Nonlinear Phenomena in Complex Systems, vol. **18**, no. 2 (2015), pp. 181 - 190

The Inverse Method Application for Non-Classical Logics

Vladimir Pavlov* and Vadim Pak†
Department of Computer Intellectual Technologies,
Saint Petersburg State Polytechnical University,
29 Politechnicheskaya Str, Saint Petersburg 195251, RUSSIA
(Received 1 April, 2015)

Maslov's inverse method is an automated theorem proving method: it can be used to develop computer programs that prove theorems automatically (such programs are called theorem provers). The inverse method can be applied to a wide range of logical calculi: propositional logic, first-order logic, intuitionistic logic, modal logics etc. We give a brief historical background of the inverse method, then discuss existing modifications and implementations of the inverse method for non-classical logics (intuitionistic logic, modal logics and some other logics).

We introduce a variant of the inverse method for intuitionistic logic - a logic that allows only constructive proofs, i.e. proofs that construct an existing mathematical object instead of just establishing the fact that such an object exists. In short, intuitionistic logic can be seen as classical logic without the law of excluded middle $A \vee \neg A$ or the law of double negation elimination $\neg\neg A \supset A$. So, classical proofs by contradiction are not allowed in intuitionistic logic. We discuss our experimental program implementation of the inverse method for intuitionistic logic: some details of implementation, results of experiments on ILTP problems (ILTP is a common library of test problems for intuitionistic theorem provers).

AMS Subject Classification: 68T15, 03B35, 03F55
Keywords: logic, automated reasoning, theorem proving, intuitionistic logic, inverse method

1. Introduction

Maslov's inverse method [1] is a forward-chaining theorem proving method (it constructs proofs in forward direction, from axioms to the goal formula).

Originally, the inverse method was developed by S. Maslov for a fragment of first-order logic. Later it was extended by its author on a broad range of logical calculi. More precisely, the method is applicable to any calculus that satisfies the subformula property (the definition is given in the following chapter).

The inverse method has not became as widespread as currently well-known theorem proving techniques: resolution-based methods (described in [2] and [3]) and tableaux methods (for example, in [4]). However, it seems that the inverse method is currently underestimated. In comparison with tableaux methods which start a proof from the goal formula and try to reduce it to the axioms, Maslov's method constructs a proof in the opposite direction. So, the inverse method can perform better on some kinds of problems that are hardly solved by tableaux methods. If compared with resolution, the inverse method has a larger area of application. That is why exploration of existing inverse method modifications and development of more efficient variants of the method is an actual research task.

2. Preliminary definitions

We assume that readers of the present article are familiar with basic concepts connected with first-order logic and formal proof methods. We recommend books [2, 4] and [3] for introduction as well as for deeper knowledge of the topic. In this chapter, we place the most important definitions for convenience. We use an abbreviation "iff" for "if and only if".

*E-mail: vlapav239@gmail.com
†E-mail: vadimpak917@gmail.com

Logical calculus is a formalization of some logical theory.

First-order logic is a logical calculus which is build upon predicates (assertions, in other words) that depend on variables.

In the current article, we use the following first-order logic language symbols: logical connectives $\neg$, $\wedge$, $\vee$, $\supset$; quantifiers $\forall$, $\exists$; logical constants $\bot$ (logical false), $\top$ (logical true); predicate symbols P, Q, R, etc.; individual constants c, d; variables x, y, z, v etc.; function symbols f, g, h; symbols for arbitrary terms (r, s, t); symbols for arbitrary formulas (A, B, C etc.).

All non-operator symbols (operators include only connectives, quantifiers and constants $\bot$, $\top$) can be enumerated by subscript indexes.

Term is a language construction defined by the following rules:

- a variable is a term;
- an individual constant is a term;
- if f is a function symbol and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.

No other terms exist except terms that can be generated by the rules listed above.

Atomic formula (or ***predicate***) is a formula of the form $P(t_1, \ldots, t_n)$, where P is a predicate symbol, $t_1, \ldots, t_n$ are terms.

Prime formula is either an atomic formula or one of the constants $\bot$, $\top$.

Literal is either an atomic formula or a negation of atomic formula.

Formula is defined by the rules:

- a prime formula is a formula;
- if A is a formula, then $\neg A$ is also a formula;
- if A and B are formulas, then $A \wedge B$, $A \vee B$, $A \supset B$ are formulas;
- if A is a formula and x is a variable, then $\exists x A$ and $\forall x A$ are formulas.

No other formulas exist except formulas that can be generated by the rules above.

An important concept in logic is ***interpretation***. Interpretations in ***classical*** and ***intuitionistic*** first-order logics are different. An interpretation of a formula F in *classical* logic consists in specifying non-empty domain D and mappings of all individual constants, functional and predicate symbols from F on the domain D.

Truth value of the formula F can be calculated for each particular interpretation according to the logical properties of operators (construction $\forall x$ is interpreted as "for all elements x from D $\exists x$ is interpreted as "exists an element x from D").

In *classical* first-order logic, a formula is ***valid*** iff it is true in all interpretations (otherwise the formula is ***invalid***).

Intuitionistic logic rejects the concepts of truth and falsity, and is usually interpreted in terms of provability: a formula F is valid iff it has a constructive proof (i.e. has a direct evidence) [3], otherwise it remains unknown whether F is valid or invalid.

Another important concept is ***derivability***. Each logical calculus includes a set of axioms and inference rules (this set is sometimes called a ***proof system***) which allow to *derive* one facts from other facts in a pure syntactical (i.e. independent of interpretation) way. A statement *derivable* iff it can be deduced from the axioms using the inference rules. The corresponding proof is called a *derivation*.

A logical theory is ***sound***, iff every derivable formula is valid. An opposite notion of ***complete*** calculus requires that every valid formula is derivable in the calculus. In classical first-order logic (as well as in intuitionistic), various *proof systems* can be used. But soundness and completeness are those properties that allow to decide whether a logical calculus is desirable or not.

Finite multiset is a collection of formulas $A_1, \ldots, A_n$ in which order of formulas is irrelevant, and which can contain multiple copies of the same formula.

Sequent is a conditional assertion of the form $\Gamma \vdash \Delta$ where Γ, Δ are finite multisets. A sequent includes two parts: an ***antecedent*** Γ, and

a ***succedent*** Δ. A sequent

$$A_1, \ldots, A_n \vdash B_1, \ldots, B_m \qquad (1)$$

can be understood as "if A_1 and ... and A_n hold, then B_1 or ... or B_m holds". Sequent 1 is equivalent to the formula

$$F = (A_1 \wedge \ldots \wedge A_n) \supset (B_1 \vee \ldots \vee B_m). \qquad (2)$$

Sequent calculus is a logical calculus where derivable objects are sequents.

Single-succedent calculus is a sequent calculus in which derivable sequents can contain at most one formula in the succedent. In ***Multi-succedent calculus*** derived sequents may contain multiple formulas in the succedent.

A logical calculus satisfies the ***subformula property*** iff for any formula that is derivable in the calculus, there exists a derivation in which all formulas are subformulas of the goal formula.

3. Historical backgrounds

The first Maslov's work is dated to 1964 [1]. In that short article, Maslov proposed the deduction search method for a first-order logic formulas of the form

$$Q_1 x_1 \ldots Q_n x_n \, (D_1 \wedge \ldots \wedge D_k) \, . \qquad (3)$$

In Formula 3, Q_1, ..., Q_n are quantifier symbols, $\{x_1, \ldots, x_n\}$ is a set of all variables in (3), all $D_i, i = 1, \ldots, k$ are disjunctions of the form $L_1 \vee \ldots \vee L_m$, where $L_1, \ldots, L_m$ are literals. The main advantage of the method was emphasized in the first Maslov's work: it essentially uses special features of the goal formula that has to be proved.

So, Maslov's invention was too outstanding discovery to be missed completely. S. Maslov and his associates continued to develop the inverse method. In the article dated to 1967 [11], Maslov extended his method from prenex formulas (3) to arbitrary first-order logic formulas with functional symbols. The ideas for that modification were rather conceptual; so, further detailing was needed.

Early publications on the inverse method [1, 11] are rather brief, and present only the general idea without important details. Furthermore, they are mostly concentrated on its theoretical aspects.

The inverse method general scheme for cut-free sequent calculi (i.e. calculi without the cut rule) satisfying *subformula property* was proposed later [12]. This general scheme can be used to specify the inverse method versions for different logical calculi including non-classical logics: modal logics, intuitionistic logic etc.

The article [12] contains detailed description of the underlying theory, presents several examples of using the method for different logics, and illustrates how to use the inverse method both as a proof-search and as a decision procedure. A decision procedure for a certain class of formulas is an algorithm that takes an arbitrary formula from this class and necessarily terminates with an answer “yes” or “no” depending on the formula derivability.

Specification of the general scheme for first-order logic with function symbols is discussed in detail in [14]. This publication covers permissible proof strategies for the inverse method and completeness proofs for different variants of the method.

Most of publications mentioned above use special terminology that was introduced by S. Maslov. There exist several other articles that describe the inverse method using terms commonly used in publications on other methods. Among them are publications in Russian ([15, 16]) and in English ([7, 10, 19]) which we recommend for introduction to the inverse method. In [16], the inverse method is outlined in appendix by S. Maslov and G. Mints. The two alternative formulations of the method for first-order logic formulas are given. In [19], G. Mints restates the same formulation of Maslov's method in a simple and understandable way, presents the short proof of its completeness, and illustrates how to obtain decidability proofs by the inverse method. V. Lifschitz in [10] reports on resolution-like inverse method modifications for propositional logic and first-order logic (the term

"resolution-like" reflects similarity to resolution-based techniques). The article also contains important background, historical information and review of results achieved by S. Maslov and his associates.

Authors of the publication [7] discuss the inverse method in detail concentrating mostly on resolution-like modifications of the method. In Section 3, they formulate the inverse method modification for refuting arbitrary first-order logic formulas and demonstrate how to reduce the inference rules number if input formulas are in a negation normal form. Section 4 describes applying Maslov's method to non-classical logics: intuitionistic logic and modal logics. In the subsequent sections, authors illustrate connections between the inverse method and resolution inferences, show how to adopt some proof search strategies from resolution to the inverse method, discuss some actual research problems connected with Maslov's method (e.g. creation efficient implementations and obtaining simpler completeness proofs). Probably, this article is the most comprehensive publication on the inverse method except the original Maslov's work [12].

On the one hand, several publications discuss theoretical applications of the inverse method connected with discovering decidable classes of formulas. S. Maslov and his colleagues used the method as a decision procedure to establish decidability of some classes which were either known or previously undiscovered [1, 11, 12], [10, 19]. On the other hand, very poor information is available about actual inverse method implementations in theorem-proving programs. The article [5] describes a program that was developed in the 1960th at the Leningrad Branch of the Steklov Mathematics Institute. That program ran on BESM-6 computer and contained about 10000 commands ([5], also [10]). Authors of the article [24] briefly describe another program named LISS that used the inverse method. In [23], a resolution-like implementation of the inverse method for intuitionistic logic is presented, and several search strategies are explored. A theorem prover Imogen [18] is claimed to be the most efficient prover for intuitionistic logic. Authors confirm this assertion by the test results on a large set of problems from the ILTP library [22]. However, Imogen prover is not presented yet in the publicly available list of ILTP tested systems and their results.

Applications of the inverse method to first-order logic with equality can be found in [6, 13]. Along with already mentioned articles, there exist other publications on the inverse method for non-classical logics, e.g. logic of bunched implications [8], and many-valued logics [9].

4. A multi-succedent inverse method calculus for intuitionistic logic

4.1. Additional Definitions

Before proceeding to the inverse method calculus, we need to introduce some more definitions. Most of these definitions can also be found in [7].

Substitution is a finite mapping from variables to terms, denoted by $\{x_1/t_1, \ldots, x_n/t_n\}$. In the definition, all variables x_i are different and $x_i \neq t_i$ for all $i = 1, \ldots, n$. ***Domain*** of a substitution θ, denoted by $dom(\theta)$, is a set $\{x_1, ..., x_n\}$. The ***empty substitution*** ε is the only substitution with empty domain.

Let E be an arbitrary expression. For a substitution $\theta = \{x_1/t_1, \ldots, x_n/t_n\}$, $E\theta$ denotes the result of *simultaneously* substituting the terms $t_1, \ldots, t_n$ for the variables $x_1, \ldots, x_n$ in the expression E. If we use a variable x in notation $E(x)$, then $E(t)$ will denote the expression $E\{x/t\}$. Notation $E[x/t]$ means exactly the same.

A construction σ_{-x} denotes the substitution with $dom(\sigma_{-x}) = dom(\sigma) \setminus \{x\}$

An *occurrence* of a variable x in expression E is called ***bound*** iff this occurrence is in construction $\forall x$ or $\exists x$ or inside the scope of such a quantifier. An occurrence of a x in E is ***free*** iff it is not bound. A variable is *free* in E iff it occurs free in E. A variable is *bound* in E iff it occurs bound in E.

A ***term*** t ***is free for*** a variable x ***in*** in a formula F, iff no variable from t becomes bound by any quantifier in F, after replacing all occurrences of v by t.

Let $var(F)$ be the set of all variables, $free(F)$ be the set of free variables in the formula F.

Ground (closed) formula F is a formula without free variables, i.e. $free(F) = \emptyset$.

A formula F is ***rectified*** iff every quantifier in F binds a different variable, and every bound variable of F not occurs free in F.

A substitution θ is a ***unifier*** of F_1 and F_2 iff $F_1\theta = F_2\theta$. Two formulas are called ***unifiable*** iff they have a unifier. A unifier θ is the ***most general unifier*** for F_1 and F_2 iff the following condition holds: for any other unifier σ there exists a substitution τ such that $\theta\tau = \sigma$.

A ***most general unifier of substitutions*** σ and τ is the most general unifier of the ordered sets $(x_1\sigma, \ldots, x_n\sigma)$ and $(x_1\tau, \ldots, x_n\tau)$, where $\{x_1, \ldots, x_n\} = dom(\sigma) \cup dom(\tau)$.

Formulas F_1 and F_2 are ***variable-disjoint*** iff $free(F_1) \cap free(F_2) = \emptyset$.

Renaming is a substitution which is a one-to-one mapping from its domain to itself. A renaming θ ***renames away*** F_1 from F_2 iff $F_1\theta$ and F_2 are variable-disjoint.

Formulas F_1, F_2 are ***weakly unifiable*** iff $F_1\theta$ and F_2 are unifiable, where θ renames away F_1 from F_2.

Immediate subformulas and ***immediate free subformulas***. If a formula F has the form $A \wedge B$, $A \vee B$, or $A \supset B$, then A and B are immediate (free) subformulas of F. Immediate (free) subformula of $\neg A$ is A. In formulas of the form $\forall x A(x)$ and $\exists x A(x)$, immediate *free* subformula is $A(x)$; immediate subformula is $A(t)$, t is an arbitrary term. So, immediate and immediate free subformulas differ only for formulas of the latter type.

Subformulas, ***free subformulas*** and their ***signs*** are defined inductively. Let F be an arbitrary formula. Then F is a (free) subformula of itself, and F is *positive* in itself. Let $G1$ be a (free) subformula of F, and $G2$ be an immediate (free) subformula of $G1$. Then $G2$ is a (free) subformula of F. The sign of $G2$ is the same as the sign of $G1$, except the cases when $G1$ has the form $G2 \supset B$ or $\neg G2$. In that cases, $G2$ has the opposite sign: if $G1$ is positive subformula in F, then $G2$ is negative subformula in F and vice versa.

As we will introduce a calculus that not contains negations as a primitive, we assume that the following abbreviations are used:

$$\neg A \equiv A \supset \bot$$
$$\top \equiv \bot \supset \bot$$

4.2. A calculus for ground formulas

The article [7] introduces the inverse method calculus for intuitionistic logic which is based on the calculus G3 by Kleene [4]. This variant allows to infer sequents that have at most one formula in the succedent. We designed another variant of the inverse method calculus based on the multi-succedent sequent calculus *m-G3i* from [3]. Multi-succedent calculus can benefit from a single-succedent calculus since proofs in the former calculus can be shorter. We applied a "universal recipe" of automated deduction from [7]. According to this recipe, the first step is to obtain a calculus for closed formulas. We developed such a sequent calculus named *m-G3i-inv-ground*.

In all the rules below, premises and conclusions are sequents. In the rule Px, P is an atomic formula. In the rules $L\exists$ and $R\forall$, the variable y satisfies the so-called eigenvariable condition: y is not free in the conclusion of the rule, and y is free for x in $A(x)$.

$$Px \quad P \vdash P \qquad L\bot \quad \bot \vdash$$

$$LC \quad \frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta} \qquad RC \quad \frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}$$

$$L\wedge_1 \quad \frac{\Gamma, A \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta} \qquad L\wedge_2 \quad \frac{\Gamma, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}$$

$$R\wedge \quad \frac{\Gamma_1 \vdash \Delta_1, A \quad \Gamma_2 \vdash \Delta_2, B}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2, A \wedge B}$$

$$L\vee \quad \frac{\Gamma_1, A \vdash \Delta_1 \quad \Gamma_2, B \vdash \Delta_2}{\Gamma_1, \Gamma_2, A \vee B \vdash \Delta_1, \Delta_2}$$

$$R\vee_1 \quad \frac{\Gamma \vdash \Delta, A}{\Gamma \vdash \Delta, A \vee B} \qquad R\vee_2 \quad \frac{\Gamma \vdash \Delta, B}{\Gamma \vdash \Delta, A \vee B}$$

$$L\supset \quad \frac{\Gamma_1 \vdash \Delta_1, A \quad \Gamma_2, B \vdash \Delta_2}{\Gamma_1, \Gamma_2, A \supset B \vdash \Delta_1, \Delta_2}$$

$$R\supset_1 \quad \frac{\Gamma \vdash B}{\Gamma \vdash A \supset B} \qquad R\supset_2 \quad \frac{\Gamma, A \vdash}{\Gamma \vdash A \supset B}$$

$$R\supset_3 \quad \frac{\Gamma, A \vdash A \supset B}{\Gamma \vdash A \supset B}$$

$$L\forall \quad \frac{\Gamma, A[x/t] \vdash \Delta}{\Gamma, \forall x A \vdash \Delta} \qquad R\forall \quad \frac{\Gamma \vdash A[x/y]}{\Gamma \vdash \forall x A}$$

$$L\exists \quad \frac{\Gamma, A[x/y] \vdash \Delta}{\Gamma, \exists x A \vdash \Delta} \qquad R\exists \quad \frac{\Gamma \vdash \Delta, A[x/t]}{\Gamma \vdash \Delta, \exists x A}$$

Calculus *m-G3i-inv-ground*

The calculus *m-G3i* is suitable for tableau-based methods that make proof search in a backward direction (from the goal formula to the axioms). Since the inverse method constructs proofs in a forward direction, we had to change some rules of the original calculus *m-G3i* to make it suitable for a forward proof search. So, the rules of these two calculi are essentially different.

Let us note the most important differences between the introduced calculus *m-G3i-inv-ground* and the calculus I_{inv} from [7] (except the difference in notation used):

- the calculus *m-G3i-inv-ground* is multi-succedent while the calculus I_{inv} is single-succedent, so there is no need to use additional rules for conjunction and disjunction;
- in the calculus *m-G3i-inv-ground*, "$\bot$"is used as a primitive instead of "$\neg$"

The calculus *m-G3i-inv-ground* is sound since all rules satisfy the following property: if all premises of a rule are intuitionistically valid, then so is a conclusion. Completeness of *m-G3i-inv-ground* can be established in the similar way as it done in [7]: by exploiting the completeness of the calculus *m-G3i* and proving that every formula derivable in *m-G3i* is also derivable in *m-G3i-inv-ground*.

4.3. A free-variable calculus

Finally, we define a free-variable inverse method calculus for intuitionistic logic *m-G3i-inv*. Suppose that we have a goal formula F that is closed and rectified. By a given formula F, we build the inverse method caclulus where all formulas in the inference rules are subformulas of F.

Sequents in the following description slightly differ from conventional sequents and have the form:

$$A_1 \cdot \theta_1, \ldots, A_n \cdot \theta_n \vdash B_1 \cdot \sigma_1, \ldots, B_m \cdot \sigma_m \quad (4)$$

In the formula 4, all A_i are negative free subformulas of F, all θ_i are substitutions ($i = 1, \ldots, n$); all B_j are positive free subformulas of F, all σ_j are substitutions ($j = 1, \ldots, m$); "$\cdot$" denotes the operation of substitution.

In the rule Px, P, and Q are weakly unifiable *atomic* subformulas of F, ρ renames away P from Q, and τ is a most general unifier of formulas $P\rho$ and Q. In each rule, the premises are variable-disjoint with each other as well as with $var(F)$.

In all the rules, a substitution θ is the most general unifier of σ_1 and σ_2. In the rules $L\exists$ and $R\forall$, the eigenvariable condition holds for the term $x\sigma$: $x\sigma$ is a *variable* not occurring *free* in the conclusion of the rule.

$$Px \qquad P \cdot \rho\tau \vdash Q \cdot \tau \qquad L\bot \qquad \bot \vdash$$

$$LC \quad \frac{\Gamma, A \cdot \sigma_1, A \cdot \sigma_2 \vdash \Delta}{\Gamma\theta, A \cdot \sigma_1\theta \vdash \Delta\theta}$$

$$RC \quad \frac{\Gamma \vdash \Delta, A \cdot \sigma_1, A \cdot \sigma_2}{\Gamma\theta \vdash \Delta\theta, A \cdot \sigma_1\theta}$$

$$L\wedge_1 \quad \frac{\Gamma, A\cdot\sigma \vdash \Delta}{\Gamma, A\wedge B\cdot\sigma \vdash \Delta} \qquad L\wedge_2 \quad \frac{\Gamma, B\cdot\sigma \vdash \Delta}{\Gamma, A\wedge B\cdot\sigma \vdash \Delta}$$

$$R\wedge \quad \frac{\Gamma_1 \vdash \Delta_1, A\cdot\sigma_1 \qquad \Gamma_2 \vdash \Delta_2, B\cdot\sigma_2}{\Gamma_1\theta, \Gamma_2\theta \vdash \Delta_1\theta, \Delta_2\theta, A\wedge B\cdot\sigma_1\theta}$$

$$L\vee \quad \frac{\Gamma_1, A\cdot\sigma_1 \vdash \Delta_1 \qquad \Gamma_2, B\cdot\sigma_2 \vdash \Delta_2}{\Gamma_1\theta, \Gamma_2\theta, A\vee B\cdot\sigma_1\theta \vdash \Delta_1\theta, \Delta_2\theta}$$

$$R\vee_1 \quad \frac{\Gamma \vdash \Delta, A\cdot\sigma}{\Gamma \vdash \Delta, A\vee B\cdot\sigma} \qquad R\vee_2 \quad \frac{\Gamma \vdash \Delta, B\cdot\sigma}{\Gamma \vdash \Delta, A\vee B\cdot\sigma}$$

$$L\supset \quad \frac{\Gamma_1 \vdash \Delta_1, A\cdot\sigma_1 \qquad \Gamma_2, B\cdot\sigma_2 \vdash \Delta_2}{\Gamma_1\theta, \Gamma_2\theta, A\supset B\cdot\sigma_1\theta \vdash \Delta_1\theta, \Delta_2\theta}$$

$$R\supset_1 \quad \frac{\Gamma \vdash B\cdot\sigma}{\Gamma \vdash A\supset B\cdot\sigma} \qquad R\supset_2 \quad \frac{\Gamma, A\cdot\sigma \vdash}{\Gamma \vdash A\supset B\cdot\sigma}$$

$$R\supset_3 \quad \frac{\Gamma, A\cdot\sigma_1 \vdash A\supset B\cdot\sigma_2}{\Gamma\theta \vdash A\supset B\cdot\sigma_2\theta}$$

$$L\forall \quad \frac{\Gamma, A\cdot\sigma \vdash \Delta}{\Gamma, \forall xA\cdot\sigma_{-x} \vdash \Delta} \qquad R\forall \quad \frac{\Gamma \vdash A\cdot\sigma}{\Gamma \vdash \forall xA\cdot\sigma_{-x}}$$

$$L\exists \quad \frac{\Gamma, A\cdot\sigma \vdash \Delta}{\Gamma, \exists xA\cdot\sigma_{-x} \vdash \Delta} \qquad R\exists \quad \frac{\Gamma \vdash \Delta, A\cdot\sigma}{\Gamma \vdash \Delta, \exists xA\cdot\sigma_{-x}}$$

Calculus *m-G3i-inv*

The calculus *m-G3i-inv* includes two axioms: Px and $L\bot$. If a sequent was derived by one of the axioms, it is called an initial sequent.

The above calculus is augmented with an implicit renaming rule (ρ is a renaming):

$$\frac{\Gamma \vdash \Delta}{\Gamma\rho \vdash \Delta\rho} \tag{5}$$

Completeness of *m-G3i-inv* can be shown by comparing it with *m-G3i-inv-ground* and using instance lemma as it done in [7].

In a program implementation of the inverse method, it can be convenient to use the stronger form of the eigenvariable condition: $x\sigma$ is a variable not occurring (*free or bound*) in the conclusion of the rule. Though the difference between the two definitions seems to be insignificant, the stronger form of the eigenvariable condition should be used with care.

For example, let us consider intuitionistically valid formula

$$G = \forall x\ \forall y\ (P(x) \supset P(x)).$$

In the formula G, the variable y is quantified but unused. We start to construct the inverse method calculus for the formula G with identifying all free subformulas of G and their signs (we give each free subformula a unique name to refer to):

Free subformula	*Sign*	*Name*
$\forall x\ \forall y\ (P(x) \supset P(x))$	*positive*	G
$\forall y\ (P(x) \supset P(x))$	*positive*	$G1$
$P(x) \supset P(x)$	*positive*	$G2$
$P(x)$	*negative*	$G3$
$P(x)$	*positive*	$G4$

Free subformulas of G

First of all, logical false constant $\bot$ not occurs in the formula G, so the axiom $L\bot$ is not applicable. Next, there are only two atomic free subformulas, namely $G3$ and $G4$. Since $G3$ and $G4$ have opposite signs and are weakly unifiable, we can apply the rule Px and derive one (and only one possible) initial sequent:

$$1.\ P(x)\cdot\varepsilon \vdash P(x)\cdot\varepsilon$$

Since G includes only one (positive) connective $\supset$ and (positive) quantifiers $\forall$, we can restrict the inverse method calculus to four selected rules: $R\supset_1$, $R\supset_2$, $R\supset_3$ and $R\forall$. The rule $R\supset_1$ can be applied to the *Sequent* 1, as subformulas $G3$ and $G4$ from the *Sequent* 1 are immediate free subformulas of $G2$. Other rules are not applicable, e.g., $R\forall$ cannot be applied because the *Sequent* 1 does not contain immediate free subformulas of G or $G1$. Before applying $R\supset_1$, we need to rename a variable x in the *Sequent* 1 as $var(G) = \{x\}$ includes this variable. So we derive the following intermediate sequent:

$$1^*.\ P(x)\cdot\{x/v1\} \vdash P(x)\cdot\{x/v1\}$$

Then we apply $R\supset_1$ directly to the *Sequent* 1*:

$$2.\ \vdash (P(x)\supset P(x))\cdot\{x/v1\}$$

A free subformula $P(x)\supset P(x)$ from the *Sequent* 2 is an immediate free subformula of $G1$. If we apply the rule $R\forall$ to the *Sequent* 2, the conclusion will have the following form:

$$3^{**}.\ \vdash \forall y\ (P(x)\supset P(x))\cdot\{x/v1\}$$

Now we need to check the stronger form of eigenvariable condition: a variable $y\{x/v1\}=y$ not occurs in a *Sequent* 3**. This condition does not hold, so we need to discard the *Sequent* 3**. No other rules can be applied to the *Sequent* 2. Indeed, rules $R\supset_2$ and $R\supset_3$ are not applicable because antecedent of the *Sequent* 2 is empty. The rule $R\supset_1$ is not applicable too: *Sequent* 2 contains the single free subformula $G2$, and $G2$ is the only free subformula of G that has the form $A\supset B$. So theorem-proving process terminates.

However, if we use the weaker eigenvariable condition, the proof can be completed in two steps:

$$3.\ \vdash \forall y\ (P(x)\supset P(x))\cdot\{x/v1\}$$

$$4.\ \vdash \forall x\ \forall y\ (P(x)\supset P(x))\cdot\varepsilon$$

A *Sequent* 3 is derived from the *Sequent* 2 by $R\forall$ (eigenvariable condition holds: a variable $y\{x/v1\}=y$ not occurs *free* in the *Sequent* 3); a *Sequent* 4 is derived from the *Sequent* 3 by $R\forall$ (eigenvariable condition also holds: a variable $x\{x/v1\}=v1$ not occurs free in the conclusion).

In summary, G cannot be proved in *m-G3i-inv* with the stronger eigenvariable condition, so the calculus becomes incomplete. To make it complete, one need to add a restriction that the goal formula does not contain unused bound variables (i.e. for any variable v that is bound by some quantifier, v occurs at least once within the scope of this quantifier). Any formula can be transformed into the appropriate form by eleminating quantifiers that bind such unused variables.

The calculus *m-G3i-inv* can be modified by adding rules for negation operators:

$$L\neg\quad \frac{\Gamma\vdash A\cdot\sigma}{\Gamma,\neg A\cdot\sigma\vdash}\qquad R\neg\quad \frac{\Gamma,A\cdot\sigma\vdash}{\Gamma\vdash\neg A\cdot\sigma}$$

One need to change axioms accordingly: remove $L\bot$; and allow P and Q in the axiom Px to be any prime formulas (i.e. either atomic or $\bot$).

The modified calculus can be more efficient on formulas with negations. We denote the inverse calculus with negation as *m-G3i-inv+*.

5. Details of experimental implementation

We extended an experimental theorem prover for classical logic ([20, 21]) with intuitionistic part. Our program is capable of proving theorems in the developed intuitionistic inverse method calculus *m-G3i-inv*, as well as in *m-G3i-inv+*. Our prover works on arbitrary intuitionistic first-order (and propositional) formulas. It can solve problems given by a set of premises $A_1,\ldots,A_n$ and a set of conclusions $B_1,\ldots,B_m$.

Since the inverse method is incomplete for proving arbitrary sequents [7], our program firstly transforms a given problem into an equivalent formula 2. The goal formula F must be closed to prove it using the inverse method (the goal formula is proved iff the sequent $\vdash F$ was derived).

Next, the goal formula is rectified: bound variables are renamed to obtain a formula where each quantifier binds a different variable. All quantifiers that bind unused variables are eliminated. For the calculus *m-G3i-inv* that has a primitive "$\bot$" instead of "$\neg$ we replace all occurrences of formulas "$\neg A$" with "$A\supset\bot$".

An implemented inference loop is a variant of the given clause algorithm [17] adapted for proving sequents. All derived sequents are divided into two sets: usable set (contains passive sequents) and set of support (with active sequents). At every iteration, the program selects the given sequent from the set of support. Then the program applies all inverse method calculus rules with the given sequent as one of the

premises; if applied rule has two premises, other premise is selected from the usable list.

We run our prover on 279 selected test problems from ILTP library [22]. Proof time for each problem was limited by 3 seconds. The prover solved 164 problems, including 104 proved theorems and 60 refuted assertions (a formula is refuted iff all possible derivations by the inverse method were made and the goal formula was not derived). All proof results were correct: only intuitionistically valid formulas were proved and only non-valid formulas were refuted. Tests were run on a computer with Intel Core 2 Duo 2.67 GHz processor and 3 Gb RAM.

Our prover is implemented in C++ and consists of the following parts:

- class library, which contains classes for performing all necessary operations on formulas and sequents;
- class that implement the inference loop;
- classes with sequent calculus specification.

All three parts have independent implementation, so one can change the inference loop algorithm (i.e. add some strategies) without affecting the sequent calculus itself; on the other hand, rules of the sequent calculus (as well as order of their application) can be changed with the inference loop remaining the same. Formulas (or sequents) transformation algorithms can also be improved without changing other two parts of the program. So, our program allows to use different inverse method calculi and compare their efficiency.

6. Conclusion

In the current article, we discussed existing applications of the inverse method for non-classical logics and presented a variant of the inverse method for intuitionistic logic. The novelty of our work is that our calculus is multi-succedent, in comparison with known inverse method calculi for intuitionistic logic (e.g. a calculus from [7]). We also suggested a modification of the calculus and illustrated some important details with an example of the proof. We presented an experimental program that can automatically prove theorems in the designed calculus.

Our future plans include implementing more powerful proof search strategies and extending our experiments on a larger set of test problems. It will be fruitful to compare different inverse method calculi, e.g. multi-succedent calculus with the single-succedent calculus from [7].

References

[1] S.Y. Maslov. The inverse method for establishing deducibility in classical predicate calculus. Doklady AN SSSR. **159**, 17-20 (1964). (in Russian). [English translation: Soviet Mathematics – Doklady. **5**].

[2] C. Chang, R. Lee. *Symbolic Logic and Mechanical Theorem Proving*. (Academic Press, New York, 1973).

[3] A.S. Troelstra, H. Schwichtenberg. *Basic Proof Theory*. 2nd ed. (Cambridge University Press, Cambridge 2000).

[4] S.C. Kleene. *Mathematical Logic*. (Wiley, New York, 1967).

[5] G. V. Davydov, S.Yu. Maslov, G.E. Mints, V.P. Orevkov, A.O. Slisenko. A machine algorithm for establishing deducibility on the basis of the inverse method. Zapiski Nauchnykh Seminarov LOMI. **16**, 8-19 (1969). (in Russian). [English translation: Seminars in Mathematics. Steklov Mathem. Inst. **16**. Also in: Siekmann, J., Wrightson, G. (eds.) *Automation of Reasoning*, vol. **2**. (Springer-Verlag, Heidelberg, 1983)].

[6] Degtyarev, A., Voronkov, A. *Equality Elimination for the Inverse Method and*

Extension Procedures. Proceedings of IJCAI-1995. Vol. **1**, pp. 342-347. (Morgan Kaufmann, San Francisco 1995).

[7] A. Degtyarev, A. Voronkov. The inverse method. In: *Handbook of Automated Reasoning*, vol. 1. Eds. A. Robinson, A. Voronkov. (Elsevier, Amsterdam, 2001). Pp. 179-272.

[8] K. Donnelly, T. Gibson, N. Krishnaswami, S. Magill, S. Park. The Inverse Method for the Logic of Bunched Implications. In: *LPAR 2004. LNAI*, vol. 3452. Eds. F. Baader, A. Voronkov. (Springer, Heidelberg, 2005). Pp. 466-480.

[9] L. Kovács, A. Mantsivoda, A. Voronkov. The Inverse Method for Many-Valued Logics. In: *12th Mexican International Conference on Artificial Intelligence MICAI 2013. LNCS*, vol. 8265. Eds. F. Castro, A. Gelbukh, M. González. (Springer, Heidelberg, 2013). Pp. 12-23.

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

[11] S.Y. Maslov. The inverse method for establishing deducibility of non-prenex formulas of predicate calculus. Doklady AN SSSR. **172**, 22-25 (1967). (in Russian). [English translation: Soviet Mathematics – Doklady. **8**].

[12] S.Y. Maslov. The inverse method for establishing deducibility for logical calculi. Logical and logical-mathematical calculus. Trudy Mat. Inst. Steklov **98**, 26-87 (1968). (in Russian). [English translation: Proc. Steklov Inst. of Mathematics. **98**. (American Math. Society, Providence, 1971)].

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

[14] S. Y. Maslov. The inverse methods and tactics for establishing deducibility for a calculus with function symbols. Trudy Matem. Inst. AN SSSR. **121**, 14-56 (1972). (in Russian). [English translation: Proc. Steklov Inst. of Mathematics. **121**. American Math. Society, Providence (1974)].

[15] S.Y. Maslov. *The Theory of Deductive Systems and Its Applications*. (Radio i svyas, Moscow 1986). (in Russian). [English translation published by MIT Press (1987)]

[16] S.Y. Maslov, G.E. Mints. The theory of proof search and the inverse method. In: Appendix A to the Russian translation of the book by Chang and Lee [2]. (Nauka, Moscow, 1983). (in Russian).

[17] W. McCune. Prover9 and Mace4. Online source: http://www.cs.unm.edu/ mccune/Prover9, 2005-2010. ,

[18] McLaughlin, S., Pfenning, F. Efficient Intuitionistic Theorem Proving with the Polarized Inverse Method. In: *CADE-22. LNCS*. Vol. 5663. Ed. R.A. Schmidt. (Springer, Heidelberg, 2009). Pp. 230-244.

[19] G. Mints. Decidability of the Class E by Maslov Inverse Method. In: *Essays Dedicated to Yuri Gurevich on the Occasion of His 70th Birthday*. LNCS, vol. 6300. Eds. A. Blass, N. Dershowitz, W. Reisig. (Springer, Heidelberg, 2010). Pp. 529-537.

[20] V. Pavlov, A. Schukin, T. Cherkasova. Exploring Automated Reasoning in First-Order Logic: Tools, Techniques and Application Areas. In: *4th International Conference, KESW 2013. CCIS*, vol. 394. Eds. P. Klinov, D. Mouromtsev. (Springer, Heidelberg, 2013). Pp. 102-116.

[21] V. Pavlov, V. Pak. The Inverse Method and First-Order Logic Theorem Proving. Nonlinear Dynamics and Applications. **20**, 127-135 (2014).

[22] T. Raths, J. Otten, C. Kreitz. The ILTP Library: Benchmarking Theorem Provers for Intuitionistic Logic. In: *Automated Reasoning with Analytic Tableaux and Related Methods, TABLEAUX 2005, LNAI*, vol. **3702**, Ed. B. Beckert. (Springer Verlag, 2005). Pp. 333-337.

[23] T. Tammet. A resolution theorem prover for intuitionistic logic. In: *CADE-13. LNCS*, vol. 1104. Eds. M.A. McRobbie, J.K. Slaney. (Springer, Heidelberg 1996). Pp. 2-16.

[24] A.A. Voronkov. Liss – The logic inference search system. In: *Proceedings. LNCS*, Kaiserslautern, FRG, July 24-27, 1990. Vol. **449**. Ed. M.E. Stickel. (Springer, Heidelberg, 1990). Pp. 677-678.