrite rule

$$ER_2 = \left\{ a.P + U \mid b.Q + V \longrightarrow^{=} a(P \mid b.Q + V) + b(a.P + U \mid Q) + [\overline{a} = b]\tau.(P \mid Q) + (U \mid V) \right.$$

together with a few rules defining the matching operator $[... = ...]$. Basically the expansion theorem says that each ground process is equivalent to a process of the form

$$a_1.P_1 + ... + a_n.P_n + \tau.Q_1 + ... + \tau.Q_m$$

The last point is the case of recursive definitions like $P \overset{def}{=} a.P + P$. Using a new constant R_i for each definition, they may be specified as non-terminating equational rules

$$ER_3 = \left\{ \mathsf{R}_i \longrightarrow a.\mathsf{R}_i + \mathsf{R}_i \right.$$

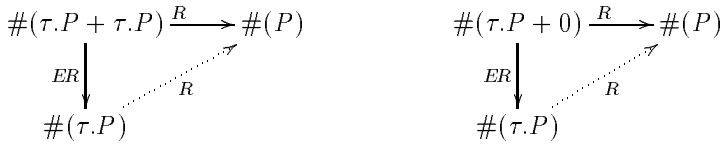
Now, considering $ER = ER_1 \cup ER_2 \cup ER_3$, the reduction relation can be specified as the single non equational rule

$$R = \left\{ \#(\tau.P + Q) \longrightarrow \#(P) \right.$$

and we have

$$P \overset{\tau}{\mapsto} Q \quad \Longleftrightarrow \quad [\#(P)]_{E \cup ER} \overset{[R]}{\longrightarrow} [\#(Q)]_{E \cup ER}$$

If we leave aside the non terminating rules for recursive definitions, theorem 2 applies and we can show that $R(ER)$ is strong coherent : there are two critical pairs modulo AC, and they verify strong coherence :



There are no critical peaks with a non linear ER rule above an R rule, since the R rule can only be applied at the topmost position of an agent. Hence the above equivalence becomes

$$P \overset{\tau}{\mapsto} Q \quad \Longleftrightarrow \quad [\#(P)]_E \overset{[R(ER)]_E}{\longrightarrow} [\#(Q')]_E \overset{[ER]_E}{\twoheadleftarrow} [\#(Q)]_E$$

which can be implemented by rewriting modulo AC.

The case of recursive definitions can be handled by applying these rules only when "needed" in order to apply a reduction rule. With such a strategy, the relation $\overset{ER}{\longrightarrow}$ terminates and our results apply. I leave aside the precise definition of such a strategy, because it requires a somewhat different set of rules in order to handle recursive definitions with a "|" operator at the top of the right-hand side.

Now let us come back to the LOTOS example. The first part of the LOTOS language is a process calculus similar to the above one (except that it allows value-passing, which implies to include binders and an operation of substitution as in the previous example). The second part allows to define arbitrary abstract data types, to be used as parameters for agents and values in messages. As usual, the definitions of abstract data types can be expressed straightforwardly by a set of equational rules.

As long as these rules do not use any of the symbols constructing CCS processes, strong coherence is preserved, since no new critical pair is created, and no problematic critical is introduced because "values" can only appear inside processes, but not the opposite.

We thus get a definition of LOTOS based on a common semantical model for both processes and data-types. This would also be possible in the framework of "classical" rewriting logic, considering all rules in *ER* as non oriented equations, but having oriented equational rules offer some advantages:

The orientation of a rule like $P + 0 \xrightarrow{=} P$ is not only useful for termination reasons, but is also a specification that extra zeroes have to be dropped, ie. one would not like to have an interpreter giving $P + 0$ as a result.

We get a "workable" definition of LOTOS, ie. one modulo an equational theory (AC) for which we know efficient matching and unification algorithms. Feeding the above rules to a rewrite interpreter like OBJ [GWM+92], Maude [Mes93] or ELAN [Vit94, KKM95] provides immediately a quite efficient interpreter for LOTOS.

An important application of this example is to provide a "clean" I/O system to such programming languages, based on a CCS-like calculus with value-passing. A prototype using explicit substitutions as in the previous example has been implemented in ELAN, including the case of recursive definitions.

5 Conclusion

We hope that the previous examples achieved their goal of convicing the reader of the effectiveness of this approach. Rewriting modulo an arbitrary big equational theory is not feasible in practice, but in all cases we have been able to find an equivalent (up to strong, weak or equational coherence) rewrite system modulo a tractable theory that can be directly fed to a rewriting interpreter.

We also provided a general framework and solutions for different problems that were handled with ad-hoc and not always satisfactory methods:

- we were able to express the LOTOS language in a unified framework.
- we showed how coherence completion is a generalisation of identity completion, and how our framework explains and provides solutions to its associated non-termination problem
- we gave a simple and general definition of the use of flat terms for implementing rewriting modulo A, AC or AC1
- we showed how the powerful procedure of completion modulo a rewrite system follows immediately from our results about coherence
- by allowing to "hide" certain reduction steps, our framework allows to have different views of derivations, and suggests a hierarchical composition of rewrite systems.
- orientation of the equational rules can also be seen as a specification of normal forms, eg. removing extra zeroes in expressions or specifying the pure λ-terms in the λv calculus.

Implementing these ideas should be quite easy by extending existing tools. Coherence completion relies on the same basic ideas than the usual Knuth-Bendix completion. Any usual rewriting interpreter is able to interpret the rewrite systems produced by coherence completion, but a little extension is needed in order to allow hierarchical composition of rewrite system and viewing of derivations at different levels.

Some future work is called for in order to relax the termination conditions that appear in most of the theorems, and find more general ways to ensure property (a) of section 3.1. Another topic worth investigating is the narrowing relation associated with the rewrite relation $\xrightarrow{[R\,(ER)]_E}$: we showed that in the case of weak coherence, when trying to find all the normal forms of a term, it is enough to reduce by R and backtrack only with the *ER* rules. This suggests a similar simplification in the case of narrowing.

References

[ACCL90] M. Abadi, L. Cardelli, P.-L. Curien, and J.-J. Lévy. Explicit substitutions. Technical Report 54, Digital Systems Research Center, February 1990. Preliminary version in *Proc. of the 17th POPL conference, Orlando (Fla., USA).*

[BB89] T. Bolognesi and E. Brinksma. Introduction to the ISO specification language LOTOS. In P. H. J. van Eijk, C. A. Vissers, and M. Diaz, editors, *The Formal Description Technique LOTOS*, pages 23–73. Elsevier Science Publishers B. V. (North-Holland), 1989.

[BBLRD95] Z. Benaissa, D. Briaud, P. Lescanne, and J. Rouyer-Degli. λv, a calculus of explicit substitutions which preserves strong normalisation. Technical Report 2477, INRIA, February 1995.

[Boc93] Alexander Bockmayr. Conditional narrowing modulo a set of equations. *Applicable Algebra in Engineering, Communication and Computation*, xx(xxx):pp, January 1993.

[DHK94] Gilles Dowek, Thérèse Hardin, and Claude Kirchner. Higher-order unification via explicit substitutions. In Denis Lugiez, editor, *Proceedings 8th International Workshop on Unification, Val d'Ajol (France).* CRIN, June 1994.

[Dom92] Eric Domenjoud. AC-unification through order-sorted AC1 unification. *Journal of Symbolic Computation*, 14:537–556, 1992.

[Gra92] B. Gramlich. Generalized sufficient conditions for modular termination of rewriting. In H. Kirchner and G. Levi, editors, *Proceedings of the 3rd Algebraic and Logic Programming Conference*, volume 632 of *LNCS*, pages 53–68. Springer-Verlag, September 1992.

[GWM+92] J. A. Goguen, T. Winkler, J. Mesegeur, K. Futatsugi, and Jouannaud J.-P. Introducing OBJ. Technical Report CSL-92-03, SRI International, 1992.

[HL89] Th. Hardin and J.-J. Lévy. A confluent calculus of substitutions. In *France-Japan Artificial Intelligence and Computer Science Symposium*, Izu, 1989.

[HY93] K. Honda and N. Yoshida. On reduction-based process semantics. In *Proc. of 13th Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 761 of *LNCS*, pages 371–387. Springer-Verlag, 1993.

[JK86] J.-P. Jouannaud and Hélène Kirchner. Completion of a set of rules modulo a set of equations. *SIAM Journal of Computing*, 15(4):1155–1194, 1986. Preliminary version in Proceedings 11th ACM Symposium on Principles of Programming Languages, Salt Lake City (USA), 1984.

[KKM95] C. Kirchner, H. Kirchner, and M.Vittek. Designing clp using computational systems. In P. Van Hentenryck and S. Saraswat, editors, *Principles and Practice of Constraint Programming.* The MIT press, 1995.

[Les94] P. Lescanne. From $\lambda\sigma$ to λv, a journey through calculi of explicit substitutions. In Hans Boehm, editor, *Proceedings of the 21st Annual ACM Symposium on Principles Of Programming Languages, Portland (Or., USA)*, pages 60–69. ACM, 1994.

[Mar93] C. Marché. *Réécriture modulo une théorie présentée par un système convergent et décidabilité du problème du mot dans certaines classes de théories équationnelles.* Thèse de Doctorat d'Université, Université de Paris-Sud, Orsay (France), October 1993.

[Mes92] J. Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science*, 96(1):73–155, 1992.

[Mes93] J. Meseguer. A logical theory of concurrent objects and its realisation in the Maude language. In Agha, Wegner, and Yonezawa, editors, *Research Directions in Object-Based Concurrency.* The MIT press, 1993.

[Mil89] R. Milner. *Communication and concurrency.* Prentice-Hall International Series in Computer Science. Prentice Hall, 1989.

[MOM93] N. Martí-Oliet and J. Meseguer. Rewriting logic as a logical and semantic framework. Technical Report CSL-93-05, SRI International, 1993.

[MS92] R. Milner and D. Sangiorgi. Barbed bisimulation. In *Procs. of ICALP'92*, volume 623 of *LNCS*, pages 685–695. Springer-Verlag, 1992.

[Rus87] M. Rusinowitch. On termination of the direct sum of term rewriting systems. *Information Processing Letters*, 26(2):65–70, 1987.

[Toy87] Y. Toyama. On the church-rosser property for the direct sum of term rewriting systens. *Journal of the ACM*, 34(1):128–143, January 1987.

[Vir92] Patrick Viry. *La réécriture concurrente.* Thèse de Doctorat d'Université, Université de Nancy 1, October 1992.

[Vir94] Patrick Viry. Rewriting: An effective model of concurrency. In *Proceedings of PARLE'94*, LNCS. Springer-Verlag, 1994.

[Vit94] Marian Vittek. *ELAN: Un cadre logique pour le prototypage de langages de programmation avec contraintes.* Thèse de Doctorat d'Université, Université Henri Poincaré - Nancy 1, October 1994.

A Appendix: Proofs

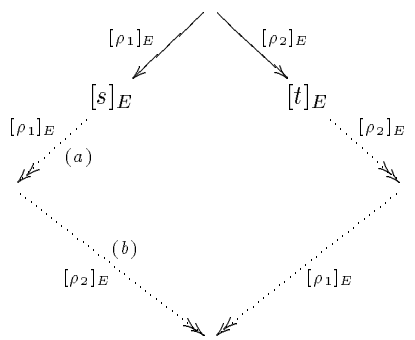
A.1 A generic permutation lemma

The permutation lemma is at the core of all the following proofs.

For a certain number of theories E including $\emptyset$, A and AC, with a suitable definition of *critical pairs modulo* E between two rewrite rewrite rules $CP_E(l_1 \to r_1, l_2 \to r_2) \subseteq T(\Sigma, X)^2$, there exists a permutation lemma stating that:

For any two rules $\rho_1 : l_1 \to r_1$ and $\rho_2 : l_2 \to r_2$, and any critical peak $\overset{[\rho_1]_E}{\longleftarrow}\overset{[\rho_2]_E}{\longrightarrow}$,

- **(superposition case)** either there exists a context $C[\,]$ such that $s \overset{E}{\twoheadleftarrow} C[u\sigma]$, $t \overset{E}{\twoheadleftarrow} C[v\sigma]$ and there is a critical pair $(u, v) \in CP_E(l_1 \to r_1, l_2 \to r_2)$,
- **(non superposition case)** either the critical peak can be closed as follows



Moreover, if l_2 is linear or if the $l_1 \to r_1$ step is not *below* the $l_2 \to r_2$ step, then the derivation (a) is empty. If r_2 is linear or if the $l_1 \to r_1$ step is not *below* the $l_2 \to r_2$ step, then the derivation (b) is a single step.
Note that the notion of applying a step below another has to be understood modulo E and may be meaningless for an arbitrary theory E, since there may not exist a suitable notion of position of a subterm in a term. Such a definition of position has been given in [Mar93] for $E = \mathsf{A}$ and $E = \mathsf{AC}$.

We do not know of any generic definition of critical pairs; a suitable one as to be provided for each different E (note that the notion of critical pairs modulo E proposed in [JK86] is not general enough in our case). Proofs of this lemma can be found for instance in [Vir92] for $E = \emptyset$ and in [Mar93] for $E = \mathsf{AC}$, where classes modulo associativity are represented by *flat terms*. The case $E = \mathsf{A}$ is proved in the same way as AC, for a slightly extended definition of critical pairs, not taking into account the equivalences induced by commutativity.

Using this lemma, checking properties such as local confluence or local coherence reduce to ensuring that the property is true in the non-superposition case, and checking that it is verified for all critical pairs, which is feasible when they are in finite number. This is the case for $E = \emptyset$, A or AC.

It remains an open question for which theories this lemma is valid, other than $\emptyset$, A or AC. However these are the most useful cases since many usual theories E can be presented by an equivalent rewrite system convergent modulo one of these three theories.

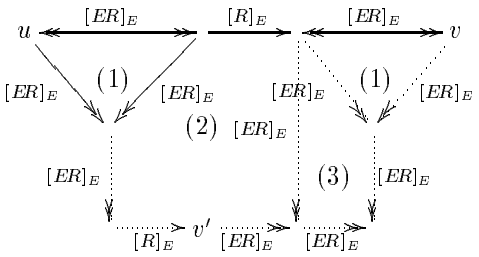
A.2 Proofs

Proof of theorem 1. In both cases, the "only if" part is trivial. For the "if" part, direction $\Longleftarrow$ is simply the definition of $\overset{[R]_{E\cup ER}}{\longrightarrow}$. Let us show the $\Longrightarrow$ part :

Strong case. Suppose that $[u]_{E\cup ER} \overset{[R]_{E\cup ER}}{\longrightarrow} [v]_{E\cup ER}$, which is equivalent to

$$[u]_E \overset{[ER]_E}{\twoheadleftarrow} \overset{[R]_E}{\longrightarrow} \overset{[ER]_E}{\twoheadleftarrow} [v]_E$$

We have :



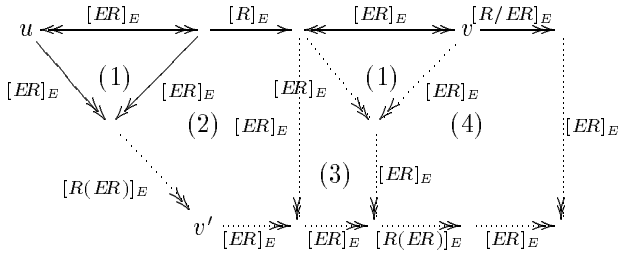
(1) confluence of $\overset{[ER]_E}{\longrightarrow}$
(2) strong coherence of $\overset{[R]_E}{\longrightarrow}$ with $\overset{[ER]_E}{\longrightarrow}$
(3) confluence of $\overset{[ER]_E}{\longrightarrow}$

Hence $[u]_E \overset{[R(ER)]_E}{\longrightarrow} [v']_E \overset{[ER]_E}{\twoheadleftarrow} [v]_E$

Weak case. Suppose that $[u]_{E\cup ER} \overset{[R]_{E\cup ER}}{\longrightarrow} [v]_{E\cup ER}$, which is equivalent to

$$[u]_E (\overset{[ER]_E}{\twoheadleftarrow} \overset{[R]_E}{\longrightarrow})^* \overset{[ER]_E}{\twoheadleftarrow} [v]_E$$

By induction over the length of the derivation, either it is empty and the theorem is trivially true, or we have :



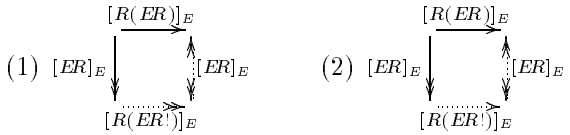
(1) confluence of $\overset{[ER]_E}{\longrightarrow}$
(2) weak coherence of $\overset{[ER]_E}{\longrightarrow}$
(3) confluence of $\overset{[ER]_E}{\longrightarrow}$
(4) induction hypothesis

Hence $[u]_E \overset{[R(ER)]_E}{\longrightarrow} [v']_E \overset{[ER]_E}{\twoheadleftarrow} [v]_E$

□

Lemma 5. *Suppose that $\overset{[ER]_E}{\longrightarrow}$ is terminating.*

If (1) $\overset{[R]_E}{\longrightarrow}$ is weak local coherent with $\overset{[ER]_E}{\longrightarrow}$, or (2) $\overset{[R]_E}{\longrightarrow}$ is strong local coherent with $\overset{[ER]_E}{\longrightarrow}$, then



and in particular, (1) $\overset{[R(ER)]_E}{\longrightarrow} \subseteq \overset{[R(ER!)]_E}{\longrightarrow}$ and (2) $\overset{[R(ER)]_E}{\longrightarrow} \subseteq \overset{[R(ER!)]_E}{\twoheadrightarrow}$.

Proof of lemma 5 Let us show the weak case ; the strong case can be shown by a simpler proof along the same lines. For a term u, define $|u|$ as the length of the longest $[ER]_E$-derivation from u to its $[ER]_E$-normal form. Consider derivations of the form

$$\delta \quad = \quad u \overset{[R/ER]_E}{\twoheadrightarrow} v \quad = \quad u = u_0 \overset{[ER]_E}{\longleftrightarrow} v_0 \overset{[R]_E}{\longrightarrow} u_1 \cdots u_{n-1} \overset{[ER]_E}{\longleftrightarrow} v_n = v$$

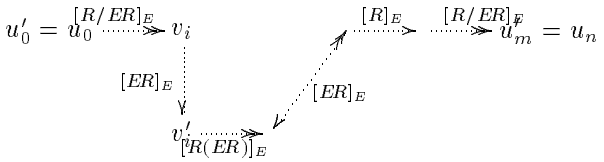
Define an ordering on such derivations as $\delta' \gg \delta''$ if $(v'_1, ..., v'_n) >_{lex} (v''_1, ..., v''_m)$ where $>_{lex}$ is the lexicographic extension (from left to right) of $>$ defined as $v'_i > v'_j$ if $|v'_i| > |v'_j|$. $\gg$ is well founded, since it is the lexicographic extension of a well founded ordering.

Starting from a derivation δ_0 from u to v, using the local coherence property, it is possible to define a sequence (δ_i) of derivations with the same source and target, thanks to the local coherence property. Given a derivation δ_k, either all the v_i's are $[ER]_E$-normal forms and the sequence is finished, either some v_i is not in $[ER]_E$-normal form and we define δ_{k+1} as follows (plain arrows stand for δ_k, dotted arrows for δ_{k+1}) :

δ_k :

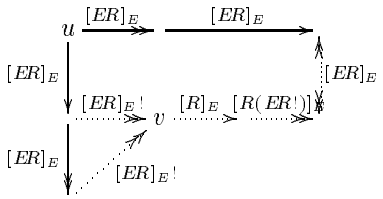
$$u_0 \xrightarrow{[R/ER]_E} v_i \xrightarrow{[R]_E} \xleftarrow{[ER]_E} \xrightarrow{[R]_E} \xrightarrow{[R/ER]_E} u_n$$

δ_{k+1} :



We have $v_j = v'_j$ for $1 \leq j < i$, and $v'_i > v_i$, hence $(v'_1, ..., v'_m) >_{lex} (v_1, ..., v_n)$ and $\delta_{k+1} \gg \delta_k$. Since $\gg$ is well founded, the sequence (δ_i) is finite. In the last derivation in the sequence, v_i's are $[ER]_E$-normal forms, otherwise the transformation would apply, hence this last derivation is of the form $u \xrightarrow{[R(ER^!)]_E} \xleftarrow{[ER]_E} v$.

Now suppose that $\xleftarrow{[ER]_E} u \xrightarrow{[ER]_E} \xleftarrow{[R]_E}$, if the $\xrightarrow{[ER]_E}$ derivation is empty, weak coherence is verified trivially, otherwise



(1) using the result above,
since $[R]_E$ is local coherent with $[ER]_E$,
(2) confluence of $[ER]_E$ (since v is a normal form)

□

Proof of theorem 2. This is a particular case of lemma 5. □