Proof-Search in Intuitionistic Logic Based on Constraint Satisfaction

Andrei Voronkov*

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

Abstract. We characterize provability in intuitionistic predicate logic in terms of derivation skeletons and constraints and study the problem of instantiations of a skeleton to valid derivations. We prove that for two different notions of a skeleton the problem is respectively polynomial and NP-complete. As an application of our technique, we demonstrate PSPACE-completeness of the prenex fragment of intuitionistic logic. We outline some applications of the proposed technique in automated reasoning.

1 Introduction

The characterization of provability for classical logic in terms of matrices was given by Kanger [9, 10] and Prawitz [19, 20] and is in a fact a reformulation of the fundamental Herbrand theorem. It is the basis of many theorem-proving methods, including the tableau method and the connection (or mating) method.

This characterization says that a formula $\exists\bar{x}\varphi(\bar{x})$, where $\varphi(\bar{x})$ is a quantifier-free formula, is provable in classical logic iff there is a *matrix* M of the form $\varphi(\bar{x}_1) \vee \ldots \vee \varphi(\bar{x}_n)$ constructed by this formula and a substitution σ such that every path in the matrix $M\sigma$ is inconsistent, i.e. contains a formula and its negation. For such a matrix M, the information on the number of quantifier duplications (i.e. the number n) is enough to find a proof using a non-deterministic polynomial algorithm (when n is fixed, the problem of the existence of such a substitution σ is NP-complete). There are many ways of constructing such a matrix and checking its paths. The way used in the tableau method can be considered as a construction of the skeleton of a derivation in the sequent calculus, and finding a substitution σ that makes this skeleton a derivation.

The situation with intuitionistic logic is more complicated. Even the propositional fragment of intuitionistic logic is PSPACE-complete due to non-invertible rules. Further, there is no prenex normal form of formulas so that the number of quantifier duplications does not give enough information about the structure of the derivation. In addition, there is no Skolem normal form and one has to deal with essentially universal quantifiers.

Our paper has two main aims. The first aim is to develop a characterization of provability in intuitionistic predicate logic. For the connection method, such

* Supported by a TFR grant

a characterization was developed by Wallen [29]. Our characterization is based on derivation skeletons and constraint satisfaction. We use constraints to design a machine-oriented version $\mathbf{LJ}_c$ of the intuitionistic sequent calculus. Our characterization of intuitionistic provability is more simple than that of Wallen [29] and can be used for constructing a family of sequent-based proof procedures, for example a tableau-based procedure, an inverse method-based procedure and their modifications on the basis of $\mathbf{LJ}_c$. The second aim of this paper is to establish complexity results for the problem of instantiation of a given derivation skeleton to a derivation (the skeleton instantiation problem). We also outline some potential applications of our results and techniques.

This paper has the following structure. Section 2 defines the sequent calculus **LJ** and skeletons of derivations in **LJ**. The proof-search in **LJ** can be considered as the search for a derivation skeleton and the instantiation of this skeleton to a valid derivation. In Section 3 we define the constraint calculus $\mathbf{LJ}_c$ and prove a theorem characterizing the relation between derivations in **LJ** and $\mathbf{LJ}_c$. In Section 4 we prove the existence of a polynomial time algorithm for constraint satisfiability. As a consequence, we characterize the complexity of the skeleton instantiation problem. Using the technique of derivation skeletons and constraints developed in previous sections, in Section 5 we prove PSPACE-completeness of the prenex fragment of intuitionistic logic. In Section 6 we briefly consider related work and some other potential applications of our technique.

2 Preliminaries. The calculus LJ

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 $\Gamma \rightarrow \varphi$ where Γ is a finite multiset of formulas and φ is a formula. For a multiset of formulas Γ and a formula φ we denote by Γ, φ the multiset obtained from Γ by adding φ. A formula is *closed* iff it has no free variables. A sequent $\Gamma \rightarrow \varphi$ is *closed* iff all formulas in Γ, φ are closed.

The notation $\rightleftharpoons$ stands for "equal by definition". $s \approx t$ denotes that terms s and t are identical. $\not\approx$ is the negation of $\approx$.

By X we denote a countably 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. The signature may contain function symbols with arity more than 0. Variables are denoted by u, v, w, x, y, z, atomic formulas by A, B, formulas by $\varphi, \chi, \psi, \rho$ and multisets of formulas by Γ, 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 $x \in X$ such that $\theta(x) \not\approx x$. The *domain* $dom(\theta)$ of the substitution θ is the set $\{x \mid x\theta \not\approx x\}$. 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 $\{x_1\theta/x_1, \ldots, x_n\theta/x_n\}$. Let E be a term, for-

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

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

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

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

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

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

In the rules $(\exists \to)$ and $(\to \forall)$ 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. The calculus **LJ**

mula, multiset of formulas or sequent. For a substitution $\theta = \{t_1/x_1, \ldots, t_n/x_n\}$, $E\theta$ 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.

2.1 The calculus LJ

The calculus **LJ** for intuitionistic logic is shown in Figure 1. All rules labelled with $(C \to)$, where C is a connective or a quantifier are called *antecedent rules,* all rules labelled with $(\to C)$ are called *succedent rules.*

Derivations in **LJ** are trees defined as usual.

2.2 Derivation skeletons

The following definition is one of the main notions studied in this paper:

Definition 1 (Weak skeleton). A *weak derivation skeleton* is a tree such that

1. Its nodes are labelled by the names of inference rules in **LJ**;
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 **LJ**.

In weak derivation skeletons, we display labels to the right of the bar.

Example 1. The following picture displays a weak derivation skeleton:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overline{}\ (\Lambda) \quad \overline{}\ (Ax)}{}\ (\supset\rightarrow)}{}\ (\wedge\rightarrow)}{}\ (\wedge\rightarrow)}{}\ (\wedge\rightarrow)}{}\ (\rightarrow\supset)$$

Weak derivation skeletons are abstractions of derivations that encode information about the structures of derivations. They abstract from the instantiation of variables in quantifier rules. Before formally defining the connection between derivations and skeletons, we introduce a stronger notion of skeletons:

Definition 2 (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.

The natural number in an antecedent rule uniquely identifies the main formula of the rule application. To formalize it, we introduce a *choice function* that selects formulas in a multiset. A *choice* function CF is a function whose arguments are a finite multiset of formulas Γ and a natural number such that, whenever Γ has k elements, the multiset $CF(\Gamma, 0), \ldots, CF(\Gamma, k-1)$ coincides with Γ. In other words, the choice function orders elements of a multiset of formulas. We assume that a choice function CF is fixed. For a finite multiset of formulas Γ with k elements and $\varphi \in \Gamma$, an *index of* φ *in* Γ is any number $n \leq k-1$ such that $CF(\Gamma, n) = \varphi$.

Definition 3 (Skeleton of a derivation). Let Π be a derivation in **LJ**. *The weak skeleton* Π' *of* Π and *the skeleton* Π'' *of* Π are defined inductively as follows:

1. If Π has the form $\dfrac{}{\Gamma, A \rightarrow A}\ (Ax)$ then Π' and Π'' are $\overline{}\ (Ax)$ and $\overline{}\ (Ax_n)$ respectively, where n is the index of A in Γ, A.
2. If Π has the form $\dfrac{}{\Gamma, \Lambda \rightarrow \varphi}\ (\Lambda)$ then both Π' and Π'' are $\overline{}\ (\Lambda)$
3. If Π has the form $\dfrac{\begin{matrix}\Pi_1 \\ \varphi, \Gamma \rightarrow \psi\end{matrix}}{\Gamma \rightarrow \varphi \supset \psi}\ (\rightarrow\supset)$ and Π_1' and Π_1'' are, respectively, the weak skeleton and the skeleton of Π_1, then Π' and Π'' are $\dfrac{\Pi_1'}{}\ (\rightarrow\supset)$ and $\dfrac{\Pi_1''}{}\ (\rightarrow\supset)$ respectively, and similar for all other succedent rules.
4. If Π has the form $\dfrac{\begin{matrix}\Pi_1 \\ \Gamma, \psi \rightarrow \chi\end{matrix} \quad \begin{matrix}\Pi_2 \\ \Gamma, \varphi \supset \psi \rightarrow \varphi\end{matrix}}{\Gamma, \varphi \supset \psi \rightarrow \chi}\ (\supset\rightarrow)$ 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 $\dfrac{\Pi_1' \quad \Pi_2'}{}\ (\supset\rightarrow)$ and $\dfrac{\Pi_1'' \quad \Pi_2''}{}\ (\supset\rightarrow_n)$ respectively, where n is the index of $\varphi \supset \psi$ in $\Gamma, \varphi \supset \psi$, and similar for all other antecedent rules.

When we display derivations, we assume that formulas in antecedents of sequents are displayed in the order of their indices, i.e. when we write $\varphi_1, \ldots, \varphi_n \to \varphi$, then the index of φ_i in $\varphi_1, \ldots, \varphi_n$ is i.

There may be many derivations of the same formula (even of a propositional one) with the same weak skeleton due to the choice of different subformulas:

Example 2. Consider the following two different derivations of the formula $((P \supset \Lambda) \wedge (Q \supset \Lambda) \wedge P \wedge Q) \supset \Lambda$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Lambda, (Q \supset \Lambda), P, Q \to \Lambda}\,(\Lambda) \quad \dfrac{}{(P \supset \Lambda), (Q \supset \Lambda), P, Q \to P}\,(Ax)}{(P \supset \Lambda), (Q \supset \Lambda), P, Q \to \Lambda}\,(\supset\to)}{(P \supset \Lambda), (Q \supset \Lambda), P \wedge Q \to \Lambda}\,(\wedge\to)}{(P \supset \Lambda), (Q \supset \Lambda) \wedge P \wedge Q \to \Lambda}\,(\wedge\to)}{(P \supset \Lambda) \wedge (Q \supset \Lambda) \wedge P \wedge Q \to \Lambda}\,(\wedge\to)}{\to ((P \supset \Lambda) \wedge (Q \supset \Lambda) \wedge P \wedge Q) \supset \Lambda}\,(\to\supset)$$

and

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{(P \supset \Lambda), \Lambda, P, Q \to \Lambda}\,(\Lambda) \quad \dfrac{}{(P \supset \Lambda), (Q \supset \Lambda), P, Q \to Q}\,(Ax)}{(P \supset \Lambda), (Q \supset \Lambda), P, Q \to \Lambda}\,(\supset\to)}{(P \supset \Lambda), (Q \supset \Lambda), P \wedge Q \to \Lambda}\,(\wedge\to)}{(P \supset \Lambda), (Q \supset \Lambda) \wedge P \wedge Q \to \Lambda}\,(\wedge\to)}{(P \supset \Lambda) \wedge (Q \supset \Lambda) \wedge P \wedge Q \to \Lambda}\,(\wedge\to)}{\to ((P \supset \Lambda) \wedge (Q \supset \Lambda) \wedge P \wedge Q) \supset \Lambda}\,(\to\supset)$$

They have the same weak skeleton shown in Example 1 but different skeletons:

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

Skeletons uniquely identify derivations of the same sequent without quantifier rules. However, there may be many derivations of the same sequent with the same skeleton due to different instantiations of terms in the quantifier rules. For example, the following two different derivations of the formula $\forall x A(x) \supset \exists x A(x)$ have the same skeleton:

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

There are several obvious procedures for constructing derivation skeletons. They can take into account invertibility and permutability of rules in **LJ** and use special strategies for non-invertible rules. We do not consider this heavily investigated subject. We shall consider the problem of instantiations of a (weak) skeleton to a derivation. We will show that this problem is decidable and characterize its complexity in Section 4. We conclude this section by formally defining the corresponding algorithmic problems:

Definition 4. 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.

3 The Constraint Calculus

In this section we define the constraint calculus $\mathbf{LJ}_c$. It can be used for defining sequent style proof procedures for automatic theorem proving in intuitionistic logic. The proof procedures based on $\mathbf{LJ}_c$ consist of two parts: the construction of a derivation in $\mathbf{LJ}_c$ and constraint satisfaction. The constraint to be checked for satisfiability is uniquely identified by the skeleton of a derivation.

In Section 3.1 we define constraints and their semantics. In Section 3.2 we introduce the constraint calculus $\mathbf{LJ}_c$ and prove its equivalence to **LJ**. Then we show that the skeleton instantiation problem can be reduced to the constraint satisfiability problem.

3.1 Constraints

Definition 5 (Constraint). *Constraints* are defined inductively as follows:

1. $\top$ and $\bot$ are constraints;
2. For any terms s and t, $s = t$ is a constraint;
3. For any terms $s, t_1, \ldots, t_n$, where $n \geq 0$, $s \not\preceq \{t_1, \ldots, t_n\}$ is a constraint;
4. If $\mathcal{C}_1$ and $\mathcal{C}_2$ are constraints, then $\mathcal{C}_1 \wedge \mathcal{C}_2$ is a constraint;
5. If $\mathcal{C}$ is a constraint and x is variable, then $\exists x \mathcal{C}$ is a constraint.

The constraints $\top$, $\bot$, $s = t$ and $s \not\preceq \{t_1, \ldots, t_n\}$ are called *atomic.*

Constraints can be considered as first order formulas in a signature whose predicate symbols are $\top$, $\bot$, $=$ and the symbols $\ldots \not\preceq \{\ldots\}$ of all arities ≥ 1. Constraints of the form $s = t$ are called *equality constraints.* We shall not distinguish the constraint $s = t$ from $t = s$.

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

$$\frac{}{\Gamma, P(s_1,\ldots,s_n) \to P(t_1,\ldots,t_n) \cdot s_1 = t_1 \wedge \ldots \wedge s_n = t_n}\ (Ax) \qquad \frac{}{\Gamma, \Lambda \to \varphi \cdot \top}\ (\Lambda)$$

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

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

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

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

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

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

For any of the rules $(\forall \to)$–$(\to \exists)$, the variable y has no free occurrences in the sequent of the conclusion of the rule. 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 $\Gamma, \exists x \varphi, \chi$.

Fig. 2. Calculus $\mathbf{LJ}_c$

Definition 6 (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. There is no substitution σ such that $\sigma \models \bot$;
3. $\sigma \models s = t$ iff $s\sigma \approx t\sigma$;
4. $\sigma \models s \not\preceq \{t_1,\ldots,t_n\}$ iff $s\sigma$ is a variable and $s\sigma$ does not occur in $t_1\sigma,\ldots,t_n\sigma$;
5. $\sigma \models \mathcal{C}_1 \wedge \mathcal{C}_2$ iff $\sigma \models \mathcal{C}_1$ and $\sigma \models \mathcal{C}_2$;
6. $\sigma \models \exists x \mathcal{C}$ iff there is a substitution τ such that $\tau \models \mathcal{C}$ and $y\tau \approx y\sigma$ for every variable y different from x.

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

3.2 Calculus $\mathbf{LJ}_c$

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

Definition 7 (Constrained sequent). A *constrained sequent* is an expression $S \cdot \mathcal{C}$ where S is a sequent and $\mathcal{C}$ is a constraint. The *constraint calculus* $\mathbf{LJ}_c$ of constrained sequents is shown in Figure 2.

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

Example 3. The example of a derivation in $\mathbf{LJ}_c$ of the formula $\exists x \forall y P(x,y) \supset \forall y \exists x P(x,y)$ with a suitable constraint:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{}{P(v,w), \forall y P(v,y) \to P(u,z) \cdot v = u \wedge w = z}\ (Ax)}{\forall y P(v,y) \to P(u,z) \cdot \exists w(v = u \wedge w = z)}\ (\forall \to)}{\forall y P(v,y) \to \exists x P(x,z) \cdot \exists u \exists w(v = u \wedge w = z)}\ (\to \exists)}{\exists x \forall y P(x,y) \to \exists x P(x,z) \cdot \exists v(v \not\preceq \{z\} \wedge \exists u \exists w(v = u \wedge w = z))}\ (\exists \to)}{\cfrac{\exists x \forall y P(x,y) \to \forall y \exists x P(x,y) \cdot \exists z(z \not\preceq \{\} \wedge \exists v(v \not\preceq \{z\} \wedge \exists u \exists w(v = u \wedge w = z)))}{\begin{array}{l}\to \exists x \forall y P(x,y) \supset \forall y \exists x P(x,y) \\ \cdot \exists z(z \not\preceq \{\} \wedge \exists v(v \not\preceq \{z\} \wedge \exists u \exists w(v = u \wedge w = z)))\end{array}}\ (\to\supset)}\ (\to \forall)$$

The resulting constraint $\exists z(z \not\preceq \{\} \wedge \exists v(v \not\preceq \{z\} \wedge \exists u \exists w(v = u \wedge w = z)))$ is satisfiable.

Example 4. The example of a derivation in $\mathbf{LJ}_c$ of the formula $\forall x \exists y P(x,y) \supset \exists y \forall x P(x,y)$ with a suitable constraint (this formula is not provable in $\mathbf{LJ}$):

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{}{P(v,w), \forall x \exists y P(x,y) \to P(u,z) \cdot v = u \wedge w = z}\ (Ax)}{\exists y P(v,y), \forall x \exists y P(x,y) \to P(u,z) \cdot \exists w(w \not\preceq \{v,u,z\} \wedge v = u \wedge w = z)}\ (\exists \to)}{\begin{array}{l}\exists y P(v,y), \forall x \exists y P(x,y) \to \forall x P(x,z) \\ \cdot \exists u(u \not\preceq \{v,z\} \wedge \exists w(w \not\preceq \{v,u,z\} \wedge v = u \wedge w = z))\end{array}}\ (\to \forall)}{\begin{array}{l}\forall x \exists y P(x,y) \to \forall x P(x,z) \\ \cdot \exists v \exists u(u \not\preceq \{v,z\} \wedge \exists w(w \not\preceq \{v,u,z\} \wedge v = u \wedge w = z))\end{array}}\ (\forall \to)}{\begin{array}{l}\forall x \exists y P(x,y) \to \exists y \forall x P(x,y) \\ \cdot \exists z \exists v \exists u(u \not\preceq \{v,z\} \wedge \exists w(w \not\preceq \{v,u,z\} \wedge v = u \wedge w = z))\end{array}}\ (\to \exists)}{\begin{array}{l}\to \forall x \exists y P(x,y) \supset \exists y \forall x P(x,y) \\ \cdot \exists z \exists v \exists u(u \not\preceq \{v,z\} \wedge \exists w(w \not\preceq \{v,u,z\} \wedge v = u \wedge w = z))\end{array}}\ (\to\supset)$$

The resulting constraint $\exists z \exists v \exists u(u \not\preceq \{v,z\} \wedge \exists w(w \not\preceq \{v,u,z\} \wedge v = u \wedge w = z))$ is unsatisfiable.

The notions of a skeleton and a weak skeleton for $\mathbf{LJ}_c$-derivations are defined in the same way as for $\mathbf{LJ}$-derivations. The following statement connects the derivability in $\mathbf{LJ}$ with the derivability in $\mathbf{LJ}_c$:

Theorem 8. *Let $\Gamma \to \varphi$ be a sequent, θ 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 $\theta \models \mathcal{C}$.*

Proof. By induction on the derivation in $\mathbf{LJ}$ (respectively, $\mathbf{LJ}_c$). 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 last rule applied in the **LJ**-derivation is (Ax):

$$\frac{}{\Gamma\theta \to P(t_1,\dots,t_n)\theta}\ (Ax)$$

Then $P(t_1,\dots,t_n)\theta \in \Gamma\theta$. It means that there is a formula $P(s_1,\dots,s_n) \in \Gamma$ such that $s_i\theta \approx t_i\theta$, for all $i \in \{1,\dots,n\}$. Define $\mathcal{C}$ as $s_1 = t_1 \wedge \dots \wedge s_n = t_n$. The following is the required $\mathbf{LJ}_c$-derivation:

$$\frac{}{\Gamma \to P(s_1,\dots,s_n)\cdot\mathcal{C}}\ (Ax)$$

The property $\theta \models \mathcal{C}$ is obvious.

2. The last rule applied in the $\mathbf{LJ}_c$-derivation is (Ax):

$$\frac{}{\Gamma, P(s_1,\dots,s_n) \to P(t_1,\dots,t_n)\cdot s_1 = t_1 \wedge \dots \wedge s_n = t_n}\ (Ax)$$

Suppose $\theta \models s_1 = t_1 \wedge \dots \wedge s_n = t_n$. Then $s_i\theta \approx t_i\theta$ for all $i \in \{1,\dots,m\}$. The required **LJ**-derivation is

$$\frac{}{\Gamma\theta, P(s_1,\dots,s_n)\theta \to P(t_1,\dots,t_n)\theta}\ (Ax)$$

3. The last rule applied in the **LJ**-derivation is $(\to\wedge)$. Then the **LJ**-derivation has the form

$$\frac{\begin{matrix}\Pi_1 \\ \Gamma\theta \to \varphi\theta\end{matrix} \quad \begin{matrix}\Pi_2 \\ \Gamma\theta \to \psi\theta\end{matrix}}{\Gamma\theta \to \varphi\theta \wedge \psi\theta}\ (\to\wedge)$$

By the induction hypothesis, there are $\mathbf{LJ}_c$-derivations Π_1' and Π_2' of the constrained sequents $\Gamma \to \varphi\cdot\mathcal{C}_1$ and $\Gamma \to \psi\cdot\mathcal{C}_2$, respectively, such that $\theta \models \mathcal{C}_1$ and $\theta \models \mathcal{C}_2$. By the definition of 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 $\mathbf{LJ}_c$-derivation is

$$\frac{\begin{matrix}\Pi_1' \\ \Gamma \to \varphi\cdot\mathcal{C}_1\end{matrix} \quad \begin{matrix}\Pi_2' \\ \Gamma \to \psi\cdot\mathcal{C}_2\end{matrix}}{\Gamma \to \varphi \wedge \psi\cdot\mathcal{C}_1 \wedge \mathcal{C}_2}\ (\to\wedge)$$

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

$$\frac{\begin{matrix}\Pi_1 \\ \Gamma \to \varphi\cdot\mathcal{C}_1\end{matrix} \quad \begin{matrix}\Pi_2 \\ \Gamma \to \psi\cdot\mathcal{C}_2\end{matrix}}{\Gamma \to \varphi \wedge \psi\cdot\mathcal{C}_1 \wedge \mathcal{C}_2}\ (\to\wedge)$$

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 **LJ**-derivations Π_1' and Π_2' of the sequents $\Gamma\theta \to \varphi\theta$ and $\Gamma\theta \to \psi\theta$, respectively. Then the required **LJ**-derivation is

$$\frac{\begin{matrix}\Pi_1' \\ \Gamma\theta \to \varphi\theta\end{matrix} \quad \begin{matrix}\Pi_2' \\ \Gamma\theta \to \psi\theta\end{matrix}}{\Gamma\theta \to \varphi\theta \wedge \psi\theta}\ (\to\wedge)$$

5. The last rule applied in the **LJ**-derivation is $(\to \forall)$. Let y be the eigenvariable of this rule and $\Sigma \to \psi$ 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 $\Sigma\{y/z\} \to \psi\{y/z\}$ is also derivable in **LJ** with the same skeleton. Hence, without loss of generality we can assume that the eigenvariable does not occur in Γ or φ. Then the **LJ**-derivation has the form

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

where θ' is a substitution such that $v\theta \approx v\theta'$ for all variables v different from y and $y\theta' \approx y$. By the induction hypothesis, there is an $\mathbf{LJ}_c$-derivation Π' of a constrained sequent $\Gamma \to \varphi\{y/x\} \cdot \mathcal{C}$ 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 $y\theta' \approx y$, 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 $\mathbf{LJ}_c$-derivation is

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

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

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

Let $\theta \models \exists y(y \not\preceq \{v_1, \ldots, v_n\} \wedge \mathcal{C})$. By the definition of constraint satisfaction, there is a substitution θ' such that $v\theta' \approx v\theta$ 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 **LJ** of the sequent $\Gamma\theta' \to \varphi\{y/x\}\theta'$. Then the required **LJ**-derivation is

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

The other cases are similar.

The following two theorems are straightforward consequences of Theorem 8:

Theorem 9. *Let S be a closed sequent and $\mathcal{S}$ be a skeleton. Then S has a derivation in* **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 $\mathcal{S}$.*

The same holds with "skeleton" replaced by "weak skeleton".

Theorem 10. *Let S be a closed sequent. Then S is derivable in* **LJ** *iff there is a satisfiable constraint $\mathcal{C}$ such that $S \cdot \mathcal{C}$ is derivable in $\mathbf{LJ}_c$.*

Note that the constraint $\mathcal{C}$ such that $S \cdot \mathcal{C}$ is derivable in $\mathbf{LJ}_c$ with a skeleton $\mathcal{S}$ is uniquely defined by S and $\mathcal{S}$ in the following sense. Suppose that the constrained sequents $S \cdot \mathcal{C}_1$ and $S \cdot \mathcal{C}_2$ have $\mathbf{LJ}_c$-derivations with the same skeleton. Then $\mathcal{C}_1$ is obtained from $\mathcal{C}_2$ by renaming bound variables.

Thus, the proof-search in intuitionistic logic can be considered as consisting of two parts: constructing a skeleton and constraint satisfaction. The two parts should not necessarily be processed in this order, for example the constraint satisfaction algorithm can check constraints after each step of the skeleton construction.

4 Constraint satisfaction

In this section we show that there is a polynomial time algorithm for the constraint satisfiability problem. In fact, a polynomial algorithm for constraint satisfiability can be defined as a modification of the unification algorithm of Martelli and Montanari [14]. Since the development is quite standard, we omit the proof of the following

Theorem 11. *There is a polynomial time algorithm checking constraint satisfiability. There is a polynomial time algorithm for the skeleton instantiation problem.*

The complete proof of this theorem may be found in [28]. Using the property that the skeleton instantiation problem only uses constraints of a special form we can even prove the existence of a linear time algorithm for constraint satisfiability.

Theorem 12. *The weak skeleton instantiation problem is NP-complete.*

Proof. Given a closed sequent S and a weak skeleton $\mathcal{S}$, one can guess the corresponding skeleton of a derivation in $\mathbf{LJ}$ using a non-deterministic polynomial time algorithm. Applying Theorem 11 we obtain that the weak skeleton instantiation problem is in NP.

To prove NP-hardness, we shall use a polynomial-time reduction of the subsumption problem which is known to be NP-complete [7]. The subsumption problem can be formulated as follows: given terms $t_1, \ldots, t_n$ and ground terms $s_1, \ldots, s_m$, is there a substitution σ such that $\{t_1\sigma, \ldots, t_n\sigma\} \subseteq \{s_1, \ldots, s_m\}$. (If such a substitution σ exists we say that $\{t_1, \ldots, t_n\}$ *subsumes* $\{s_1, \ldots, s_m\}$.)

Let $t_1, \ldots, t_n$ be terms and $s_1, \ldots, s_m$ be ground terms. Let $v_1, \ldots, v_k$ be all variables of $t_1, \ldots, t_n$ and P be a unary predicate symbol. Consider the closed sequent $P(s_1), \ldots, P(s_m) \to \exists v_1 \ldots \exists v_k (P(t_1) \wedge \ldots \wedge P(t_n))$ (we could as well consider the formula $P(s_1) \wedge \ldots \wedge P(s_m) \supset \exists v_1 \ldots \exists v_k (P(t_1) \wedge \ldots \wedge P(t_n))$). Consider the weak skeleton

$$\left.\begin{array}{c}
\dfrac{\overline{}\;(Ax) \qquad \overline{}\;(Ax)}{}\;(\rightarrow \wedge) \\
\vdots \\
\dfrac{\overline{}\;(Ax) \qquad \overline{}\;(\rightarrow \wedge)}{}\;(\rightarrow \wedge)
\end{array}\right\} n-1 \text{ applications of } (\rightarrow \wedge)$$

$$\left.\begin{array}{c}
\overline{}\;(\rightarrow \exists) \\
\vdots \\
\overline{}\;(\rightarrow \exists)
\end{array}\right\} k \text{ applications of } (\rightarrow \exists)$$

Denote $P(s_1), \ldots, P(s_m)$ by Γ. Any derivation with this weak skeleton has the form

$$\begin{array}{c}
\dfrac{\dfrac{}{\Gamma \rightarrow P(t_{n-1})\sigma}\;(Ax) \qquad \dfrac{}{\Gamma \rightarrow P(t_n)\sigma}\;(Ax)}{\Gamma \rightarrow P(t_{n-1})\sigma \wedge P(t_n)\sigma}\;(\rightarrow \wedge) \\
\vdots \\
\dfrac{\dfrac{\dfrac{}{\Gamma \rightarrow P(t_1)\sigma}\;(Ax) \qquad \Gamma \rightarrow P(t_2)\sigma \wedge \ldots \wedge P(t_n)\sigma}{\Gamma \rightarrow P(t_1)\sigma \wedge \ldots \wedge P(t_n)\sigma}\;(\rightarrow \wedge)}{\Gamma \rightarrow \exists v_k (P(t_1) \wedge \ldots \wedge P(t_n))\sigma}\;(\rightarrow \exists) \\
\vdots \\
\dfrac{\Gamma \rightarrow \exists v_2 \ldots \exists v_k (P(t_1) \wedge \ldots \wedge P(t_n))\sigma}{\Gamma \rightarrow \exists v_1 \ldots \exists v_k (P(t_1) \wedge \ldots \wedge P(t_n))}\;(\rightarrow \exists)
\end{array}$$

where σ is a substitution. Note that the top sequents of such derivations are axioms of **LJ** iff $\{t_1\sigma, \ldots, t_n\sigma\} \subseteq \{s_1, \ldots, s_m\}$. Thus, a derivation with this weak skeleton exists iff $\{t_1, \ldots, t_n\}$ subsumes $\{s_1, \ldots, s_m\}$.

5 PSPACE-completeness of the prenex fragment of intuitionistic logic

As an application of our technique, we prove PSPACE-completeness of the prenex fragment of intuitionistic logic.

The calculus $\mathbf{LJ}_c$ is not the most efficient computationally. In order to make it a useful procedure for theorem proving in intuitionistic logic, we have to modify $\mathbf{LJ}_c$. For example, if we know that above some application R of an inference rule there are no applications of quantifier rules, we can modify the rule $(\supset\rightarrow)$ so that the derivation above R in $\mathbf{LJ}_c$ will be loop-free. This technique yields a decision procedure for intuitionistic propositional logic (see Vorob'ev [24, 25], Lincoln, Scedrov, and Shankar [12], Dyckhoff [5], Hudelmaier [8]). We show how

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

$$\frac{\Gamma \to \varphi \quad \Gamma \to \psi}{\Gamma \to \varphi \wedge \psi}\ (GI1\wedge) \qquad \frac{\Gamma, \varphi \supset \chi \to \varphi \quad \Gamma, \psi \supset \chi \to \psi}{\Gamma, \varphi \wedge \psi \supset \chi \to \varphi \wedge \psi}\ (GI2\wedge)$$

$$\frac{\Gamma \to \varphi}{\Gamma \to \varphi \vee \psi}\ (GI1\vee) \qquad \frac{\Gamma \to \psi}{\Gamma \to \varphi \vee \psi}\ (GI1\vee)$$

$$\frac{\Gamma, \varphi \supset P, \psi \supset P, P \supset \chi \to \varphi}{\Gamma, (\varphi \vee \psi) \supset \chi \to \varphi \vee \psi}\ (GI2\vee) \qquad \frac{\Gamma, \varphi \supset P, \psi \supset P, P \supset \chi \to \psi}{\Gamma, (\varphi \vee \psi) \supset \chi \to \varphi \vee \psi}\ (GI2\vee)$$

$$\frac{\Gamma, \varphi \to \psi}{\Gamma \to \varphi \supset \psi}\ (GI1 \supset) \qquad \frac{\Gamma, \varphi, \psi \supset \chi \to \psi}{\Gamma, (\varphi \supset \psi) \supset \chi \to \varphi \supset \psi}\ (GI2 \supset)$$

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

$$\frac{\Gamma, A, \psi \to \chi}{\Gamma, A, A \supset \psi \to \chi}\ (GE \supset P) \qquad \frac{\Gamma, \varphi \supset (\psi \supset \chi) \to \rho}{\Gamma, (\varphi \wedge \psi) \supset \chi \to \rho}\ (GE \supset \wedge)$$

$$\frac{\Gamma, \varphi \supset P, \psi \supset P, P \supset \chi \to \rho}{\Gamma, (\varphi \vee \psi) \supset \chi \to \rho}\ (GE \supset \vee) \qquad \frac{\Gamma, \varphi, \psi \supset \chi \to \psi \quad \Gamma, \chi \to \rho}{\Gamma, (\varphi \supset \psi) \supset \chi \to \rho}\ (GE \supset\supset)$$

The rules satisfy the following restrictions:

1. Conclusions of $(GI1)$-rules must not be of the form $\Gamma, \varphi' \supset \psi' \to \varphi'$;
2. In every rule where P occurs, P is a propositional variable not occurring in the conclusion of the rule;
3. In the rule $(GE \supset \wedge)$ we have $\rho \neq \varphi \wedge \psi$, in the rule $(GE \supset \vee)$ we have $\rho \neq \varphi \vee \psi$, in the rule $(GE \supset\supset)$ we have $\rho \neq \varphi \supset \psi$.

Fig. 3. Calculus **LG**

this technique together with our constraint-based technique proves PSPACE-completeness of the prenex fragment of intuitionistic logic. In fact, we only sketch proofs due to the lack of space.

First, we recall some techniques of Hudelmaier [8]. The sequent calculus **LG** is shown in Figure 3. Hudelmaier proves [8] that this calculus is sound and complete for propositional intuitionistic logic. In addition, the backward search with backtracking of all possible proofs in **LG** of a given propositional formula φ gives us a decision procedure for intuitionistic logic working in space polynomial in the size of φ. We shall use these properties of **LG** and the constraint satisfaction technique of Section 4 to prove PSPACE-completeness of the prenex fragment of intuitionistic logic.

We introduce a technical definition.

Definition 13. For any quantifier-free formulas φ, ψ, the constraint $\mathcal{E}_{\varphi=\psi}$ is defined inductively as follows:

1. If φ has the form $\varphi_1 \wedge \varphi_2$ and ψ has the form $\psi_1 \wedge \psi_2$, then $\mathcal{E}_{\varphi=\psi} \rightleftharpoons \mathcal{E}_{\varphi_1=\psi_1} \wedge$

$\mathcal{E}_{\varphi_2=\psi_2}$;

2. If φ has the form $\varphi_1 \vee \varphi_2$ and ψ has the form $\psi_1 \vee \psi_2$, then $\mathcal{E}_{\varphi=\psi} \rightleftharpoons \mathcal{E}_{\varphi_1=\psi_1} \wedge \mathcal{E}_{\varphi_2=\psi_2}$;
3. If φ has the form $\varphi_1 \supset \varphi_2$ and ψ has the form $\psi_1 \supset \psi_2$, then $\mathcal{E}_{\varphi=\psi} \rightleftharpoons \mathcal{E}_{\varphi_1=\psi_1} \wedge \mathcal{E}_{\varphi_2=\psi_2}$;
4. If φ has the form $P(s_1, \ldots, s_n)$ and ψ has the form $P(t_1, \ldots, t_n)$, then $\mathcal{E}_{\varphi=\psi} \rightleftharpoons s_1 = t_1 \wedge \ldots \wedge s_n = t_n$;
5. In all other cases $\mathcal{E}_{\varphi=\psi} \rightleftharpoons \bot$.

The calculus $\mathbf{LG}_c$ of constraint sequents is shown in Figure 4.

$$\frac{}{\Gamma, A \to A' \cdot \mathcal{E}_{A=A'}}\ (Ax) \qquad \frac{}{\Gamma, \Lambda \to \varphi \cdot \top}\ (\Lambda)$$

$$\frac{\Gamma \to \varphi \cdot \mathcal{C}_1 \quad \Gamma \to \psi \cdot \mathcal{C}_2}{\Gamma \to \varphi \wedge \psi \cdot \mathcal{C}_1 \wedge \mathcal{C}_2}\ (GI1\wedge)$$

$$\frac{\Gamma, \varphi \supset \chi \to \varphi \cdot \mathcal{C}_1 \quad \Gamma, \psi \supset \chi \to \psi \cdot \mathcal{C}_2}{\Gamma, \varphi \wedge \psi \supset \chi \to \varphi' \wedge \psi' \cdot \mathcal{C}_1 \wedge \mathcal{C}_2 \wedge \mathcal{E}_{\varphi=\varphi'} \wedge \mathcal{E}_{\psi=\psi'}}\ (GI2\wedge)$$

$$\frac{\Gamma \to \varphi \cdot \mathcal{C}}{\Gamma \to \varphi \vee \psi \cdot \mathcal{C}}\ (GI1\vee) \qquad \frac{\Gamma \to \psi \cdot \mathcal{C}}{\Gamma \to \varphi \vee \psi \cdot \mathcal{C}}\ (GI1\vee)$$

$$\frac{\Gamma, \varphi \supset P, \psi \supset P, P \supset \chi \to \varphi \cdot \mathcal{C}}{\Gamma, (\varphi \vee \psi) \supset \chi \to \varphi' \vee \psi' \cdot \mathcal{C} \wedge \mathcal{E}_{\varphi=\varphi'} \wedge \mathcal{E}_{\psi=\psi'}}\ (GI2\vee)$$

$$\frac{\Gamma, \varphi \supset P, \psi \supset P, P \supset \chi \to \psi \cdot \mathcal{C}}{\Gamma, (\varphi \vee \psi) \supset \chi \to \varphi' \vee \psi' \cdot \mathcal{C} \wedge \mathcal{E}_{\varphi=\varphi'} \wedge \mathcal{E}_{\psi=\psi'}}\ (GI2\vee)$$

$$\frac{\Gamma, \varphi \to \psi \cdot \mathcal{C}}{\Gamma \to \varphi \supset \psi \cdot \mathcal{C}}\ (GI1 \supset) \qquad \frac{\Gamma, \varphi, \psi \supset \chi \to \psi \cdot \mathcal{C}}{\Gamma, (\varphi \supset \psi) \supset \chi \to \varphi' \supset \psi' \cdot \mathcal{C} \wedge \mathcal{E}_{\varphi=\varphi'} \wedge \mathcal{E}_{\psi=\psi'}}\ (GI2 \supset)$$

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

$$\frac{\Gamma, A, \psi \to \chi \cdot \mathcal{C}}{\Gamma, A, A' \supset \psi \to \chi \cdot \mathcal{C} \wedge \mathcal{E}_{A=A'}}\ (GE \supset P) \qquad \frac{\Gamma, \varphi \supset (\psi \supset \chi) \to \rho \cdot \mathcal{C}}{\Gamma, (\varphi \wedge \psi) \supset \chi \to \rho \cdot \mathcal{C}}\ (GE \supset \wedge)$$

$$\frac{\Gamma, \varphi \supset P, \psi \supset P, P \supset \chi \to \rho \cdot \mathcal{C}}{\Gamma, (\varphi \vee \psi) \supset \chi \to \rho \cdot \mathcal{C}}\ (GE \supset \vee)$$

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

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

The rules satisfy both the restrictions of $\mathbf{LG}$ and the restrictions of $\mathbf{LJ}_c$.

Fig. 4. Calculus $\mathbf{LG}_c$

Using the same considerations as in Theorem 8, and completeness of $\mathbf{LG}$ we can prove the following

Theorem 14. *Let φ be a quantifier-free formula and θ be a substitution. Then the following conditions are equivalent:*

1. *There is a derivation of $\rightarrow \varphi\theta$ in* **LJ***;*
2. *There is a constraint $\mathcal{C}$ and a derivation of $\rightarrow \varphi \cdot \mathcal{C}$ in $\mathbf{LG}_c$ such that $\theta \models \mathcal{C}$.*

In fact, the condition on skeletons formulated in Theorem 8 for **LJ** and $\mathbf{LJ}_c$ is not true for **LG** and $\mathbf{LG}_c$: the calculus $\mathbf{LG}_c$ gives "more" derivations than **LG**.

Definition 15 (Prenex formula). A formula is called prenex iff it has the form $Q_1x_1 \dots Q_nx_n\varphi$, where every Q_i is a quantifier and φ is a quantifier-free formula.

Theorem 14 and considerations similar to those of Theorem 8 prove

Theorem 16. *Let φ be a closed prenex formula. Then the sequent $\rightarrow \varphi$ is derivable in* **LJ** *iff there is a satisfiable constraint $\mathcal{C}$ such that $\rightarrow \varphi \cdot \mathcal{C}$ is derivable in $\mathbf{LG}_c$.*

We note that $\mathbf{LG}_c$ is in general incomplete: this theorem only states its completeness for prenex formulas.

Theorem 17. *The prenex fragment of intuitionistic logic is PSPACE-complete.*

Proof. The complete proof of this statement is quite lengthy, so we only sketch the proof. Let φ be any prenex formula. Considerations similar to that of Hudelmaier [8] prove that the length of any branch in an $\mathbf{LG}_c$-derivation of a sequent $\rightarrow \varphi$ is linear in the size of φ. One can also note that the size of constraints in any $\mathbf{LG}_c$-derivation of this sequent can be bounded by a polynomial in the size of φ by replacing constraints of the form $\mathcal{C} \wedge \mathcal{C}$ by $\mathcal{C}$. These two facts together, Theorem 16 on completeness of $\mathbf{LG}_c$ for prenex formulas and Theorem 11 on constraint satisfaction show that backward derivations in $\mathbf{LG}_c$ with backtracking and a suitable constraint simplification mechanism give a polynomial space decision procedure for **LJ**. This shows that the prenex fragment of intuitionistic logic is in PSPACE.

Since intuitionistic propositional logic is already PSPACE-hard (Statman [22]), we conclude that the prenex fragment of intuitionistic logic is PSPACE-hard, and hence PSPACE-complete.

This result is interesting because the prenex fragment of classical logic (even without function symbols) is undecidable. In fact, in classical logic every formula is effectively equivalent to a prenex formula.

6 Conclusion and related work

The idea to use constraints instead of substitutions is used in many areas of automated deduction. It is presented in the unification algorithm of Martelli and Montanari [14], it is used in the semantics of logic programs, for example in Gabbrielli, Dore and Levi [6], and Maher [13]. It is implemented in the theorem proving system based on the sequent calculus for classical logic (Anufriev and Aselderov [1], Degtyarev, Zhezherun and Lyaletski [3], see also the survey by Degtyarev and Voronkov [2]). The use of constraints in automated deduction is investigated in many papers (see the survey by Kirchner [11]). In this article, we consider the use of constraints in intuitionistic logic. In particular, we obtain some complexity results for the problem of the instantiation of derivation skeletons and for the prenex fragment of intuitionistic logic.

It is generally well-known how to design sequent-based proof-procedures for intuitionistic logic. For example, a matrix characterization has been developed by Wallen [29], a tableau-based proof procedure has been described by Shankar [21] and systems based on the inverse method have been given in Voronkov [26] and Tammet [23]. Related logical papers are Vorob'ev [24, 25], Kanger [9], Maslov [15], Dyckhoff [5], Hudelmaier [8] and Miglioli, Moscato and Ornaghi [16]. The novelty of our paper is an abstract characterization of provability and related complexity results.

This characterization and related complexity results can be used in several ways not pursued in the above mentioned papers. Let us first consider the complexity results. Both in Shankar [21] and Voronkov [26] the substitutions satisfying constraints are constructed in an "eager" way. Shankar [21] uses "local skolemization" to instantiate essentially universally quantified variables. Voronkov [26] uses ordinary variables which is possible due to the *locality* of the inverse method. In both cases, it is known that the resulting substitution may have size exponential in the size of the set of constraints. Thus, even for "almost propositional" fragments of intuitionistic logic, like the prenex fragment, both methods yield a procedure using exponential space. (In fact, the method of Shankar [21] does not even give a decision procedure, unless it is suitable modified.) Our technique allows us to find a polynomial space decision procedure for this fragment.

In the matrix characterization of Wallen [29], provability is characterized in terms of a multiplicity (the number of copies of subformulas to be used), existence of a substitution and certain ordering restrictions. The ordering restrictions are supposed to encode the order of the application of inference rules in **LJ**. For a given multiplicity, it is not clear what is the complexity of the problem of the existence of a substitution satisfying the ordering restrictions. For example, Otten and Kreitz [18] use an algorithm based on a string unification algorithm. The complexity of full string unification is not known and currently known upper bounds yield several exponents. (It is not clear whether Otten and Kreitz [18]) use full string unification, or only some special case.) In addition, the Wallen's characterization does not give a decision algorithm even for propositional intuitionistic logic.

Another interesting point in the characterization of provability using constraints is that such a characterization can be generalized to include built-in relations, for example arithmetical relations, in intuitionistic theorem provers.

In [27] the technique of this paper is developed for intuitionistic logic with equality. In [4] we prove some complexity results related to intuitionistic logic with equality. In the case of arbitrary built-in predicates, it is hard to expect either the tableau method or the inverse method to eagerly find a substitution closing a derivation. The constraint-handling mechanism for built-in relations can be integrated into our constraint satisfaction algorithms.

In our paper, the use of constraints instead of explicit substitutions gives a polynomial time procedure instead of an exponential time procedure. When it comes to built-in relations, the use of constraints can result in more essential speedup as noted in Nieuwenhuis and Rubio [17]. That paper has shown how to use an NP-complete constraint satisfaction problem instead of constructing a double exponential number of minimal AC-unifiers.

Our technique can be used for other non-classical logics, for example some modal logics. Our characterization of provability (Theorem 8) and related complexity results (Theorems 11 and 12) are also true for classical logic.

References

1. F.V. Anufriev and Z.M. Aselderov. The algorithm of evidence (in Russian). *Kibernetika*, (5):29–60, 1972.
2. A. Degtyarev and A. Voronkov. Equality control methods in machine theorem proving. *Cybernetics*, 22(3):298–307, 1986.
3. A.I. Degtyarev, A.P. Zhezherun, and A.V. Lyaletski. Deductive tools of a mathematical text processing system (in Russian). *Kibernetika*, (5):105–108, 1978.
4. A. Degtyarev and A. Voronkov. Skolemization and decidability problems for fragments of intuitionistic logic. To appear in *LICS'96*, 10 pages, 1996.
5. R. Dyckhoff. Contraction-free sequent calculi for intuitionistic logic. *Journal of Symbolic Logic*, 57:795–807, 1992.
6. M. Gabbrielli, G.M. Dore, and G. Levi. Observable semantics for constraint logic programs. *Journal of Logic and Computation*, 5(2):133–171, 1995.
7. M.R. Garey and D.S. Johnson. *Computers and Intractability.* Freeman, San Francisco, 1979.
8. J. Hudelmaier. An $O(n \log n)$-space decision procedure for intuitionistic propositional logic. *Journal of Logic and Computation*, 3(1):63–75, 1993.
9. S. Kanger. *Provability in Logic*, volume 1 of *Studies in Philosophy.* Almqvist and Wicksell, Stockholm, 1957.
10. 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.
11. H. Kirchner. On the use of constraints in automated deduction. In A. Podelski, editor, *Constraint Programming: Basics and Tools*, volume 910 of *Lecture Notes in Computer Science*, pages 128–146. Springer Verlag, 1995.
12. P. Lincoln, A. Scedrov, and N. Shankar. Linearizing intuitionistic implication. In *Proc. IEEE Conference on Logic in Computer Science (LICS)*, pages 51–62, 1991.
13. M.J. Maher. A CLP view of logic programming. In *Proc. Conf. on Algebraic and Logic Programming*, volume 632 of *Lecture Notes in Computer Science*, pages 364–383, October 1992.

14. A. Martelli and U. Montanari. An efficient unification algorithm. *ACM Transactions on Programming Languages and Systems*, 4(2):258–282, 1982.
15. 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.
16. P. Miglioli, U. Moscato, and M. Ornaghi. An improved refutation system for intuitionistic predicate logic. *Journal of Automated Reasoning*, 13:361–373, 1994.
17. R. Nieuwenhuis and A. Rubio. AC-superposition with constraints: no AC-unifiers needed. In A. Bundy, editor, *Automated Deduction — CADE-12. 12th International Conference on Automated Deduction.*, volume 814 of *Lecture Notes in Artificial Intelligence*, pages 545–559, Nancy, France, June/July 1994.
18. J. Otten and C. Kreitz. A connection based proof method for intuitionistic logic. In P. Baumgartner, R. Hähnle, and J. Posegga, editors, *Theorem Proving with Analytic Tableaux and Related Methods*, number 918 in Lecture Notes in Computer Science, pages 122–137, Schloß Rheinfels, St. Goar, Germany, May 1995.
19. D. Prawitz. An improved proof procedure. In J. Siekmann and G. Wrightson, editors, *Automation of Reasoning. Classical Papers on Computational Logic*, volume 1, pages 162–201. Springer Verlag, 1983. Originally appeared in 1960.
20. D. Prawitz. A proof procedure with matrix reduction. In M. Laudet, L. Lacombe, L. Nolin, and M Schutzenberger, editors, *Symp. on Automatic Demonstration*, volume 125 of *Lecture Notes in Mathematics*, pages 207–213. Springer Verlag, 1970.
21. 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.
22. R. Statman. Intuitionistic logic is polynomial-space complete. *Theoretical Computer Science*, 9, 1979.
23. T. Tammet. A resolution theorem prover for intuitionistic logic. *Submitted.*
24. N.N. Vorob'ev. The derivability problem in the constructive propositional calculus with strong negation. *Dokl. Akad. Nauk SSSR (in Russian)*, 58:689–692, 1952.
25. N.N. Vorob'ev. A new derivability algorithm in the constructive propositional calculus. In *Trudy Math. Inst. Steklov*, volume 52. Moscow, 1958.
26. 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.
27. 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.
28. A. Voronkov. Proof-search in intuitionistic logic based on the constraint satisfaction. UPMAIL Technical Report 120, Uppsala University, Computing Science Department, January 1996.
29. L.A. Wallen. *Automated Deduction in Nonclassical Logics.* MIT Press, 1990.