# Proof-Search in Intuitionistic Logic with Equality, or Back to Simultaneous Rigid $E$-Unification

Andrei Voronkov*

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

**Abstract.** We characterize provability in intuitionistic logic with equality in terms of a constraint calculus. This characterization uncovers close connections between provability in intuitionistic logic with equality and solutions to simultaneous rigid $E$-unification. We show that the problem of existence of a sequent proof with a given skeleton is polynomial-time equivalent to simultaneous rigid $E$-unifiability. This gives us a proof procedure for intuitionistic logic with equality modulo simultaneous rigid $E$-unification. We also show that simultaneous rigid $E$-unifiability is polynomial-time reducible to intuitionistic logic with equality. Thus, any proof procedure for intuitionistic logic with equality can be considered as a procedure for simultaneous rigid $E$-unifiability. In turn, any procedure for simultaneous rigid $E$-unifiability gives a procedure for establishing provability in intuitionistic logic with equality.

## 1 Introduction

Any automated theorem proving method for predicate logic provides a way of instantiating variables in proofs. In modern theorem proving methods, the variable instantiation is usually controlled by a suitable unification algorithm. In this paper we analyze the problem of variable instantiation in intuitionistic logic with equality. Our analysis shows the close connections between the variable instantiation problem and simultaneous rigid $E$-unification introduced by Gallier, Raatz and Snyder [11]. Furthermore, our results imply that handling simultaneous rigid $E$-unification is unavoidable in automatic proof procedures for intuitionistic logic with equality.

For classical logic, there are many methods of variable instantiation depending on the proof procedure. The main methods (in the chronological order) are the following.

1. The early methods used saturation algorithms: substitutions of all possible (ground) terms for variables. The same effect is achieved by the use of the *functional reflexivity axiom* $t \approx t$ (Robinson and Wos [23], also Loveland [15] and Fitting [10] in the context of model elimination and the tableau

* Supported by a TFR grant

method) or by the *functional substitutivity axioms:* both axioms allow us to replace variables by arbitrary terms. This way of instantiation is applicable to arbitrary theorem proving procedures, for example resolution-based or tableau-based procedures.
2. Kanger [12] proposed to instantiate variables by subterms of terms already occurring in the (partial) derivation. This method was characterized by Matulis [18] and Maslov [17] as *minus-normalization.* The applicability of minus-normalization has been demonstrated for proof procedures based on sequent calculi, but it can as well be applied to resolution- and paramodulation-based proof procedures.
3. Resolution and *paramodulation* originally introduced by Robinson and Wos [23], and its version *superposition* use most-general unifiers to instantiate variables. However, most general unifiers have been shown useful only in indirect proof procedures: resolution and the inverse method (see Voronkov [29], Degtyarev and Voronkov [8])[2]).
4. Plotkin [22] proposed to use *E-unification.* However, $E$-unification is applicable only in very restricted cases.
5. Gallier, Raatz and Snyder [11] proposed to use *simultaneous rigid E-unification* to instantiate variables in matrix-based methods, including the connection (or the mating) method and semantic tableaux. However, our recent results on the undecidability of simultaneous rigid $E$-unification [4, 5] show that this approach can hardly be an advantage over methods using most general unifiers.
6. Degtyarev and Voronkov [3, 6, 7] proposed a combination of two kinds of unification for matrix methods based on so-called *equality elimination.* This approach consists of using most general unifiers in the superposition rules and *subset unification* to close tableau branches (subset unifiability is NP-complete).

For intuitionistic logic with equality, no machine-oriented characterization of provability is known. Of course, we can characterize provability in terms of arbitrary substitutions or the functional reflexivity axiom but this is hardly efficient. There are no analogues of resolution for intuitionistic logic. The inverse method augmented with paramodulation is incomplete, as implied by Example 1 below. This example also demonstrates non-applicability of equality elimination to intuitionistic logic.

The aim of this paper is to give a machine-oriented characterization of provability for intuitionistic logic with equality. Our characterization is based on simultaneous rigid $E$-unification. Using a description of provability in terms of derivation skeletons and constraints, we show that the existence of a derivation with a given skeleton is polynomial-time equivalent to simultaneous rigid $E$-unifiability. We also show that simultaneous rigid $E$-unifiability is polynomial time reducible to intuitionistic logic with equality. These two observations

[2] The inverse method was originally described by Maslov [16], but Maslov has never explicitly considered how to instantiate variables.

show that a (semi-decision) algorithm for simultaneous rigid $E$-unifiability is "necessary and sufficient" for designing automated reasoning procedures for intuitionistic logic with equality.

This paper is structured as follows. Section 2 introduces main notions and defines simultaneous rigid $E$-unification. Section 3 defines a sequent calculus $\mathbf{LJ}^=$ for intuitionistic logic with equality. We introduce derivation skeletons and the skeleton instantiation problem: the problem of the existence of a derivation with a given skeleton. We show that simultaneous rigid $E$-unifiability is polynomial time reducible to intuitionistic logic with equality and to the skeleton instantiation problem. In Section 4 we introduce the constraint calculus $\mathbf{LJ}^=_c$ which is a constraint-based modification of $\mathbf{LJ}^=$. We also prove a general theorem about the connection between derivability in $\mathbf{LJ}^=$ and that in $\mathbf{LJ}^=_c$. In Section 5 we prove that the skeleton instantiation problem is polynomial-time equivalent to simultaneous rigid $E$-unifiability. In Section 6 we briefly consider related work. The extended version of this paper containing all proofs is available as a technical report [31] that can be obtained at `http://www.csd.uu.se/~thomasl/reports.html` or by anonymous ftp at `ftp.csd.uu.se` in directory `pub/papers/reports`.

## 2 Preliminaries

*Formulas* are defined as usual, using atomic formulas, the constant $\Lambda$, the connectives $\wedge, \vee, \supset$ and the quantifiers $\forall, \exists$. Negation $\neg\varphi$ can be introduced as $\varphi \supset \Lambda$. A *sequent* is an expression of the form $\Gamma \rightarrow \varphi$ where $\Gamma$ is a finite sequence of formulas and $\varphi$ is a formula. A formula is *closed* iff it has no free variables. A sequent $\Gamma \rightarrow \varphi$ is *closed* iff all formulas in $\Gamma, \varphi$ are closed. We assume that all free variables in formulas are different from all bound variables.

The notation $\rightleftharpoons$ stands for "equal by definition". An *equation* is a formula of the form $s{\approx}t$. An expression is *ground* iff it has no variables.

By $X$ we denote a countable infinite set of variables. The set of all terms in a signature $\Sigma$ with variables in $X$ is denoted by $T_\Sigma(X)$. By $T_\Sigma$ we denote $T_\Sigma(\emptyset)$, i.e. the set of all ground terms in the signature $\Sigma$. *Constants* are regarded as function symbols of arity 0. Variables are denoted by $u, v, w, x, y, z$, maybe with indices. We assume that all formulas use variables in the set $X$.

A *substitution* is any mapping $\theta$ from $X$ to $T_\Sigma(X)$ such that there is only a finite number of variables $x$ such that $\theta(x) \neq x$. The *domain* $dom(\theta)$ of the substitution $\theta$ is the set $\{x \mid x\theta \neq x\}$. The *range* $ran(\theta)$ of the substitution $\theta$ is defined by $ran(\theta) \rightleftharpoons \{x\theta \mid x \in dom(\theta)\}$. Substitutions $\theta$ with the domain $x_1, \ldots, x_n$ are denoted by $\{x_1\theta/x_1, \ldots, x_n\theta/x_n\}$. Let $E$ be a term, formula, sequence of formulas or sequent. For a substitution $\theta = \{t_1/x_1, \ldots, t_n/x_n\}$, $E\theta$ denotes the result of *applying $\theta$ to $E$*, i.e. the replacement of all free occurrences of $x_i$ by $t_i$.

Let $\varphi$ be a formula all whose free variables are $x_1, \ldots, x_n$, in the order of their occurrences in $\varphi$. By $\forall\varphi$ we denote the formula $\forall x_1 \ldots \forall x_n \varphi$, and similar for $\exists\varphi$. We identify formulas equal up to renaming of bound variables.

**Definition 1 (Simultaneous rigid $E$-unification)** A *rigid equation* is an expression $E \vdash_\forall s{\approx}t$, where $E$ is a finite set of equations. A *solution to a rigid* *equation* $\{s_1 \approx t_1, \ldots, s_n \approx t_n\} \vdash_\forall s \approx t$ is any substitution $\sigma$ such that the formula $\forall(s_1\sigma \approx t_1\sigma \wedge \ldots \wedge s_n\sigma \approx t_n\sigma \supset s\sigma \approx t\sigma)$ is true[3]. A *system of rigid equations* is a finite set of rigid equations. A *solution to a system of rigid equations* $S$ is any substitution that is a solution to every rigid equation in $S$. The problem of solvability of rigid equations is known as *rigid* $E$*-unification.* The problem of solvability of systems of rigid equations is known as *simultaneous rigid* $E$*-unification*[4].

We shall sometimes write the left-hand side of a rigid equation as a *sequence* of equations, for example $x \approx a \vdash_\forall g(x) \approx x$ instead of $\{x \approx a\} \vdash_\forall g(x) \approx x$.

Consider two examples

*Example 1* The system of two rigid equations[5]

$$a \approx b \vdash_\forall g(x, u, v) \approx g(y, f(c), f(d))$$
$$c \approx d \vdash_\forall g(u, x, y) \approx g(v, f(a), f(b))$$

has one solution $\{f(a)/x, f(b)/y, f(c)/u, f(d)/v\}$. This system of rigid equations has an interesting property: we cannot find its solution using V-paramodulation of Chang and Lee [13] (later described as rigid paramodulation by Plaisted [21]) and most general solutions to the equations in the right-hand sides of $\vdash_\forall$.

The following result is proven by Degtyarev and Voronkov [4, 5]:

**Theorem 2** *Simultaneous rigid* $E$*-unifiability is undecidable.*

In view of this result, when we mention procedures or algorithms for simultaneous rigid $E$-unification, we mean semi-decision procedures or algorithms.

## 3 The calculus $\mathbf{LJ}^=$

In this section we define the sequent calculus $\mathbf{LJ}^=$ for intuitionistic logic with equality and two notions of derivation skeletons. We also prove some properties of derivations in $\mathbf{LJ}^=$.

The calculus $\mathbf{LJ}^=$ for intuitionistic logic with equality derives sequents. The rules of the calculus are shown in Figure 1. Derivations in $\mathbf{LJ}^=$ are trees defined as usual.

[3] I.e. this formula is provable in first-order (classical or intuitionistic) logic. Equivalently, we can say that $s\sigma$ and $t\sigma$ lie in the same class of the congruence generated by $\{s_1\sigma \approx t_1\sigma, \ldots, s_n\sigma \approx t_n\sigma\}$.

[4] We slightly changed the terminology compared to [11]. Our notion of a rigid equation corresponds to the notion of a rigid $E$-unification problem of [11]. Our notion of a system of rigid equations corresponds to the notion of a simultaneous rigid $E$-unification problem of [11]. We introduced a new terminology to avoid the ambiguous use of the word "problem" that can also be understood as the algorithmic problem of finding a solution to a rigid equation.

[5] This system was proposed by Anatoli Degtyarev to show that paramodulation-based methods are incomplete for simultaneous rigid $E$-unification.

$$\frac{}{\varGamma, A, \varDelta \to A}\ (Ax) \qquad \frac{}{\varGamma, \varLambda, \varDelta \to \varphi}\ (\varLambda) \qquad \frac{}{\varGamma \to t{\approx}t}\ ({\approx})$$

$$\frac{\varGamma\{t/x\}, s{\approx}t, \varDelta\{t/x\} \to \chi\{t/x\}}{\varGamma\{s/x\}, s{\approx}t, \varDelta\{s/x\} \to \chi\{s/x\}}\ ({\approx_1}) \qquad \frac{\varGamma\{s/x\}, s{\approx}t, \varDelta\{s/x\} \to \chi\{s/x\}}{\varGamma\{t/x\}, s{\approx}t, \varDelta\{t/x\} \to \chi\{t/x\}}\ ({\approx_2})$$

$$\frac{\varGamma, \varphi, \psi, \varDelta \to \chi}{\varGamma, \varphi \wedge \psi, \varDelta \to \chi}\ (\wedge \to) \qquad \frac{\varGamma \to \varphi \quad \varGamma \to \psi}{\varGamma \to \varphi \wedge \psi}\ (\to \wedge)$$

$$\frac{\varGamma, \varphi, \varDelta \to \chi \quad \varGamma, \psi, \varDelta \to \chi}{\varGamma, \varphi \vee \psi, \varDelta \to \chi}\ (\vee \to) \qquad \frac{\varGamma \to \varphi}{\varGamma \to \varphi \vee \psi}\ (\to \vee_1) \qquad \frac{\varGamma \to \psi}{\varGamma \to \varphi \vee \psi}\ (\to \vee_2)$$

$$\frac{\varGamma, \psi, \varDelta \to \chi \quad \varGamma, \varphi \supset \psi, \varDelta \to \varphi}{\varGamma, \varphi \supset \psi, \varDelta \to \chi}\ (\supset\to) \qquad \frac{\varphi, \varGamma \to \psi}{\varGamma \to \varphi \supset \psi}\ (\to\supset)$$

$$\frac{\varGamma, \varphi\{t/x\}, \forall x\varphi, \varDelta \to \chi}{\varGamma, \forall x\varphi, \varDelta \to \chi}\ (\forall \to) \qquad \frac{\varGamma \to \varphi\{y/x\}}{\varGamma \to \forall x\varphi}\ (\to \forall)$$

$$\frac{\varGamma, \varphi\{y/x\}, \varDelta \to \chi}{\varGamma, \exists x\varphi, \varDelta \to \chi}\ (\exists \to) \qquad \frac{\varGamma \to \varphi\{t/x\}}{\varGamma \to \exists x\varphi}\ (\to \exists)$$

In this figure, $A$ ranges over atomic formulas, $\varphi, \psi, \chi$ range over arbitrary formulas and $\varGamma, \varDelta$ range over finite sequences of formulas. In the rules $(\to \forall)$ and $(\exists \to)$ the variable $y$ has no free occurrences in the conclusions of the rules. The variable $y$ is called the *eigenvariable* of these rules.

**Fig. 1.** Calculus $\mathbf{LJ}^{=}$

There are two major kinds of automated theorem proving methods based on the proof-search in sequent calculi: the tableau method and the inverse method. The inverse method tries to construct proofs in the direction from axioms to the goal. The tableau method tries to construct proofs in the direction from the goal to axioms. The calculus $\mathbf{LJ}^{=}$ is formulated in a way more suitable for a tableau proof procedure. We can also define a similar calculus for the inverse method, but designing a particular *procedure* is not the aim of this paper.

The notion of a skeleton provides an abstraction of a family of derivations.

**Definition 3 (Skeleton)** A *derivation skeleton* is a tree such that

1. Its nodes are labelled by the names of inference rules in $\mathbf{LJ}^{=}$, except for $({\approx_1})$ and $({\approx_2})$;
2. The number of parents of a node labelled by a name of an inference rule is equal to the number of the premises in such a rule in $\mathbf{LJ}^{=}$;
3. All nodes labelled by antecedent rules or $(Ax)$ are additionally labelled by a natural number.

We display derivation skeletons as derivations with omitted sequents, nodes are denoted by horizontal bars with labels displayed to the right of the bar.

**Definition 4 (Skeleton of a derivation)** The skeleton of a derivation $\varPi$ is obtained from $\varPi$ by removing all sequents and all applications of rules $({\approx_1})$ and

$(\approx_2)$. The natural number in an antecedent rule or in an axiom $(Ax)$ represents the index of the main formula (counting from 0) of this rule in the antecedent of the conclusion of the rule. In other words, this number is the number of formulas in $\Gamma$ in the corresponding rules of $\mathbf{LJ}^=$.

A very formal definition of the skeleton of a derivation may be found in [31]. We always display the number as an index.

*Example 2* The following is the derivation skeleton of the derivation of Figure 2.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\quad}(\approx)}{\quad}(\to\exists)}{\quad}(\exists\to_1)\quad\dfrac{\dfrac{}{\quad}(\approx)}{\quad}(\to\vee_2)}{\quad}(\supset\to_1)}{\quad}(\forall\to_1)}{\quad}(\to\exists)}{\quad}(\exists\to_0)\quad\dfrac{\dfrac{}{\quad}(\approx)}{\quad}(\to\vee_1)}{\quad}(\supset\to_0)}{\quad}(\forall\to_0)}{\quad}(\to\exists)}{\quad}(\to\supset)$$

The following are *not* derivation skeletons:

$$\dfrac{\dfrac{}{\quad}(\approx)}{\quad}(\to\wedge) \qquad \text{and} \qquad \dfrac{\dfrac{}{\quad}(\approx)}{\quad}(\approx_2)$$

In the first figure, the node labeled $(\to\wedge)$ has one parent instead of two. The second figure contains a node labeled $(\approx_2)$ that is not allowed in skeletons.

We introduce an important notion

**Definition 5 (Regular derivation)** A derivation in $\mathbf{LJ}^=$ is called *regular* iff above any application of $(\approx_1)$ or $(\approx_2)$ there can only be applications of the rules $(\approx_1)$, $(\approx_2)$, $(Ax)$, $(\Lambda)$ and $(\approx)$.

An example of a non-regular derivation in $\mathbf{LJ}^=$ is given in Figure 2.

Derivation skeletons are abstractions of derivations that encode information about the structures of derivations. They abstract from the following information:

1. The instantiation of variables in quantifier rules;
2. The order of applications of the equality rules $(\approx_1)$ and $(\approx_2)$.

In particular, skeletons are not able to distinguish regular derivations from non-regular derivations.

The following theorem shows that the order of applications of $(\approx_1)$ and $(\approx_2)$ is, in some sense, unimportant. Similar theorems for various formalizations of intuitionistic logic with equality were stated and/or proven by Orevkov [20] and other authors.

This figure shows a derivation of the formula $\forall x(x{\approx}a \lor x{\approx}b \supset \exists y(f(x,y){\approx}b)) \supset \exists u\exists v\exists w(f(f(u,v),w){\approx}b)$. This derivation is not regular. It has the skeleton of Example 2. We denote by $\varphi$ the formula $\forall x(x{\approx}a \lor x{\approx}b \supset \exists y(f(x,y){\approx}b))$. We also denote by $\cdots$ some irrelevant parts of the sequents.

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{f(a,y){\approx}b, f(b,z){\approx}b, \varphi \to b{\approx}b}\,(\approx)}{f(a,y){\approx}b, f(b,z){\approx}b, \varphi \to f(b,z){\approx}b}\,(\approx_2)}{f(a,y){\approx}b, f(b,z){\approx}b, \varphi \to \exists w(f(b,w){\approx}b)}\,(\to\exists)}{f(a,y){\approx}b, \exists y(f(b,y){\approx}b), \varphi \to \exists w(f(b,w){\approx}b)}\,(\exists\to) \quad \dfrac{\dfrac{}{\cdots \to b{\approx}b}\,(\approx)}{\cdots \to b{\approx}a \lor b{\approx}b}\,(\to\lor_2)}{f(a,y){\approx}b, b{\approx}a \lor b{\approx}b \supset \exists y(f(b,y){\approx}b), \varphi \to \exists w(f(b,w){\approx}b)}\,(\supset\to)}{f(a,y){\approx}b, \varphi \to \exists w(f(b,w){\approx}b)}\,(\forall\to)
$$

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{f(a,y){\approx}b, \varphi \to \exists w(f(b,w){\approx}b)}{f(a,y){\approx}b, \varphi \to \exists w(f(f(a,y),w){\approx}b)}\,(\approx_1)}{f(a,y){\approx}b, \varphi \to \exists v\exists w(f(f(a,v),w){\approx}b)}\,(\to\exists)}{\exists y(f(a,y){\approx}b), \varphi \to \exists v\exists w(f(f(a,v),w){\approx}b)}\,(\exists\to) \quad \dfrac{\dfrac{}{\cdots \to a{\approx}a}\,(\approx)}{\cdots \to a{\approx}a \lor a{\approx}b}\,(\to\lor_1)}{a{\approx}a \lor a{\approx}b \supset \exists y(f(a,y){\approx}b), \varphi \to \exists v\exists w(f(f(a,v),w){\approx}b)}\,(\supset\to)}{\varphi \to \exists v\exists w(f(f(a,v),w){\approx}b)}\,(\forall\to)}{\varphi \to \exists u\exists v\exists w(f(f(u,v),w){\approx}b)}\,(\to\exists)}{\to \varphi \supset \exists u\exists v\exists w(f(f(u,v),w){\approx}b)}\,(\to\supset)
$$

**Fig. 2.** A non-regular derivation in $\mathbf{LJ}^=$

**Theorem 6** *Let $S$ be a sequent and $\mathcal{S}$ be a skeleton. The following conditions are equivalent:*

1. *There is a derivation of $S$ with the skeleton $\mathcal{S}$.*
2. *There is a regular derivation of $S$ with the skeleton $\mathcal{S}$.*

*As a consequence, any derivable sequent has a regular derivation.*

In this paper, we shall consider the problem of instantiation of a skeleton to a derivation:

**Definition 7** The *skeleton instantiation problem* is the following problem. Given a closed sequent $S$ and a skeleton $\mathcal{S}$, whether there is a derivation in $\mathbf{LJ}^=$ of $S$ with the skeleton $\mathcal{S}$.

Note that the condition on $S$ to be closed is inessential since one can always replace free variables of a sequent by fresh constants.

In [30] we proved that, for formulas without equality the skeleton instantiation problem can be solved in polynomial time.

For formulas with equality the skeleton instantiation problem is undecidable. This follows from the following statement:

**Theorem 8** *Simultaneous rigid E-unifiability is polynomial-time reducible to any of the following two problems:*

1. *The problem of derivability in* $\mathbf{LJ}^=$;
2. *The skeleton instantiation problem.*

This statement and Theorem 2 yield

**Theorem 9** *The skeleton instantiation problem is undecidable.*

We note that this undecidability statement is only proved for the case when the signature $\Sigma$ in which formulas are written contains binary function symbols (we come back to this topic at the end of Section 5).

## 4 The Constraint Calculus

In this section we define the constraint calculus $\mathbf{LJ}^=_c$. This calculus can be used for defining sequent style proof procedures for intuitionistic logic with equality. The proof procedures based on $\mathbf{LJ}^=_c$ consist of two parts: the construction of a derivation skeleton and the constraint satisfaction part. The constraint to be satisfied is computed from the skeleton of a derivation.

**Definition 10 (Constraints)** *Constraints* are defined inductively as follows:

1. $\top$ is a constraint;
2. For any terms $t, t_1, \ldots, t_n$, $t \not\preceq \{t_1, \ldots, t_n\}$ is a constraint;
3. Any rigid equation $s_1 \approx t_1, \ldots, s_n \approx t_n \vdash_\forall s \approx t$ is a constraint;
4. If $\mathcal{C}_1, \mathcal{C}_2$ are constraints, then $\mathcal{C}_1 \wedge \mathcal{C}_2$ is a constraint;
5. If $\mathcal{C}$ is a constraint, $x$ is variable, then $\exists x \mathcal{C}$ is a constraint.

Constraints can be regarded as first order formulas using the atomic formulas $\top$, $t \not\preceq \{t_1, \ldots, t_n\}$ and $E \vdash_\forall s{\approx}t$ so that $\ldots \not\preceq \{\ldots\}$ and $\ldots \vdash_\forall \ldots$ are considered as a family of predicate symbols of arbitrary arities. Then, we can speak about the set $var(\mathcal{C})$ of *free variables* of a constraint $\mathcal{C}$.

Constraints describe the domain of substitutions. If a constraint $\mathcal{C}$ is true on a substitution $\sigma$ we say that $\sigma$ *satisfies* $\mathcal{C}$:

**Definition 11 (Constraint satisfiability)** The notion "a substitution $\sigma$ satisfies the constraint $\mathcal{C}$" (denoted $\sigma \models \mathcal{C}$) is defined inductively as follows:

1. $\sigma \models \top$ for every substitution $\sigma$;
2. $\sigma \models E \vdash_\forall s{\approx}t$ iff $\sigma$ is a solution to the rigid equation $E \vdash_\forall s{\approx}t$;
3. $\sigma \models t \not\preceq \{t_1, \ldots, t_n\}$ iff $t\sigma$ is a variable and $t\sigma$ does not occur in $t_1\sigma, \ldots, t_n\sigma$;
4. $\sigma \models \mathcal{C}_1 \wedge \mathcal{C}_2$ iff $\sigma \models \mathcal{C}_1$ and $\sigma \models \mathcal{C}_2$;
5. $\sigma \models \exists x \mathcal{C}$ iff there is a substitution $\tau$ such that $\tau \models \mathcal{C}$ and $y\tau = y\sigma$ for every variable $y$ different from $x$.

Constraints $\mathcal{C}_1$ and $\mathcal{C}_2$ are *equivalent* iff for every substitution $\sigma$ we have $\sigma \models \mathcal{C}_1$ iff $\sigma \models \mathcal{C}_2$. A constraint $\mathcal{C}$ is *satisfiable* iff there is a substitution $\sigma$ satisfying $\mathcal{C}$.

$$\frac{}{\Gamma, A(s_1,\ldots,s_n), \Delta \to A(t_1,\ldots,t_n)\cdot E \vdash_{\forall} s_1{\approx}t_1 \wedge \ldots \wedge E \vdash_{\forall} s_n{\approx}t_n}\ (Ax)$$

$$\frac{}{\Gamma, \Lambda, \Delta \to \varphi \cdot \top}\ (\Lambda) \qquad \frac{}{\Gamma \to s{\approx}t \cdot E \vdash_{\forall} s{\approx}t}\ (\approx)$$

$$\frac{\Gamma, \varphi, \psi, \Delta \to \chi \cdot \mathcal{C}}{\Gamma, \varphi \wedge \psi, \Delta \to \chi \cdot \mathcal{C}}\ (\wedge \to) \qquad \frac{\Gamma \to \varphi \cdot \mathcal{C}_1 \quad \Gamma \to \psi \cdot \mathcal{C}_2}{\Gamma \to \varphi \wedge \psi \cdot \mathcal{C}_1 \wedge \mathcal{C}_2}\ (\to \wedge)$$

$$\frac{\Gamma, \varphi, \Delta \to \chi \cdot \mathcal{C}_1 \quad \Gamma, \psi, \Delta \to \chi \cdot \mathcal{C}_2}{\Gamma, \varphi \vee \psi, \Delta \to \chi \cdot \mathcal{C}_1 \wedge \mathcal{C}_2}\ (\vee \to)$$

$$\frac{\Gamma \to \varphi \cdot \mathcal{C}}{\Gamma \to \varphi \vee \psi \cdot \mathcal{C}}\ (\to \vee_1) \qquad \frac{\Gamma \to \psi \cdot \mathcal{C}}{\Gamma \to \varphi \vee \psi \cdot \mathcal{C}}\ (\to \vee_2)$$

$$\frac{\Gamma, \psi, \Delta \to \chi \cdot \mathcal{C}_1 \quad \Gamma, \varphi \supset \psi, \Delta \to \varphi \cdot \mathcal{C}_2}{\Gamma, \varphi \supset \psi, \Delta \to \chi \cdot \mathcal{C}_1 \wedge \mathcal{C}_2}\ (\supset\to) \qquad \frac{\varphi, \Gamma \to \psi \cdot \mathcal{C}}{\Gamma \to \varphi \supset \psi \cdot \mathcal{C}}\ (\to\supset)$$

$$\frac{\Gamma, \varphi\{y/x\}, \forall x\varphi, \Delta \to \chi \cdot \mathcal{C}}{\Gamma, \forall x\varphi, \Delta \to \chi \cdot \exists y\mathcal{C}}\ (\forall \to) \qquad \frac{\Gamma \to \varphi\{y/x\} \cdot \mathcal{C}}{\Gamma \to \forall x\varphi \cdot \exists y(y \not\preceq \{v_1,\ldots,v_n\} \wedge \mathcal{C})}\ (\to \forall)$$

$$\frac{\Gamma, \varphi\{y/x\}, \Delta \to \chi \cdot \mathcal{C}}{\Gamma, \exists x\varphi, \Delta \to \chi \cdot \exists y(y \not\preceq \{v_1,\ldots,v_n\} \wedge \mathcal{C})}\ (\exists \to) \qquad \frac{\Gamma \to \varphi\{y/x\} \cdot \mathcal{C}}{\Gamma \to \exists x\varphi \cdot \exists y\mathcal{C}}\ (\to \exists)$$

In the rule $(Ax)$, $A$ is a predicate symbol different from equality and $E$ is the set of all equations in $\Gamma, \Delta$. In the rule $(\approx)$, $E$ is the set of all equations in $\Gamma$. In the rules $(\forall \to)$–$(\to \exists)$ the variable $y$ has no free occurrences in the sequent in the conclusion of the rules. In the rule $(\to \forall)$, $v_1,\ldots,v_n$ are all free variables of $\Gamma, \forall x\varphi$. In the rule $(\exists \to)$, $v_1,\ldots,v_n$ are all free variables of $\Gamma, \exists x\varphi, \Delta, \chi$.

**Fig. 3.** Calculus $\mathbf{LJ}_c^=$

The calculus $\mathbf{LJ}_c^=$ derives *constrained sequents*:

**Definition 12** A *constrained sequent* is an expression $S \cdot \mathcal{C}$ where $S$ is a sequent, $\mathcal{C}$ is a constraint. The *constraint calculus* $\mathbf{LJ}_c^=$ of constrained sequents is shown in Figure 3.

The notion of a skeleton for $\mathbf{LJ}_c^=$-derivations is similar to that of $\mathbf{LJ}^=$-derivations.

Consider two examples of derivations in $\mathbf{LJ}_c^=$:

*Example 3* The following is a derivation in $\mathbf{LJ}_c^=$ of the formula $\forall x(x{\approx}a \vee x{\approx}b \supset \exists y(f(x,y){\approx}b)) \supset \exists u\exists v\exists w(f(f(u,v),w){\approx}b)$ (see Figure 2) with a suitable constraint. As in Figure 2, we denote the formula $\forall x(x{\approx}a \vee x{\approx}b \supset \exists y(f(x,y){\approx}b))$ by $\varphi$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\varphi \to f(f(u,v),w){\approx}b \cdot \vdash_{\forall} f(f(u,v),w){\approx}b}\ (\approx)}{\varphi \to \exists w(f(f(u,v),w){\approx}b) \cdot \exists w(\vdash_{\forall} f(f(u,v),w){\approx}b)}\ (\to \exists)}{\varphi \to \exists v\exists w(f(f(u,v),w){\approx}b) \cdot \exists v\exists w(\vdash_{\forall} f(f(u,v),w){\approx}b)}\ (\to \exists)}{\varphi \to \exists u\exists v\exists w(f(f(u,v),w){\approx}b) \cdot \exists u\exists v\exists w(\vdash_{\forall} f(f(u,v),w){\approx}b)}\ (\to \exists)}{\to \varphi \supset \exists u\exists v\exists w(f(f(u,v),w){\approx}b) \cdot \exists u\exists v\exists w(\vdash_{\forall} f(f(u,v),w){\approx}b)}\ (\to\supset)$$

The resulting constraint $\exists u \exists v \exists w(\vdash_\forall f(f(u,v),w) \approx b)$ of this derivation is evidently unsatisfiable.

*Example 4* Another proof of this formula (with the same skeleton as that of the derivation of Figure 2) is

$$
\dfrac{}{\begin{array}{c} f(x_1,y_1)\approx b, f(x_2,y_2)\approx b, \varphi \rightarrow \\ f(f(u,v),w)\approx b \cdot \mathcal{C}_1 \end{array}}\ (\approx)
$$

$$
\dfrac{\begin{array}{c} f(x_1,y_1)\approx b, f(x_2,y_2)\approx b, \varphi \rightarrow \\ f(f(u,v),w)\approx b \cdot \mathcal{C}_1 \end{array}}{\begin{array}{c} f(x_1,y_1)\approx b, f(x_2,y_2)\approx b, \varphi \rightarrow \\ \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_2 \end{array}}\ (\rightarrow \exists)
$$

$$
\dfrac{\begin{array}{c} f(x_1,y_1)\approx b, f(x_2,y_2)\approx b, \varphi \rightarrow \\ \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_2 \end{array}}{\begin{array}{c} f(x_1,y_1)\approx b, \exists y(f(x_2,y)\approx b), \varphi \rightarrow \\ \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_4 \end{array}}\ (\exists \rightarrow)
\qquad
\dfrac{\dfrac{}{\cdots \rightarrow x_2\approx b \cdot \mathcal{C}_3}\ (\approx)}{\cdots \rightarrow x_2\approx a \vee x_2\approx b \cdot \mathcal{C}_3}\ (\rightarrow \vee_2)
$$

$$
\dfrac{\begin{array}{c} f(x_1,y_1)\approx b, \exists y(f(x_2,y)\approx b), \varphi \rightarrow \\ \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_4 \end{array} \qquad \cdots \rightarrow x_2\approx a \vee x_2\approx b \cdot \mathcal{C}_3}{\begin{array}{c} f(x_1,y_1)\approx b, x_2\approx a \vee x_2\approx b \supset \exists y(f(x_2,y)\approx b), \varphi \rightarrow \\ \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_5 \end{array}}\ (\supset\rightarrow)
$$

$$
\dfrac{\dfrac{\begin{array}{c} f(x_1,y_1)\approx b, x_2\approx a \vee x_2\approx b \supset \exists y(f(x_2,y)\approx b), \varphi \rightarrow \\ \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_5 \end{array}}{f(x_1,y_1)\approx b, \varphi \rightarrow \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_6}\ (\forall \rightarrow)}{\begin{array}{c} f(x_1,y_1)\approx b, \varphi \rightarrow \\ \exists v \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_7 \end{array}}\ (\rightarrow \exists)
$$

$$
\dfrac{\begin{array}{c} f(x_1,y_1)\approx b, \varphi \rightarrow \\ \exists v \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_7 \end{array}}{\begin{array}{c} \exists y(f(x_1,y)\approx b), \varphi \rightarrow \\ \exists v \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_9 \end{array}}\ (\exists \rightarrow)
\qquad
\dfrac{\dfrac{}{\cdots \rightarrow x_1\approx a \cdot \mathcal{C}_8}\ (\approx)}{\cdots \rightarrow x_1\approx a \vee x_1\approx b \cdot \mathcal{C}_8}\ (\rightarrow \vee_1)
$$

$$
\dfrac{\dfrac{\dfrac{\dfrac{\begin{array}{c} \exists y(f(x_1,y)\approx b), \varphi \rightarrow \\ \exists v \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_9 \end{array} \qquad \cdots \rightarrow x_1\approx a \vee x_1\approx b \cdot \mathcal{C}_8}{x_1\approx a \vee x_1\approx b \supset \exists y(f(x_1,y)\approx b), \varphi \rightarrow \exists v \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_{10}}\ (\supset\rightarrow)}{\varphi \rightarrow \exists v \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_{11}}\ (\forall \rightarrow)}{\varphi \rightarrow \exists u \exists v \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_{12}}\ (\rightarrow \exists)}{\rightarrow \varphi \supset \exists u \exists v \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_{12}}\ (\rightarrow\supset)
$$

Here the constraints $\mathcal{C}_1$–$\mathcal{C}_{12}$ are as follows:

$$
\begin{array}{ll}
\mathcal{C}_1 & \rightleftharpoons f(x_1,y_1)\approx b, f(x_2,y_2)\approx b \vdash_\forall f(f(u,v),w)\approx b \\
\mathcal{C}_2 & \rightleftharpoons \exists w(f(x_1,y_1)\approx b, f(x_2,y_2)\approx b \vdash_\forall f(f(u,v),w)\approx b) \\
\mathcal{C}_3 & \rightleftharpoons f(x_1,y_1)\approx b \vdash_\forall x_2\approx b \\
\mathcal{C}_4 & \rightleftharpoons \exists y_2(y_2 \not\preceq \{x_1,y_1,x_2,u,v\} \wedge \exists w(f(x_1,y_1)\approx b, f(x_2,y_2)\approx b \vdash_\forall f(f(u,v),w)\approx b)) \\
\mathcal{C}_5 & \rightleftharpoons \mathcal{C}_4 \wedge \mathcal{C}_3 \\
\mathcal{C}_6 & \rightleftharpoons \exists x_2(\mathcal{C}_4 \wedge \mathcal{C}_3) \\
\mathcal{C}_7 & \rightleftharpoons \exists v \exists x_2(\mathcal{C}_4 \wedge \mathcal{C}_3) \\
\mathcal{C}_8 & \rightleftharpoons \vdash_\forall x_1\approx a \\
\mathcal{C}_9 & \rightleftharpoons \exists y_1(y_1 \not\preceq \{x_1,u\} \wedge \exists v \exists x_2(\mathcal{C}_4 \wedge \mathcal{C}_3)) \\
\mathcal{C}_{10} & \rightleftharpoons \exists y_1(y_1 \not\preceq \{x_1,u\} \wedge \exists v \exists x_2(\mathcal{C}_4 \wedge \mathcal{C}_3)) \wedge \vdash_\forall x_1\approx a \\
\mathcal{C}_{11} & \rightleftharpoons \exists x_1(\exists y_1(y_1 \not\preceq \{x_1,u\} \wedge \exists v \exists x_2(\mathcal{C}_4 \wedge \mathcal{C}_3)) \wedge \vdash_\forall x_1\approx a) \\
\mathcal{C}_{12} & \rightleftharpoons \exists u \exists x_1(\exists y_1(y_1 \not\preceq \{x_1,u\} \wedge \exists v \exists x_2(\mathcal{C}_4 \wedge \mathcal{C}_3)) \wedge \vdash_\forall x_1\approx a)
\end{array}
$$

The resulting constraint of this derivation $\mathcal{C}_{12}$ is satisfiable. To establish this, we first note that $\mathcal{C}_{12}$ is satisfiable iff such is the following constraint $\mathcal{C}$ obtained from $\mathcal{C}_{12}$ be removing all quantifiers:

$$
\begin{array}{l}
f(x_1,y_1)\approx b, f(x_2,y_2)\approx b \vdash_\forall f(f(u,v),w)\approx b \ \wedge f(x_1,y_1)\approx b \vdash_\forall x_2\approx b \ \wedge \\
y_2 \not\preceq \{x_1,y_1,x_2,u,v\} \wedge \vdash_\forall x_1\approx a \ \wedge y_1 \not\preceq \{x_1,u\}
\end{array}
$$

This quantifier-free constraint is satisfied by the substitution $\{a/u, a/x_1, y_1/v, b/x_2, y_2/w\}$.

The following statement connects the derivability in $\mathbf{LJ}^=$ with the derivability in $\mathbf{LJ}_c^=$:

**Theorem 13** *Let $\Gamma \to \varphi$ be a sequent, $\theta$ be a substitution and $\mathcal{S}$ be a skeleton. Then the following conditions are equivalent:*

1. *There is a derivation of $\Gamma\theta \to \varphi\theta$ in $\mathbf{LJ}^=$ with the skeleton $\mathcal{S}$;*
2. *There is a constraint $\mathcal{C}$ and a derivation of $\Gamma \to \varphi \cdot \mathcal{C}$ in $\mathbf{LJ}_c^=$ with the skeleton $\mathcal{S}$ such that $\theta \models \mathcal{C}$.*

The following two theorems are straightforward consequences of Theorem 13:

**Theorem 14** *Let $S$ be a closed sequent and $\mathcal{S}$ be a skeleton. Then $S$ has a derivation in $\mathbf{LJ}^=$ with the skeleton $\mathcal{S}$ iff there is a satisfiable constraint $\mathcal{C}$ such that $S \cdot \mathcal{C}$ has a derivation in $\mathbf{LJ}_c^=$ with the skeleton $\mathcal{S}$.*

**Theorem 15** *Let $S$ be a closed sequent. Then $S$ is derivable in $\mathbf{LJ}^=$ iff there is a satisfiable constraint $\mathcal{C}$ such that $S \cdot \mathcal{C}$ is derivable in $\mathbf{LJ}_c^=$.*

Note that the set of constraints $\mathcal{C}$ such that $S \cdot \mathcal{C}$ is derivable in $\mathbf{LJ}_c^=$ is uniquely defined by the skeleton of a derivation of $S \cdot \mathcal{C}$ in $\mathbf{LJ}_c^=$ up to renaming of bound variables. Thus, the proof-search in intuitionistic logic can be considered as consisting of two parts: constructing a skeleton and the constraint satisfaction. The two parts should not necessarily be processed in this order, as the constraint satisfaction algorithm can try to check constraints after each step of the skeleton construction.

According to Theorem 9, the skeleton instantiation problem is undecidable which causes doubts about the efficiency of such procedures. In classical logic, there are many complete methods for reasoning with equality which do not use simultaneous rigid $E$-unification. However, for intuitionistic logic, there is a simple reduction of simultaneous rigid $E$-unifiability to the provability problem, and hence handling of simultaneous rigid $E$-unification is inevitable for designing a complete procedure. In addition, some special cases of simultaneous rigid $E$-unification are decidable [2] (for example, the function-free case is NP-complete) which allows one to use decision procedures for these fragments.

## 5 Constraint satisfiability and simultaneous rigid $E$-unification

The main result of this section is that the skeleton instantiation problem is polynomial-time equivalent to simultaneous rigid $E$-unifiability. This shows close relationships between the two problems. In fact, in the proof of this statement we give a structure-preserving translations between the two problems.

Note that any system or rigid equations $\{R_1, \ldots, R_n\}$ is equivalent to the constraint $R_1 \wedge \ldots \wedge R_n$ in the sense that any solution to $\{R_1, \ldots, R_n\}$ satisfies $R_1 \wedge \ldots \wedge R_n$ and vice versa. Thus, simultaneous rigid $E$-unification can be regarded as a special case of constraint satisfaction. Using some properties of simultaneous rigid $E$-unification [31], we can prove

**Theorem 16** *Let $\Sigma$ be any signature having a countable number of constants and maybe some other function symbols. The following three problems are polynomial-time equivalent:*

1. *Simultaneous rigid $E$-unifiability in the signature $\Sigma$;*
2. *The constraint satisfiability problem in the signature $\Sigma$;*
3. *The skeleton instantiation problem in the signature $\Sigma$.*

Countable number of constants in this theorem is needed because some reductions may introduce new constants.

This theorem is also true for classical logic, if we change the system $\mathbf{LJ}^=$ to the corresponding classical system allowing for more than one formula in the succedent. However, for classical logic we cannot show a straightforward reduction of simultaneous rigid $E$-unification to the provability problem. Indeed, there are many methods of automatic theorem proving in classical logic with equality not using simultaneous rigid $E$-unification. Our results show that the (direct or indirect) use of simultaneous rigid $E$-unification is unavoidable in theorem proving in intuitionistic logic with equality.

Theorem 16 has several important consequences related to the skeleton instantiation problem. Although simultaneous rigid $E$-unifiability is, in general, undecidable, its undecidability is only known for the case when the signature $\Sigma$ in which formulas are written contains binary function symbols (see proofs of the undecidability by Degtyarev and Voronkov [4, 5], Plaisted [21] and Veanes [27] for an overview of such proofs). Let us make some comments about results on the decidability of simultaneous rigid $E$-unification in various signatures $\Sigma$. We assume that $\Sigma$ contains a countable number of constants.

1. If all function symbols in $\Sigma$ are constants, then simultaneous rigid $E$-unification in this signature is decidable and NP-complete [2].
2. If $\Sigma$ contains at least one function symbols of arity $> 1$, then simultaneous rigid $E$-unification in this signature is undecidable [4, 27].
3. If $\Sigma$ contains one unary function symbol and no function symbols of arity $> 1$, then simultaneous rigid $E$-unification in this signature is decidable [2]. No complexity results are known and the decision algorithm uses a (non-trivial) algorithm for the Diophantine problem for addition and divisibility [1]. There is hardly a simple algorithm for simultaneous rigid $E$-unification in the signature $\Sigma$ since there the Diophantine problem for addition and divisibility can be reduced to it in an obvious way.
4. If $\Sigma$ contains more than one unary function symbol and no function symbols of arity $> 1$, then the decidability of simultaneous rigid $E$-unification in this signature is an open problem [2]. If it is decidable, then algorithms should

be extremely non-trivial because there is an obvious reduction of the word equation solvability (unification under associativity) to simultaneous rigid $E$-unification in the signature $\Sigma$.

By Theorem 16, all these results immediately imply corresponding results for the skeleton instantiation problem. In particular, for signatures with one unary function symbols and no function symbols of arity $> 1$, the skeleton instantiation problem is decidable. Thus, algorithms based on skeleton construction + skeleton instantiation can be implemented for such signatures.

## 6 Conclusion and related work

There are not so many papers on intuitionistic logic with equality. The decidability problems for intuitionistic logic with equality are studied in [14, 19, 20]. Lifschitz [14] proves the undecidability of some systems of intuitionistic equality. Mints [19] introduces a system of Skolem functions for formulas of intuitionistic logic restricting the application of quantifier rules. Orevkov [20] gives a partial classification of decidable pseudo-prenex[6] fragments of intuitionistic logic with and without equality. All these papers do not use function symbols, except for constants. The decidability of the prenex fragment of intuitionistic logic with equality and without function symbols follows from any of [19, 20]. In [9] Degtyarev and Voronkov prove some decidability and complexity results for prenex formulas of intuitionistic logic with and without equality.

Voda and Komara [28] consider so-called *Herbrand skeletons* (formulas obtained after a fixed number of quantifier duplications in a prefix skolemized formula) and prove that the corresponding skeleton instantiation problem is undecidable.

Tammet [26] implemented a theorem prover for intuitionistic logic based on the inverse method. At the moment the prover does not have equality, but Tammet is going to include equality as well (private communication). Our results provide a basis for including equality in such provers. Sahlin, Franzén and Haridi [24] implemented a theorem prover for intuitionistic logic where a non-standard formalization of equality is used. For example, in their logical system the formula $a \approx b \supset c \approx d$ is provable.

The results of our paper explain the (so far) unsuccessful attempts to incorporate equality in intuitionistic logic. To add equality to proof procedures for intuitionistic logic one has to develop algorithms for simultaneous rigid $E$-unifiability. No such algorithms are known yet.

We did not give any concrete procedure for proof search modulo simultaneous rigid $E$-unification. In fact, it is easy to construct such a procedure using rules of our constraint calculus in the backward direction. Such procedures are standard in tableau-based theorem provers, see e.g. Shankar [25], with the exception that in presence of equality we cannot use (even dynamic) skolemization.

[6] A *pseudo-prenex* is a finite sequence of $\forall x_i$, $\exists x_i$ and $\neg\neg$.

Since our technique does not always exploit particular properties of intuitionistic logic, many results of this paper may be reformulated for other non-classical logic, for example predicate variants of modal logics S4 and K.

**Acknowledgments**

I thank Anatoli Degtyarev for many helpful comments.

## References

1. A. Beltyukov. Decidability of the universal theory of natural numbers with addition and divisibility (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 60:15–28, 1976. English translation in: Journal of Soviet Mathematics.
2. A. Degtyarev, Yu. Matiyasevich, and A. Voronkov. Simultaneous rigid $E$-unification and related algorithmic problems. In *LICS'96*, page 11, 1996.
3. A. Degtyarev and A. Voronkov. Equality elimination for semantic tableaux. UPMAIL Technical Report 90, Uppsala University, Computing Science Department, December 1994.
4. A. Degtyarev and A. Voronkov. Simultaneous rigid $E$-unification is undecidable. UPMAIL Technical Report 105, Uppsala University, Computing Science Department, May 1995. To appear in Theoretical Computer Science.
5. A. Degtyarev and A. Voronkov. Reduction of second-order unification to simultaneous rigid $E$-unification. UPMAIL Technical Report 109, Uppsala University, Computing Science Department, June 1995.
6. A. Degtyarev and A. Voronkov. A new procedural interpretation of Horn clauses with equality. In Leon Sterling, editor, *Proceedings of the Twelfth International Conference on Logic Programming*, pages 565–579. The MIT Press, 1995.
7. A. Degtyarev and A. Voronkov. General connections via equality elimination. In M. De Glas and Z. Pawlak, editors, *Second World Conference on the Fundamentals of Artificial Intelligence (WOCFAI-95)*, pages 109–120, Paris, July 1995. Angkor.
8. A. Degtyarev and A. Voronkov. Equality elimination for the inverse method and extension procedures. In C.S. Mellish, editor, *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, volume 1, pages 342–347, Montréal, August 1995.
9. A. Degtyarev and A. Voronkov. Skolemization and decidability problems for fragments of intuitionistic logic. *Submitted*, 10 pages, 1996.
10. M. Fitting. *First Order Logic and Automated Theorem Proving.* Springer Verlag, New York, 1990.
11. J.H. Gallier, S. Raatz, and W. Snyder. Theorem proving using rigid $E$-unification: Equational matings. In *Proc. IEEE Conference on Logic in Computer Science (LICS)*, pages 338–346. IEEE Computer Society Press, 1987.
12. 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.
13. R.C.T. Lee and C.L. Chang. *Symbolic Logic and Mechanical Theorem Proving.* Academic Press, 1973.

14. V. Lifschitz. The decidability problem for some constructive theories of equality (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 4:78–85, 1967.
15. D.W. Loveland. *Automated Theorem Proving: a Logical Basis.* North Holland, 1978.
16. S.Yu. Maslov. An inverse method for establishing deducibility of nonprenex formulas of the predicate calculus. In J.Siekmann and G.Wrightson, editors, *Automation of Reasoning (Classical papers on Computational Logic)*, volume 2, pages 48–54. Springer Verlag, 1983.
17. S.Yu. Maslov. An invertible sequential variant of constructive predicate calculus (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 4, 1967. English Translation in: Seminars in Mathematics: Steklov Math. Inst. 4, Consultants Bureau, NY-London, 1969, p.36–42.
18. V.A. Matulis. On variants of classical predicate calculus with the unique deduction tree (in Russian). *Soviet Mathematical Doklady*, 148:768–770, 1963.
19. G.E. Mints. Collecting terms in the quantifier rules of the constructive predicate calculus (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 4:78–85, 1967.
20. V.P. Orevkov. Solvable classes of pseudo-prenex formulas (in Russian). *Zapiski Nauchnyh Seminarov LOMI*, 60:109–170, 1976. English translation in: Journal of Soviet Mathematics.
21. D.A. Plaisted. Special cases and substitutes for rigid $E$-unification. Technical Report MPI-I-95-2-010, Max-Planck-Institut für Informatik, November 1995.
22. G. Plotkin. Building-in equational theories. In Meltzer and Michie, editors, *Machine Intelligence*, volume 7, pages 73–90. Edinburgh University Press, Edinburgh, 1972.
23. G. Robinson and L.T. Wos. Paramodulation and theorem-proving in first order theories with equality. In Meltzer and Michie, editors, *Machine Intelligence*, volume 4, pages 135–150. Edinburgh University Press, Edinburgh, 1969.
24. D. Sahlin, T. Franzén, and S. Haridi. An intuitionistic predicate logic theorem prover. *Journal of Logic and Computation*, 2(5):619–656, 1992.
25. N. Shankar. Proof search in the intuitionistic sequent calculus. In D. Kapur, editor, *11th International Conference on Automated Deduction*, volume 607 of *Lecture Notes in Artificial Intelligence*, pages 522–536, Saratoga Springs, NY, USA, June 1992. Springer Verlag.
26. T. Tammet. A resolution theorem prover for intuitionistic logic. This volume.
27. M. Veanes. Undecidability proofs of simultaneous rigid $E$-unification. Upmail technical report, Uppsala University, Computing Science Department, 1996. To appear.
28. P.J. Voda and J. Komara. On Herbrand skeletons. Technical report, Institute of Informatics, Comenius University Bratislava, July 1995.
29. A. Voronkov. Theorem proving in non-standard 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.
30. A. Voronkov. Proof-search in intuitionistic logic based on the constraint satisfaction. UPMAIL Technical Report 120, Uppsala University, Computing Science Department, January 1996.
31. A. Voronkov. On proof-search in intuitionistic logic with equality, or back to simultaneous rigid $E$-Unification. UPMAIL Technical Report 121, Uppsala University, Computing Science Department, January 1996.