A Quantifier Elimination Algorithm in a FOL with Equality

Luís Damas
Nelma Moreira
University of Porto, Campo Alegre 823, 4100 Porto, Portugal
{luis,nam}@ncc.up.pt

December 1991

1 Introduction

Let $Vars$ be a countable set of variables x, y, z, ... and Fun be a countable set of n-ary function symbols f, g, h, $l \ldots$,$n \geq 0$. If f has arity n we will denote it by f^n,whenever necessary. If n=0 f is called *atomic* and we will omit its arity. Let T be the term algebra over Fun and $Vars$ and T_0 be the corresponding set of ground terms.

Let C be a first order language over T with equality as the only predicate symbol.

We introduce a path notation p to a allow reference to a specific argument of a complex term avoiding the need for introducing existencial quantifiers and extraneous variables. So we extend C to expressions involving paths, which are called *values*. If t denotes terms, p paths, v values and c formulas of C we have:

$$
\begin{array}{lll}
t & ::= & x \\
 & & f^n(t_1, \ldots, t_n) \qquad n \geq 0 \\
p & ::= & \varepsilon \\
 & & pf^n\pi_i \qquad 1 \leq i \leq n \\
v & ::= & t \\
 & & v.p \\
 & & \perp \\
c & ::= & t.pf^n \\
 & & v = v \\
 & & false \\
 & & \neg c \\
 & & c \wedge c \\
 & & c \vee c \\
 & & c \rightarrow c \\
 & & \exists x\ c \\
 & & \forall x\ c
\end{array}
$$

where

$$
\begin{array}{rcl}
c \rightarrow c & \equiv & \neg c \vee c \\
\forall x \ c & \equiv & \neg \exists x \ \neg c
\end{array}
$$

In the above definitions π_i denotes de ith projection of a term. Formulas of the form $t.pf^n$ will be called *path formulas.* Given a formula c, $Vars(c)$ will denote the set of variables occurring in c. Given v and v', $v \equiv v'$ if they are the same value.

We take as semantic model for C the Herbrand model with the usual semantics for the logic connectives. Given a term t its denotation is

$$[\![t]\!] = \{St \in T_0\}$$

where S is a substitution of terms for variables. We consider two formulas are *equivalent* concerning the semantic model.

2 Reduced formulas

Definition 1 (Slots) *A value s is a* slot *if*

$$s \quad := \quad x \mid x.p$$

where x denotes variables and p paths.

Definition 2 *The length of a* path *p, $|p|$ is inductively defined by:*

1. $|\epsilon| = 0$
2. $|pf^n\pi_i| = 1 + |p|$

Definition 3 (Reduced values) *A value v is* reduced *if it is either a term or a slot.*

Proposition 1 *Any value can be rewritten to an equivalent* reduced value *or $\perp$ using the following rewriting rules:*

$$
\begin{array}{rcl}
f(t_1, \ldots, t_n).f^n\pi_i p & \longrightarrow & t_i.p \\
f(t_1, \ldots, t_n).g^k\pi_i p & \longrightarrow & \perp \\
 & & \textit{if } f^n \neq g^k
\end{array}
$$

Proof.

If v is a term or a slot then it is already in *reduced form.* Otherwise it is of the form $f^n(t1, ..., tn).p$. If $n = 0$ then if $p = \epsilon$ v is *reduced form* else v is rewritten to $\perp$. If $n > 0$ let $m = |p|$ be the *length* of p. Applying the rewritting rules at most m times we obtain a term or a slot or $\perp$. •

Definition 4 (Reduced equalities) *An equality c is in* reduced form *if it is*

$$c \quad ::= \quad s = s' \mid s = a$$

where a is an atomic term.

Proposition 2 *An equality $c := v = v'$ where v and v' are* reduced values*, can be rewritten to an equivalent conjuction of* reduced equalities *or $false$ or $\neg false$ using the following rewriting rules:*

$$\begin{array}{lcll}
v = v' & \longrightarrow & false & \textit{if either } v \textit{ or } v' \textit{ is } \perp \\
v = v' & \longrightarrow & false & \textit{if } v \textit{ and } v' \textit{ are atomic and } v \neq v' \\
v = v' & \longrightarrow & \neg false & \textit{if } v \textit{ and } v' \textit{ are the same value} \\
f(t_1, \ldots, t_n) = f(u_1, \ldots, u_n) & \longrightarrow & t_1 = u_1 \ \wedge \ \ldots & \wedge \ t_n = u_n \\
f(t_1, \ldots, t_n) = g(u_1, \ldots, u_n) & \longrightarrow & false & \\
x.p = f(t_1, \ldots, t_n) & \longrightarrow & x.pf^n\pi_1 = t_1 & \wedge \ \ldots \ \wedge \ x.pf^n\pi_n = t_n, \ n > 0 \\
t = s & \longrightarrow & s = t & \textit{if } t \textit{ is not a variable}
\end{array}$$

Definition 5 (Reduced formulas) *A formula c is in* reduced form *it has only reduced equalities and path formulas of the form $x.pf^n$.*

Proposition 3 *Any formula c can be converted to* reduced form *by reducing all equalities and for each* path formula*, $t.pf^n$ reducing the value $t.p$ and using the following rewriting rules:*

$$\begin{array}{rcl}
\perp.f^n & \longrightarrow & false \\
f(t_1, \ldots, t_n).f^n & \longrightarrow & \neg false \\
f(t_1, \ldots, t_n).g^k & \longrightarrow & false \\
& & \textit{if } f^n \neq g^k
\end{array}$$

Definition 6 (Prenex reduced formulas) *A formula is in* prenex reduced form *if it is a reduced formula in prenex normal form*

$$Q_1 x_1 \ldots Q_n x_n \, c$$

where $n \geq 0$, c is a reduced formula and $Q_i \equiv \exists$ or $Q_i \equiv \forall$, for $i = 1, \ldots, n$.

Definition 7 (Normal formulas) *A formula c is a* normal formula *if it is a* prenex reduced formula *in disjunctive normal form.*

$$\begin{array}{lcl}
c & ::= & cc \\
& & cc \vee cc \\
& & \exists x \ c \\
& & \forall x \ c \\
cc & ::= & ci \\
& & ci \wedge ci \\
ci & ::= & s = s' \\
& & s = a \\
& & x.pf^n \\
& & \neg ci
\end{array}$$

Reduced equalities and path formulas $x.pf^n$ will be called *positive literals* and its negations *negative literals*.

3 Elimination of existencial quantifiers

Definition 8 *Let p be a path and v a value, $p(v)$ is defined inductively by:*

1. $\epsilon(v) = v$
2. $f^n.\pi_i.p(v) = f^n(z_1, \ldots, z_{i-1}, p(v), z_{i+1}, \ldots, z_n)$ *where* $z_1, \ldots, z_n$ *are new variables*

Note that if v is a term also is $p(v)$.

Definition 9 *The* compatiblity *of Two paths p and q, $p \approx q$ is defined as follows:*

1. $\epsilon \approx \epsilon$
2. $f^n\pi_i p \approx f^n\pi_i q$ *iff* $p \approx q$
3. $f^n\pi_i p \approx f^n\pi_j q$ *and* $i \neq j$

If two paths p and q are *compatible* then the terms $p(z)$ and $q(z)$ are *unifiable*, where z is a new variable.

3.1 Elimination Algorithm

Let $[v/x]c$ be the formula obtainned from c replacing all free ocurrences of x with v.

Considering the rewriting rules of figure 3.1, the elimination algorithm is as follows:

Input: A formula c

Output: A quantifier free formula or report failure.

Step 1 Put c in normal form.

Step 2 If there exists quantified variables then select the innermost quantification else go to step 4. If x is universally quantified, $\forall xc$ (with c quantifier free) then proceed to step 3 with $\exists x \neg c$ and in the end of step 3 negate the resulting formula; otherwise go to step 3.

Step 3 If necessary apply Rule I. For every disjunct of the form $\exists x\ c$:

1. Apply rule II.
2. Or:
   (a) find a formula of the form $x = v$ or $v = x$ and apply a rewrite rule of group I (rearranging if necessary the conjuncts).
   (b) if not found, find one of the form $x.p = v$ or $v = x.p$ or $x.pf^n$ and apply a rewrite rule of group II.
   (c) if x only appears in negative literals then, if the Herbrand universe, $\mathcal{H}$ is infinite, either we can reduce that formulas to $\neg false$ or $false$ by applying a rewrite rule group III to each one.

Rule I

$$\exists x\ c_1 \vee c_2 \quad \longrightarrow \quad \exists x\ c_1 \vee \exists x\ c_2$$

Rule II

$$\exists x\ c \quad \longrightarrow \quad c \quad \text{if } x \text{ does not occur free in } c$$

Group I

$$\begin{array}{lll l}
\exists x\ x = u & \longrightarrow & \neg false & \text{if } u \text{ is atomic or a variable} \\
\exists x\ x = y.pf^n\pi_i & \longrightarrow & y.pf^n & \text{if } x \not\equiv y \\
\exists x\ x = u \wedge c & \longrightarrow & [u/x]c & \text{if } u \text{ is atomic or a variable} \\
\exists x\ x = y.pf^n\pi_i \wedge c & \longrightarrow & [y.pf^n\pi_i/x]c \wedge y.pf^n & \text{if } x \not\equiv y
\end{array}$$

Group II

$$\begin{array}{lll r}
\exists x\ x.p = u & \longrightarrow & \neg false & \\
 & & \text{if } u \not\equiv x \text{ and } u \text{ is atomic or a variable} & \\
\exists x\ x.p = x.q & \longrightarrow & \neg false & \text{if } p \approx q \\
\exists x\ x.p = y.qf^n\pi_i & \longrightarrow & y.qf^n & \text{if } x \not\equiv y \\
\exists x\ x.p = y.q \wedge c & \longrightarrow & \exists z \exists z_1 \ldots \exists z_m [p(z)/x](z = y.q \wedge c) & \\
 & & \text{where if } x \equiv y \text{ then } p \approx q & \\
\exists x\ x.p = u \wedge c & \longrightarrow & \exists z_1 \ldots \exists z_m [p(u)/x]c & \\
 & & \text{if } u \equiv x \text{ and } u \text{ is atomic or a variable} & \\
\exists x\ x.pf^n \wedge c & \longrightarrow & \exists z_1 \ldots \exists z_m [p(f(z_1, \ldots, z_n))/x]c & m \geq n
\end{array}$$

Group III

$$\begin{array}{lll}
\exists x\ \neg x = x & \longrightarrow & false \\
\exists x\ \neg x.p = v & \longrightarrow & \neg false \quad \text{if } x \text{ does not occur free in } v \\
\exists x\ \neg x.pf^n & \longrightarrow & \neg false
\end{array}$$

Figure 1: Rewriting rules for quantifier elimination.

In any case perform step 1 to the resulting formula. Go to step 2.

3. Otherwise halt with failure.

Step 4 If rule I has been applied, produce a final normal fomula and halt.

In order to prove the correctness of the algorithm we define a norm on paths, slots and formulas. These norms are related to the number of new variables that can be introduced.

Definition 10 (Norms) *1. Let p be a path, the* norm *of p, $\|p\|$, is defined as follows:*

(a) $\|\epsilon\| = 0$

(b) $\|pf^n\pi_i\| = (n-1) + \|p\|$

2. Let s be a slot, the norm *of s, $\|s\|$, is defined as:*

(a) $\|x\| = 1$

(b) $\|x.p\| = \|p\| + 1$

3. (a) Let c be a positive literal*, the norm of $\|c\|$ is define as:*

i. $\|s = s_1\| = \|s\| + \|s_1\|$

ii. $\|s = a\| = \|s\|$ *where a is atomic*

iii. $\|x.pf^n\| = \|x.p\| + n$

*(b) Let x be a variable and c be a*positive literal *the norm of c with respect to x, $\|c : x\|$ is defined by:*

i. $\|c : x\|$ *if x does not occur in c*

ii. $\|s = s_1\| = \|s\| + \|s_1\|$ *is x occurs in s and s_1*

iii. $\|s = s_1\| = \|s\|$ *is x occurs only in s*

iv. $\|s = s_1\| = \|s_1\|$ *is x occurs only in s_1*

v. $\|x.p = a : x\| = \|x.p\|$ *where a is atomic*

vi. $\|x.pf^n\| = \|x.p\| + n$

4. (a) Let c be formula and n be the number of positive literals *of c,c_i. Given a variable x, the norm of c with respect to x, $\|c : x\|$, is defined as follows:*

$$\|c : x\| = \sum_{i=1}^{n} \|c_i : x\|$$

where some p_i can be ϵ.

(b) Let c be a formula, the norm of c, $\|c\|$, is defined by:

$$\|c\| = \sum_{x \in Vars(c)} \|c : x\|$$

The idea of this last definition is that new existencial quantified variables can only be produced by positive literals. The norm of slots in negative literals can be arbitratly great.

Lema 1 *Let $cc \equiv \exists x\ c$ be a formula such that c is a quantifier free formula. The elimination algorithm converts cc to an equivalent quantifier free normal formal c'.*

Proof.

Step 1 produces a normal formula $cc' \equiv \exists x\ c'$, c' quantifier free. Rule I transforms cc' in a disjunction of formulas $\exists x\ c_i$. It suffices to show that x is eliminate from each one.

The rewriting rules ensure that x is eliminated from c_i and that equivalence is preserved. If no rule can be applied then failure is reported (cases where x appears in the right hand of equalities).

As in the elimination process new existencial quantified variables are introduced, we must show that the process always halts.

If rule II is applied, the algorithm halts and the result is a quantifier free formula. In the same way, if a rewrite rule of group I, finitly many of group III or one of the first three rules of group II is applied, x is eliminated, the substitutions do not introduce more variables or positive literals and the algorithm obviously halts with a quantifier free (normal) formula.

Otherwise, let $\|c_i\| = m$ and $\|c_i : x\| = k$, $k \leq m$.

We prove by induction on k.

If $k = 1$ then one of the rules of group I must apply.

Suppose valid for any value less than k.

Let $x.p_m = u$ be the first occurence of x in positive literals of c_i, where $m = \|p_m\|$.

Suppose, without loss of generality that c_i is $\exists x\ x.p_m = u \wedge c_1$. Let $c_j^p, j = 1, \ldots, n$ be the positive literals of c_1 and $p_m(u) = t_x$. Then,

$$\begin{array}{rcl} \exists x\, x.p_m = u \wedge c_1 & \longrightarrow & \exists z_1 \ldots \exists z_m\, [t_x/x]c_1 \\ & \longrightarrow & \exists z_1 \ldots \exists z_m c_2 \end{array}$$

where c_2 is the resulting reduced formula.

Each $c_j^p, j = 1, \ldots, n$ was rewritten in one of the ways shown in figure 3.1. Then we can ensure that:

$$\sum_{i=1}^{m} \|c_2 : z_i\| < \|c_1 : x\| + (m+1) = \|c_i : x\| = k$$

So, every z_i has a norm that is less than k and by inductive hypothesis every each z_i is eliminated and as no more positive literals are added the process is finite and the algorithm halts.

If the first occurrence of x in positive literals of c_i, was in $x.p = y.q$ or $x.pf^n$ the proof was similar.

•

$$
\begin{array}{lllll}
[t_x/x]x.p = a & \rightarrow & t_x.p = a & & \\
& \rightarrow & z_i.q_i = a & \|p\| > \|q_i\| & \\
& \rightarrow & z_i = a & & \\
& \rightarrow & b = a & \rightarrow & false \\
& & & \rightarrow & \neg false \\
& \rightarrow & \perp = a & \rightarrow & false
\end{array}
$$

$$
\begin{array}{lllll}
[t_x/x]x.p = y.q & \rightarrow & t_x.p = y.q & & \\
& \rightarrow & z_i.q_i = y.q & & \|p\| > \|q_i\| \\
& \rightarrow & z_i = y.q & & \\
& \rightarrow & b = y.q & \rightarrow & y.q = b \\
& \rightarrow & \perp = y.q & \rightarrow & false
\end{array}
$$

$$
\begin{array}{lllll}
[t_x/x]x.pf^n & \rightarrow & t_x.pf^n & & \\
& \rightarrow & z_i.q_i f^n & & \|p\| > \|q_i\| \\
& \rightarrow & z_i.f^n & & \\
& \rightarrow & g^k(z_i, \ldots, z_j).f^n & \rightarrow & false \\
& \rightarrow & b.f^n & \rightarrow & false \\
& \rightarrow & \perp.f^n & \rightarrow & false
\end{array}
$$

Figure 2: Reducing positive literals

Theorem 1 *The elimination algorithm converts any formula c into an equivalent quantifier free formula or halts with failure.*

Proof.

As the number of quantified variables is finite, it suffices to sucessively apply lemma 1 to the innermost quantification. •

4 Related Work

D.Smith in [7] presents an algorithm for reducing sets of universally quantified disequalities to solved form based on an algorithm for existencially quantified equations do to [5] and [4].

References

[1] Damas, Luís, Nelma Moreira and Giovanni B. Varile, 1991. *The Formal and Processing Models of CLG*, in Proceedings of the EACL'91, Berlim.

[2] Dörre, J. and William Rounds, 1990. *On Subsumption and Semiunification in Features Algebras.*

[3] Johnson, Mark, 1988. *Attribute-Value Logic and the Theory of Grammar*, number 16 in CSLI Lecture Notes, Center for the Study of Language and Information, Standford, CA.

[4] Lassez, J-L., M. J. Maher and K.Marriott, 1988. Unification Revisited, in *Fundations of Deductive Databases and Logic Programming.* M. Kaufmann.

[5] Maher, Micheal J., 1988. *Complete axiomatizations of the algebras of finite, rational and infinite trees.* Research Report, IBM, Thomas J. Watson Research Center.

[6] Smolka, G., 1989. *Feature Constraint Logics for Unification Grammars*, LILOG Report 93, IWBS, IBM Deutschland.

[7] Smith, Donald, 1991. Constraint Operations for CLP($\mathcal{FT}$). In *Proceedings of the ICLP91*, MIT Press.