Equality Elimination for the Inverse Method and Extension Procedures

Anatoli Degtyarev*
Computing Science Department
Uppsala University
Box 311, S-751 05 Uppsala,
Sweden

Andrei Voronkov*
Computing Science Department
Uppsala University
Box 311, S-751 05 Uppsala,
Sweden

Abstract

We demonstrate how to handle equality in the inverse method using *equality elimination.* In the equality elimination method, proofs consist of two parts. In the first part we try to solve equations obtaining so called *solution clauses.* Solution clauses are obtained by a very refined strategy — basic superposition with selection function. In the second part, we perform the usual sequent proof search by the inverse method. Our approach is called equality elimination because we eliminate all occurrences of equality in the first part of the proof. Unlike the previous approach proposed by Maslov, our method uses most general substitutions, ordering restrictions and selection functions.

We also note that this technique is directly applicable to *extension procedures,* like the *connection method.* Unlike other approaches, we do not require the use of rigid or mixed *E*-unification.

1 The inverse method

The inverse method of theorem proving in sequent calculi has been proposed by Maslov in the 1960s. The method is based on the bottom-up1 search in sequent calculi. The inverse method is completely local [Maslov and Mints, 1983; Degtyarev and Voronkov, 1994b] and can be efficiently implemented both for classical and non-classical logics [Voronkov, 1992]. In terms of efficiency, it is competitive with resolution2. The inverse method requires no normal forms which can be an advantage for interactive provers.

The introduction of equality in the inverse method has not yet received proper attention. In this paper we consider the inverse method with equality. Our approach is based on equality elimination.

The structure of this paper is the following. In this section we briefly discuss introduction of equality in the inverse method in general. In Section 2 we introduce main definitions and notation. In Section 3 we define the equality elimination method and give several examples. In Section 4 we show how one can generalize our technique to the connection method.

The first natural generalization of the inverse method to include equality has been made by Maslov [Maslov, 1971]. It was based on theorems proven in [Kanger, 1983; Lifschitz, 1968] about the specialization of proofs in the sequent calculus with equality (but without free variables!). According to these theorems, all equality reasoning steps can be moved on top of the proof so that they precede all other steps. This gives the characterization of the inverse method with equality which in fact coincides with hyperresolution (on classical logic), but with a different initial set of clauses (closed collections in Maslov's terminology). While in the inverse method without equality the initial sequents correspond to axioms of the sequent calculus, in the equality case initial sequents are those derivable from axioms exclusively by equality rules. [Kanger, 1983] called such sequents "directly demonstratable". In later papers the role of these sequents play (instantiated) equational mated sets [Gallier, 1992] or E-complementary eq-connections [Bibel, 1987]. We call their analog *solution clauses.*

Let us come back to results on the specialization of sequent proofs. Is it possible to generalize these results on proofs with free variables (metavariables in Maslov's terminology)? In this case equality rules become paramodulation rules. Direct lifting from the ground case is now impossible without functional reflexivity axioms. As we shall show, the use of free variables is possible both for the traditional formulation of the inverse method and for the formulation proposed in our paper.

In the original presentation of the inverse method by Maslov, unification has been made by using arbitrary (not only most general) substitutions which made lifting from the ground case unnecessary3. Even in the paper

* Supported by a grant from the Swedish Institute.
*On leave from Kiev University.
t Supported by a TFR grant.

1 Bottom-up search means the search from axioms to the goal.

2In fact, the inverse method can be simulated by resolution using structure-preserving clause-form translation [Maslov, 1983; Boy de la Tour, 1990].

3In terms of resolution, such a treatment of unification gives *unrestricted resolution* [Lloyd, 1987] — a calculus with too high non-determinism.

[Maslov and Mints, 1983] evaluated in [Lifschitz, 1989] as a "very clean explanation of the inverse method" it was only noted that "likewise the definition of a most general unifier, at each step one could select minimal substitutions". However, the proof of completeness in the [Maslov and Mints, 1983] reformulation of the inverse method is made using unrestricted substitutions given by an arbitrary saturation of the Herbrand universe. Arbitrary substitutions are also used in [Lifschitz, 1989] where the most general unifiers are only used for factoring.

An alternative approach would be to take into account the results of [Kanger, 1983; Lifschitz, 1968] on the specialization of sequent proofs. These results allow one to restrict oneself to simultaneous paramodulation only. "Simultaneous" means that in the application of the equality rules from the conclusion to the premise *all* occurrences of the "into-term" are replaced by the "from-term". As it has been shown in [Benanav, 1990], for simultaneous paramodulation lifting is possible without the functional reflexivity axioms.

We do not, however, use simultaneous paramodulation here. Instead, we use very refined strategies of dealing with equality — orderings, basic restriction and selection function. It happens that these severe restrictions on the equality part of the proof is enough to preserve completeness using most general unifiers.

Our method is also directly applicable to so called "extension procedures" [Prawitz, 1983], including tableaux methods [Fitting, 1990; Degtyarev and Voronkov, 1994b], mating or connection methods [Andrews, 1986; Bibel, 1987; Degtyarev and Voronkov, 1995] and their generalization — consolution [Eder, 1991]. We shall illustrate such an application of our method in Section 4.

The major difference of our treatment of equality reasoning for the connection method is that instead of checking paths through an extension of the input formula for E-complementarity (i.e. for the existence of a rigid E-unifier) we first generate a set of E-complementary paths using ordinary unification, and then look for an appropriate extension. Thus, we split extension procedures into two separate processes: equality elimination and search for an extension. The two processes are connected by solution clauses generated in the equality elimination part. This allows us to use strong sides of both equality reasoning methods known so far and extension procedures.

2 Preliminaries

We present here a brief overview of notions and preliminary definitions necessary for the paper. We assume the basic knowledge of substitutions and unification.

Let Σ be a signature and X a set of variables. $T(\Sigma, X)$ will denote the set of all terms in the signature Σ with variables from X.

A *literal* is either an atomic formula or a negation of an atomic formula. We shall always write $s \neq t$ for the literal $\neg(s = t)$.

A *clause* is a finite set $\{L_1, \ldots, L_n\}$ of literals, denoted $L_1, \ldots, L_n$. If L is a literal, C a clause, L, C will denote the clause $\{L\} \bigcup C$. An *equation* is an expression $s = t$, where $s, t \in T(\Sigma, X)$. By a *ground expression* (i.e. term, equation, clause etc.) we mean an expression containing no variables. We write $A[s]$ to indicate that an expression A contains s as a subexpression and denote by $A[t]$ the result of replacing particular occurrences of s in A by t. By $A\sigma$ we denote the result of applying the substitution σ to A.

We shall sometimes denote tuples of equations $s_1 = t_1, \ldots, s_n = t_n$ by $\bar{s} = \bar{t}$. The overbar notation $\bar{t}$ can also be used to denote sequences of terms. Substitutions θ with the domain $x_1, \ldots, x_n$ will be denoted by $[x_1\theta/x_1, \ldots, x_n\theta/x_n]$.

Let $\succ$ be a partial ordering on $T(\Sigma, X)$. It is called a *reduction ordering* iff

1. $\succ$ is well-founded;
2. if $s \succ t$ then $u[s\sigma] \succ u[t\sigma]$, for all terms s, t, u and substitutions σ.

We assume that reduction orderings are total on ground terms of $T(\Sigma, X)$.

Following [Bachmair *et al.*, 1992] we distinguish terms occurring in the original formula from terms introduced by substitution by using *closures*, i.e. pairs $C \cdot \sigma$, where C is a clause, σ a substitution. The clause C will often be identified with the closure $C \cdot \varepsilon$, where ε is the empty substitution. For any expression E, the set $Var(E)$ is defined as the set of all variables occurring in E. Two closures $C_1 \cdot \sigma_1$ and $C_2 \cdot \sigma_2$ are *variants* iff C_1 is a variant of C_2 and $C_1\sigma_1$ is a variant of $C_2\sigma_2$.

When we use notation $\models$ or $\vdash$, they state for truth and provability *in classical first order logic with equality*, respectively.

A formula is in *skolemized negation normal form* iff it is constructed from literals using connectives $\wedge, \vee$ and the quantifier $\exists$. There is a structure-preserving translation of formulas without equivalences into formulas in skolemized negation normal form with the same number of occurrences of atoms.

3 The equality elimination method

For the rest of this section we assume that γ is a closed formula in skolemized negation normal form to be proven (the "goal"). We assume that all different occurrences of quantifiers in γ bind different variables. For example, γ cannot have the form $\exists x A \vee \exists x B$. *All formulas in this section are assumed to be subformulas of* γ. We shall identify subformulas and superformulas with their *occurrences* in γ. For example, in the formula γ of the form $A \wedge (A \vee B)$ the second occurrence of A is considered a subformula of $(A \vee B)$, but the first occurrence of A is not. The occurrence of a subformula φ of γ is called *conjunctive* iff it is the occurrence in a subformula $\varphi \wedge \psi$ or in $\psi \wedge \varphi$. A *conjunctive superformula* of φ is a superformula[4] ψ of φ that is conjunctive. The *least conjunctive superformula* of φ is the conjunctive superformula ψ of φ such that any other conjunctive superformula of φ is a superformula of ψ. Let us note that conjunctive superformulas do not necessarily exist. For

[4] φ is a superformula of ψ iff ψ is a subformula of φ (not necessarily proper).

example, γ has no conjunctive superformula. Any formula having a conjunctive superformula has the unique least conjunctive superformula.

We can enumerate all conjunctive subformulas $\gamma_1, \dots, \gamma_n$ of γ, for example in the order of their occurrences in γ. Thus we can unambiguously use "the kth conjunctive (sub)formula" γ_k of γ.

Let $A_1, \dots, A_n$ be a set of predicate symbols not occurring in γ. We say that $A_k(x_1, \dots, x_m)$ is *the γ-name for a subformula* φ of γ iff

1. The least conjunctive superformula of φ is γ_k.
2. $x_1, \dots, x_m$ are all free variables of γ_k in the order of their occurrences in γ_k.

If a γ-name of a formula φ exists, then it is unique. Note that different formulas may have the same γ-names. Also note that some subformulas of γ do not have γ-names. We can use *the set of γ-names* of a subformula. The set of γ-names of a formula φ is either $\emptyset$ or a singleton $\{A_k(x_1, \dots, x_m)\}$.

The following example illustrates least conjunctive superformulas and γ-names:

Example 3.1 *Let γ be the formula*

$$(\exists x(A(x) \wedge (B(x) \vee \exists y C(x,y))) \wedge \exists z D(z)) \vee E$$

All subformulas of γ and their γ-names are shown in Figure 1 at the end of this paper.

Lemma 3.1 *Let φ be a subformula of γ. Then*

1. *If ψ is the least conjunctive superformula of φ then $\models \forall(\varphi \supset \psi)$.*
2. *If there is no conjunctive superformula of φ then $\models \forall(\varphi \supset \gamma)$.*

This lemma partially explains the need for introducing least conjunctive superformulas. There are deterministic chains of inferences in sequent systems, where formulas with $\vee$ or $\exists$ occur. For example,

$$\cfrac{\cfrac{\Gamma \to \Delta, \varphi}{\Gamma \to \Delta, \varphi \vee \psi}}{\Gamma \to \Delta, \exists x(\varphi \vee \psi)}$$

By restricting ourselves to conjunctive superformulas only, we eliminate these deterministic chains, making them in one step.[5]

Now we can formulate our deductive system for the inverse method. The proof-search consists of two parts: the equality solution part and the sequent proof part. We encode formulas by their γ-names. Then sequents (i.e. sets of formulas) become clauses (i.e. sets of literals, obtained as the union of corresponding sets of γ-names). In the first part of the proof, we try to solve equalities in the formula, generating a set of closures not containing equality. A closure not containing equality will be called a *solution clause*. Solution clauses $C \cdot \sigma$ will always be written in the ordinary clause notation, i.e. as $C\sigma$. This set of solution clauses is obtained by basic superposition from *initial closures* defined below. In the second part, the usual sequent style proof search by the inverse method is performed. The solution clauses generated in the first part are used as the initial clauses in the sequent part. Hence the name "equality elimination" for our method: we eliminate equality in the first part of the proof.

Initial closures are generated according to one of the three rules:

1. Whenever a literal $s \neq t$ occurs in γ and C is the set of γ-names for this occurrence of $s \neq t$, the closure $s = t, C \cdot \varepsilon$ is an initial closure.
2. Whenever $s = t$ occurs in γ and C is the set of γ-names for this occurrence of $s = t$, the closure $s \neq t, C \cdot \varepsilon$ is an initial closure.
3. Let literals $P(s_1, \dots, s_n)$ and $\neg P(t_1, \dots, t_n)$, where P is different from $=$, occur in γ and where C_1, C_2 are their sets of γ-names. Let the substitution σ rename variables such that variables of $P(s_1, \dots, s_n)\sigma, C_1\sigma$ and $P(t_1, \dots, t_n), C_2$ are disjoint. Then the closure $s_1\sigma \neq t_1, \dots, s_n\sigma \neq t_n, C_1\sigma, C_2 \cdot \varepsilon$ is an initial closure.

The equality elimination method consists of five inference rules. The *basic superposition* rules are defined as in [Bachmair *et al.*, 1992]. We always assume that premises of rules have disjoint variables (this can be achieved by using variants). The first three rules derive closures from the initial closures defined above. Applications of the first three rules form the *equational solution part of the proof.* The aim of the equational solution part of the proof is to generate *solution clauses* that are used as axioms of the sequent part of the proof. The sequent part of the proof uses the last two rules. The aim of the sequent part of the proof is to derive the empty clause.

Basic right superposition

$$\frac{(s = t, C) \cdot \sigma_1 \qquad (u[s'] = v, D) \cdot \sigma_2}{(u[t] = v, C, D) \cdot \sigma_1\sigma_2\rho}$$

where

1. ρ is a most general unifier of $s\sigma_1$ and $s'\sigma_2$;
2. $t\sigma_1\rho \not\succeq s\sigma_1\rho$;
3. $v\sigma_2\rho \not\succeq u[s']\sigma_2\rho$;
4. s' is not a variable.

(We call attention to the fact that variables of the premises must be disjoint.)

Basic left superposition

[5]This simple but powerful idea of restricting to conjunctive superformulas (see [Voronkov, 1992]) has been successfully implemented in the theorem prover described in [Voronkov, 1990] and in a theorem prover for intuitionistic logic implemented by T. Tammet (private communications). In the framework of tableau theorem proving there were many papers using the idea of permutability of inference rules in sequent calculi (see e.g. [Shankar, 1992]). In fact, the least conjunctive superformula of φ is a superformula ψ of φ provable from φ and such that all inference rules applied in the proof of ψ from φ are permutable with all other rules. The use of conjunctive superformulas allows one to get rid of non-conjunctive subformulas before the proof-search, unlike the dynamic use of permutabilities as in [Shankar, 1992].

$$\frac{(s = t, C) \cdot \sigma_1 \qquad (u[s'] \neq v, D) \cdot \sigma_2}{(u[t] \neq v, C, D) \cdot \sigma_1\sigma_2\rho}$$

with the same conditions as for basic right superposition and one additional restriction: $u[s'] \neq v$ must be the leftmost disequation in the second premise.[6]

Equality solution

$$\frac{(s \neq t, C) \cdot \sigma}{C \cdot \sigma\rho}$$

where ρ is a most general unifier of $s\sigma, t\sigma$ and $s \neq t$ is the leftmost disequation in the premise.

Conjunction rule This rule is used in the sequent part of the proof, thus it is only applied to *clauses*, containing no equality. Let $\varphi \wedge \psi$ be a subformula of γ, and N_φ, N_ψ and $N_{\varphi\wedge\psi}$ be sets of γ-names of φ, ψ and $\varphi \wedge \psi$, respectively. Let $\bar{x}$ be all variables of $\varphi \wedge \psi$. Let clauses Γ, Δ have no occurrences of equality. Then the following is a conjunction rule:

$$\frac{\Gamma, N_\varphi[\bar{s}/\bar{x}] \qquad \Delta, N_\psi[\bar{t}/\bar{x}]}{(\Gamma, \Delta, N_{\varphi\wedge\psi}[\bar{s}/\bar{x}])\theta}$$

where θ is a most general unifier of $\bar{s}$ and $\bar{t}$.

Factoring rule This rule is also used in the sequent part of the proof, thus it is only applied to *clauses*, containing no equality. Let the clause Γ, A, B have no occurrences of equality. Then the following is a factoring rule

$$\frac{\Gamma, A, B}{(\Gamma, A)\theta}$$

where θ is a most general unifier of A, B.

The following theorems are true about the method:

Theorem 3.1 (Soundness) *If there is a derivation of the empty clause from the initial closures then γ is provable.*

Theorem 3.2 (Completeness) *If γ is provable then there is a derivation of the empty clause from the initial closures.*

Proofs of these two theorems may be found in [Degtyarev and Voronkov, 1994c].

Consider an example (the formula is taken from [Maslov, 1971]):

Example 3.2 *The formula to be proven is*

$$\exists x((a \neq x \vee \neg G(b) \vee G(f(f(x)))) \wedge (\neg G(f(x)) \vee x = b))$$

There are two conjunctive subformulas with the following names:

$$A_1(x) \equiv a \neq x \vee \neg G(b) \vee G(f(f(x)))$$
$$A_2(x) \equiv \neg G(f(x)) \vee x = b$$

The corresponding conjunction rule is

$$\frac{\Gamma, A_1(s) \qquad \Delta, A_2(t)}{(\Gamma, \Delta)\theta}$$

where θ is a most general unifier of s, t.

The initial closures are

1. $a = x, A_1(x) \cdot \varepsilon$
2. $x \neq b, A_2(x) \cdot \varepsilon$
3. $b \neq f(f(x)), A_1(x), A_1(y) \cdot \varepsilon$
4. $f(x) \neq f(f(y)), A_2(x), A_1(y) \cdot \varepsilon$

The first two closures are generated by equations in the goal, the last two by pairs $G, \neg G$.

As the order $\succ$ we consider the recursive path ordering with $f > b > a$. The equality elimination part of the proof consists of applications of basic superposition and equality solution:

5. $A_2(x) \cdot [b/x]$ (equality solution from 2)
6. $b \neq a, A_1(x), A_1(y), A_1(z) \cdot [f(f(y))/x]$
   (left superposition from 1,3)
7. $a \neq a, A_1(x), A_1(y), A_1(z), A_1(v) \cdot [f(f(y))/x, b/v]$
   (left superposition from 1,6)
8. $A_1(x), A_1(y), A_1(z), A_1(v) \cdot [f(f(y))/x, b/v]$
   (equality solution from 7)
9. $A_2(x), A_1(y) \cdot [f(y)/x]$
   (equality solution from 4)

The clauses which can be used in the sequent part of the proof are the following:

5'. $A_2(b)$
8'. $A_1(f(f(y))), A_1(y), A_1(z), A_1(b)$
9'. $A_2(f(y)), A_1(y)$

The sequent part of the proof only uses the conjunction rule and the factoring rule:

10. $A_1(f(f(y))), A_1(y), A_1(b)$
    (factoring from 8')
11. $A_1(f(f(b))), A_1(b)$
    (factoring from 10)
12. $A_1(f(f(b)))$ (conjunction rule from 5',11)
13. $A_1(f(b))$ (conjunction rule from 9',12)
14. $A_1(b)$ (conjunction rule from 9',13)
15. $\square$ (conjunction rule from 5',14)

There are many optimizations not considered in this paper. The most powerful one is *subsumption* that can be applied in both parts of the proof in order to prevent generation of unnecessary clauses. Subsumption for the basic strategy has been explained in [Degtyarev, 1982; Bachmair *et al.*, 1992]. In the sequent part of the proof we can use ordinary subsumption, or even a stronger subsumption for sequent proofs proposed in [Voronkov, 1992].

6 According to our definitions, a clause is a set of literals, so the use of the leftmost disequation is not quite correct, but this restriction can easily be formalized using the selection mechanism [Bachmair and Ganzinger, 1994]. The proof-theoretic justification of this possibility is given in [Degtyarev *et al.*, 1995].

4 The connection method

The equality elimination method is very general and can be applied to other automated reasoning methods, for example the connection method. In order to illustrate it, we consider Example 3.2 in the connection method. We shall refer to definitions and statements from [Gallier, 1992]. The connection method is considered in more detail in [Degtyarev and Voronkov, 1995].

As in [Gallier, 1992], we try to establish unsatisfiability of the negation of the goal formula

$$\forall x((a = x \land G(b) \land \neg G(f(f(x)))) \lor (G(f(x)) \land x \neq b))$$

Denote the matrix of this formula via $M(x)$. According to Theorems 4.3 and 7.7 of [Gallier, 1992] this formula is inconsistent iff there is a path-acceptable equational mating for some amplification $M(x_1) \land \ldots \land M(x_n)$. Consider the set of vertical paths through this amplification. According to Definition 4.4 of [Gallier, 1992], the set of vertical paths $vp(A)$ through amplification A is the set of sets of literals defined inductively as follows:

$vp(A) = \{\{A\}\}$, if A is a literal;
$vp(B \land C) = \{\pi_1 \bigcup \pi_2 \mid \pi_1 \in vp(B), \pi_2 \in vp(C)\}$;
$vp(B \lor C) = vp(B) \bigcup vp(C)$

Denote

$$A_1(x) = \{a = x, G(b), \neg G(f(f(x)))\}$$
$$A_2(x) = \{G(f(x)), x \neq b\}$$

From the above definition of vp it follows that the set $vp(M(x_1) \land \ldots \land M(x_n))$ has the form

$$\{A_{i_1}(x_1) \bigcup \ldots \bigcup A_{i_n}(x_n) \mid 1 \leq j \leq n \text{ and } i_j \in \{1,2\}\}$$

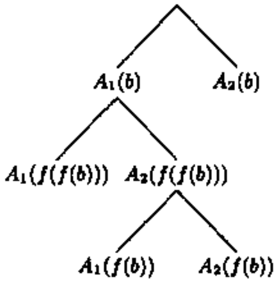
Note now that the solution clauses used in Example 3.2 define the following set of inconsistent instantiated partial paths:

$$A_2(b)$$
$$A_1(b) \bigcup A_1(y) \bigcup A_1(f(f(y))) \bigcup A_1(z)$$
$$A_1(x) \bigcup A_2(f(x))$$

Indeed, any of these partial paths contains an inconsistent instantiated mated set (see [Gallier, 1992]) of $M(x_1) \land \ldots \land M(x_n)$. For example, for the second of these partial paths it will be the instantiated mated set

$$\{a = b, a = f(f(y)), G(b), \neg G(f(f(y)))\}$$

The composition of these instantiated mated sets forms an equational mating. It remains to find an instantiated amplification for which this mating is path acceptable. Such amplification can be extracted from the sequent part of the proof. The instantiated amplification for our example is shown in the following picture:



Thus, in this amplification $n = 3$, $x_1 = b$, $x_2 = f(f(b))$ and $x_3 = f(b)$. Note that this tree can be easily changed into a tableau proof of γ.

5 Conclusion

The equality elimination method is a general method of reasoning in first-order logic with equality. It is based on the following general version of Herbrand theorem [Degtyarev and Voronkov, 1995]: a formula γ is provable iff there is a matrix M of γ and a substitution σ such that every path through $M\sigma$ contains an instance of a solution clause. Equality elimination was originally introduced in [Degtyarev and Voronkov, 1994a] as a method of handling equality in logic programs. Later, it has been applied to the tableau method [Degtyarev and Voronkov, 1994b] and the connection method [Degtyanev and Voronkov, 1995]. In [Degtyarev *et al,* 1995] a combination of equality elimination and *basic folding* has been introduced which allows to transform equational logic programs into recursive logic programs without equality.

References

[Andrews, 1986] P.B. Andrews. *An introduction to type theory: to truth through proof.* Academic Press, 1986.

[Bachmair and Ganzinger, 1994] L. Bachmair and H. Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation,* 4(3):217-247, 1994.

[Bachmair *et al,* 1992] L. Bachmair, H. Ganzinger, C. Lynch, and W. Snyder. Basic paramodulation and superposition. In D. Kapur, editor, *11th International Conference on Automated Deduction,* volume 607 of *Lecture Notes in Artificial Intelligence,* pages 462-476, Saratoga Springs, NY, USA, June 1992. Springer Verlag.

[Benanav, 1990] D. Benanav. Simultaneous paramodulation. In M.E. Stickel, editor, *Proc. 10th Int. Conf. on Automated Deduction,* volume 449 of *Lecture Notes in Artificial Intelligence,* pages 442-455, 1990.

[Bibel, 1987] W. Bibel. *Automated theorem proving.* Vieweg Verlag, 1987. 2nd edition.

[Boy de la Tour, 1990] T. Boy de la Tour. Minimizing the number of clauses by renaming. In M.E. Stickel, editor, *Proc. 10th CADE,* volume 449 of *Lecture Notes in Artificial Intelligence,* pages 558-572, 1990.

[Degtyarev and Voronkov, 1994a] A. Degtyarev and A. Voronkov. A new procedural interpretation of Horn clauses with equality. UPMAIL Technical Report 89, Uppsala University, Computing Science Department, November 1994. Also to be published in Proc. ICLP'95.

[Degtyarev and Voronkov, 1994b] A. Degtyarev and A. Voronkov. Equality elimination for semantic tableaux. UPMAIL Technical Report 90, Uppsala University, Computing Science Department, December 1994.

[Degtyarev and Voronkov, 1994c] A. Degtyarev and A. Voronkov. Equality elimination for the inverse method and extension procedures. UPMAIL Technical Report 92, Uppsala University, Computing Science Department, December 1994.

[Degtyarev and Voronkov, 1995] A. Degtyarev and A. Voronkov. General connections via equality elimination. UPMAIL Technical Report 93, Uppsala University, Computing Science Department, January 1995. Also to be published in Proc. WOCFAF95.

[Degtyarev et a/., 1995] A. Degtyarev, Yu. Koval, and A. Voronkov. Handling equality in logic programming via basic folding. UPMAIL Technical Report 101, Uppsala University, Computing Science Department, March 1995.

[Degtyarev, 1982] A. Degtyarev. On the forms of inference in calculi with equality and paramodulation. In Yu.V. Kapitonova, editor, Automation of Research in Mathematics, pages 14-26. Institute of Cybernetics, Kiev, Kiev, 1982.

[Eder, 1991] E. Eder. Consolution and its relation with resolution. In Proc. IJCAI'91, pages 132-136, 1991.

[Fitting, 1990] M. Fitting. First Order Logic and Automated Theorem Proving. Springer Verlag, New York, 1990.

[Gallier, 1992] J. Gallier. Unification procedures in automated deduction methods based on matings: a survey. In M. Nivat and A. Podelski, editors, Tree Automata and Languages, pages 439-485. Elsevier Science, 1992.

[Kanger, 1983] S. Kanger. A simplified proof method for elementary logic. In J. Siekmann and G. Wrightson, editors, Automation of Reasoning. Classical Papers on Computational Logic, volume 1, pages 364-371. Springer Verlag, 1983. Originally appeared in 1963.

[Lifschitz, 1968] V. Lifschitz. Specialized forms of derivation in predicate calculus with equality and functional symbols (in Russian). In Trudy MIAN, volume 98, pages 5-25. 1968. English translation in: Proc. Steklov Institute of Math., AMS, Providence, RI, 1971.

[Lifschitz, 1989] V. Lifschitz. What is the inverse method? Journal of Automated Reasoning, 5(1):1-23, 1989.

[Lloyd, 1987] J.W. Lloyd. Foundations of Logic Programming (2nd edition). Springer Verlag, 1987.

[Maslov and Mints, 1983] S.Yu. Maslov and G. Mints. The proof-search theory and the inverse method (in Russian). In Mints G., editor, Mathematical Logic and Automatic Theorem Proving, pages 291-314. Nauka, Moscow, 1983.

[Maslov, 1971] S.Yu. Maslov. The generalization of the inverse method to predicate calculus with equality (in Russian). Zapiski Nauchnyh Seminarov LOMI, 20:80-96, 1971. English translation in: Journal of Soviet Mathematics 1, no. 1.

[Maslov, 1983] S.Yu. Maslov. Relationship between tactics of the inverse method and the resolution method. In J.Siekmann and G.Wrightson, editors, Automation of Reasoning (Classical papers on Computational Logic), volume 2, pages 264-272. Springer Verlag, 1983. Originally appeared in 1963.

[Prawitz, 1983] D. Prawitz. An improved proof procedure. In J. Siekmann and G. Wrightson, editors, Automation of Reasoning. Classical Papers on Computational Logic, volume 1, pages 162-201. Springer Verlag, 1983. Originally appeared in 1960.

[Shankar, 1992] N. Shankar. Proof search in the intuitionistic sequent calculus. In 12th International Conference on Automated Deduction, Lecture Notes in Computer Science, 1992.

[Voronkov, 1990] A. Voronkov. LISS - the logic inference search system. In Mark Stickel, editor, Proc. 10th Int. Conf. on Automated Deduction, volume 449 of Lecture Notes in Computer Science, pages 677-678, Kaiserslautern, Germany, 1990. Springer Verlag.

[Voronkov, 1992] A. Voronkov. Theorem proving in nonstandard logics based on the inverse method. In D. Kapur, editor, 11th International Conference on Automated Deduction, volume 607 of Lecture Notes in Artificial Intelligence, pages 648-662, Saratoga Springs, NY, USA, June 1992. Springer Verlag.

Subformula	least conjunctive superformula	set of γ-names
$(\exists x(A(x) \wedge (B(x) \vee \exists y C(x,y))) \wedge \exists z D(z)) \vee E$		$\emptyset$
$\exists x(A(x) \wedge (B(x) \vee \exists y C(x,y))) \wedge \exists z D(z)$		$\emptyset$
$\exists x(A(x) \wedge (B(x) \vee \exists y C(x,y)))$	$\exists x(A(x) \wedge (B(x) \vee \exists y C(x,y)))$	$\{A_1\}$
$A(x) \wedge (B(x) \vee \exists y C(x,y))$	$\exists x(A(x) \wedge (B(x) \vee \exists y C(x,y)))$	$\{A_1\}$
$A(x)$	$A(x)$	$\{A_2(x)\}$
$B(x) \vee \exists y C(x,y)$	$B(x) \vee \exists y C(x,y)$	$\{A_3(x)\}$
$B(x)$	$B(x) \vee \exists y C(x,y)$	$\{A_3(x)\}$
$\exists y C(x,y)$	$B(x) \vee \exists y C(x,y)$	$\{A_3(x)\}$
$C(x,y)$	$B(x) \vee \exists y C(x,y)$	$\{A_3(x)\}$
$\exists z D(z)$	$\exists z D(z)$	$\{A_4\}$
$D(z)$	$\exists z D(z)$	$\{A_4\}$
E		$\emptyset$

Figure 1: Least conjunctive superformulas and γ-names.