E-Unification via Constrained Rewriting

Jacques Chabin, Siva Anantharaman, Pierre Réty
LIFO, Dépt. Math-Info.
Université d'Orléans
45067 Orléans Cedex 02 (Fr.)
e-mail: {chabin, siva, rety}@univ-orleans.fr

Abstract

Basic narrowing is known to be adequate for solving equations with respect to an equational theory E, when E admits a finite canonical rewrite system. If E is not so well-behaved, one may still combine ordered completion with basic narrowing, to deduce a complete procedure for E-unification. The situation is unexpectedly more complicated when we try to combine ordered *basic* completion with basic narrowing : the lemma of Hullot *ceases to be true* in its usual form. The objective of this paper is to show that such a combination of basic narrowing and ordered basic completion does exist, and provides a complete tool for E-unification. This is obtained via an inference system covering both ordered completion and narrowing in a setup of constrained rewriting, which is an appropriate formalism for expressing the basic character of both. Initial equations with equality or ordering constraints are allowed; our approach also incorporates sufficiently many simplification steps so as to be able to provide 'basic' canonical systems for all classical examples (e.g. groups).

1 Introduction

For any given equational theory E, one knows that ordered completion provides a complete tool for proving valid (equational) E-theorems. This follows from the fact that it generates a (not necessarily infinite) final system which is ground canonical, if the reduction ordering employed is a cso (i.e. total on ground terms). The same fact also leads to a rather loose conclusion that narrowing techniques can be combined with ordered completion, to provide a complete tool for E-unification (see e.g. [8]).

But narrowing has an inherent tendency to get into infinite derivations, so the notion of 'basic narrowing' was introduced to cut down some of these infinite branches (cf. [4]). A similar notion of 'basic completion' was introduced more recently (cf. [10]), with the hope that the completion process would be more economical this way. Basic narrowing (resp. basic completion) usually employs 'marking' techniques to localize the positions brought about by earlier substitutions, which are to be avoided in the later steps of narrowing (resp. superposition). But the formalism of rewriting under *constraints*, as defined in [5] is ideally suited for expressing this 'basic' character without any artifices of marking, and [6] gave the first 'complete' inference system for basic paramodulation, in such a setup.

Now if we consider that rewriting originated with Knuth's attempt to find complete sets of reductions for groups, it is legitimate to affirm that a principal factor for judging the efficiency of a completion inference system is its capacity to produce a finite (and reasonably small) canonical system for groups. This being achievable only with a serious dose of simplification, inference systems for basic completion in general have poor performances in this respect : for instance, simplifications in [6] are quite impractical, and no convincing implementation of this work seems available yet, as far as we know. The situation gets worse when we try combine narrowing with this approach of basic completion. In fact, if the definition of constrained narrowing is not a problem, it turns out that the so-called Lemma of Hullot - which serves as a link between the rewriting and narrowing steps, to ensure subsequently the completeness of narrowing w.r.t. unification - is in general *false*, in this setup of constraints (see Section 4 below).

Our objective in this work is therefore the obtention of an inference system for basic (ordered) completion via constrained rewriting, which behaves satisfactorily w.r.t. all the above requirements : complete for equational deduction w.r.t. the theory E, complete for E-unification via basic narrowing, and allowing sufficiently many simplification steps (while remaining 'basic') to be able to produce finite canonical systems for all classical cases. Our approach is such, that we can also allow the initial equations to be constrained; but we will be assuming these constraints to be only of the *equality, or ordering* type.

In our inference system given below, we assume tacitly (as in all the relevant earlier works) that any *equality constraint* can be reduced into a 'solved form' when appropriate, by an independent algorithm. Here are the major conclusions obtained :

- If τ is any E-solution of a unificand $s \stackrel{?}{=} t$, then our basic narrowing (under basic ordered completion) generates a σ more general than τ, in finite time.
- The unificands can always be kept in normal form.
- If the initial equations are without constraints, the superposition in our ordered paramodulation inferences is basic.
- If the initial equations are constrained, then we may need to resort to some 'propagation' in the sense of [5] to remain complete; but this is needed *only* w.r.t. the *initial constraints*, and can be kept in some sense minimal.

The ordered completion part of our approach may thus be viewed as extending the results of [6],[7], to include initial (ordering) constraints, and a lot more of simplification steps. Our simplification shares the same philosophy as that in [11], though not identical.

This paper is organized as follows. Section 2 introduces all the needed notions; among which a rather sophisticated matching alogorithm for constrained terms, essential for understanding our 'basic simplification' steps. A few examples are also given so as to motivate the formulation of our inference system given in Section 3. The proofs of completeness of this inference system figure in Sections 4, 5. Section 6 gives some more examples treated with the help of UREVEAL, a soft implementing our approach. The concluding section outlines among others possible directions of further work.

2 Preliminaries

Throughout this paper the symbol '=' will stand for syntactic equality. We denote by '>' a fixed, complete simplification ordering on terms (i.e. total on ground terms; in short a *cso*). An equality constraint is an equality of the form $t = t'$; an ordering constraint is an inequality of the form $t > t'$. We will need to distinguish between such constraints present at the initial level, and those introduced at later stages by our inference steps. For this reason, two further comparison symbols, namely '$=_n$ and '$>_n$', will be employed below to express the same kind of relation between terms, but in the non-initial constraints (see the definitions below for details). A constraint in general is a finite conjonction of all such particular types, and will always figure inscribed in a : $[\![ ]\!]$. Given a constraint $[\![c]\!]$, $SOL(c)$ denotes the set of all substitutions τ such that $\tau(c)$ evaluates to 'true'.

A constrained term is a pair (t, T), where t is a (usual) term and T is a constraint; we agree to write this as $t[\![T]\!]$. A constrained equation is a pair of constrained terms, expressed under the form $u \doteq v[\![T]\!]$. The generic word 'term' (resp. 'equation') will cover both the unconstrained and constrained term (resp. equation). Since the process of orienting an equation does not play any role in the theoretical part of this work, we will consider 'rule' and 'equation' to be synonyms.

A unificand is a pair of terms with a constraint, expressed here under the form : $s \stackrel{?}{=} t\ [\![c]\!]$. Given any set of rules R, a solution (wrt R) for such a unificand is, by definition a substitution $\sigma \in SOL(c)$, such that $\sigma s =_R \sigma t$.

An unconstrained term t' is a *propagation* of $t[\![T]\!]$ iff exists $\sigma \in SOL(T)$, such that $\sigma t = t'$. Notation : $t' = \bar{t}$.

The notion of a rewrite-step from a term $t[\![T]\!]$ using a constrained equation is more intricate to define; this is so since the adoption of a too 'bland' (straightforward) approach may not allow sufficiently many simplifications and will lead to inefficiency. For this we will need the notion of a c-match on constrained terms. Intuitively : a c-match from $g[\![c]\!]$ onto $t[\![T]\!]$ is a match from a propagation of the former, on a propagtion of the latter, The formal definition of a c-match bewtween two constrained terms will be given in the following subsection; to prepare the reader for that we introduce the notion of the 'propagated part' (in short p-part) of a substitution σ. An assignment $z \leftarrow a$ is in the p-part of σ wrt a term $t[\![T]\!]$ iff it is applied directly on t (and not 'adjoined' to the constraint T); while the non-propagated part (in short np-part) concerns assignments *not* applied to t but adjoined to T (as an additional equality constraint). The respective notations will be : $\leftarrow_p$, $\leftarrow_{np}$; and σ_p, σ_{np} will denote the corresponding parts of σ.

2.1 The notions of c-match, Constrained Rewriting, and Constrained Narrowing

In what follows, for any term t we will be denoting by $V(t)$ the set of all variables in t; this notion is extended naturally to any constrained rule. If C is a constraint, thus $V(C)$ is the set of variables appearing in C; we will denote by $V_a(C)$ the set of *assigned variables* in C, i.e., the set $\{x \in V(C) \mid C \text{ is of the form } C' \wedge (x = t)\}$.

Let us view the usual matching on terms $\lhd$ as a binary function from the set of (constrained) terms, with values in the set of all substitutions. We want to construct a similar binary function, denoted $\lhd_c$, which will give us the above mentioned c-matches. Given any pair of (constrained) terms $g[\![C]\!]$, $t[\![C']\!]$ (assumed to have no common variables), a c-match from the first on the second is the subsitution resulting from the application of the following 'algorithm' (where, for commodity, we denote by $\lhd_?$ *either* of these two symbols $\lhd, \lhd_c$).

1) Decomposition

$$f(s_1, .., s_n)\ [\![C]\!]\ \lhd_?\ f(s'_1, .., s'_n)\ [\![C']\!]\ \vdash\ (s_1\ [\![C]\!]\ \lhd_?\ s'_1\ [\![C']\!])\ \wedge \ldots \wedge\ (s_n\ [\![C]\!]\ \lhd_?\ s'_n\ [\![C']\!])$$

2) Collision

$$f(s_1, .., s_n)\ [\![C]\!]\ \lhd_?\ g(s'_1, .., s'_k)\ [\![C']\!]\ \vdash\ Fail$$

3) Left-Instantiation

$$x \; [\![C]\!] \; \lhd_c \; t' \; [\![C']\!] \; \vdash \; \overline{x} \; \lhd_c \; t' \; [\![C']\!]$$

if $x \in V_a(C)$, and t' **not** a variable

4a) (Propagated) Solution

$$x \; [\![C]\!] \; \lhd_c \; t' \; [\![C']\!] \; \vdash \; \{x \leftarrow_p t'\}$$

if x is a variable not in $V_a(C)$

4b) (Non-Propagated) Solution

$$x \; \lhd \; t' \; \vdash \; \{x \leftarrow_{np} t'\}$$

5) Right-Instantiation

$$t \; [\![C]\!] \; \lhd_c \; y \; [\![C']\!] \; \vdash \; \overline{t} \; \lhd \; \overline{y}$$

if y is a variable

This 'algorithm' always terminates, and there are two cases :

- we get 'Fail' : then there is no c-match,

- or we get a set of the form $\{x_1 \leftarrow s_1, ..., x_n \leftarrow s_n\}$ (where some assignements are propagated, others not). Here we get a c-match *provided* $\forall i, j, (x_i = x_j) \Rightarrow (s_i = s_j)$; else there is no c-match.

Definition 1 *A term t $[\![T]\!]$ rewrites into t' $[\![T']\!]$, at position p by an equation $g \doteq d$ $[\![c]\!]$ iff :*

- *i) there exists c-match σ from g $[\![c]\!]$ on $t|_p$ $[\![T]\!]$: $g \; [\![c]\!] \; \lhd_c \; t|_p \; [\![T]\!]$,*
- *ii) σ is in $SOL(c)$,* **and** *$\sigma g > \sigma d$,*
- *iii) and $t' = t[p \leftarrow \sigma_p d]$, $T' = T \wedge \sigma_{np}$.*

We will write : $t \; [\![T]\!] \longrightarrow_{[p,(g \doteq d \; [\![c]\!]),\sigma]} t' \; [\![T']\!]$; or just : $t \; [\![T]\!] \longrightarrow_R t' \; [\![T']\!]$, if the rule, the position, and the match are indifferent.

Example . Consider the term $f(g(a), x)$ $[\![x = g(b)]\!]$, and the rule $f(y, z) \doteq i(y, z)$ $[\![z = g(z')]\!]$. One may be tempted to rewrite the given term, with the 'usual match' $\{y \leftarrow g(a), \; z \leftarrow g(b)\}$, into the term $i(y, z)$ $[\![y = g(a) \wedge z = g(b)]\!]$. But then this will give rise to the problem that '$g(a)$' is now 'protected'. But if we use the above definition of constrained rewriting, we get the c-match : $\{y \leftarrow_p g(a), \; z' \leftarrow_{np} b\}$. But since this must **also** satisfy the constraint on the matching term the final substitution obtained is :

$$\{y \leftarrow_p g(a), \; z \leftarrow_{np} g(z'), \; z' \leftarrow_{np} b\}$$

and the rewritten term is : $i(g(a), z)$ $[\![z = g(b)]\!]$. □

The notion of constrained narrowing on a (constrained) term turns out to be much simpler.

Definition 2 *(Constrained narrowing) Let s_1 $[\![c_1]\!]$ and s_2 $[\![c_2]\!]$ be two terms. Then s_1 $[\![c_1]\!]$ narrows into s_2 $[\![c_2]\!]$ at position p, under the equation $l \doteq r$ $[\![c]\!]$, iff :*

- *p is a non-variable position of s_1, and $V(l \doteq r \; [\![c]\!]) \cap V(s_1 \; [\![c_1]\!]) = \emptyset$.*
- *$s_2 = s_1[p \leftarrow r]$ and $c_2 = c \wedge c_1 \wedge (l >_n r) \wedge (s_1|_p =_n l)$*

Notation : $s_1 \llbracket c_1 \rrbracket \leadsto_{[p,(l \doteq r \llbracket c \rrbracket)]} s_2 \llbracket c_2 \rrbracket$.

The so-called 'narrowing substitution' does not explicitly appear in this definition; but gets generated as the most general solution of the eventual equality constraints. Moreover the substitution part being in the constraint, will never get narrowed further; for this reason constrained narrowing is basic in the sense of [4]. In practice, narrowing steps as above will be considered *only when the constraint c_2 is satisfiable.* We will also employ a notion of *normalizing narrowing* : which corresponds to normalizing a term obtained after narrowing. Notation : $s \llbracket c \rrbracket \leadsto\!\!\!\leadsto_E s' \llbracket c' \rrbracket$. It amounts to the same as saying $s \llbracket c \rrbracket \leadsto_E s'' \llbracket c' \rrbracket \longrightarrow^*_R s' \llbracket c' \rrbracket$ where s' is a normal form of s'', wrt a given set of rules R. This is not exactly the same notion as the one defined in [9].

Example 1 . Consider the one rule system :

$$g(x', f(y')) \doteq f(x') \llbracket y' > x' \rrbracket$$

under an lpo (= lexico-path-ordering), where $g > f > h$.

Thus we have $g(y, f(h(y))) \llbracket c \rrbracket \leadsto f(x') \llbracket (g(y, f(h(y))) =_n g(x', f(y'))) \wedge (y' > x') \wedge c \rrbracket$.
But also $g(h(y), f(y)) \llbracket c \rrbracket \leadsto f(x') \llbracket (g(h(y), f(y)) =_n g(x', f(y'))) \wedge (y' > x') \wedge c \rrbracket$.

The last derivation will get eventually eliminated since the constraint becomes unsatisfiable.

Reduction and narrowing are defined on constrained equations in a similar manner; for instance we will write $s \doteq t \llbracket c \rrbracket \leadsto_E s' \doteq t' \llbracket c' \rrbracket$ iff :

$$s \llbracket c \rrbracket \leadsto_E s' \llbracket c' \rrbracket \text{ and } t' = t, \textbf{ or, } t \llbracket c \rrbracket \leadsto_E t' \llbracket c' \rrbracket \text{ and } s' = s$$

Constrained narrowing on a unificand is defined in the same way.

2.2 The notions of quasi-normal and free rewrite steps

These are the notions where our basic completion procedure will differ from that of [2]; they are essential if we want our narrowing to be complete. The point is that if we allow our above-defined notion of rewriting to simplify (and then replace) rules in the 'current system', it turns out that Hullot's lifting lemma no longer holds : i.e. to say a rewrite-step on an instantiated term can no longer be lifted to a narrowing step (employing the same rule) on the original term. (See section 4). Evidently this is not what we are looking for.

On the other hand if we do not allow any simplification at all, then it is easy to convince onself that completion would run amuck (in all non-trivial cases), and so narrowing would be highly inefficient too. The notions presented in this subsection are meant to preserve us from either of these disasters.

Definition 3 *A rewrite-step $t \llbracket T \rrbracket \longrightarrow_{[p,(g \doteq d \llbracket c \rrbracket),\sigma]} t' \llbracket T' \rrbracket$ is said to be :*

- **free** *iff :*

  - $V(d) \setminus V(g)$ has **no** *variables appearing in c,* **and**

  - for every $x \in V_a(c)$, and occurrence u of $x \in g$. $t|_{pu}$ is a variable.
- **quasi-normal** *wrt a set of rules S iff the c-match σ is in S-normal form on the variables of $V(d) \setminus V(g)$.*

(A term t $[\![T]\!]$ is said to be in 'normal form' wrt a set of rules S iff no rewriting is possible on t $[\![T]\!]$ with the rules in S).

An example should help in understanding the free-rewrite-step notion above, that we we will be employing wrt *rule-simplifications.*

Example . Suppose we have a current rule

$$f(h(y)) \doteq g(y) \ [\![y = i(z)]\!]$$

in our system, and that two further rules get generated of the form :

$$f(x) \doteq c \ [\![x = h(i(a))]\!]$$
$$f(h(i(a)) \doteq c$$

The substitution $\{z \leftarrow_{np} a, y \leftarrow_{np} i(z)\}$ defines a c-match from $f(h(y))$ $[\![y = i(z)]\!]$ onto $f(x)$ $[\![x = h(i(a))]\!]$ and em allows the simplification of the first of these latter rules by the 'initial' rule. But though there exists a 'possible' rewrite-step of the last rule by the initial rule, this is **not a free** rewrite-step so this simplification is disallowed. Intuitively this is understandable : since such a simplification would have replaced the 'unprotected' $f(h(i(a))$ by $g(y)$ $[\![y = i(z) \wedge z = a]\!]$, a term where $i(a)$ would be henceforth under 'protection'.

(Compare our approach with the notion of T being 'dominated' by c as defined in [2]; cf. section 4 to localize the differences).

The main features ensuring the success of our approach can now be made explicit :

- Allow the deletion of a current rule *only when the rewrite step employed is free*,

- Show that Hullot's lifting lemma holds on quasi-normal rewrite-steps, **and** that such rewrite-steps suffice to normalize any given term. (See Section 4).

3 Inference rules for constrained equational completion and unification

Definition 4 *Let $s_1 \doteq t_1$ $[\![c_1]\!]$, $s_2 \doteq t_2$ $[\![c_2]\!]$ be two constrained rules. Then the set of critical pairs from the latter into the former is the set of all rules of the form :*

$$s_1[u \leftarrow t_2] \doteq t_1 \ [\![c_1 \wedge c_2 \wedge (s_1|_u =_n s_2) \wedge (s_1 >_n t_1) \wedge (s_2 >_n t_2)]\!]$$

where u is any arbitrary non-variable *position in s_1.*

(Observe that in the above definitions, the 'new' constraints are expressed - as pointed out earlier - with the help of the symbols $=_n, >_n$).

If S is any given set of rules, and $s_1 \doteq t_1$ $[\![c_1]\!]$ any given equation, let us denote by $CP\{s_1 \doteq t_1 \ [\![c_1]\!], S\}$ the set of all critical pairs from the given equation into any equation of S or from any equation of S into this equation, or from the given equation into itself. For any term t, $t\downarrow_S$ will denote some S-normal form of t.

The following definition is meant to play a limited 'practical' role.

Definition 5 *Given any constraint c, we define $SOL_o(c)$ to be the set of all substitutions which are solutions of the 'old part' of c, i.e. of the constraint obtained from c by eliminating all its '$=_n$', and '$>_n$'-components,* provided c is satisfiable*; otherwise $SOL_o(c)$ is defined to be the empty set.*

We will also employ below the notion of 'minimality' of a solution of a constraint. It is the usual notion on substitutions, applied here w.r.t. the set of variables appearing in the constraint : if σ, τ are solutions of a constraint c, then σ is 'more general than' τ iff $\exists \alpha \neq id$ such that $\tau = \alpha \circ \sigma$ on $V(c)$. We will say that $\tau \in SOL(c)$ is minimal iff there is no element in $SOL(c)$ more general than τ.

This notion of minimality is easily decidable wrt the 'syntactic-equality' constraints; but is obviously not so wrt an ordering constraint. We will content ourselves here with an operational approach (see illustrative Example 2 below).

Our inference rules work on triples (E, R, G), where :

- E is a set of (constrained) equations. (At the beginning of the procedure, it's the set of all - possibly constrained - input equations).
- R is a set of (constrained) equations. (At the beginning, it is empty).
- G is a set of unificands, or goals. (At the beginning it contains a single unificand with an empty constraint).

1) Choose and Deduce

$$\frac{(E \cup \{s \doteq t \; [\![c]\!]\}, R, G)}{(E \cup CP\{s \doteq t \; [\![c]\!], R\}, R \cup \{s \doteq t \; [\![c]\!]\}, G)}$$

2) (Minimal) Propagation

$$\frac{(E, R \cup \{s \doteq t \; [\![c]\!]\}, G)}{(E \cup \{\sigma(s \doteq t \; [\![c]\!])\}, R \cup \{s \doteq t \; [\![c]\!]\}, G)} \qquad \text{if } c \text{ is satisfiable, } s \neq t, \text{ and } \sigma \text{ is } \textbf{minimal} \text{ in } SOL_o(c).$$

3) Deletion of an equation

$$\frac{(E, R \cup \{s \doteq t \; [\![c]\!]\}, G)}{(E, R, G)} \qquad \text{if } s = t \textit{ or } c \textit{ is unsatisfiable.}$$

4) Constrained narrowing

$$\frac{(E, R, G \cup \{s \stackrel{?}{=} t \; [\![c]\!]\})}{(E, R, G \cup \{s \stackrel{?}{=} t \; [\![c]\!] \;,\; s' \stackrel{?}{=} t' \; [\![c']\!]\})} \qquad \text{if } (s \stackrel{?}{=} t \; [\![c]\!]) \;\rightsquigarrow_{(R)}\; (s' \stackrel{?}{=} t' \; [\![c']\!]).$$

5r) Simplify a Rule

$$\frac{(E \cup \{s \doteq t[\![c]\!]\}, R, G) \quad OR \quad (E, R \cup \{s \doteq t[\![c]\!]\}, G)}{(E \cup \{s' \doteq t[\![c']\!]\}, R, G)}$$

if $s \longrightarrow_{E \cup R} s'$, by a **free** rewrite step.

5u) Simplify a Unificand

$$\frac{(E, R, G \cup \{s \stackrel{?}{=} t \; [\![c]\!]\})}{(E, R, G \cup \{s' \stackrel{?}{=} t' \; [\![c']\!]\})} \qquad \text{if } (s' \stackrel{?}{=} t' \; [\![c']\!]) = (s \stackrel{?}{=} t \; [\![c]\!])\downarrow_R.$$

6) Generation of solution

$$\frac{(E, R, G \cup \{s \stackrel{?}{=} t \; [\![c]\!]\})}{(E, R, G \cup \{s \stackrel{?}{=} t \; [\![c]\!] \;,\; \emptyset \; [\![c \wedge (s = t)]\!]\})} \qquad \text{if } s \textit{ is syntactically unifiable with } t.$$

7) Deletion of a unificand

$$\frac{(E, R, G \cup \{s \stackrel{?}{=} t \ \llbracket c \rrbracket\})}{(E, R, G)} \qquad \text{if } c \textit{ is unsatisfiable.}$$

8) Simplify a solution

$$\frac{(E, R, G \cup \{\emptyset \ \llbracket c \wedge (s >_n t) \rrbracket\})}{(E, R, G \cup \{\emptyset \ \llbracket c \rrbracket\})} \qquad \text{if } c \wedge (s >_n t) \textit{ is satisfiable.}$$

Some comments are appropriate here.

i) In this setup, R represents the set of rules having already served for superposition, and E is the set of remaining rules. Thus if completion stops, E will have gotten emptied.

ii) Inference rule 8) plays a very practical role : it says that the role of the $>_n$-constraints, *on the goal*, is limited to the level of 'satisfiability' and *not* in generating the narrowing substitution. That they *have* to be satisfied implies for instance that an 'unorientable' rule cannot be employed either way indefinitely, thus leading into a cycle. On the other hand a non-ground substitution has 'in general' not much chance of being solution of an ordering constraint; this explains why we solve a constraint for the narrowing substitution without taking into account the (new) ordering constraints. [We cannot drop however the 'old' ordering constraints for evident reasons]. See illustrative Examples 8 below. Observe finally that dropping the $>_n$ constraints, while solving for the narrowing substitution, creates no problem of 'correction' w.r.t. E-unification.

iii) In general our 'minimal propagation' is much less expensive than the schematization of [5], even when this latter is applied only wrt the initial constraints (cf. example below). We do *not* try to 'expand' *a priori* the initial constraints (say as in [7]), in particular since such an expansion on an ordering constraint may lead to an infinite system.

Example 2 Consider the one rule system :

$$f(x) \doteq g(x) \ \llbracket x > b \rrbracket$$

under an lpo ordering where $f > g > a > b$. Now $\{x \leftarrow a, x \leftarrow f(x), x \leftarrow g(x)\}$ is a 'complete' set of 'minimal' solutions for the constraint $x > b$. So our minimal propagation generates only the following rules :

$$f(a) \doteq g(a), \quad f(g(x)) \doteq g(g(x)), \quad f(f(x)) \doteq g(f(x)). \qquad \Box$$

We will be considering below (as usual) derivation chains $(E_i, R_i, G_i)_{i \geq 0}$, where each i-th triple is obtained from the $(i-1)$-th by one inference step. We will be assuming that our derivation chain is **fair** in the following sense :

- $E_\infty = \emptyset$, **and**, for any rule $l \doteq r \ \llbracket T \rrbracket$ in R_∞ and any $n \geq 0$, the ouput of 'minimal propagation' on $l \doteq r \ \llbracket T \rrbracket$, (resp. the set $CP\{l \doteq r \ \llbracket T \rrbracket, R_n\}$ is contained in some set R_r (resp. some R_s),

- for any $n \geq 0$, any rule $g \doteq d \ \llbracket c \rrbracket$ in R_∞, and a unificand in G_n, the result of (possible) constrained narrowing from the rule considered into the unificand, is in some goal-set G_r.

We first show that the ground confluence of the final system R_∞ can be obtained 'with some particularities'. An appropriate reformulation of Hullot's lemma will then allow us to conclude the completeness of our narrowing.

4 Ground Confluence with Reference to Narrowing

In what follows, we will denote by S the final set R_∞, obtained by our basic completion procedure from E_0. We first observe that the ground confluence of S is provable along the same lines as employed in [2] : *a priori* we allow less 'simplifications' on the 'current' rules than their approach, so their 'completeness' cannot be lost in our context.

As a matter of fact, ground confluence of S can also be proved along the following lines (cf. the full version of this paper). Consider an S-proof between two ground terms, containing a peak of the form :

$$s' \longleftarrow_{[p,(g \doteq d\ [\![c]\!]),\sigma]} \quad s \longrightarrow_{[q,(l \doteq r\ [\![c']\!]),\theta]} \quad s"$$

The only case to study really is e.g. when the second step rewrites under a 'variable position' in g. If x is the variable concerned in g, and is unconstrained, then our peak can be reduced easily along the 'classical' lines; so assume that $x \in V(c)$. Now if all the components of c concerning this x are 'old', we may as well employ a rule propagated from the first rule, to get rid of the fact that x is contrained; so assume that c has a 'new' component concerning x. This means that $g \doteq d\ [\![c]\!]$ has been generated as a critical pair between two earlier rules, at least one of which has a superposition instance with $l \doteq r\ [\![c']\!]$. One shows that the rule resulting from this overlap can be used to reduce the above peak.

For narrowing purposes however, just proving ground confluence of S this way turns out to be insufficient. Recall that the completeness of (classical) narrowing was proved by Hullot by exploiting the fact that a (classical) rewrite step on an instantiated term can be lifted to a narrowing step (using the same rule) on the original term. Such a conclusion turns out to be (in general) false in the context of basic completion, even if the rewrite step considered is innermost.

4.1 Hullot's lemma under constraints is false in general

The following Example is also meant to show why our rule deletion is restricted to the case of 'free rewrite-steps'.

Example 3 Consider the system :

$$\begin{array}{ll} (1) & f(x,c) \doteq x \\ (2) & f(i(y),z) \doteq g(y) \\ (3) & i(a) \doteq a \end{array}$$

under an lpo where : $f > g > c > i > a$.

Superposition between (1) and (2) gives the equation :

$$\begin{array}{ll} (4) & g(y) \doteq x\ [\![(x =_n i(y))]\!] \end{array}$$

Now the term $g(a)$ is rewritten by (4) into $i(a)$, with the match $\tau_1 = \{x \leftarrow i(a), y \leftarrow a\}$, and $i(a)$ is rewritten by (3) into a (both are innermost rewrite steps).

$$\begin{array}{lclcl} g(a) & \rightarrow_{(4)} & i(a) & \rightarrow_{(3)} & a \\ \uparrow \rho_0 & & \uparrow \rho_1 & & \\ g(y) & \rightsquigarrow_{(4)} & x\ [\![x =_n i(y)]\!] & & \end{array}$$

Now $g(a)$ can be narrowed, by using the same rule (4), into $x\ [\![x =_n i(y)]\!]$, but this cannot be narrowed further.

This already shows that Hullot's lifting lemma cannot be enounced under its 'usual' form in our context. But there is an additional point here. Suppose we had allowed 'unrestrictive' deletion of a current rule, whenever this gets simplified in the sense of our definitions of Section 2 ; in such a case we could have simplified rule (2) by (4), and replace it by :

$$(2') \qquad f(i(y), z) \doteq x \ [\![x = i(y)]\!]$$

If that were the case, the *only* way to normalize $g(a)$ would have been as above, and Hullot's lemma would be irremediably lost !

This explains why our Inference rule for rule-simplification is retrictive : in our approah (2) is **not** simplifiable into $(2')$ (rewriting *not* free). This gives the possibilty of generating two more new rules :

$$((3) \text{ into } (2)) : \quad (5) \qquad f(a, z) \doteq g(y) \ [\![y = a]\!]$$

$$((5) \text{ into } (1)) : \quad (6) \qquad g(y) \doteq x \ [\![y = a \wedge x = a]\!]$$

And rule (6) gives a single rewrite-step from $g(a)$ to a.

Hullot's difficulty in this Example comes from the fact that the rewrite-step on $g(a)$ was *not* S-quasi-normal. □

To prove that S-quasi-normal rewrite-steps are sufficient to normalize any (ground) term, we first need a lemma.

Lemma 1 *Let* $s \longrightarrow_{[p,(l \doteq r \ [\![c]\!]),\tau]} t$ *be an innermost S-quasi-normal rewrite step. Then τ is in fact in S-normal form on $V(r)$.*

Proof : We just have to consider τ on any variable $x \in V(l) \cap V(r)$.

Case 1. If l is a non-variable, $s|_p = \tau l$ is of size > 1, and all its proper sub-terms are irreducible, since our rewrite-step is innermost. So τx is S-irreducible.

Case 2. If l is a variable, then $l = x$; this case is impossible since x is also in $V(r)$ and $\tau l > \tau r$ under our cso. □

Remark. It follows from the proof of this lemma, that any *innermost S-quasi-normal* step has an S-normal match, *when the rule employed at this step is not a single variable to the left.* The combination 'innermost+S-quasi-normal' can thus be seen as generalizing the classical notion of rewrite-step with an S-normal-match.

4.2 S-quasi-normal Reduction steps Suffice

The following lemma is a key step prove this.

Lemma 2 *Let $rr \equiv g \doteq d \ [\![T]\!]$ be a rule deduced as a critical pair between two 'earlier' rules $g_1 \doteq d_1 \ [\![T_1]\!]$, $g_2 \doteq d_2 \ [\![T_2]\!]$, of S. Assume τ to be a ground substitution on the variables of rr, such that : $\tau g > \tau d$, and τ is a solution of $[\![T]\!]$. Then at least one of the following two conclusions holds :*

- *The term τg can be rewritten by a rule (in S) by an S-quasi-normal step.*
- *Or, there is a rule of the form $g \doteq d \ [\![\overline{T}]\!]$, obtained by basic superpositions between the two 'earlier' rules, and those employed in going from τ to an S-normal form $\overline{\tau}$ of τ* wrt the variables in $V(d) \setminus V(g)$, *such that $\overline{\tau}$ is a solution of the* 'equality' part *of $\overline{T}$.*

The complete details of the proof of this lemma are to be found in the thesis of J. Chabin, under preparation. We give here only the main ideas (similar to those employed in our above mentioned proof of ground confluence), that we illustrate first with an example.

Example 4 Consider E be the system :

$$
\begin{array}{ll}
(1) & f(h(x,y),x) \doteq f(a,y) \\
(2) & f(z,b) \doteq g(z) \\
(3a) & h(u,a) \doteq i(u,c) \\
(3b) & i(b,c) \doteq e
\end{array}
$$

under an lpo, where $f > g > h > i > a > b > c > e$. By superposition from (1) into (2), we get :

$$(4) \qquad f(a,y) \doteq g(z) \; [\![z = h(b,y)]\!]$$

Consider the ground term $t = f(a,a)$, rewritten by rule (4) into the term $g(z)$ $[\![z = h(b,y) \wedge y = a]\!]$ (our 'algorithm' gives the c-match $\{y \leftarrow a\}$) ; this is not an S-quasi-normal step, since $h(b,a)$ is reducible.

Now $h(b,a) \rightarrow_{(3a)} i(b,c) \rightarrow_{(3b)} e$, is an innermost S-rewrite-chain to a normal-form of $h(b,a)$. But then $(3a)$ superposes into rule (1) to give :

$$(5a) \qquad f(i(x,c),x) \doteq f(a,y) \; [\![y = a]\!]$$

and $(3b)$ overlaps into $(5a)$ to give :

$$(5b) \qquad f(e,x) \doteq f(a,y) \; [\![x = b \wedge y = a]\!]$$

But then this rule $(5b)$ overlaps into rule (2) to give :

$$(6) \qquad f(a,y) \doteq g(z) \; [\![y = a \wedge z = e]\!]$$

And this last rule rewrites our $t = f(a,a)$ into $g(z)$ $[\![z = e]\!]$, which is a (single) quasi-normal rewrite-step. $\square$

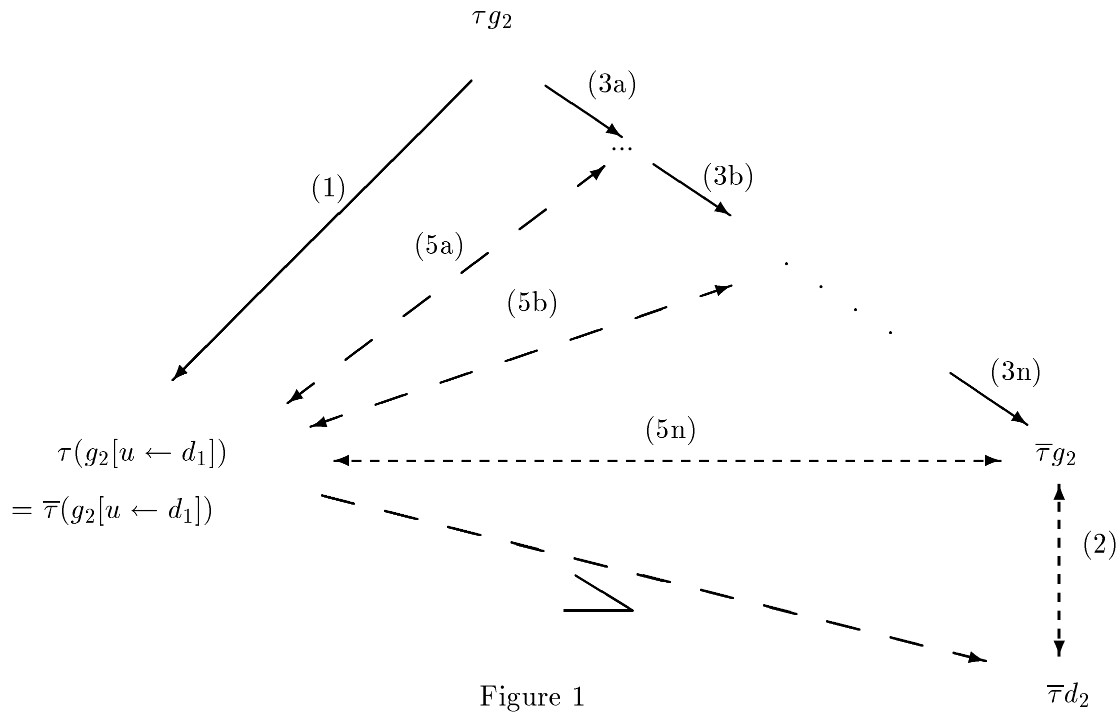

Figure 1

We are going to show that the general situation is quite similar, and can be represented by the following 'pyramidal diagram', where the slant arrows to the right correspond to the rewrite-steps 'suitably normalizing' the c-match used in the given initial rewrite-step. In our above example, this is obtained in two steps (so $n = b$), and the bottom triangle corresponds to the superposition giving the 'final rule'.

Proof : (of the Lemma - Sketch). The reasoning goes along the same lines as in the above example. Consider for example the case where the critical pair considered is of the form :

$$g = g_2[u \leftarrow d_1], d = d_2, T = T_1 \wedge T_2 \wedge (g_1 >_n d_1) \wedge (g_2 >_n d_2) \wedge (g_2|_u =_n g_1)$$

(or a rule gotten by simplifying such a rule).

If the rewrite step on τg defined by our rule and the match τ, is not S-quasi-normal, this means that there exist variables x of the *right-hand-side* d of the given rule (wrt τ), and not in g, such that τx can be rewritten by a rule in S. *We may assume that no normal form* $\overline{\tau}$ *of* τ *wrt the variables in* $V(d) \setminus V(g)$ *is a solution for the 'equality' part of* T (otherwise we may choose this as match, and we are done). The idea is then that the rules that lead us from τ to such an S-normal form suitably chosen, induce a 'sequence' of overlap instances with the earlier rules, at the end of which we must arrive at one of the two conclusions of our lemma. One proves in fact that for achieving this it suffices to *choose an innermost* S*-rewrite-chain from* τ *to an* S*-normal form* $\overline{\tau}$, wrt the variables in $V(d) \setminus V(g)$. If during such a normalization. at some stage, we get a rule $(5i), i \in \{a..n\}$, which rewrites *from* $\overline{\tau}g$, we have the first conclusion of the lemma.

In the situation where we have gone on upto $\overline{\tau}g_2$, *and* $\overline{\tau}g_2 > \overline{\tau}d_2$, we get an overlap from $(5n)$ into the 'earlier' rule (2), and then the resulting new rule corresponds to the second conclusion of the lemma. (This was what happened in Example 4 above).

The case where we have : $g = d_2, d = g_2[u \leftarrow d_1]$, is entirely similar. Only the roles of (1) and (2) get inverted in the above described deduction-schema. □

Proposition 1 *Let* t *be any given ground term. Then there is a chain of innermost and* S*-quasi-normal rewrite-steps from* t *to an* S*-normal form* $t\!\downarrow$*.*

Proof : If t is minimal, then t is S-normal, and there's nothing to prove. Assume then there is an S-rewrite-step from t to some t'. Then we will show that there is an innermost S-quasi-normal rewrite-step from t to some t"; we will be through by noetherian induction.

Write then $t \longrightarrow_{[p,(g \doteq d\ [\![T]\!]),\tau]} t'$. We may assume that this (assumed) rewrite-step is innermost in t; if this is already an S-quasi-normal step, we are done. So let's assume that it is not; this implies that for some variable x in d, not in g, τx is S-reducible. The only problematic case is when no S-normal form $\overline{\tau}$ of τ, wrt $V(d) \setminus V(g)$, is solution for T. Thus if T is empty, we are through; also, if the entire constraint T is 'old', our rewrite-step on t can be done via an appropriate propagated rule, which will be without constraint, and we will be through again. Remains then to consider the case where T has a 'new' component.

But then the rule concerned in our rewrite-step has been deduced as a critical pair between two 'earlier' rules; by applying the conclusions of lemma 2, we may also assume that an S-normal form $\overline{\tau}$ of τ, wrt $V(d) \setminus V(g)$, satisfies the 'equality part' of T. We can then conclude by induction on the length of the 'new' component in T (details in the thesis of J. Chabin). □

5 Completeness of Narrowing.

In this section we suppose our constrained system S to be confluent (and noetherian) on ground terms wrt the cso $>$. We first prove an appropriate version of Hullot's lifting lemma. For reasons of redaction, the substitutions in this section will have 'propagted' and 'non-propagated' parts (cf. Section 2); if $\sigma = \sigma_p \wedge \sigma_{np}$, is such a decomposition of a substitution σ, applying it to the constrained term t $[\![T]\!]$ gives (by definition) the constrained term $\sigma_p t$ $[\![T \wedge \sigma_{np}]\!]$. For two constrained terms t_1 $[\![c_1]\!]$, t_2 $[\![c_2]\!]$, with 'disjoint variables', we will say that they are 'equal' iff for any solution σ of $c_1 \wedge c_2$, we have σt_1 $[\![c_1]\!] = \sigma t_2$ $[\![c_2]\!]$ *syntactically.*

Lemma 3 (Hullot under Constraints) *Let t_0 $[\![c_0]\!]$ be a term, $l \doteq r$ $[\![c]\!]$ any rule in S and ρ_0 a substitution, such that $\rho_0 \in SOL(c_0)$ and ρ_0 is in S-normal form on the variables of $V(t_0)$. If $\rho_0 t_0 \longrightarrow_{[p,(l \doteq r,[\![c]\!]),\beta]} t'_1$ $[\![T']\!]$ is an innermost S-quasi-normal rewrite-step, then t_0 $[\![c_0]\!] \rightsquigarrow_{[p,(l \doteq r,[\![c]\!])]} t_1$ $[\![c_1]\!]$ and there exists a substitution $\rho_1 \in SOL(c_1)$ such that :*

- *t'_1 $[\![T']\!] = \rho_1 t_1$, and $\rho_1|_{V(t_1)}$ is S-normal,*
- *$\rho_1 = \rho_0[D(\rho_0)]$ and $D(\rho_0) \subseteq D(\rho_1)$.*

That is, such that the following diagram commutes :

$$\begin{array}{ccc} \rho_0 t_0 & \longrightarrow_{[p,(l \doteq r,[\![c]\!]),\beta]} & t'_1 \ [\![T']\!] \\ \uparrow \rho_0 & & \uparrow \rho_1 \\ t_0 \ [\![c_0]\!] & \rightsquigarrow_{[p,(l \doteq r,[\![c]\!])]} & t_1 \ [\![c_1]\!] \end{array}$$

Proof : We may assume that $V(l \doteq r\ [\![c]\!]) \cap (V(t_0\ [\![c_0]\!]) \cup D(\rho_0)) = \emptyset$. The substitution ρ_0 being assumed S-normal on $V(t_0)$, suppose $\rho_0 t_0$ is rewritten at position p with an equation of S; then p is a non-variable position in t_0. We have : t_0 $[\![c_0]\!] \rightsquigarrow_{[p,(l \doteq r,[\![c]\!])]} t_1$ $[\![c_1]\!]$ with $t_1 = t_0[p \leftarrow r]$ and $c_1 = c_0 \wedge c \wedge (l >_n r) \wedge (t_0|_p =_n l)$.

Let $\rho_1 = \beta \cup \rho_0$; this makes sense since $D(\beta) \cap D(\rho_0) = \emptyset$. Let us consider $x \in V(t_1)$. There are two possibilities :

- either $x \in V(t_0)$, and as ρ_0 is S-normal on $V(t_0)$, $\rho_1(x)$ is S-normal too,

- or $x \in V(r)$; in this case β is S-normal on $V(r)$ by Lemma 1, and consequently $\rho_1(x)$ is S-normal. Which proves first that ρ_1 is S-normal on $V(t_1)$.

In such a situation we have : $\circ$ $\rho_1 \in SOL(c_0)$ since $\rho_0 \in SOL(c_0)$
$\circ$ $\rho_1 \in SOL(c)$ since $\beta \in SOL(c)$
$\circ$ $\rho_1 l = \beta l > \beta r = \rho_1 r$ so $\rho_1 \in SOL(l > r)$
And finally $\circ$ $\rho_1(t_0|_p) = (\rho_0 t_0)|_p = \beta l = \rho_1 l$; thus $\rho_1 \in SOL(t_0|_p =_n l)$

That proves $\rho_1 \in SOL(c_1)$. One deduces immediately : $\rho_1(t_1\ [\![c_1]\!]) = t'_1\ [\![T']\!]$. It is clear that $\rho_1 = \rho_0[D(\rho_0)]$ and $D(\rho_0) \subseteq D(\rho_1)$ by the definition of our ρ_1. □

Here is an illustrative example for the above lemma.

Example . Consider the one rule system :

$$f(x, y) \doteq h(x, y)\ [\![y = g(z)]\!]$$

under an lpo, where $f > g > h > a$.

Consider the constrained term : $f(g(x'),y')\ [\![y' = g(z')]\!]$. Then the constrained ('ground') term $f(g(a),y')\ [\![y' = g(a)]\!]$ may be seen as its 'image' under the substitution $\rho_0 : \{x' \leftarrow_p a, z' \leftarrow_{np} a\}$. Now our rule rewrites this ground term into $t_1'\ [\![T']\!] = h(g(a),y)\ [\![y = g(a)]\!]$ under the c-match : $\{x \leftarrow_p g(a), y \leftarrow_{np} g(a)\}$. We indeed get a narrowing lift as below :

$$\begin{array}{ccc} f(g(a),y')\ [\![y' = g(a)]\!] & \longrightarrow_{[(l \doteq r, [\![c]\!]),\beta]} & h(g(a),y)\ [\![y = g(a)]\!] \\ \uparrow \rho_0 & & \uparrow \rho_1 \\ f(g(x'),y')\ [\![y' = g(z')]\!] & \rightsquigarrow_{(l \doteq r, [\![c]\!])} & t_1\ [\![c_1]\!] \end{array}$$

where $l \doteq r, [\![c]\!]$ is our rule, $t_1\ [\![c_1]\!] = h(x,y)\ [\![x' = g(x) \wedge y' = y \wedge y = g(z) \wedge y' = g(z')]\!]$, and ρ_1 is indeed the substitution $\beta \cup \rho_0$. $\Box$.

From this point on the proof of the completeness of narrowing (wrt unification) is 'standard'. Suppose that the given initial unificand $u \stackrel{?}{=} v$ has a solution; then there exists a substitution τ such that $\tau u \leftrightarrow_S \tau v$. By skolemizing the variables we may assume that τ is ground, so due to ground confluence of S, there exists an S-valley from τu to τv; we may now suppose that this valley has only innermost S-quasi-normal steps (Proposition 1). It is then immediate that these steps can be 'lifted' by (constrained) narrowing, to yield a substitution σ (in general more general than τ) such that $\sigma u =_S \sigma v$. Our theorem below is just the formal description of all this.

Theorem 1 (Narrowing is Complete) *Let $s_0 \stackrel{?}{=} t_0\ [\![c_0]\!]$ be a constrained equation, and suppose there exists a substitution $\rho_0 \in SOL(s_0 \stackrel{?}{=} t_0\ [\![c_0]\!])$, i.e. $\rho_0 s_0 =_S \rho_0 t_0$ and $\rho_0 \in SOL(c_0)$, such that :*

- *ρ_0 is S-normal on $V(s_0 \stackrel{?}{=} t_0)$,*
- *$\rho_0 s_0$ and $\rho_0 t_0$ are ground.*

Then there exists a derivation : $(s_0 \stackrel{?}{=} t_0\ [\![c_0]\!]) \rightsquigarrow^*_S (s_n \stackrel{?}{=} t_n\ [\![c_n]\!])$ and a substitution $\rho_n \in SOL(c_n)$ such that $\rho_n s_n = \rho_n t_n$ and ρ_n coïncides with ρ_0 on $D(\rho_0)$.

Proof : The proof is by noetherian induction on the (multiset) pair $\{\rho_0 s_0, \rho_0 t_0\}$. We have $\rho_0 s_0$ and $\rho_0 t_0$ ground, both having the same S-normal form; so we may assume that there is an S-valley between the two, where all the S-steps are innermost and S-quasi-normal (cf. Propostion 1).

Case 1. Where $\rho_0 s_0 = \rho_0 t_0$. Since $\rho_0 \in SOL(c_0)$, there is nothing to prove here.

Case 2. Let $(\rho_0 s_0 \stackrel{?}{=} \rho_0 t_0) \longrightarrow_S (s_1' \stackrel{?}{=} t_1')$ be the first (innermost and S-quasi-normal) reduction step of our unificand, in the given valley. By Hullot's lemma we have $(s_0 \stackrel{?}{=} t_0\ [\![c_0]\!]) \rightsquigarrow_S (s_1 \stackrel{?}{=} t_1\ [\![c_1]\!])$ and there exists a substitution $\rho_1 \in SOL(c_1)$, S-normal on $V(s_1 \stackrel{?}{=} t_1)$ such that $(s_1' \stackrel{?}{=} t_1') = \rho_1(s_1 \stackrel{?}{=} t_1)$, $\rho_1 = \rho_0[D(\rho_0)]$ and $D(\rho_0) \subseteq D(\rho_1)$.

Moreover, $(s_1 \stackrel{?}{=} t_1\ [\![c_1]\!]) \longrightarrow^*_S (s_1{\downarrow} \stackrel{?}{=} t_1{\downarrow}\ [\![c_1]\!])$; so, $(s_1' \stackrel{?}{=} t_1') = \rho_1(s_1 \stackrel{?}{=} t_1) \longrightarrow^*_S \rho_1(s_1{\downarrow} \stackrel{?}{=} t_1{\downarrow})$. And we get the following situation :

$$\begin{array}{ccccc} \rho_0(s_0 \stackrel{?}{=} t_0) & \rightarrow_S & s_1' \stackrel{?}{=} t_1' & \rightarrow^*_S & \rho_1(s_1{\downarrow} \stackrel{?}{=} t_1{\downarrow}) \\ \uparrow \rho_0 & & \uparrow \rho_1 & & \uparrow \rho_1 \\ s_0 \stackrel{?}{=} t_0\ [\![c_0]\!] & \rightsquigarrow_S & s_1 \stackrel{?}{=} t_1\ [\![c_1]\!] & \rightarrow^*_S & s_1{\downarrow} \stackrel{?}{=} t_1{\downarrow}\ \ [\![c_1]\!] \\ & & & & \\ s_0 \stackrel{?}{=} t_0\ [\![c_0]\!] & & \rightsquigarrow_S & & s_1{\downarrow} \stackrel{?}{=} t_1{\downarrow}\ \ [\![c_1]\!] \end{array}$$

Let us prove now that $s_1\!\downarrow \stackrel{?}{=} t_1\!\downarrow [\![c_1]\!]$ and ρ_1 verify again the hypotheses of our theorem. Now $\rho_1 \in SOL(c_1)$, $\rho_0 s_0$ is ground and $\rho_0 s_0 \rightarrow^*_S \rho_1 s_1\!\downarrow$; so $\rho_1 s_1\!\downarrow$ is ground, as well as $\rho_1 t_1\!\downarrow$. But S is (assumed) confluent and noetherian on ground terms, thus $\rho_1(s_1\!\downarrow \stackrel{?}{=} t_1\!\downarrow) \longrightarrow^*_S (u \stackrel{?}{=} u)$. Which means that $\rho_1 s_1\!\downarrow =_S \rho_1 t_1\!\downarrow$, therefore $\rho_1 \in SOL(s_1\!\downarrow \stackrel{?}{=} t_1\!\downarrow [\![c_1]\!])$ and ρ_1 is S-normal on $V(s_1 \stackrel{?}{=} t_1) \supseteq V(s_1\!\downarrow \stackrel{?}{=} t_1\!\downarrow)$.

We may apply the induction hypothesis on the pair $(s'_1 = \rho_1(s_1)) \stackrel{?}{=} (t'_1 = \rho_1(t_1))$ which is smaller than the pair we started with. So there exists a normalizing narrowing derivation $s_1\!\downarrow \stackrel{?}{=} t_1\!\downarrow [\![c_1]\!] \quad \rightsquigarrow^*_S \; s_n \stackrel{?}{=} t_n \; [\![c_n]\!]$ and a substitution $\rho_n \in SOL(c_n)$ such that $\rho_n s_n = \rho_n t_n$, and $\rho_n = \rho_1[D(\rho_1)]$. As $\rho_1 = \rho_0[D(\rho_0)]$ and $D(\rho_0) \subseteq D(\rho_1)$, we may deduce that $\rho_n = \rho_0[D(\rho_0)]$. □

6 Further Examples, Comments and Implementation

The following example shows that, in actual practice, narrowing may often need much fewer equations than ground confluence.

Example 5 Consider the (non-ground-confluent) system E :

$$\begin{array}{ll} (1) & f(x) \doteq b \; [\![x = g(h(y))]\!] \\ (2) & h(a) \doteq b \\ (3) & g(a) \doteq b \end{array}$$

under an lpo where : $f > g > h > a > b$.

Let's try to solve : $f(g(x)) \stackrel{?}{=} b$, wrt E. The unificand can get narrowed by rule (3) or rule (1). We get the following situation (where for readability's sake we have suppressed the 'ordering constraints', which are all trivially satisfied).

i) $f(g(x)) \stackrel{?}{=} b) \quad \rightsquigarrow_{(3)} \quad (f(b) \stackrel{?}{=} b \; [\![g(x) =_n g(a)]\!] \quad \rightsquigarrow_{(1)}$

$$b \stackrel{?}{=} b \; [\![(g(x) =_n g(a)) \wedge (f(x') =_n f(b)) \wedge (x' = g(h(y))]\!]$$

This leads to no solution since the final constraint is unsatisfiable.

ii) $f(g(x)) \stackrel{?}{=} b \quad \rightsquigarrow_{(1)} \quad b \stackrel{?}{=} b \; [\![(f(g(x) =_n f(x')) \wedge (x' = g(h(y))]\!]$

Which gives the solution $\{x \leftarrow h(y)\}$. □

The following example highlights the practical role played by inference rule $n^o 8$, without which narrowing wouldn't find the non-ground solutions here.

Example 6 Consider now the AC system E :

$$\begin{array}{ll} (1) & x * y \doteq y * x \\ (2) & (x * y) * z \doteq x * (y * z) \end{array}$$

under an lpo where : $* > a > b$ and $*$ has the left-to-right status.

Let's try to find the (non-ground) solution $x = a * u, y = b * z$ for the unificand $(x*b)*z \stackrel{?}{=} (a*y)*u$, wrt E (where u, z are variables). We get the following situation (where for readability, we have *not* accumulated the constraints successively, and suppressed the trivially satisfied 'ordering' constraints) :

$((x * b) * z \stackrel{?}{=} (a * y) * u)$
$\rightsquigarrow_{(2)} (x_1 * (y_1 * z_1)) * z \stackrel{?}{=} (a * y) * u \; [\![(x =_n x_1 * y_1) \wedge (z_1 =_n b)]\!]$
$\rightsquigarrow_{(1)} (x_1 * (y_2 * x_2)) * z \stackrel{?}{=} (a * y) * u \; [\![(x_2 =_n y_1) \wedge (y_2 =_n z_1) \wedge (x_2 * y_2 >_n y_2 * x_2)]\!]$
$\rightsquigarrow_{(2)} x_3 * (y_3 * z_3) \stackrel{?}{=} (a * y) * u \; [\![(x_3 =_n x_1) \wedge (y_3 =_n y_2 * x_2) \wedge (z_3 =_n z)]\!]$
$\rightsquigarrow_{(2)} x_3 * (x_4 * (y_4 * z_4)) \stackrel{?}{=} (a * y) * u \; [\![(y_3 =_n x_4 * y_4) \wedge (z_4 =_n z_3)]\!]$
$\rightsquigarrow_{(1)} x_3 * (x_4 * (y_5 * x_5)) \stackrel{?}{=} (a * y) * u \; [\![(x_5 =_n y_4) \wedge (y_5 =_n z_4) \wedge (x_5 * y_5 >_n y_5 * x_5)]\!]$
$\rightsquigarrow_{(2)} x_3 * (x_4 * (y_5 * x_5)) \stackrel{?}{=} x_6 * (y_6 * z_6) \; [\![(x_6 =_n a) \wedge (y_6 =_n y) \wedge (z_6 =_n u)]\!]$
$\rightsquigarrow_{(2)} x_3 * (x_4 * (y_5 * x_5)) \stackrel{?}{=} x_6 * (x_7 * (y_7 * z_7)) \; [\![(y_6 =_n x_7 * y_7) \wedge (z_7 =_n z_6)]\!]$

If we apply '*Generation of solution*' we get the constraint : $[\![(x_3 =_n x_6) \wedge (x_4 =_n x_7) \wedge (y_5 =_n y_7) \wedge (x_5 =_n z_7) \wedge (x =_n x_1 * y_1) \wedge (z_1 =_n b) \wedge (x_2 =_n y_1) \wedge (y_2 =_n z_1) \wedge (x_3 =_n x_1) \wedge (y_3 =_n y_2 * x_2) \wedge (z_3 =_n z) \wedge (y_3 =_n x_4 * y_4) \wedge (z_4 =_n z_3) \wedge (x_5 =_n y_4) \wedge (y_5 =_n z_4) \wedge (x_6 =_n a) \wedge (y_6 =_n y) \wedge (z_6 =_n u) \wedge (y_6 =_n x_7 * y_7) \wedge (z_7 =_n z_6) \wedge (x_2 * y_2 >_n y_2 * x_2) \wedge (x_5 * y_5 >_n y_5 * x_5)]\!]$.

We get $x \leftarrow x_1 * y_1 = x_3 * x_2 = x_6 * y_4 = a * x_5 = a * z_7 = a * z_6 = a * u$, and $y \leftarrow y_6 = x_7 * y_7 = x_4 * y_5 = y_2 * z_4 = z_1 * z_3 = b * z$; which gives the solution $\{x \leftarrow a * u, y \leftarrow b * z\}$ (after '*Simplify a solution*', because the constraint is satisfiable for $\{u \leftarrow a, z \leftarrow b\}$ for example). □

Remark 1. Keeping our unificands normalized means in effect that the 'narrowing substitution', when calculated, is in normal form. This leads to the practical consequence that narrowing along branches with non-normal constraints is not needed.

Example 7 Consider the system E :

$$(1) \qquad h(0, x) \doteq x$$
$$(2) \qquad f(0) \doteq 0$$

under an lpo where : $f > h > 0$. And let's try to solve : $h(x, f(x)) \stackrel{?}{=} 0$, wrt E. The unificand can get narrowed by rule(1) or rule (2). We get the following situations :

i) $h(x, f(x)) \stackrel{?}{=} 0 \;\; \rightsquigarrow_{(1)} \;\; x_1 \stackrel{?}{=} 0 \; [\![(h(x, f(x)) =_n h(0, x_1)) \wedge (h(0, x_1) >_n x_1)]\!]$

Which gives the unificand $x_1 \stackrel{?}{=} 0 \; [\![(x =_n 0) \wedge (f(0) =_n x_1)]\!]$). If we had normalized the constraint, we would have found the solution $\{x \leftarrow 0\}$.

ii) But this solution is in fact calculated by another branch :

$$h(x, f(x)) \stackrel{?}{=} 0 \;\; \rightsquigarrow_{(2)} \;\; h(x, 0) \stackrel{?}{=} 0 \; [\![f(x) =_n f(0)]\!]$$
$$\rightsquigarrow_{(1)} \;\; x_1 \stackrel{?}{=} 0 \; [\![(f(x) =_n f(0)) \wedge (h(x, 0) =_n h(0, x_1))]\!] \qquad \square.$$

Remark 2. It is generally believed that the approach via constraints may play a useful role in equational deduction with 'equality modulo AC'; the reason being that several of the unification branches may get hopefully eliminated, the constraints concerned becoming 'obviously unsatisfiable'. We may point out here that our inference system would remain valid when the 'equality' is modulo AC (instead of being syntactic), under the provisos :

i) that the underlying reduction ordering is an AC-compatible cso,

ii) whenever a unification process is assumed done, the possibility of 'multiple output' is taken into account, by the generation of more than one new rules or unificands.

Remark 3. In general we need to superpose with the propagated rules generated by Inference Rule $n^o 2$ (which explains why such rules are put into the set E, and not into R). Here is an easy example to get convinced of this fact : If $E = \{f(x) \doteq y \; [\![x = i(y)]\!], \; i(0) \doteq 0\}$, we have : $f(0) =_E 0$, and this cannot be proved without such a superposition.

6.1 UREVEAL : Implementation of the approach

UREVEAL is a prototype version of the theorem-prover REVEAL, destined as a tool for unification, via the above described mechanism of constrained ordered completion+narrowing. (Both are developed at the LIFO, are written in C, and are available through public ftp). Despite the fact that we (a priori) allow for fewer rule simplification steps than in [2], our implementation seems quite efficient as is shown by the Examples below.

Example 8 (A Comaparison between usual completion and our 'basic' completion procedure).

For the standard 3-rule specification for groups, our (basic) completion procedure generates a final system of 10 rules, exactly comparable with that obtained via non-basic completion; the number of critical pairs calculated seems to be the same either way (note however that this is very strategy dependent).

On the contrary, here is an Example for which 'basic' completion generates fewer intermediary rules than usual completion (this would suggest that there might be strategies which might render basic completion in a sense more economical).

Consider the system :

$$E \; : \; \begin{cases} g(f(x)) \doteq g(x) & (1) \\ f(i1(x, h(y))) \doteq a & (2) \\ i1(0, x) \doteq x & (3) \\ h(0) \doteq 0 & (4) \end{cases}$$

under an lpo where $f > i1 > h > 0 > a$.

Usual completion computes the set of folowing rules :

(1)	$g(f(x)) \to g(x)$	
(2)	$f(i1(x, h(y))) \to a$	
(3)	$i1(0, x) \to x$	
(4)	$h(0) \to 0$	
(5)	$g(i1(x, h(y))) \to g(a)$	
(6)	$f(i1(x, 0)) \to a$	
(7)	$f(h(x)) \to a$	
(8)	$g(h(x)) \to g(a)$	
(9)	$f(0) \to a$	
(10)	$g(0) \to g(a)$	
(11)	$g(i1(x, 0)) \to g(a)$	
(12)	$f(0) \doteq a$	simplified with (9)
(13)	$g(0) \doteq g(a)$	simplified with (10)
(14)	$g(0) \doteq g(a)$	simplified with (10)
(15)	$g(i1(x, 0)) \doteq g(a)$	simplified with (11)
(16)	$g(h(x)) \doteq g(a)$	simplified with (8)

Basic completion computes only the first 11 critical pairs :

(1) $g(f(x)) \rightarrow g(x)$ $[\![empty]\!]$
(2) $f(i1(x, h(y))) \rightarrow a$ $[\![empty]\!]$
(3) $i1(0, x) \rightarrow x$ $[\![empty]\!]$
(4) $h(0) \rightarrow 0$ $[\![empty]\!]$
(5) $g(x) \rightarrow g(a)$ $[\![x = i1(y, h(z))]\!]$
(6) $f(x) \rightarrow a$ $[\![x = h(y)]\!]$
(7) $f(i1(x, 0)) \rightarrow a$ $[\![empty]\!]$
(8) $g(x) \rightarrow g(a)$ $[\![x = h(y)]\!]$
(9) $f(x) \rightarrow a$ $[\![x = 0]\!]$
(10) $g(x) \rightarrow g(a)$ $[\![x = i1(y, 0)]\!]$
(11) $g(x) \rightarrow g(a)$ $[\![x = 0]\!]$

Example 9 Here is a Prolog type problem, where narrowing does a much better job than Prolog. Consider the system :

$$E \ : \ \begin{cases} inf(x, x) \doteq false & (1) \\ inf(b, a) \doteq false & (2) \\ inf(c, b) \doteq false & (3) \\ inf(d, c) \doteq false & (4) \\ inf(a, b) \doteq true & (5) \\ inf(b, c) \doteq true & (6) \\ inf(c, d) \doteq true & (7) \\ et(inf(x, z), et(inf(x, y), inf(y, z))) \doteq et(inf(x, y), inf(y, z)) & (8) \\ et(true, x) \doteq x & (9) \\ et(false, x) \doteq false & (10) \\ et(x, true) \doteq x & (11) \\ et(x, false) \doteq false & (12) \end{cases}$$

under an lpo where $et > inf > a > b > c > d > true > false$

For the unificand $inf(a, x) \stackrel{?}{\doteq} true$, UREVEAL computes the following solutions (after constrained completion).

```
*** NEW * Solution ***  :  [[ x =n b]]

*** NEW * Solution ***  :  [[ x =n c]]

*** NEW * Solution ***  :  [[ x =n d]]
```

Example 10 (Distributive Theory) Consider the theory E with the 2 rules :

$$F(x, G(y, z)) = G(F(x, y), F(x, z))$$

$$F(G(x, y), z) = G(F(x, z), F(y, z))$$

and the unificand $F(a, x) \stackrel{?}{\doteq} F(x, y)$.

The solutions below, quickly obtained by UREVEAL, show why there are inifinitely many solutions for our unificand here :

*** NEW * Solution *** : $[[x =_n a \wedge y =_n a]]$

*** NEW * Solution *** : $[[x =_n G(a,a) \wedge y =_n a]]$

*** NEW * Solution *** : $[[x =_n G(G(a,a),a) \wedge y =_n a]]$

*** NEW * Solution *** : $[[x =_n G(a,G(a,a)) \wedge y =_n a]]$

*** NEW * Solution *** : $[[x =_n G(G(a,a),G(a,a)) \wedge y =_n a]]$

7 Conclusion

We believe that our inference rules give a more efficient method for E-unification, than the usual approaches of Gallier-Snyder-Dougherty. The performance of our narrowing can (and has to) be improved still further :

- by introducing suitable goal-directed strategies (as in [1]) in particular to have a better control on the generation of the propagated rules, or for better rule selection for the narrowing steps,

- and also by making use of a suitable 'graph of operators' à la Dershowitz (cf. [3]).

These aspects will be developed in the thesis of J. Chabin, as well as the extensions needed to cover the case of 'AC-equality', and applications to PROLOG programs seen as rewrite programs. A word perhaps about basic superposition. From our experiments so far, we have not been to conclude that this gives systematically a more economical completion process as compared to the usual 'full simplification strategy'; in particular, it seems a tough job to be able to apply 'indexing' techniques for basic simplification. However, for E-unification via narrowing wrt theories not admitting a finite canonical system, we may *have to* generate new rules, therefore not generating too many 'unnecessary' intermediary rules may turn out to be helpful. Finally even if we choose to drop the basic aspect in our ordered completion, the 'basicity' of our narrowing part - which is quite independent of that of superposition - has obviously to be maintained for reasons of efficiency.

References

[1] S. ANANTHARAMAN and N. ANDRIANARIVELO. Heuristic criteria in refutational theorem proving. In *PROCEEDINGS OF DISCO'90*, volume 429 of *Lecture Notes in Computer Science*. Spring-Verlag, 1990.

[2] L. BACHMAIR, H. GANZINGER, C. LYNCH, and W. SNYDER. Basic paramodulation and superposition. In *Proceedings 11th International Conference on Automated Deduction, Albany (NY, USA)*, 1992.

[3] N. DERSHOWITZ and G. SIVAKUMAR. Goal-directed equation solving. In *PROCEEDINGS OF THE SEVENTH NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, pages 166–170, ST. PAUL, MN (USA), August 1988.

[4] J.-M. HULLOT. Canonical forms and unification. In *Proceedings 5th International Conference on Automated Deduction, Les Arcs (France)*, pages 318–334, July 1980.

[5] C. KIRCHNER, H. KIRCHNER, and M. RUSINOWITCH. Deduction with symbolic constraints. *Revue d'Intelligence Artificielle*, 4(3):9–52, 1990. SPECIAL ISSUE ON AUTOMATIC DEDUCTION.

[6] R. NIEUWENHUIS and A. RUBIO. Theorem Proving with Ordering Constrained Clauses. In *Proceedings 11th International Conference on Automated Deduction, Albany (NY, USA)*, volume 617 of *Lecture Notes in Artificial Intelligence*. Spring-Verlag, 1992.

[7] R. NIEUWENHUIS and A. RUBIO. Basic superposition is complete. In *Proceedings of the European Symposium on Programming*, Rennes, France, February 1992.

[8] A OHSUGA and K. SAKAI. Complete equational unification based on an extension of the knuth-bendix completion procedure. In K. Schulz, editor, *Proc. of the First International Workshop on Word Equations and Related Topics*, volume 572 of *Lecture Notes in Computer Science*, pages 197–209. Spring-Verlag, October 1990.

[9] P. RÉTY. Improving basic narrowing. In P. LESCANNE, editor, *Proceedings 2nd Conference on Rewriting Techniques and Applications, Bordeaux (France)*, volume 256 of *Lecture Notes in Computer Science*, pages 228–241, BORDEAUX (FRANCE), May 1987. Spring-Verlag.

[10] W. SNYDER and C. LYNCH Basic paramodulation. In *Research Report, Boston University (1991), Abstracts - UNIF'91, Barbizon (France)*

[11] W. SNYDER and C. LYNCH Redundancy Criteria for Constrained Completion. In *Proc. of the International Conference on Rewriting, RTA-93 (Montreal), to appear.*