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

Andrei Voronkov (`voronkov@csd.uu.se`)*
Computing Science Department, Uppsala University, Box 311, S 751 05, Uppsala, Sweden

1995/04/07

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.

Keywords: Intuitionistic logic, equality, rigid E-unification

Table of Contents

* Supported by Swedish TFR grants

© 1998 *Kluwer Academic Publishers. Printed in the Netherlands.*

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 article we analyze the problem of variable instantiation in intuitionistic logic with equality. Our analysis shows close connections between the variable instantiation problem and simultaneous rigid E-unification introduced by Gallier, Raatz and Snyder [12]. 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 [33]; also Loveland [23] in the context of model elimination; and Fitting [11] in the context of the tableau 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 [17, 18] proposed to instantiate variables by subterms of terms already occurring in the (partial) derivation. This method was characterized as *minus-normalization* by Matulis [26] and Maslov [24]. 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 (Robinson [34]) and *paramodulation* originally introduced by Robinson and Wos [33], or *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 (for the latter see Voronkov [42]; Degtyarev and Voronkov [3][1]).

4. Plotkin [32] proposed to use *E-unification*. However, E-unification is applicable only in some restricted cases.

[1] The inverse method was originally described by Maslov [25], but he never considered explicitly how to instantiate variables.

5. Gallier, Raatz and Snyder [12] 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, the recent results on the undecidability of simultaneous rigid E-unification by Degtyarev and Voronkov [9] show that this approach can hardly be an advantage over methods using most general unifiers.

6. Degtyarev and Voronkov [7, 5, 4] 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 2.2 below. This example also demonstrates non-applicability of equality elimination to intuitionistic logic.

The aim of this article 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 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 article is structured as follows. Section 2 introduces main notions and defines simultaneous rigid E-unification. Section 3 defines a sequent calculus $\mathbf{LJ}^{\approx}$ for intuitionistic logic with equality. In Section 3.1 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 $\mathbf{LJ}^{\approx}_c$ which is a constraint-based modification of $\mathbf{LJ}^{\approx}$. We also prove a general theorem about the connection between derivability in $\mathbf{LJ}^{\approx}$ and that in $\mathbf{LJ}_c^{\approx}$. In Section 5 we prove that the skeleton instantiation problem is polynomial-time equivalent to simultaneous rigid E-unifiability. In Section 7 we briefly consider related work.

This article is an extended version of [44].

2. Preliminaries

Formulas are defined as usual, using atomic formulas, the constant Λ, 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 $\Gamma \to \varphi$ where Γ is a finite sequence of formulas and φ is a formula. A formula is *closed* if it has no free variables. A sequent $\Gamma \to \varphi$ is *closed* if all formulas in Γ, φ are closed. We assume that all free variables in formulas are different from all bound variables.

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

By X we denote a countable infinite set of variables. The set of all terms in the signature Σ with variables in X is denoted by $T_\Sigma(X)$. By T_Σ we denote $T_\Sigma(\emptyset)$, i.e. the set of all ground terms in the signature Σ. *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 θ 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 θ is the set $\{x \mid x\theta \neq x\}$. The *range* $ran(\theta)$ of the substitution θ is defined by $ran(\theta) \rightleftharpoons \{x\theta \mid x \in dom(\theta)\}$. Substitutions θ with the domain $x_1, \ldots, x_n$ are denoted by $\{x_1 \mapsto x_1\theta, \ldots, x_n \mapsto x_n\theta\}$. Let E be a term, formula, sequence of formulas or sequent. For a substitution $\theta = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$, $E\theta$ denotes the result of *applying θ to E*, i.e. renaming of those bound variables in E that coincide with a variable occurring in any t_i and the replacement of all free occurrences of x_i by t_i.

Let φ be a formula all whose free variables are $x_1, \ldots, x_n$, in the order of their occurrences in φ. 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.

2.1. Simultaneous rigid E-unification

In this section we define simultaneous rigid E-unification originally introduced in Gallier, Raatz and Snyder [12].

Definition 2.1. (Simultaneous rigid E-unification)
We call a *rigid equation* any 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 σ 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 valid[2]. 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*[3].

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 2.2. The system of two rigid equations[4]

$$\begin{array}{l} a \approx b \vdash_\forall g(x,u,v) \approx g(y,f(c),f(d)) \\ c \approx d \vdash_\forall g(u,x,y) \approx g(v,f(a),f(b)) \end{array}$$

has one solution $\{x \mapsto f(a), y \mapsto f(b), u \mapsto f(c), v \mapsto f(d)\}$. This system of rigid equations has an interesting property: we cannot find its solution using V-paramodulation of Lee and Chang [19] (later described as rigid paramodulation by Plaisted [31]) and most general solutions to the equations in the right-hand sides of $\vdash_\forall$.

Example 2.3. The system of two rigid equations

$$\begin{array}{l} h(x_1) \approx a, h(x_2) \approx a, h(a) \approx y \vdash_\forall y \approx g(h(y)) \\ h(x_1) \approx a, h(x_2) \approx a, h(b) \approx f(y) \vdash_\forall y \approx g(f(y)) \end{array}$$

has an infinite number of solutions. Among these solutions are the substitutions of the form $\{x_1 \mapsto a, x_2 \mapsto b, y \mapsto g(h^n(a))\}$, for every $n \geq 0$.

[2] 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 $\{s_1\sigma \approx t_1\sigma, \ldots, s_n\sigma \approx t_n\sigma\}$.

[3] We slightly changed the terminology compared to Gallier, Raatz and Snyder [12]. Our notion of a rigid equation corresponds to the notion of a rigid E-unification problem of [12]. Our notion of a system of rigid equations corresponds to the notion of a simultaneous rigid E-unification problem of [12]. 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.

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

The following result is proven in Degtyarev and Voronkov [9]:

THEOREM 2.4. *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.

A more precise formulation of the undecidability results is due to Veanes [39]:

THEOREM 2.5. *Simultaneous rigid E-unifiability in any signature with one function symbol of arity ≥ 2 and at least one constant is undecidable.*

The decidability of simultaneous rigid E-unification in the signatures with unary function symbols is an open problem (see Gurevich and Voronkov [16]).

3. The calculus $\mathbf{LJ}^{\approx}$

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

Definition 3.1. (Calculus $\mathbf{LJ}^{\approx}$) The calculus $\mathbf{LJ}^{\approx}$ for intuitionistic logic with equality derives sequents. The rules of the calculus are shown in Figure 3. Derivations in $\mathbf{LJ}^{\approx}$ are trees defined as usual.

We formulate $\mathbf{LJ}^{\approx}$ as a system without the cut rule. It is not difficult to prove that cut is admissible in $\mathbf{LJ}^{\approx}$ using the general cut-elimination theorem given in Sazonov [36], Buss [1] or Girard [14]).

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 $\mathbf{LJ}^{\approx}$ 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 article.

The following simple statement connects the derivability in $\mathbf{LJ}^{\approx}$ with solutions to rigid equations.

LEMMA 3.2. *Let σ be a substitution. The following conditions are equivalent:*

$$\frac{}{\Gamma, A, \Delta \to A}\ (Ax) \qquad \frac{}{\Gamma, \Lambda, \Delta \to \varphi}\ (\Lambda) \qquad \frac{}{\Gamma \to t \approx t}\ (\approx)$$

$$\frac{\Gamma\{x \mapsto t\}, s\approx t, \Delta\{x \mapsto t\} \to \chi\{x \mapsto t\}}{\Gamma\{x \mapsto s\}, s\approx t, \Delta\{x \mapsto s\} \to \chi\{x \mapsto s\}}\ (\approx_1)$$

$$\frac{\Gamma\{x \mapsto s\}, s\approx t, \Delta\{x \mapsto s\} \to \chi\{x \mapsto s\}}{\Gamma\{x \mapsto t\}, s\approx t, \Delta\{x \mapsto t\} \to \chi\{x \mapsto t\}}\ (\approx_2)$$

$$\frac{\Gamma, \varphi, \psi, \Delta \to \chi}{\Gamma, \varphi \wedge \psi, \Delta \to \chi}\ (\wedge \to) \qquad \frac{\Gamma \to \varphi \quad \Gamma \to \psi}{\Gamma \to \varphi \wedge \psi}\ (\to \wedge)$$

$$\frac{\Gamma, \varphi, \Delta \to \chi \quad \Gamma, \psi, \Delta \to \chi}{\Gamma, \varphi \vee \psi, \Delta \to \chi}\ (\vee \to)$$

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

$$\frac{\Gamma, \psi, \Delta \to \chi \quad \Gamma, \varphi \supset \psi, \Delta \to \varphi}{\Gamma, \varphi \supset \psi, \Delta \to \chi}\ (\supset\to) \qquad \frac{\varphi, \Gamma \to \psi}{\Gamma \to \varphi \supset \psi}\ (\to\supset)$$

$$\frac{\Gamma, \varphi\{x \mapsto t\}, \forall x\varphi, \Delta \to \chi}{\Gamma, \forall x\varphi, \Delta \to \chi}\ (\forall \to) \qquad \frac{\Gamma \to \varphi\{x \mapsto y\}}{\Gamma \to \forall x\varphi}\ (\to \forall)$$

$$\frac{\Gamma, \varphi\{x \mapsto y\}, \Delta \to \chi}{\Gamma, \exists x\varphi, \Delta \to \chi}\ (\exists \to) \qquad \frac{\Gamma \to \varphi\{x \mapsto t\}}{\Gamma \to \exists x\varphi}\ (\to \exists)$$

In this figure, A ranges over atomic formulas, φ, ψ, χ range over arbitrary formulas and Γ, Δ 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.

Figure 1. Calculus $\mathbf{LJ}^{\approx}$

1. *The sequent* $s_1\sigma\approx t_1\sigma, \ldots, s_n\sigma\approx t_n\sigma \to s\sigma\approx t\sigma$ *is derivable in* $\mathbf{LJ}^{\approx}$*;*
2. *The sequent* $s_1\sigma\approx t_1\sigma, \ldots, s_n\sigma\approx t_n\sigma \to s\sigma\approx t\sigma$ *is derivable in* $\mathbf{LJ}^{\approx}$ *using only* $(\approx)$, $(\approx_1)$ *and* $(\approx_2)$*;*
3. σ *is a solution to the rigid equation* $s_1\approx t_1, \ldots, s_n\approx t_n \vdash_\forall s\approx t$.
   Proof. Standard.

Note. To prove this lemma, it is enough to have a restricted system where the rules $(\approx_1)$ and $(\approx_2)$ only replace terms in the succedents of the sequents. The completeness of the system with such restricted rules for intuitionistic logic with equality is proven by a number of authors (e.g. Orevkov [28]).

3.1. Derivation skeletons

In this section we study two kinds of skeletons of $\mathbf{LJ}^{\approx}$-derivations. The notion of a skeleton provides an abstraction of a family of derivations.

Definition 3.3. (Weak skeleton) A *weak derivation skeleton* is a tree such that

1. Its nodes are labelled by the names of inference rules in $\mathbf{LJ}^{\approx}$, 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 $\mathbf{LJ}^{\approx}$.

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

Example 3.4. The following is a weak derivation skeleton. This weak skeleton corresponds to derivations of Example 3.10 in some precise sense defined below (Definition 3.6).

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

The following are *not* weak derivation skeletons:

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

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 a stronger notion of skeletons:

Definition 3.5. (Skeleton) A *skeleton* is defined similar to the weak skeleton with the following difference: all nodes labelled by antecedent rules or (Ax) are additionally labelled by a natural number.

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 $(\approx_2)$.

We shall illustrate this in Examples 3.7–3.10 below.

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 Γ in the corresponding rules of $\mathbf{LJ}^{\approx}$ (Figure 3). We always display the number as an index.

The weak skeleton of a derivation Π is obtained from Π by removing all sequents and all applications of rules $(\approx_1)$ and $(\approx_2)$. More formally, the skeleton and the weak skeleton of a derivation are defined as follows:

Definition 3.6. (Skeleton of a derivation) Let Π be a derivation in $\mathbf{LJ}^{\approx}$. *The weak skeleton* Π' *of* Π and *the skeleton* Π'' *of* Π are defined inductively as follows:

1. If Π has the form $\dfrac{}{\Gamma, A, \Delta \to A}\,(Ax)$ then Π' and Π'' are $\overline{\quad}\,(Ax)$ and $\overline{\quad}\,(Ax_n)$ respectively, where n is the number of formulas in Γ.

2. If Π has the form $\dfrac{}{\Gamma, \Lambda, \Delta \to \varphi}\,(\Lambda)$ then both Π' and Π'' are $\overline{\quad}\,(\Lambda)$.

3. If Π has the form $\dfrac{}{\Gamma \to t\approx t}\,(\approx)$ then both Π' and Π'' are $\overline{\quad}\,(\approx)$.

4. If Π has the form $\dfrac{\begin{matrix}\Pi_1\\ \Gamma\{x \mapsto t\}, s\approx t, \Delta\{x \mapsto t\} \to \chi\{x \mapsto t\}\end{matrix}}{\Gamma\{x \mapsto s\}, s\approx t, \Delta\{x \mapsto s\} \to \chi\{x \mapsto s\}}\,(\approx_1)$ and Π_1' and Π_1'' are, respectively, the weak skeleton and the skeleton of Π_1, then $\Pi' \rightleftharpoons \Pi_1'$ and $\Pi'' \rightleftharpoons \Pi_1''$, and similar for the $(\approx_2)$-rule.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Lambda, (B \supset \Lambda), A, B \to \Lambda}\,(\Lambda) \qquad \dfrac{}{(A \supset \Lambda), (B \supset \Lambda), A, B \to A}\,(Ax)}{(A \supset \Lambda), (B \supset \Lambda), A, B \to \Lambda}\,(\supset\to)}{(A \supset \Lambda), (B \supset \Lambda), A \wedge B \to \Lambda}\,(\wedge \to)}{(A \supset \Lambda), (B \supset \Lambda) \wedge A \wedge B \to \Lambda}\,(\wedge \to)}{(A \supset \Lambda) \wedge (B \supset \Lambda) \wedge A \wedge B \to \Lambda}\,(\wedge \to)}{\to ((A \supset \Lambda) \wedge (B \supset \Lambda) \wedge A \wedge B) \supset \Lambda}\,(\to\supset)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{(A \supset \Lambda), \Lambda, A, B \to \Lambda}\,(\Lambda) \qquad \dfrac{}{(A \supset \Lambda), (B \supset \Lambda), A, B \to B}\,(Ax)}{(A \supset \Lambda), (B \supset \Lambda), A, B \to \Lambda}\,(\supset\to)}{(A \supset \Lambda), (B \supset \Lambda), A \wedge B \to \Lambda}\,(\wedge \to)}{(A \supset \Lambda), (B \supset \Lambda) \wedge A \wedge B \to \Lambda}\,(\wedge \to)}{(A \supset \Lambda) \wedge (B \supset \Lambda) \wedge A \wedge B \to \Lambda}\,(\wedge \to)}{\to ((A \supset \Lambda) \wedge (B \supset \Lambda) \wedge A \wedge B) \supset \Lambda}\,(\to\supset)$$

Figure 2. Two derivations of a propositional formula

5. If Π has the form $\dfrac{\overset{\Pi_1}{\varphi, \Gamma \to \psi}}{\Gamma \to \varphi \supset \psi}\,(\to\supset)$ and Π_1' and Π_1'' are, respectively, the weak skeleton and the skeleton of Π_1, then Π' and Π'' are $\dfrac{\Pi_1'}{}\,(\to\supset)$ and $\dfrac{\Pi_1''}{}\,(\to\supset)$ respectively, and similar for all other succedent rules.

6. If Π has the form $\dfrac{\overset{\Pi_1}{\Gamma, \psi, \Delta \to \chi} \quad \overset{\Pi_2}{\Gamma, \varphi \supset \psi, \Delta \to \varphi}}{\Gamma, \varphi \supset \psi, \Delta \to \chi}\,(\supset\to)$ and Π_1', Π_2' and Π_1'', Π_2'' are, respectively, the weak skeletons and the skeletons of Π_1, Π_2, then Π' and Π'' are $\dfrac{\Pi_1' \quad \Pi_2'}{}\,(\supset\to)$ and $\dfrac{\Pi_1'' \quad \Pi_2''}{}\,(\supset\to_n)$ respectively, where n is the number of formulas in Γ, and similar for all other antecedent rules.

The following example shows that the notion of a weak skeleton abstracts from some details of the structure of derivations which is explicitly presented in skeletons.

Example 3.7. Consider the two derivations of the propositional formula $((A \supset \Lambda) \wedge (B \supset \Lambda) \wedge A \wedge B) \supset \Lambda$ given in Figure 3.1. They have the same weak skeleton

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{}\,(\Lambda)\qquad\dfrac{}{}\,(Ax)}{}\,(\supset\to)}{}\,(\wedge\to)}{}\,(\wedge\to)}{}\,(\wedge\to)}{}\,(\to\supset)$$

but different skeletons:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{}\,(\Lambda)\qquad\dfrac{}{}\,(Ax_2)}{}\,(\supset\to_0)}{}\,(\wedge\to_2)}{}\,(\wedge\to_1)}{}\,(\wedge\to_0)}{}\,(\to\supset)\qquad\qquad\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{}\,(\Lambda)\qquad\dfrac{}{}\,(Ax_3)}{}\,(\supset\to_1)}{}\,(\wedge\to_2)}{}\,(\wedge\to_1)}{}\,(\wedge\to_0)}{}\,(\to\supset)$$

Skeletons uniquely identify derivations of propositional formulas. However, there may be many derivations of the same non-propositional formula (even without equality) with the same skeleton due to different instantiations of terms in the quantifier rules.

Example 3.8. The following two different derivations of the formula $\forall x A(x) \supset \exists x A(x)$ have the same skeleton:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A(x),\forall xA(x)\to A(x)}\,(Ax)}{\forall xA(x)\to A(x)}\,(\forall\to)}{\forall xA(x)\to\exists xA(x)}\,(\to\exists)}{\to\forall xA(x)\supset\exists xA(x)}\,(\to\supset)\qquad\qquad\dfrac{\dfrac{\dfrac{\dfrac{}{A(f(a)),\forall xA(x)\to A(f(a))}\,(Ax)}{\forall xA(x)\to A(f(a))}\,(\forall\to)}{\forall xA(x)\to\exists xA(x)}\,(\to\exists)}{\to\forall xA(x)\supset\exists xA(x)}\,(\to\supset)$$

Finally, skeletons abstract from the order of applications of the rules $(\approx_1)$ and $(\approx_2)$. Before giving an example, we introduce an important notion:

Definition 3.9. (Regular derivation)
A derivation in $\mathbf{LJ}^{\approx}$ is called *regular* if above any application of $(\approx_1)$ or $(\approx_2)$ there can only be applications of the rules $(\approx_1)$, $(\approx_2)$, (Ax), (Λ) and $(\approx)$.

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{$(\approx)$}
\UnaryInfC{$f(a,y)\approx b, f(b,z)\approx b, \varphi \to f(b,z)\approx f(b,z)$}
\RightLabel{$(\approx_2)$}
\UnaryInfC{$f(a,y)\approx b, f(b,z)\approx b, \varphi \to f(b,z)\approx b$}
\RightLabel{$(\approx_1)$}
\UnaryInfC{$f(a,y)\approx b, f(b,z)\approx b, \varphi \to f(f(a,y),z)\approx b$}
\RightLabel{$(\to \exists)$}
\UnaryInfC{$f(a,y)\approx b, f(b,z)\approx b, \varphi \to \exists w(f(f(a,y),w)\approx b)$}
\RightLabel{$(\exists \to)$}
\UnaryInfC{$f(a,y)\approx b, \exists y(f(b,y)\approx b), \varphi \to \exists w(f(f(a,y),w)\approx b)$}
\AxiomC{}
\RightLabel{$(\approx)$}
\UnaryInfC{$\cdots \to b\approx b$}
\RightLabel{$(\to \vee_2)$}
\UnaryInfC{$\cdots \to b\approx a \vee b\approx b$}
\RightLabel{$(\supset\to)$}
\BinaryInfC{$f(a,y)\approx b, b\approx a \vee b\approx b \supset \exists y(f(b,y)\approx b), \varphi \to \exists w(f(f(a,y),w)\approx b)$}
\RightLabel{$(\forall \to)$}
\UnaryInfC{$f(a,y)\approx b, \varphi \to \exists w(f(f(a,y),w)\approx b)$}
\RightLabel{$(\to \exists)$}
\UnaryInfC{$f(a,y)\approx b, \varphi \to \exists v\exists w(f(f(a,v),w)\approx b)$}
\RightLabel{$(\exists \to)$}
\UnaryInfC{$\exists y(f(a,y)\approx b), \varphi \to \exists v\exists w(f(f(a,v),w)\approx b)$}
\AxiomC{}
\RightLabel{$(\approx)$}
\UnaryInfC{$\cdots \to a\approx a$}
\RightLabel{$(\to \vee_1)$}
\UnaryInfC{$\cdots \to a\approx a \vee a\approx b$}
\RightLabel{$(\supset\to)$}
\BinaryInfC{$a\approx a \vee a\approx b \supset \exists y(f(a,y)\approx b), \varphi \to \exists v\exists w(f(f(a,v),w)\approx b)$}
\RightLabel{$(\forall \to)$}
\UnaryInfC{$\varphi \to \exists v\exists w(f(f(a,v),w)\approx b)$}
\RightLabel{$(\to \exists)$}
\UnaryInfC{$\varphi \to \exists u\exists v\exists w(f(f(u,v),w)\approx b)$}
\RightLabel{$(\to\supset)$}
\UnaryInfC{$\to \varphi \supset \exists u\exists v\exists w(f(f(u,v),w)\approx b)$}
\end{prooftree}
$$

In this figure, we denote by φ the formula

$$\forall x(x\approx a \vee x\approx b \supset \exists y(f(x,y)\approx b)).$$

We also denote by $\cdots$ some irrelevant parts of the sequents.

Figure 3. A regular derivation of a formula with equality

total.tex; 28/08/1998; 14:44; p.12

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

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

In this figure, we denote by φ 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.

Figure 4. A nonregular derivation of a formula with equality

total.tex; 28/08/1998; 14:44; p.13

Example 3.10. Consider two derivations of the formula

$$\forall x(x \approx a \vee x \approx b \supset \exists y(f(x,y) \approx b)) \supset \exists u \exists v \exists w(f(f(u,v),w) \approx b)$$

given in Figures 3 and 4. Both derivations have the weak skeleton of Example 3.4. They have the same skeleton and use the same instantiation of variables. However, the proofs are different because of the order of applications of the rules $(\approx_1)$ and $(\approx_2)$. Moreover, the first derivation is regular while the second is not regular.

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 Kanger [18] and Lifschitz [21].

THEOREM 3.11. *Let S be a sequent and $\mathcal{S}$ be a skeleton (respectively, a weak skeleton). The following conditions are equivalent:*

1. *There is a derivation of S with the skeleton (respectively, the weak skeleton) $\mathcal{S}$;*

2. *There is a regular derivation of S with the skeleton (respectively, the weak skeleton) $\mathcal{S}$.*

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

Proof. An application of the rule $(\approx_1)$ or $(\approx_2)$, where the variable x has n occurrences in Γ, Δ, χ of the rule, can be replaced by n applications of the same rule. Thus, we can assume that in any application of $(\approx_1)$ or $(\approx_2)$, the variable x has precisely one occurrence in Γ, Δ, χ. Also, we can assume that x does not occur in the derivation, since otherwise x can be replaced by any variable not occurring in the derivation.

The proof of this theorem is based on proof transformations moving applications of $(\approx_1)$ and $(\approx_2)$ up in the derivation. For simplicity, we only consider one case.

Suppose that an application of $(\approx_2)$ is immediately above an application of $(\rightarrow \exists)$, and the occurrence of the variable x is in the succedent of the sequent:

$$\frac{\Gamma, s \approx t, \Delta \rightarrow (\exists y \varphi)\{x \mapsto s\}}{\Gamma, s \approx t, \Delta \rightarrow (\exists y \varphi)\{x \mapsto t\}}\ (\approx_2)$$

Without loss of generality we can assume that y does not occur in s, t. Since x does not occur in the derivation, this can be rewritten as

$$\frac{\Gamma, s{\approx}t, \Delta \to \exists y(\varphi\{x \mapsto s\})}{\Gamma, s{\approx}t, \Delta \to (\exists y\varphi)\{x \mapsto t\}}\ (\approx_2)$$

Then the fragment of the derivation consisting of the two rule applications has the form

$$\cfrac{\cfrac{\Gamma, s{\approx}t, \Delta \to (\varphi\{x \mapsto s\})\{y \mapsto r\}}{\Gamma, s{\approx}t, \Delta \to \exists y(\varphi\{x \mapsto s\})}\ (\to \exists)}{\Gamma, s{\approx}t, \Delta \to (\exists y\varphi)\{x \mapsto t\}}\ (\approx_2)$$

Since x does not occur in the derivation and y does not occur in s, t, the sequent $\Gamma, s{\approx}t, \Delta \to (\varphi\{x \mapsto s\})\{y \mapsto r\}$ is equal to $\Gamma, s{\approx}t, \Delta \to (\varphi\{y \mapsto r\})\{x \mapsto s\}$. Then this fragment of the derivation can be replaced by

$$\cfrac{\cfrac{\Gamma, s{\approx}t, \Delta \to (\varphi\{x \mapsto s\})\{y \mapsto r\}}{\Gamma, s{\approx}t, \Delta \to (\varphi\{x \mapsto t\})\{y \mapsto r\}}\ (\approx_2)}{\Gamma, s{\approx}t, \Delta \to \exists y(\varphi\{x \mapsto t\})}\ (\to \exists)$$

The proof for applications of other rules is similar.

In this article, we shall consider the problem of instantiation of a (weak) skeleton to a derivation:

Definition 3.12. The *(weak) skeleton instantiation problem* is the following problem. Given a closed sequent S and a (weak) skeleton $\mathcal{S}$, whether there is a derivation in **LJ** of S with the (weak) 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 [43] we proved that, for formulas without equality both problems are decidable. In addition, for formulas without equality

1. The weak skeleton instantiation problem is NP-complete;

2. The skeleton instantiation problem is in PTIME.

For formulas with equality both problems are undecidable. This follows from the following statement:

THEOREM 3.13. *Simultaneous rigid E-unifiability is polynomial-time reducible to any of the following three problems:*

1. *The problem of derivability in* $\mathbf{LJ}^{\approx}$*;*

2. *The weak skeleton instantiation problem;*

3. *The skeleton instantiation problem.*

Proof. Consider any system R of rigid equations of the form

$$\begin{array}{ccccc} s_{11}{\approx}t_{11} & \ldots & s_{1n_1}{\approx}t_{1n_1} & \vdash_{\forall} & s'_1{\approx}t'_1 \\ \vdots & & \vdots & & \vdots \\ s_{k1}{\approx}t_{k1} & \ldots & s_{kn_k}{\approx}t_{kn_k} & \vdash_{\forall} & s'_k{\approx}t'_k \end{array} \tag{1}$$

Let $x_1, \ldots, x_m$ be all variables occurring in R. Denote by φ_R the formula

$$\begin{array}{ll} \exists x_1 \ldots \exists x_m(& (s_{11}{\approx}t_{11} \wedge \ldots \wedge s_{1n_1}{\approx}t_{1n_1} \supset s'_1{\approx}t'_1) \quad \wedge \\ & \qquad\qquad \ldots \qquad\qquad\qquad\qquad\qquad\quad \wedge \\ & (s_{k1}{\approx}t_{k1} \wedge \ldots \wedge s_{kn_k}{\approx}t_{kn_k} \supset s'_k{\approx}t'_k) \quad) \end{array}$$

(We assume that $\wedge$ associates to the right.) Evidently, φ_R can be constructed from R using a polynomial-time algorithm. By considering all possible derivations of $\rightarrow \varphi_R$ in $\mathbf{LJ}^{\approx}$, it is easy to see that φ_R is derivable in $\mathbf{LJ}^{\approx}$ if there is a substitution σ such that the formula ψ of the form

$$\begin{array}{ll} (& (s_{11}{\approx}t_{11} \wedge \ldots \wedge s_{1n_1}{\approx}t_{1n_1} \supset s'_1{\approx}t'_1) \quad \wedge \\ & \qquad\qquad \ldots \qquad\qquad\qquad\qquad\qquad\quad \wedge \\ & (s_{k1}{\approx}t_{k1} \wedge \ldots \wedge s_{kn_k}{\approx}t_{kn_k} \supset s'_k{\approx}t'_k) \quad)\sigma \end{array}$$

is derivable in $\mathbf{LJ}^{\approx}$. Using the fact that the rules $(\rightarrow \wedge)$, $(\wedge \rightarrow)$ and $(\rightarrow\supset)$ are invertible[5] we can prove that ψ is derivable in $\mathbf{LJ}^{\approx}$ if and only if it has a derivation with the skeleton $\mathcal{S}$ of the form

$$\dfrac{\mathcal{S}_1 \qquad \dfrac{\mathcal{S}_2 \qquad \dfrac{\vdots}{\dfrac{\mathcal{S}_{k-1} \quad \mathcal{S}_k}{} \;(\rightarrow \wedge)}}{} \;(\rightarrow \wedge)}{} \;(\rightarrow \wedge)$$

where $\mathcal{S}_j$ are skeletons

[5] A rule is invertible if the derivability of the conclusion of the rule implies the derivability of all premises of the rule.

$$
\begin{array}{l}
\overline{\qquad} \ (\approx) \\
\overline{\qquad} \ (\wedge \to_{n_j-1}) \\
\vdots \\
\overline{\qquad} \ (\wedge \to_0) \\
\overline{\qquad} \ (\to\supset)
\end{array}
$$

It is easy to check that there is a derivation of ψ with the skeleton $\mathcal{S}$ if and only if for all $j \in \{1,\ldots,k\}$ the sequent

$$s_{j1}\sigma{\approx}t_{j1}\sigma,\ldots,s_{jn_j}\sigma{\approx}t_{jn_j}\sigma \to s'_j\sigma{\approx}t'_j\sigma$$

is derivable in $\mathbf{LJ}^{\approx}$. By Lemma 3.2 we obtain that there is a derivation of ψ with the skeleton $\mathcal{S}$ if and only if σ is a solution to R.

This proves that simultaneous rigid E-unifiability is polynomial-time reducible to the derivability in $\mathbf{LJ}^{\approx}$. By considering the formula φ_R and the skeleton

$$
\left.
\begin{array}{l}
\dfrac{\mathcal{S}}{\qquad} \ (\to \exists) \\
\vdots \\
\overline{\qquad} \ (\to \exists)
\end{array}
\right\} m \text{ applications of } (\to \exists)
$$

and using the same arguments as above, we prove that simultaneous rigid E-unifiability is polynomial-time reducible to the skeleton instantiation problem, and similar for the weak skeleton instantiation problem.

This statement and Theorem 2.5 yield

THEOREM 3.14. *The weak skeleton instantiation problem and the skeleton instantiation problem for any signature having a function symbol of arity ≥ 2 are undecidable.*

4. The Constraint Calculus

In this section we define the constraint calculus $\mathbf{LJ}^{\approx}_c$. This calculus can be used for defining sequent style proof procedures for intuitionistic logic with equality. The proof procedures based on $\mathbf{LJ}_c^{\approx}$ 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.

In Section 4.1 we define constraints and their semantics. In Section 4.2 we introduce the constrained calculus $\mathbf{LJ}_c^{\approx}$ and prove its equivalence to $\mathbf{LJ}^{\approx}$.

4.1. Constraints

Definition 4.1. (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 $E \vdash_\forall s \approx t$ such 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 σ we say that σ *satisfies* $\mathcal{C}$:

Definition 4.2. (Constraint satisfiability) The notion "a substitution σ satisfies the constraint $\mathcal{C}$" (denoted $\sigma \models \mathcal{C}$) is defined inductively as follows:

1. $\sigma \models \top$ for every substitution σ;
2. $\sigma \models E \vdash_\forall s \approx t$ if σ is a solution to the rigid equation $E \vdash_\forall s \approx t$;
3. $\sigma \models t \not\preceq \{t_1, \ldots, t_n\}$ if $t\sigma$ is a variable and $t\sigma$ does not occur in $t_1\sigma, \ldots, t_n\sigma$;
4. $\sigma \models \mathcal{C}_1 \wedge \mathcal{C}_2$ if $\sigma \models \mathcal{C}_1$ and $\sigma \models \mathcal{C}_2$;
5. $\sigma \models \exists x \mathcal{C}$ if there is a substitution τ 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* if for every substitution σ we have $\sigma \models \mathcal{C}_1$ if and only if $\sigma \models \mathcal{C}_2$. A constraint $\mathcal{C}$ is *satisfiable* if there is a substitution σ satisfying $\mathcal{C}$.

4.2. The calculus $\mathbf{LJ}_c^{\approx}$

In this section we introduce the calculus $\mathbf{LJ}_c^{\approx}$ of *constrained sequents.* Before giving its inference rules, we introduce the following notation. For a sequence of formulas Γ, denote by $\Gamma_{\approx}$ the set of formulas

$$\{s \approx t \mid s \approx t \in \Gamma\}.$$

Definition 4.3. A *constrained sequent* is an expression $S \cdot \mathcal{C}$ where S is a sequent, $\mathcal{C}$ is a constraint. The *constraint calculus* $\mathbf{LJ}_c^{\approx}$ of constrained sequents is shown in Figure 5.

The notions of a skeleton and a weak skeleton for $\mathbf{LJ}_c^{\approx}$-derivations are similar to those of $\mathbf{LJ}^{\approx}$-derivations.

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

Example 4.4. The following is a derivation in $\mathbf{LJ}_c^{\approx}$ 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 Example 3.10) with a suitable constraint. As in Example 3.10, we denote by φ the formula $\forall x(x{\approx}a \vee x{\approx}b \supset \exists y(f(x,y){\approx}b))$.

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

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 4.5. Another derivation of this formula (with the skeleton of the derivations of Example 3.10) is given in Figure 6. The resulting constraint $\mathcal{C}_{12}$ of this derivation is satisfiable. To establish this, we first note that $\mathcal{C}_{12}$ is satisfiable if so is the following constraint $\mathcal{C}$ obtained from $\mathcal{C}_{12}$ be removing all quantifiers:

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

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

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

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

$$\frac{\Gamma, \varphi, \Delta \to \chi \cdot \mathcal{C}_1 \quad \Gamma, \psi, \Delta \to \chi \cdot \mathcal{C}_2}{\Gamma, \varphi \vee \psi, \Delta \to \chi \cdot \mathcal{C}_1 \wedge \mathcal{C}_2}\ (\vee \to) \qquad \frac{\Gamma \to \varphi \cdot \mathcal{C}}{\Gamma \to \varphi \vee \psi \cdot \mathcal{C}}\ (\to \vee_1) \quad \frac{\Gamma \to \psi \cdot \mathcal{C}}{\Gamma \to \varphi \vee \psi \cdot \mathcal{C}}\ (\to \vee_2)$$

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

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

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

In the rule (Ax), A is a predicate symbol different from equality. 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 $\Gamma, \exists x\varphi, \Delta, \chi$.

Figure 5. Calculus $\mathbf{LJ}_c^{\approx}$

total.tex; 28/08/1998; 14:44; p.20

$$
\dfrac{
\dfrac{\dfrac{\dfrac{}{f(x_1,y_1)\approx b, f(x_2,y_2)\approx b, \varphi \to f(f(u,v),w)\approx b \cdot \mathcal{C}_1}\;(\approx)}{f(x_1,y_1)\approx b, f(x_2,y_2)\approx b, \varphi \to \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_2}\;(\to \exists)}{f(x_1,y_1)\approx b, \exists y(f(x_2,y)\approx b), \varphi \to \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_4}\;(\exists \to)
\qquad
\dfrac{\dfrac{}{\cdots \to x_2\approx b \cdot \mathcal{C}_3}\;(\approx)}{\cdots \to x_2\approx a \vee x_2\approx b \cdot \mathcal{C}_3}\;(\to \vee_2)
}{
\dfrac{f(x_1,y_1)\approx b, x_2\approx a \vee x_2\approx b \supset \exists y(f(x_2,y)\approx b), \varphi \to \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_5}{f(x_1,y_1)\approx b, \varphi \to \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_6}\;(\forall \to)
}\;(\supset\to)
$$

$$
\dfrac{
\dfrac{\dfrac{f(x_1,y_1)\approx b, \varphi \to \exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_6}{f(x_1,y_1)\approx b, \varphi \to \exists v\exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_7}\;(\to \exists)}{\exists y(f(x_1,y)\approx b), \varphi \to \exists v\exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_9}\;(\exists \to)
\qquad
\dfrac{\dfrac{}{\cdots \to x_1\approx a \cdot \mathcal{C}_8}\;(\approx)}{\cdots \to x_1\approx a \vee x_1\approx b \cdot \mathcal{C}_8}\;(\to \vee_1)
}{
\dfrac{\dfrac{\dfrac{x_1\approx a \vee x_1\approx b \supset \exists y(f(x_1,y)\approx b), \varphi \to \exists v\exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_{10}}{\varphi \to \exists v\exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_{11}}\;(\forall \to)}{\varphi \to \exists u\exists v\exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_{12}}\;(\to \exists)}{\to \varphi \supset \exists u\exists v\exists w(f(f(u,v),w)\approx b) \cdot \mathcal{C}_{12}}\;(\to\supset)
}\;(\supset\to)
$$

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

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

Figure 6. Derivation in $\mathbf{LJ}^{\approx}_{c}$

This quantifier-free constraint is satisfied by the substitution

$$\{u \mapsto a, x_1 \mapsto a, v \mapsto y_1, x_2 \mapsto b, w \mapsto y_2\}.$$

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

THEOREM 4.6. *Let $\Gamma \to \varphi$ be a sequent, θ be a substitution and $\mathcal{S}$ be a skeleton (respectively, weak skeleton). Then the following conditions are equivalent:*

1. *There is a derivation of $\Gamma\theta \to \varphi\theta$ in $\mathbf{LJ}^{\approx}$ with the skeleton (respectively, weak skeleton) $\mathcal{S}$;*

2. *There is a constraint $\mathcal{C}$ and a derivation of $\Gamma \to \varphi \cdot \mathcal{C}$ in $\mathbf{LJ}^{\approx}_c$ with the skeleton (respectively, weak skeleton) $\mathcal{S}$ such that $\theta \models \mathcal{C}$.*

Proof. We only prove the claim for skeletons, then the result for weak skeletons will be a straightforward consequence. By Theorem 3.11, we can restrict ourselves to regular $\mathbf{LJ}^{\approx}$-derivations. The proof is carried out by induction on the regular derivations in $\mathbf{LJ}^{\approx}$ (respectively, derivations in $\mathbf{LJ}^{\approx}_c$). Consider several cases depending on the last inference rule applied. In all cases the condition on the skeletons to be the same is straightforward.

1. The $\mathbf{LJ}^{\approx}$-derivation of a sequent $\Gamma\theta \to s\theta{\approx}t\theta$ consists of an application of the rule $(\approx)$ followed by any number of $(\approx_1)$ or $(\approx_2)$. By Lemma 3.2, θ is a solution to the rigid equation $\Gamma_{\approx} \vdash_{\forall} s{\approx}t$. The following is the required $\mathbf{LJ}^{\approx}_c$-derivation:

$$\frac{}{\Gamma \to s{\approx}t \cdot \Gamma_{\approx} \vdash_{\forall} s{\approx}t}\ (\approx)$$

2. The $\mathbf{LJ}^{\approx}_c$-derivation consists of one application of $(\approx)$:

$$\frac{}{\Gamma \to s{\approx}t \cdot \Gamma_{\approx} \vdash_{\forall} s{\approx}t}\ (\approx)$$

Suppose $\theta \models \Gamma_{\approx} \vdash_{\forall} s{\approx}t$. It follows from Lemma 3.2 that there is an $\mathbf{LJ}^{\approx}$-derivation of the sequent $\Gamma\theta \to s\theta{\approx}t\theta$ consisting only of applications of the rules $(\approx)$, $(\approx_1)$ and $(\approx_2)$.

3. The last rule applied in the $\mathbf{LJ}^{\approx}$-derivation is $(\rightarrow \wedge)$. Then the $\mathbf{LJ}^{\approx}$-derivation has the form

$$\frac{\begin{array}{cc}\Pi_1 & \Pi_2 \\ \Gamma\theta \rightarrow \varphi\theta & \Gamma\theta \rightarrow \psi\theta\end{array}}{\Gamma\theta \rightarrow \varphi\theta \wedge \psi\theta}\ (\rightarrow \wedge)$$

By the induction hypothesis, there are $\mathbf{LJ}_c^{\approx}$-derivations Π_1' and Π_2' of constrained sequents $\Gamma \rightarrow \varphi \cdot \mathcal{C}_1$ and $\Gamma \rightarrow \psi \cdot \mathcal{C}_2$, respectively, such that $\theta \models \mathcal{C}_1$ and $\theta \models \mathcal{C}_2$. By the definition of the constraint satisfaction, $\theta \models \mathcal{C}_1 \wedge \mathcal{C}_2$. Define $\mathcal{C}$ as $\mathcal{C}_1 \wedge \mathcal{C}_2$. Then the required $\mathbf{LJ}_c^{\approx}$-derivation is

$$\frac{\begin{array}{cc}\Pi_1' & \Pi_2' \\ \Gamma \rightarrow \varphi \cdot \mathcal{C}_1 & \Gamma \rightarrow \psi \cdot \mathcal{C}_2\end{array}}{\Gamma \rightarrow \varphi \wedge \psi \cdot \mathcal{C}_1 \wedge \mathcal{C}_2}\ (\rightarrow \wedge)$$

4. The last rule applied in the $\mathbf{LJ}_c^{\approx}$-derivation is $(\rightarrow \wedge)$. Then the $\mathbf{LJ}_c^{\approx}$-derivation has the form

$$\frac{\begin{array}{cc}\Pi_1 & \Pi_2 \\ \Gamma \rightarrow \varphi \cdot \mathcal{C}_1 & \Gamma \rightarrow \psi \cdot \mathcal{C}_2\end{array}}{\Gamma \rightarrow \varphi \wedge \psi \cdot \mathcal{C}_1 \wedge \mathcal{C}_2}\ (\rightarrow \wedge)$$

Let $\theta \models \mathcal{C}_1 \wedge \mathcal{C}_2$. Then $\theta \models \mathcal{C}_1$ and $\theta \models \mathcal{C}_2$. By the induction hypothesis, there are $\mathbf{LJ}^{\approx}$-derivations Π_1' and Π_2' of the sequents $\Gamma\theta \rightarrow \varphi\theta$ and $\Gamma\theta \rightarrow \psi\theta$, respectively. Then the required $\mathbf{LJ}^{\approx}$-derivation is

$$\frac{\begin{array}{cc}\Pi_1' & \Pi_2' \\ \Gamma\theta \rightarrow \varphi\theta & \Gamma\theta \rightarrow \psi\theta\end{array}}{\Gamma\theta \rightarrow \varphi\theta \wedge \psi\theta}\ (\rightarrow \wedge)$$

5. The last rule applied in the $\mathbf{LJ}^{\approx}$-derivation is $(\rightarrow \forall)$. Let y be the eigenvariable of this rule and $\Sigma \rightarrow \psi$ be the premise of the rule. Then it is easy to see that for any variable z not occurring in Σ or ψ, the sequent $\Sigma\{z \mapsto y\} \rightarrow \psi\{z \mapsto y\}$ is also derivable in $\mathbf{LJ}^{\approx}$ with the same skeleton. Hence, without loss of generality we can assume that the eigenvariable does not occur in Γ or φ. Then the $\mathbf{LJ}^{\approx}$-derivation has the form

$$\frac{\begin{array}{c}\Pi\\ \Gamma\theta' \to \varphi\{x \mapsto y\}\theta'\end{array}}{\Gamma\theta \to (\forall x\varphi)\theta}\ (\to \forall)$$

where θ' is a substitution such that $v\theta = v\theta'$ for all variables v different from y and $y\theta' = y$. By the induction hypothesis, there is an $\mathbf{LJ}_c^{\approx}$-derivation Π' of a constrained sequent $\Gamma \to \varphi\{x \mapsto y\} \cdot \mathcal{C}$ such that $\theta' \models \mathcal{C}$. Let $v_1, \ldots, v_m$ be all free variables of $\Gamma, \forall x\varphi$. Since y does not occur in Γ, φ and $y\theta' = y$, then $\theta' \models y \not\preceq \{v_1, \ldots, v_m\} \wedge \mathcal{C}$. It follows that $\theta \models \exists y(y \not\preceq \{v_1, \ldots, v_m\} \wedge \mathcal{C})$. The required $\mathbf{LJ}_c^{\approx}$-derivation is

$$\frac{\begin{array}{c}\Pi'\\ \Gamma \to \varphi\{x \mapsto y\} \cdot \mathcal{C}\end{array}}{\Gamma \to \forall x\varphi \cdot \exists y(y \not\preceq \{v_1, \ldots, v_m\} \wedge \mathcal{C})}\ (\to \forall)$$

6. The last rule applied in the $\mathbf{LJ}_c^{\approx}$-derivation is $(\to \forall)$. Then the $\mathbf{LJ}_c^{\approx}$-derivation has the form

$$\frac{\begin{array}{c}\Pi\\ \Gamma \to \varphi\{x \mapsto y\} \cdot \mathcal{C}\end{array}}{\Gamma \to \forall x\varphi \cdot \exists y(y \not\preceq \{v_1, \ldots, v_n\} \wedge \mathcal{C})}\ (\to \forall)$$

Let $\theta \models \exists y(y \not\preceq \{v_1, \ldots, v_n\} \wedge \mathcal{C})$. By the definition of the constraint satisfaction, there is a substitution θ' such that $v\theta' = v\theta$ for all variables different from y and $\theta' \models y \not\preceq \{v_1, \ldots, v_m\} \wedge \mathcal{C}$. It means that $\theta' \models \mathcal{C}$ and $y\theta'$ is a variable not occurring in $v_1\theta', \ldots, v_m\theta'$. By the induction hypothesis, there is derivation Π' in $\mathbf{LJ}^{\approx}$ of the sequent $\Gamma\theta' \to \varphi\{x \mapsto y\}\theta'$. Then the required $\mathbf{LJ}^{\approx}$-derivation is

$$\frac{\begin{array}{c}\Pi'\\ \Gamma\theta' \to \varphi\{x \mapsto y\}\theta'\end{array}}{\Gamma\theta \to (\forall x\varphi)\theta}$$

The other cases are similar.

The following two theorems are straightforward consequences of Theorem 4.6:

THEOREM 4.7. *Let S be a closed sequent and $\mathcal{S}$ be a skeleton (respectively, a weak skeleton). Then S has a derivation in* $\mathbf{LJ}^{\approx}$ *with the skeleton (respectively, the weak skeleton) $\mathcal{S}$ if and only if there is a satisfiable constraint $\mathcal{C}$ such that $S \cdot \mathcal{C}$ has a derivation in* $\mathbf{LJ}_c^{\approx}$ *with the skeleton (respectively, the weak skeleton) $\mathcal{S}$.*

THEOREM 4.8. *Let S be a closed sequent. Then S is derivable in* $\mathbf{LJ}^{\approx}$ *if and only if there is a satisfiable constraint $\mathcal{C}$ such that $S \cdot \mathcal{C}$ is derivable in* $\mathbf{LJ}_c^{\approx}$*.*

Note that the set of constraints $\mathcal{C}$ such that $S \cdot \mathcal{C}$ is derivable in $\mathbf{LJ}_c^{\approx}$ with a skeleton $\mathcal{S}$ is uniquely defined by the skeleton $\mathcal{S}$ 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 3.14, 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 derivability 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 (Degtyarev, Matiyasevich and Voronkov [2], Gurevich and Voronkov [16] and Veanes [40]). 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. Since both problems are undecidable, it shows close relationships between the two problems. In fact, we give structure-preserving translations between the two problems.

We start with several technical definitions and statements. Let Σ be any signature containing at least one constant. *We assume that all terms of this section are terms in* Σ. For any signature Δ, we denote by Δ^* an extension of Δ by a countable set of new constants $c_1, c_2 \ldots$.

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.

We call a constraint $\mathcal{C}$ *refined* if for any two different occurrences of quantifiers $\exists x$ and $\exists y$ in $\mathcal{C}$ we have $x \neq y$. Evidently, any constraint can be transformed to an equivalent refined one by renaming bound variables. *For the rest of this section we assume that all constraints we consider are refined.*

We say that a constraint $\mathcal{C}'$ is obtained from a constraint $\mathcal{C}$ be *quantifier removal* if $\mathcal{C}'$ is a quantifier-free constraint obtained from $\mathcal{C}$ by removal of all quantifiers $\exists x$. For example, if $\mathcal{C}$ is $\exists z(z \approx a \wedge \exists y(y \approx f(u)))$, then after quantifier removal we obtain $z \approx a \wedge y \approx f(u)$.

LEMMA 5.1. *Let $\mathcal{C}$ be a refined constraint and $\mathcal{C}'$ be obtained from $\mathcal{C}$ by quantifier removal. Then $\mathcal{C}$ and $\mathcal{C}'$ are equi-satisfiable.*

Proof. Straightforward.

We implicitly used this lemma when we established the constraint satisfiability in Example 4.5.

Definition 5.2. (Special constraint) A constraint $\mathcal{C}$ is called *special* if any occurrence of $\ldots \not\preceq \{\ldots\}$ in $\mathcal{C}$ is an occurrence in a subformula of $\mathcal{C}$ of the form $\exists x(x \not\preceq \{v_1, \ldots, v_n\} \wedge \mathcal{C}')$, where $v_1, \ldots, v_n$ are variables and all free variables of $\mathcal{C}'$ are among $x, v_1, \ldots, v_n$.

Note that for any constraint sequent $S \cdot \mathcal{C}$ derivable in $\mathbf{LJ}_c^{\approx}$, the constraint $\mathcal{C}$ is special.

We shall introduce some technical definitions and prove several technical lemmas.

Definition 5.3. (Equivalent substitutions) Let X be a set of variables and θ_1, θ_2 be substitutions. Then θ_1 and θ_2 are called *equivalent w.r.t. X*, denoted $\theta_1 \equiv_X \theta_2$, if there are two substitutions τ_1, τ_2 such that for every variable $x \in X$ we have $x\theta_1 = x\theta_2\tau_2$ and $x\theta_1\tau_1 = x\theta_2$.

LEMMA 5.4. *Let $\theta_1 \equiv_X \theta_2$ and $\mathcal{C}$ be a constraint all whose free variables are among X. Then $\theta_1 \models \mathcal{C}$ if and only if $\theta_2 \models \mathcal{C}$.*

Proof. By straightforward induction on the structure of $\mathcal{C}$.

LEMMA 5.5. *Let $\mathcal{C}$ be a constraint all whose free variables are among $x, v_1, \ldots, v_n$ and θ be any substitution such that $\theta \models x \not\preceq \{v_1, \ldots, v_n\} \wedge$*

$\mathcal{C}$. Let y be any variable not occurring in $x\theta, v_1\theta, \ldots, v_n\theta$ and the substitution θ' be defined as follows:

$$\theta'(z) \rightleftharpoons \begin{cases} \theta(z), & \text{if } z \neq x \\ y & \text{otherwise} \end{cases}$$

Then $\theta' \models x \not\preceq \{v_1, \ldots, v_n\} \wedge \mathcal{C}$.

Proof. Since $x\theta$ is a variable not occurring in $v_1\theta, \ldots, v_n\theta$, we have $\theta \equiv_{\{x, v_1, \ldots, v_n\}} \theta'$. Then apply Lemma 5.4.

LEMMA 5.6. *Let $\mathcal{C}$ be a satisfiable special refined constraint. Let $\mathcal{C}'$ be the quantifier-free constraint obtained from $\mathcal{C}$ by quantifier removal. Let $\{x_1, \ldots, x_m\}$ be the set of all variables x such that $\mathcal{C}'$ contains a constraint of the form $x \not\preceq \{t_1, \ldots, t_n\}$. Then $\mathcal{C}'$ has a solution θ with the following property:*

1. *For any variable y occurring in $\mathcal{C}'$ we have $y\theta \in T_\Sigma(\{x_1, \ldots, x_m\})$;*

2. *For every $i \in \{1, \ldots, m\}$ we have $x_i\theta = x_i$.*

Proof. Let σ be any substitution satisfying $\mathcal{C}'$. Let Y be the set of terms $\{x_1\sigma, \ldots, x_m\sigma\}$. Since $\sigma \models \mathcal{C}'$, all terms in Y are variables. By Lemma 5.5 we can assume that the variables $x_1\sigma, \ldots, x_m\sigma$ are pairwise disjoint. Let $Y = \{y_1, \ldots, y_m\}$. Take any constant $c \in \Sigma$ and define the substitution τ as follows: for any variable x,

$$x\tau \rightleftharpoons \begin{cases} x_i, & \text{if } x = y_i \\ c, & \text{if } x \in var(\mathcal{C}') \cup var(ran(\sigma)) \setminus Y \\ x, & \text{otherwise} \end{cases}$$

Finally, define $\theta \rightleftharpoons \sigma\tau$. It is easy to check that θ satisfies the claim.

Let Δ be any finite set of function symbols with at least one constant, c be any constant in Δ and t be a term not necessarily in the signature Δ. Define the rigid equation

$$\mathcal{R}_{t,\Delta} \rightleftharpoons \{f(c, \ldots, c) \approx c \mid f \in \Delta\} \vdash_\forall t \approx c$$

LEMMA 5.7. *For every substitution θ, θ is a solution to $\mathcal{R}_{t,\Delta}$ if and only if $t\theta \in T_\Delta$.*

Proof. Obvious.

THEOREM 5.8. *Simultaneous rigid E-unifiability in the signature Σ^* is polynomial-time equivalent to the satisfiability problem for special constraints in the signature Σ^*.*

Proof. As we noted, simultaneous rigid E-unifiability is a special case of constraint satisfiability. It remains to show a polynomial-time reduction of constraint satisfiability to simultaneous rigid E-unifiability.

Let $\mathcal{C}$ be a special constraint. Without loss of generality we can assume that $\mathcal{C}$ is refined. By Lemma 5.1 we can assume that $\mathcal{C}$ is quantifier-free. Let $\{x_1, \ldots, x_m\}$ be the set of all variables x such that $\mathcal{C}$ contains a constraint of the form $x \not\preceq \{t_1, \ldots, t_n\}$. Let $c_1, \ldots, c_m$ be constants occurring in Σ^* but not in Σ. For any $i \in \{1, \ldots, n\}$ denote by Σ_{-i} the signature $\Sigma \cup \{c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_n\}$.

Let $R_{\mathcal{C}}$ be the system of rigid equations consisting of

1. All rigid equations occurring in $\mathcal{C}$;
2. For every $i \in \{1, \ldots, m\}$, the rigid equation $\vdash_\forall x_i \approx c_i$;
3. For every constraint $x_i \not\preceq \{v_1, \ldots, v_n\}$ occurring in $\mathcal{C}$, the set of rigid equations $\mathcal{R}_{v_j, \Sigma_{-i}}$, where $j \in \{1, \ldots, n\}$.

Obviously, $R_{\mathcal{C}}$ can be constructed by $\mathcal{C}$ in polynomial time. We prove that $\mathcal{C}$ is satisfiable if and only if $R_{\mathcal{C}}$ is solvable.

Define the substitution $\tau \rightleftharpoons \{x_1 \mapsto c_1, \ldots, x_m \mapsto c_m\}$. We prove the following claim:

(I) For any substitution θ the following conditions are equivalent:

   a) $\theta \models \mathcal{C}$ and θ satisfies the conditions of Lemma 5.6;
   b) $\theta\tau$ is a solution to $R_{\mathcal{C}}$.

   We only prove the ($\Rightarrow$) direction. The reverse direction is similar. Suppose that $\theta \models \mathcal{C}$ and θ satisfies the conditions of Lemma 5.6.

   a) Since $\theta \models \mathcal{C}$, evidently $\theta\tau$ satisfies all rigid equations in $\mathcal{C}$.
   b) Consider any $i \in \{1, \ldots, m\}$. Since θ satisfies the conditions of Lemma 5.6, we have $x_i\theta = x_i$. Hence, $x_i\theta\tau = c_i$. Then $\theta\tau \models \vdash_\forall x_i \approx c_i$.
   c) Let $x_i \not\preceq \{v_1, \ldots, v_n\}$ be a constraint occurring in $\mathcal{C}$. Since θ satisfies the conditions of Lemma 5.6, we have $x_i\theta = x_i$. Hence, x_i does not occur in $v_1\theta, \ldots, v_n\theta$. Since all $v_j\theta$ are terms in $T_\Sigma(\{x_1, \ldots, x_m\})$, we have that c_i does not occur in the terms $v_1\theta\tau, \ldots, v_n\theta\tau$. Hence, for any $j \in \{1, \ldots, n\}$ we have $v_i\theta\tau \in T_{\Sigma_{-i}}$. By Lemma 5.7, $\theta\tau \models \mathcal{R}_{v_j, \Sigma_{-i}}$.

Claim (I) is proven.

Suppose now that $\mathcal{C}$ is satisfiable. By Lemma 5.6, there is substitution θ satisfying conditions of this lemma and such that $\theta \models \mathcal{C}$. By claim (I), $R_{\mathcal{C}}$ is solvable.

Let $R_{\mathcal{C}}$ be solvable and σ be any solution to $R_{\mathcal{C}}$. Take any substitution θ such that $\theta\tau = \sigma$. By claim (I) we have $\theta \models \mathcal{C}$. We have shown that $\mathcal{C}$ is satisfiable if and only if $R_{\mathcal{C}}$ is solvable. Thus, the satisfiability problem for special constraints is polynomial-time reducible to simultaneous rigid E-unifiability.

The aim of this theorem was to show the close connection between constraint satisfaction and simultaneous rigid E-unification. Of course in practice, instead of reduction of constraint satisfaction to simultaneous rigid E-unification, one can develop special algorithms for constraint satisfaction hoping that they may be more "efficient" or more intuitive compared to the one using the above reduction to simultaneous rigid E-unification.

Combining Theorems 4.6 and 5.8, we obtain

THEOREM 5.9. *The skeleton instantiation problem in the signature Σ^* is polynomial-time equivalent to simultaneous rigid E-unifiability in the signature Σ^*.*

This theorem is also true for classical logic, if we change the system $\mathbf{LJ}^{\approx}$ 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 5.9 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 Σ in which formulas are written contains binary function symbols (see proofs of the undecidability in Degtyarev and Voronkov [9, 8], Plaisted [31] and Veanes [39, 40]). In Degtyarev, Matiyasevich and Voronkov [2] it is proven that simultaneous rigid E-unifiability in the signatures having at most one unary function symbol and no function symbols of arity ≥ 2 is decidable which implies the following result:

THEOREM 5.10. *Let Σ be a signature consisting of any number of constants and at most one unary function symbol. The following problems restricted to formulas of the signature Σ are decidable:*

1. *Simultaneous rigid E-unifiability;*

2. *The weak skeleton instantiation problem;*

3. *The skeleton instantiation problem.*

Proof. For simultaneous rigid E-unifiability in Σ^* it was proven in Degtyarev, Matiyasevich and Voronkov [2]. Theorem 5.9 implies that the skeleton instantiation problem in this signature is decidable. Since any instance of the weak skeleton instantiation problem can be effectively reduced to a disjunction of instances of the skeleton instantiation problem, the weak skeleton instantiation problem for this signature is decidable.

6. Modal logics with equality

There is no uniform viewpoint on the use of quantification in modal logics. Neither is there a uniform viewpoint on the use of equality. The survey Garson [13] contains some material on this topic. Without going into many details, we show that the above considerations about intuitionistic logic can as well be applied to some formalizations of various modal logics. We consider modal logics in which equality is axiomatized by the rules $(\approx)$, $(\approx_1)$ and $(\approx_2)$, like in $\mathbf{LJ}^{\approx}$. As an example we take $\mathbf{S4}^{\approx}$, which is the extension of the first-order version of $\mathbf{S4}$ by these three equality rules. This section is slightly informal and may be skipped by readers who are not familiar with modal logics.

Formulas in $\mathbf{S4}^{\approx}$ are constructed in the same way as formulas in $\mathbf{LJ}^{\approx}$ but can also use unary operators $\Box$ and $\Diamond$. A *sequent* in $\mathbf{S4}^{\approx}$ is any expression $\Gamma \to \Delta$, where Γ and Δ are sequences of formulas.

For any sequence of formulas Γ denote by $\Gamma_\Box$ the sequence of formulas obtained from Γ by removing all formulas not having the form $\Box\varphi$ and deleting $\Box$ in front of all remaining formulas. For example, if Γ is $\varphi_1 \wedge \varphi_2, \Box\psi_1, \Diamond\chi, \Box\psi_2$, then $\Gamma_\Box$ is ψ_1, ψ_2. Similarly, we define $\Gamma_\Diamond$.

Definition 6.1. (Calculus $\mathbf{S4}^{\approx}$)
The calculus $\mathbf{S4}^{\approx}$ for modal logic with equality derives sequents. The rules of the calculus are shown in Figure 6. Derivations in $\mathbf{S4}^{\approx}$ are trees defined as usual.

$$\frac{}{\Gamma, A, \Delta \to \Sigma, A, \Xi}\ (Ax) \qquad \frac{}{\Gamma, \Lambda, \Delta \to \Sigma}\ (\Lambda)$$

$$\frac{}{\Gamma \to \Sigma, t{\approx}t, \Xi}\ (\approx)$$

$$\frac{\Gamma\{x \mapsto t\}, s{\approx}t, \Delta\{x \mapsto t\} \to \Sigma\{x \mapsto t\}}{\Gamma\{x \mapsto s\}, s{\approx}t, \Delta\{x \mapsto s\} \to \Sigma\{x \mapsto s\}}\ (\approx_1)$$

$$\frac{\Gamma\{x \mapsto s\}, s{\approx}t, \Delta\{x \mapsto s\} \to \Sigma\{x \mapsto s\}}{\Gamma\{x \mapsto t\}, s{\approx}t, \Delta\{x \mapsto t\} \to \Sigma\{x \mapsto t\}}\ (\approx_2)$$

$$\frac{\Gamma, \varphi, \psi, \Delta \to \Sigma}{\Gamma, \varphi \wedge \psi, \Delta \to \Sigma}\ (\wedge \to) \qquad \frac{\Gamma \to \Sigma, \varphi, \Xi \quad \Gamma \to \Sigma, \psi, \Xi}{\Gamma \to \Sigma, \varphi \wedge \psi, \Xi}\ (\to \wedge)$$

$$\frac{\Gamma, \varphi, \Delta \to \Sigma \quad \Gamma, \psi, \Delta \to \Sigma}{\Gamma, \varphi \vee \psi, \Delta \to \Sigma}\ (\vee \to) \qquad \frac{\Gamma \to \Sigma, \varphi, \psi, \Xi}{\Gamma \to \Sigma, \varphi \vee \psi, \Xi}\ (\to \vee)$$

$$\frac{\Gamma, \psi, \Delta \to \Sigma \quad \Gamma, \Delta \to \Sigma, \varphi}{\Gamma, \varphi \supset \psi, \Delta \to \Sigma}\ (\supset\to) \qquad \frac{\varphi, \Gamma \to \Sigma, \psi, \Xi}{\Gamma \to \Sigma, \varphi \supset \psi, \Xi}\ (\to\supset)$$

$$\frac{\Gamma, \varphi\{x \mapsto t\}, \forall x\varphi, \Delta \to \Sigma}{\Gamma, \forall x\varphi, \Delta \to \Sigma}\ (\forall \to) \qquad \frac{\Gamma \to \Sigma, \varphi\{x \mapsto y\}, \Xi}{\Gamma \to \Sigma, \forall x\varphi, \Xi}\ (\to \forall)$$

$$\frac{\Gamma, \varphi\{x \mapsto y\}, \Delta \to \Sigma}{\Gamma, \exists x\varphi, \Delta \to \Sigma}\ (\exists \to) \qquad \frac{\Gamma \to \Sigma, \varphi\{x \mapsto t\}, \exists x\varphi, \Xi}{\Gamma \to \Sigma, \exists x\varphi, \Xi}\ (\to \exists)$$

$$\frac{\Gamma, \varphi, \Delta \to \Sigma}{\Gamma, \Box\varphi, \Delta \to \Sigma}\ (\Box \to) \qquad \frac{\Gamma_{\Box} \to \Sigma_{\Diamond}, \varphi, \Delta_{\Diamond}}{\Gamma \to \Sigma, \Box\varphi, \Delta}\ (\to \Box)$$

$$\frac{\Gamma_{\Box}, \varphi, \Delta_{\Box} \to \Sigma}{\Gamma, \Diamond\varphi, \Delta \to \Sigma_{\Diamond}}\ (\Diamond \to) \qquad \frac{\Gamma \to \Sigma, \varphi, \Xi}{\Gamma \to \Sigma, \Diamond\varphi, \Xi}\ (\to \Diamond)$$

The rules of this calculus are subject to the same conditions as those of $\mathbf{LJ}^{\approx}$.

Figure 7. Calculus $\mathbf{S4}^{\approx}$

Consider again any system R of rigid equations of the form (1) and the formula ψ_R of the form

$$\begin{array}{cc}(s_{11}{\approx}t_{11} \wedge \ldots \wedge s_{1n_1}{\approx}t_{1n_1} \supset s'_1{\approx}t'_1) & \wedge \\ \ldots & \wedge \\ (s_{k1}{\approx}t_{k1} \wedge \ldots \wedge s_{kn_k}{\approx}t_{kn_k} \supset s'_k{\approx}t'_k) & \end{array}$$

Let $\varphi \rightleftharpoons \exists x_1 \ldots \exists x_m \Box \psi_R$. Any derivation of $\to \varphi$ in $\mathbf{S4}^{\approx}$ has the following form, for a suitable substitution σ:

$$\left.\begin{array}{c}\left.\begin{array}{c}\vdots \\ \to \psi_R\sigma\end{array}\right\} \text{derivation of } \to \psi_R\sigma \\ \hline \to \Sigma, \Box\psi_R\sigma, \Xi \quad (\to \Box) \\ \vdots \\ \to \exists x_1 \ldots \exists x_m \Box\psi_R\end{array}\right\} \text{derivation using only rules } \to \exists$$

Again, such a derivation exists if and only if σ is a solution to R. Thus, nearly every result asserted above for $\mathbf{LJ}^{\approx}$, can be reformulated in a suitable way for $\mathbf{S4}^{\approx}$.

We can note that this phenomenon: the necessity of handling simultaneous rigid E-unification will arise in almost any logic whose semantics is based on possible worlds. The same trick that we used for modal logics above can, for instance, be made for some temporal logics.

7. Conclusion and related work

There are not so many papers on intuitionistic logic with equality. Decision problems for intuitionistic logic with equality are studied in [20, 27, 29]. Lifschitz [20] proves the undecidability of some systems of intuitionistic equality. Mints [27] introduces a system of Skolem functions for formulas of intuitionistic logic restricting the application of quantifier rules. Orevkov [29] 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. Degtyarev and Voronkov [6] prove some decidability and complexity results for prenex formulas of intuitionistic logic with and without equality.

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

Voda and Komara [41] consider so-called *Herbrand skeletons* (formulas obtained after a fixed number of quantifier duplications in a prenex skolemized formula) and claim that the corresponding skeleton instantiation problem is undecidable. Their results are improved by Gurevich and Veanes [15].

Tammet [38] describes a theorem prover for intuitionistic logic based on the inverse method. Currently, 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 [35] 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 article 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, except for ones based on a straightforward enumeration described by Petermann [30].

Our results also show a fundamental difference between handling equality in proof procedures for classical and non-classical logics. For classical logic, a number of methods not using simultaneous rigid E-unification have been developed. For methods based on sequent calculi, the equality elimination method has been developed by Degtyarev and Voronkov [7, 4, 3]. Recently, Degtyarev and Voronkov [10] proved that an incomplete but terminating procedure for rigid E-unification gives a complete procedure for classical logic with equality. It follows from the results of this article that none of these methods is applicable to intuitionistic logic.

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 [37] or Lincoln and Shankar [22], with the exception that in presence of equality we cannot use (even dynamic) skolemization.

Acknowledgements

I thank Anatoli Degtyarev for many helpful comments.

References

1. Buss, S.: 1986, *Bounded Arithmetic*, Vol. 3 of *Studies in Proof Theory*. Napoly: Bibliopolis.
2. Degtyarev, A., Y. Matiyasevich, and A. Voronkov: 1996, 'Simultaneous Rigid E-Unification and Related Algorithmic Problems'. In: *Eleventh Annual IEEE Symposium on Logic in Computer Science (LICS'96)*. New Brunswick, NJ, pp. 494–502.
3. Degtyarev, A. and A. Voronkov: 1995a, 'Equality Elimination for the Inverse Method and Extension Procedures'. In: C. Mellish (ed.): *Proc. International Joint Conference on Artificial Intelligence (IJCAI)*, Vol. 1. Montréal, pp. 342–347.
4. Degtyarev, A. and A. Voronkov: 1995b, 'General Connections via Equality Elimination'. In: M. De Glas and Z. Pawlak (eds.): *Second World Conference on the Fundamentals of Artificial Intelligence (WOCFAI-95)*. Paris, pp. 109–120.
5. Degtyarev, A. and A. Voronkov: 1995c, 'A New Procedural Interpretation of Horn Clauses with Equality'. In: L. Sterling (ed.): *Proceedings of the Twelfth International Conference on Logic Programming*. pp. 565–579.
6. Degtyarev, A. and A. Voronkov: 1996a, 'Decidability Problems for the Prenex Fragment of Intuitionistic Logic'. In: *Eleventh Annual IEEE Symposium on Logic in Computer Science (LICS'96)*. New Brunswick, NJ, pp. 503–512.
7. Degtyarev, A. and A. Voronkov: 1996b, 'Equality Elimination for the Tableau Method'. In: J. Calmet and C. Limongelli (eds.): *Design and Implementation of Symbolic Computation Systems. International Symposium, DISCO'96*, Vol. 1128 of *Lecture Notes in Computer Science*. Karlsruhe, Germany, pp. 46–60.
8. Degtyarev, A. and A. Voronkov: 1996c, 'Simultaneous Rigid E-Unification is Undecidable'. In: H. Kleine Büning (ed.): *Computer Science Logic. 9th International Workshop, CSL'95*, Vol. 1092 of *Lecture Notes in Computer Science*. Paderborn, Germany, September 1995, pp. 178–190.
9. Degtyarev, A. and A. Voronkov: 1996d, 'The Undecidability of Simultaneous Rigid E-Unification'. *Theoretical Computer Science* **166**(1–2), 291–300.
10. Degtyarev, A. and A. Voronkov: 1996e, 'What You Always Wanted to Know About Rigid E-Unification'. In: J. Alferes, L. Pereira, and E. Orlowska (eds.): *Logics in Artificial Intelligence. European Workshop, JELIA'96*, Vol. 1126 of *Lecture Notes in Artificial Intelligence*. Évora, Portugal, pp. 50–69.
11. Fitting, M.: 1990, *First Order Logic and Automated Theorem Proving*. New York: Springer Verlag.
12. Gallier, J., S. Raatz, and W. Snyder: 1987, 'Theorem Proving Using Rigid E-Unification: Equational Matings'. In: *Proc. IEEE Conference on Logic in Computer Science (LICS)*. pp. 338–346.
13. Garson, J.: 1984, 'Quantification in Modal Logic'. In: D. Gabbay and F. Guenther (eds.): *Handbook in Philosophical Logic*, Vol. II. D. Reidel Publishing Company, Chapt. II.5, pp. 249–307.
14. Girard, J.-Y.: 1987, *Proof Theory and Logical Complexity*, Studies in Proof Theory. Napoly: Bibliopolis.
15. Gurevich, Y. and M. Veanes: 1997, 'Some Undecidable Problems Related to the Herbrand Theorem'. UPMAIL Technical Report 138, Uppsala University, Computing Science Department.
16. Gurevich, Y. and A. Voronkov: 1997, 'Monadic Simultaneous Rigid E-Unification and Related Problems'. In: P. Degano, R. Corrieri, and A. Marchetti-Spaccamella (eds.): *Automata, Languages and Programming. 24th*

International Colloquium, ICALP'97, Vol. 1256 of *Lecture Notes in Computer Science*. Bologna, Italy, pp. 154–165.

17. Kanger, S.: 1957, *Provability in Logic*, Vol. 1 of *Studies in Philosophy*. Stockholm: Almqvist and Wicksell.
18. Kanger, S.: 1983, 'A Simplified Proof Method for Elementary Logic'. In: J. Siekmann and G. Wrightson (eds.): *Automation of Reasoning. Classical Papers on Computational Logic*, Vol. 1. Springer Verlag, pp. 364–371. Originally appeared in 1963.
19. Lee, R. and C. Chang: 1973, *Symbolic Logic and Mechanical Theorem Proving*. Academic Press.
20. Lifschitz, V.: 1967, 'Problem of Decidability for Some Constructive Theories of Equalities (in Russian)'. *Zapiski Nauchnyh Seminarov LOMI* **4**, 78–85. English Translation in: Seminars in Mathematics: Steklov Math. Inst. 4, Consultants Bureau, NY-London, 1969, p.29–31.
21. Lifschitz, V.: 1968, 'Specialized Forms of Derivation in Predicate Calculus with Equality and Functional Symbols (in Russian)'. In: *Trudy MIAN*, Vol. 98. pp. 5–25. English translation in: Proc. Steklov Institute of Math., AMS, Providence, RI, 1971.
22. Lincoln, P. and N. Shankar: 1994, 'Proof Search in First-Order Linear Logic and Other Cut-Free Sequent Calculi'. In: S. Abramski (ed.): *Ninth Annual Symposium on Logic in Computer Science*. Paris, France, pp. 282–291.
23. Loveland, D.: 1978, *Automated Theorem Proving: a Logical Basis*. North Holland.
24. Maslov, S.: 1967, 'Invertible Sequential Variant of Constructive Predicate Calculus (in Russian)'. *Zapiski Nauchnyh Seminarov LOMI* **4**, 96–111. English Translation in: Seminars in Mathematics: Steklov Math. Inst. 4, Consultants Bureau, NY-London, 1969, p.36–42.
25. Maslov, S.: 1983, 'An Inverse Method for Establishing Deducibility of Nonprenex Formulas of the Predicate Calculus'. In: J.Siekmann and G.Wrightson (eds.): *Automation of Reasoning (Classical papers on Computational Logic)*, Vol. 2. Springer Verlag, pp. 48–54.
26. Matulis, V.: 1963, 'On variants of classical predicate calculus with the unique deduction tree (in Russian)'. *Soviet Mathematical Doklady* **148**, 768–770.
27. Mints, G.: 1967, 'Choice of Terms in Quantifier Rules of Constructive Predicate Calculus (in Russian)'. *Zapiski Nauchnyh Seminarov LOMI* **4**, 78–85. English Translation in: Seminars in Mathematics: Steklov Math. Inst. 4, Consultants Bureau, NY-London, 1969, p.43–46.
28. Orevkov, V.: 1974, 'A Specialized form of derivation in Gentzen calculi and its applications (in Russian)'. *Zapiski Nauchnyh Seminarov LOMI* **32**, 110–118.
29. Orevkov, V.: 1976, 'Solvable Classes of Pseudo-Prenex Formulas (in Russian)'. *Zapiski Nauchnyh Seminarov LOMI* **60**, 109–170. English translation in: Journal of Soviet Mathematics.
30. Petermann, U.: 1994, 'A Complete Connection Calculus with Rigid E-Unification'. In: *JELIA'94*, Vol. 838 of *Lecture Notes in Computer Science*. pp. 152–166.
31. Plaisted, D.: 1995, 'Special Cases and Substitutes for Rigid E-Unification'. Technical Report MPI-I-95-2-010, Max-Planck-Institut für Informatik.
32. Plotkin, G.: 1972, 'Building-in Equational Theories'. In: B. Meltzer and D. Michie (eds.): *Machine Intelligence*, Vol. 7. Edinburgh University Press, pp. 73–90.

33. Robinson, G. and L. Wos: 1969, 'Paramodulation and Theorem-Proving in First Order Theories with Equality'. In: B. Meltzer and D. Michie (eds.): *Machine Intelligence*, Vol. 4. Edinburgh University Press, pp. 135–150.
34. Robinson, J.: 1965, 'A Machine-Oriented Logic Based on the Resolution Principle'. *Journal of the Association for Computing Machinery* **12**(1), 23–41.
35. Sahlin, D., T. Franzén, and S. Haridi: 1992, 'An Intuitionistic Predicate Logic Theorem Prover'. *Journal of Logic and Computation* **2**(5), 619–656.
36. Sazonov, V.: 1980, 'A Logical Approach to the Problem "P=NP?"'. In: *Mathematical Foundations of Computer Science*, Vol. 88 of *Lecture Notes in Computer Science*. pp. 562–575.
37. Shankar, N.: 1992, 'Proof Search in the Intuitionistic Sequent Calculus'. In: D. Kapur (ed.): *11th International Conference on Automated Deduction*, Vol. 607 of *Lecture Notes in Artificial Intelligence*. Saratoga Springs, NY, USA, pp. 522–536.
38. Tammet, T.: 1996, 'A Resolution Theorem Prover for Intuitionistic Logic'. In: M. McRobbie and J. Slaney (eds.): *Automated Deduction — CADE-13*, Vol. 1104 of *Lecture Notes in Computer Science*. New Brunswick, NJ, USA, pp. 2–16.
39. Veanes, M.: 1996, 'Uniform Representation of Recursively Enumerable Sets with Simultaneous Rigid E-Unification'. UPMAIL Technical Report 126, Uppsala University, Computing Science Department.
40. Veanes, M.: 1997, 'On Simultaneous Rigid E-Unification'. Ph.D. thesis, Uppsala University.
41. Voda, P. and J. Komara: 1995, 'On Herbrand Skeletons'. Technical report, Institute of Informatics, Comenius University Bratislava.
42. Voronkov, A.: 1992, 'Theorem Proving in Non-standard Logics Based on the Inverse Method'. In: D. Kapur (ed.): *11th International Conference on Automated Deduction*, Vol. 607 of *Lecture Notes in Artificial Intelligence*. Saratoga Springs, NY, USA, pp. 648–662.
43. Voronkov, A.: 1996a, 'Proof Search in Intuitionistic Logic Based on Constraint Satisfaction'. In: P. Miglioli, U. Moscato, D. Mundici, and M. Ornaghi (eds.): *Theorem Proving with Analytic Tableaux and Related Methods. 5th International Workshop, TABLEAUX '96*, Vol. 1071 of *Lecture Notes in Artificial Intelligence*. Terrasini, Palermo Italy, pp. 312–329.
44. Voronkov, A.: 1996b, 'Proof Search in Intuitionistic Logic with Equality, or Back to Simultaneous Rigid E-Unification'. In: M. McRobbie and J. Slaney (eds.): *Automated Deduction — CADE-13*, Vol. 1104 of *Lecture Notes in Computer Science*. New Brunswick, NJ, USA, pp. 32–46.