THE INVERSE METHOD FOR SOLVING ARTIFICIAL INTELLIGENCE PROBLEMS IN THE FRAMEWORKS OF LOGIC-OBJECTIVE APPROACH AND BOUNDS OF ITS NUMBER OF STEPS

Tatiana Kosovskaya, Nina Petukhova

***Abstract**: The paper is devoted to the modifying of Maslov inverse method for a special form of predicate formulas used in the solution of artificial intelligence problems. An algorithm of the inverse method application for such type formulas is justified. Upper and lower bounds of the number of steps in such an application are obtain. Upper bounds coincide with those of other deduction algorithms, but the exhaustion is greatly reduced while construction particular derivation.*

***Keywords**: artificial intelligence, pattern recognition, predicate calculus, inverse method of S.Yu.Maslov, complexity theory*.

***ACM Classification Keywords**: I.2.4 ARTIFICIAL INTELLIGENCE Knowledge Representation Formalisms and Methods – Predicate logic, I.5.1 PATTERN RECOGNITION Models – Deterministic, F.2.2 Nonnumerical Algorithms and Problems – Complexity of proof procedures.*

Introduction

One of the most thought out derivation methods for predicate calculus is the inverse method proposed by S.Yu. Maslov [Orevkov, 2003]. In [Kosovskaya, 2011] it is shown that many of artificial intelligence problems are reduced to the proof of a special type formulas. A simplification of the inverse method for the formulas of such a type is presented in this paper.

Bounds of the number of steps for solving an artificial intelligence problems (namely the problem of logic-objective recognition) using the Maslov inverse method are considered in this paper. The problem is the following [Kosovskaya, Timofeev, 1985]

Let Ω be a collection of finite sets $\omega=\{a_1,\ldots,a_k\}$ which will be called objects. Any subset τ of ω will be called its part. A partition $\Omega = \cup_{j=1}^{K}\Omega_j$ of the set Ω on K (possibly intersected) classes is done. A set of predicates $p_1,\ldots,p_n$ characterizes the properties of and relationships between the elements of ω.

Logical description $S(\omega)$ of the object ω is a set of all true constant formulas of the type $p_i(\bar{\tau})$ or $\neg p_i(\bar{\tau})$ calculated for all possible parts τ of the object ω.

Here and below the notation $\bar{x}$ is used for a list of elements of a finite set x, corresponding to some permutation of its elements. The fact that the list $\bar{x}$ elements are the elements of y will be written in the form $x \subseteq y$. In order to write that list of values for variables $\bar{x}$ that satisfy the formula $A(\bar{x})$ are different the notation $\exists \bar{x}_{\neq} A(\bar{x})$ will be used. instead of the formula

$$\exists x_1 ... \exists x_n \left(\&_{i=1}^{n-1} \&_{j=i+1}^{n} (x_i \neq x_j) \& A(x_1, ..., x_n) \right).$$

It is shown in [Kosovskaya, 2011] that the most Artificial Intelligence problems may be reduced to the proof of the formula of the type

$$S(\omega) \Rightarrow \exists \bar{x}_{\neq} A_j(\bar{x}), \quad (1)$$

where $A_j(\bar{x})$ is an elementary conjunction of atomic formulas with predicates $p_1,\ldots,p_n$. This problem is known to be an NP-hard problem [Kosovskaya, 2007]. The proof of the sequent (1) is equivalent to the proof of the formula

$$(\& S(\omega)) \rightarrow \exists \bar{x}_{\neq} A(\bar{x}),$$

where $(\&S(\omega))$ is a notification for a conjunction of all formulas from the set $S(\omega)$.

This formula can be reduced using the equivalent transformations to formula of the form

$$\forall a_1, ..., a_k \exists x_1, ..., x_n \left(\&_{i=1}^{\alpha} D_i(a_1, ..., a_k, x_1, ..., x_n) \right),$$

where D_i is $\vee \neg S(\omega) \vee P_{k_i}(\bar{x})$ and the notation $\vee \neg S(\omega)$ means a disjunction of negations of formulas of $S(\omega)$, and the solution of which will be considered in the paper.

The Maslov Inverse Method and its Modification

We modify the Maslov inverse method described in [Orevkov, 2003] for the formulas of the form

$$\forall a_1, ..., a_k \exists x_1, ..., x_n \left(\&_{i=1}^{\alpha} D_i(a_1, ..., a_k, x_1, ..., x_n) \right). \quad (2)$$

The original inverse method is formulated for the formulas of the form

$$\forall z_1, ..., z_k \exists (x_1, ..., x_n)_{\neq} \forall y_1, ..., y_m \left(\&_{i=1}^{\alpha} D_i(z_1, ..., z_k, x_1, ..., x_n, y_1, ..., y_m) \right),$$

special case of which are the formulas considered in this paper. In our case there are no variables which are universally quantified at the inner and, respectively, the operations associated with them should be omitted.

Variables universally quantified at the outer will be considered as constants.

Below the definitions without numbering are the definitions from [Orevkov, 2003]. For the considered case definitions in the paper are numbered.

Definition. $x<y$ if there exists a formula $D_i(a_1, ..., a_k, u_1, ..., u_n, d_1, ..., d_m)$ from a list of formulas Γ of the form $D_i(a_1, ..., a_k, u_1, ..., u_n, d_1, ..., d_m)$ such that x coincides with one of the variables u_i, and y coincides with one of the variables d_j.

Definition: List Γ of the formulas of the form $D_i(a_1, ..., a_k, c_1, ..., c_n, d_1, ..., d_m)$ is called **admissible** under the following conditions.

1. For each formula D_i from Γ variables $a_1,...a_k, d_1,...,d_m$ are mutually distinct.

2. Whatever be the formulas $D_i(a_1, ..., a_k, u_1, ..., u_n, d_1, ..., d_m)$ and $D_r(a_1, ..., a_k, v_1, ..., v_n, c_1, ..., c_m)$ from Γ if a variable d_l $(1 \leq l \leq m)$ coincides with a variable c_p $(1 \leq p \leq m)$ then $l = p$.

3. It is impossible to find a string of variables $x_1, ..., x_l$ $(l \geq 1)$ appearing in the formulas of Γ such that $x_1 < x_2 < ... < x_l < x_1$.

Since a_k are constants and d_m are absent in (2) then the conditions 1. – 3. hold for any D_i. Hence we can formulate the following definition.

Definition 1. Any list Γ of formulas of the form $D_i(a_1,...,a_k,c_1,...,c_n)$ is **admissible** for the formulas of the form (2).

Definition: An admissible list Γ of formulas of the form $D_i(a_1,...,a_k,c_1,...,c_n,d_1,...,d_m)$ is an **F-set** under the following conditions.

1. Formulas of the list Γ are not repeated.

2. Whatever be the formulas $D_i(a_1,...,a_k,u_1,...,u_n,d_1,...,d_m)$ and $D_r(a_1,...,a_k,v_1,...,v_n,c_1,...,c_m)$ from Γ if the sets $d_1,...,d_m$ and $c_1,...,c_m$ have a common variable then D_r and D_i coincide.

Definition 2. If the formulas in a list Γ of formulas of the form $D_i(a_1,...,a_k,c_1,...,c_n)$ are not repeated then this list is an **F-set**.

Definition: Let $\alpha_1,...,\alpha_l$ and $\beta_1,...,\beta_l$ be the lists of variables and constants that can coincide with each other and with the constants from the list $a_1,...,a_k$. Let us consider the system of equalities

$$\begin{cases} \alpha_1 = \beta_1 \\ \quad \cdots \\ \alpha_l = \beta_l \end{cases}.$$

Let $u_1,...,u_p$ be a list without repetitions of all variables which differ from constants $a_1,...,a_k$ that appear in equalities of the system (3). The system (3) is called a **system of equations in the variables** $u_1,...,u_p$. Any set of variables $(\gamma_1,...,\gamma_p)$ such that after simultaneous replacement of variables $u_1,...,u_p$ by their values in the solution $(\gamma_1,...,\gamma_p)$ the left and right parts of each equation of the system coincide is called a **solution of system of equations** (3). We say that the **solution σ_1 absorbs the solution σ_2** if σ_2 may be received from σ_1 by replacing some of the basic variables by other variables. **The solution** of the system of equations (3) is called **universal** if it absorbs all the solutions of this system.

For the considered case this definition takes the form

Definition 3. Let $\alpha_1,...,\alpha_l$ be a list of constants from the list $a_1,...,a_k$ and $\beta_1,...,\beta_l$ be a list of some variables and constants from the same list $a_1,...,a_k$. Consider the system of equalities

$$\begin{cases} \alpha_1 = \beta_1 \\ \quad \cdots \\ \alpha_l = \beta_l \end{cases}. \qquad (3)$$

Let $u_1,...,u_p$ be a list without repetitions of all variables included in the equation system (3). System (3) is called a **system of equations with variables** $u_1,...,u_p$. Any set of constants $(\gamma_1,...,\gamma_p)$ from the list $a_1,...,a_k$ such that after simultaneous replacement of variables $u_1,...,u_p$ by their values in the solution $(\gamma_1,...,\gamma_p)$ the left and the right parts of each equation of the system coincide is called a **solution of system of equations** (3). **The system of equations** (3) **has no solution** if the list $(\gamma_1,...,\gamma_p)$ has repetitions.

In the case considered in the paper two variables in systems of equations cannot be compared (since different variables may have only different values according to the formulation of the problem). So the definitions of an absorbing and universal solutions are not used in the paper.

Procedure 1 (identifying of variables with constants). Let *Γ* be a list of formulas of the form $D_i(a_1,...,a_k,u_1,...,u_n)$, S be a system of equations with variables $u_1,...,u_p$ as unknowns, *σ* be a solution of this system. The procedure for identifying of variables with constants in the list of *Γ* according to the system *S* consists in the replacing of variables $u_1,...,u_p$ by their values from the solution of *σ* in all formulas from the list *Γ*.

Procedure of identifying formulas. Let *Γ* be a list of formulas in the form $D_i(a_1,...,a_k,c_1,...,c_n,d_1,...,d_m)$ and includes formulas $D_i(a_1,...,a_k,u_1,...,u_n,d_1,...,d_m)$ and $D_r(a_1,...,a_k,v_1,...,v_n,c_1,...,c_m)$. The procedure of identifying of these formulas in the list *Γ* consists in application to the list *Γ* of procedure of identifying of variables according to the system of equations in variables

$$\begin{cases} u_1 = v_1 \\ \quad \dots \\ u_n = v_n \\ d_1 = c_1 \\ \quad \dots \\ d_m = c_m \end{cases}.$$

If the system has a solution then the determined procedure succeeds.

In the considered case there are no variables $d_1,...,d_m$ and $c_1,...,c_m$ and the procedure will not be considered, as in the considered case different variables have different values. Hence the system has no solution.

The procedure of transformation of a list of formulas to an F-set. Let *Γ* be a list of formulas of the form $D_i(a_1,...,a_k,c_1,...,c_n,d_1,...,d_m)$. The procedure of transformation of a list of formulas *Γ* to an F-set consists in the sequential execution of the following operations.

- Check whether the list is admissible. If it is so then go to the next item. Otherwise, the procedure defined ends without a result.
- Reduce all the repetitions of formulas in the list.
- Check whether the list is an F-set. If it is so then the processed list is the result of the determined procedure. Otherwise, proceed to the next step.
- Find such formulas $D_i(a_1,...,a_k,u_1,...,u_n,d_1,...,d_m)$ and $D_r(a_1,...,a_k,v_1,...,v_n,c_1,...,c_m)$ in the list that sets $d_1,...,d_m$ and $c_1,...,c_m$ have a common variable. Apply the procedure of identifying of these formulas. If it succeeds then go to the next step. Otherwise, the defined procedure ends without a result.
- Check if there is a repetition of formulas in the resulting list of the previous step. If so, then go to step 1. Otherwise, the defined procedure ends without a result.

In our case, the first point should not be performed as any list is admissible. The fourth point also should not be performed. The fifth one in the absence of the fourth one repeats the second, so the procedure of transformation of the list of formulas in F-sets takes the form.

Procedure 2 (transformation of list of formulas to an F-sets). Let Γ be a list of formulas of the form $D_i(a_1,...,a_k,u_1,...,u_n)$. The transformation of the list Γ to an F-set consists in deleting repetitions of formulas in this list.

The procedure of gluing of formulas in F-sets. Let Γ be an F-set and Γ includes formulas *A* and *B*. The procedure of gluing of formulas *A* and *B* in the list Γ is sequential execution of the following operations.

- Apply the procedure of identification of formulas *A* and *B*. If it succeeds then go to the next step. Otherwise, the defined procedure ends without a result.
- Apply to the list of formulas obtained in the previous step the procedure for transformation of a list of formulas to an F-set.

In our case, this procedure will not be used because there is not used the procedure of identifying of formulas (the first step). The second step (without execution of the first step) is fulfilled automatically.

The procedure of constructing a closed F-sets: Let $D_i(a_1,...,a_k,t_1,...,t_n,d_1,...,d_m)$ and $D_r(a_1,...,a_k,v_1,...,v_n,c_1,...,c_m)$ be formulas of the form $D_i(a_1,...,a_k,c_1,...,c_n,d_1,...,d_m)$ where all the variables $a_1,...,a_k$, $t_1,...,t_p$, $d_1,...,d_m$, $v_1,...,v_p$ and $c_1,...,c_m$ are different. Denote the first formula by means of *A* and the second one by *B*. Assume that *A* contains an atomic formula $P(t_1, ..., t_s)$ as a disjunct and *B* contains the negation of the formula $P(v_1, ..., v_s)$, where P is an s -ary predicate. The procedure for constructing a closed F-set according to the pairs of formulas *A*, $P(t_1, ..., t_s)$ and *B*, $\neg P(v_1, ..., v_s)$ is the sequential execution the following operations.

1 Apply the procedure of identifying of variables to the list $\langle A, B\rangle$ according to the system of equations in variables

$$\begin{cases} t_1 = v_1 \\ \quad \dots \\ t_s = v_s \end{cases}.$$

2 If the identifying procedure is successful then use the procedure of transformation of the resulting list of formulas in F-set.

Procedure 3 (construction of a closed F-set for formulas of the form (2)). Let $D_i(a_1,...,a_k,t_1,...,t_n)$ and $D_r(a_1,...,a_k,v_1,...,v_n)$ be the formulas of the form (2), where $t_1,...,t_p$ and $v_1,...,v_n$ are variables and $a_1,...,a_k$ are constants. Denote the first formula by means of *A*, and the second one by *B*. We will assume that *A* contains atomic formula $P(t_1, ..., t_s)$ as a disjunct, and B contains the negation of the formula $P(a_1, ..., a_s)$, where *P* is an s-ary predicate. The procedure for constructing a closed F-sets of pairs of formulas *A*, $P(t_1, ..., t_s)$ and *B*, $\neg P(a_1,..., a_s)$ is the sequential execution of the following operations.

1. Apply the procedure of identifying of variables to the list $\langle A, B\rangle$ according to the system of equations in variables

$$\begin{cases} t_1 = a_1 \\ \quad \dots \\ t_s = a_s \end{cases}.$$

2. If the identifying procedure is successful then use the procedure of transformation of the resulting list of formulas in F-set.

Definition 4: F-set is called a **closed** one if it may be obtained by applying the procedure of constructing a closed F-set with some pairs of formulas *A, P (t1, ..., ts)* and *B, ⊣P(a1,..., as)*.

Procedure of application of Rule B to an F-set. Consider a system of δ F-sets

$$\begin{cases} \Gamma_1, \quad D_1\left(a_1,...,a_k,c_1^1,...,c_n^1,d_1^1,...,d_m^1\right) \\ \qquad\qquad ... \\ \Gamma_\delta, \quad D_\delta\left(a_1,...,a_k,c_1^\delta,...,c_n^\delta,d_1^\delta,...,d_m^\delta\right) \end{cases}, \tag{4}$$

where $\Gamma_1,\ldots,\Gamma_\delta$ are the lists of formulas of form $D_i\left(a_1,...,a_k,c_1,...,c_n,d_1,...,d_m\right)$. If any two of these sets contain a common basic variable then rename one of them to a new variable. In such a way we will achieve that F-sets (4) will not contain common basic variables. Let us apply procedure of identifying of the variables of the lists $\Gamma_1,\ldots,\Gamma_\delta$ according to the system of equations in variables

$$\begin{cases} c_1^1 = c_1^2 = ... = c_1^\delta \\ \qquad ... \\ c_n^1 = c_n^2 = ... = c_n^\delta \\ d_1^1 = d_1^2 = ... = d_1^\delta \\ \qquad ... \\ d_m^1 = d_m^2 = ... = d_m^\delta \end{cases} \tag{5}$$

and then the procedure of transformation of list of formulas in F-sets to the resulting list. Constructed in such a way F-set Σ is the result of application of the rule B to F-sets (4), if the values of variables $d_1^1, d_2^1, ..., d_m^1$ in the universal solution σ of (5) satisfy the following conditions.

1. They are distinct.
2. They are not included in the formula of Σ.
3. They differ from the values $c_1^1, c_1^2, ..., c_1^\delta, ..., c_n^1, c_n^2, ..., c_n^\delta$ in the solution variables σ.

As in our case, there are no variables $d_1^1, d_1^2, ..., d_1^\delta$, ..., $d_n^1, d_n^2, ..., d_n^\delta$ this rule has the following form:

Procedure 4 of application of Rule B to F-sets. Consider a system of δ F-sets

$$\begin{cases} \Gamma_1, \quad D_1\left(a_1,...,a_k,c_1^1,...,c_n^1\right) \\ \qquad\qquad ... \\ \Gamma_\delta, \quad D_\delta\left(a_1,...,a_k,c_1^\delta,...,c_n^\delta\right) \end{cases}, \tag{6}$$

where $\Gamma_1,...,\Gamma_\delta$ are lists of the formulas of form $D_i\left(a_1,...,a_k,u_1,...,u_n\right)$. If any two of these sets contain a common variable then rename it to a new variable. In such a way it will be achieved that F-sets (6) do not contain common variables. Let us apply procedure of identifying of the variables to the list of $\Gamma_1,...,\Gamma_\delta$ according to the system of equations in the variables

$$\begin{cases} c_1^1 = c_1^2 = ... = c_1^\delta \\ \qquad ... \\ c_n^1 = c_n^2 = ... = c_n^\delta \end{cases} \tag{7}$$

and then procedure of transformation of list of formulas in F-sets must be applied to the resulting list. Constructed in this way F-set Σ is the result of application of rule B to F-sets (6).

Since at the very beginning of the first application of this rule we have renamed all the variables then all variables in the system of equations (7) are different. So contrary to the condition that different variables have different values in the solution of the system does not arise. While solving equations (7) the variables are renamed again to the original names and δ F-sets are combined into a single F-set so the application of the rule B to the formulas considered in this paper can be reduced to a simple unification of δ monomial F-sets in a δ-member F-set. For simplicity, we will use this procedure in such a way: we should rewrite one δ-member F-set instead of δ monomial F-sets.

Theorem [Orevkov, 2003]**.** The formula F is provable in a predicate calculus if and only if an empty F-set □ is derivable in the calculus of favorable sets.

This calculus is given by following **rules A, B and the rule of permutation**

Rule A: Closed F-set is favorable.

Rule B: F-set is favorable if the procedure of rule B applying is successful.

Rule of permutation: A permutation of formulas in a favorable F-set is a favorable F-set.

Now, on the basis of the presented procedures and rules we can formulate the algorithm for searching an inference of a formula of the form (2)

$$\forall a_1,\ldots,a_k \exists (x_1,\ldots,x_n)_{\neq} \left(\mathop{\&}_{i=1}^{\alpha} D_i(a_1,\ldots,a_k,x_1,\ldots,x_n) \right)$$

using the tactic of the inverse method and compare it with the algorithm of proof search using tactics of the resolution method.

Algorithm of Formula Derivation Based on Maslov Inverse Method

Definition 5. F-set is called **empty** if all formulas in it have no variables and are tautological.

Definition 6. F-set is called a **deadlock one** if it includes at least one formula that has no variables and is false or it is neither a tautology nor a contradiction.

Algorithm *Alg* of the formula $\forall a_1,\ldots,a_k \exists (x_1,\ldots,x_n)_{\neq} \left(\mathop{\&}_{i=1}^{\alpha} D_i(a_1,\ldots,a_k,x_1,\ldots,x_n) \right)$ **proof .**

1. F-sets $\{D_1(a_1,\ldots,a_k, x_1,\ldots,x_n), \ldots, D_\alpha(a_1,\ldots,a_k, x_1,\ldots,x_n)\}$ are written down according to the original formula.
2. Apply the procedure for constructing a closed F-set as follows:

   2.1. We are looking for such elementary disjunctions D_j and D_r which contain s-ary predicate symbol P and its negation (with may be different sets of variables or constants as arguments). Let it be an atomic formula $P(t_1, \ldots, t_s)$ and the negation of atomic formula $\neg P(v_1, \ldots, v_s)$. One of the lists $t_1, \ldots, t_s$ or $v_1, \ldots, v_s$ must be a list only of constants. If the other list contains a constant then the two lists at the same position should have the same constant. For example, if one of the lists has the form (a_2, a_4, a_3, a_1) then the other one must contain the constant a_2 or a variable on the first place, the constant a_1 or a variable on the second place, and so on.

   2.2. Solve the system of equations which identifies the lists of variables and constants $t_1,\ldots,t_s$ and $v_1,\ldots,v_s$. If this system has a solution then go to step 2.3. Otherwise go to step 2.1. as follows: look for another version appropriate for formula or circuit considered suitable for the new closed formula.

2.3. Delete repetition of formulas (if they exist) in the resulting F-set.

2.4. Verify whether an empty set is derived. If it is so, the algorithm run stops. Otherwise, if there exist a formula in the F-set to which a rule for constructing a closed F-set can be applied then go to step 3. If a deadlock set is received then go to step 3.

3. Undo one action, and execute step 2. as follows: look for another version appropriate for formula or circuit considered suitable for the new closed formula. If the application of step 2. does not give the new assign values to variables, step 3 is applied once more. Apply rule 3. until the empty set is derived or an opportunity to cancel the action runs out.
4. If the combination for the closure of F-sets is over, and the result is not obtained, then the formula is not derivable

Estimates of the Number of Steps of Algorithm Run

Let $\forall a_1,...,a_k \exists (x_1,...,x_n)_{\neq} \left(\overset{\delta}{\underset{i=1}{\&}} D_i(a_1,...,a_k,x_1,...,x_n) \right)$ be a considered formula, where k is a number of constants; n be the number of variables ($k \geq n$ since values of different variables must differ); D_i has the form $\vee \neg S(a_1,...,a_k) \vee P_{k_i}(\overline{x})$ (where $\vee \neg S\ (\omega)$ means a disjunction of negations of formulas of $S\ (\omega)$) and the solution of which will be considered in the paper.

The execution of every of the following operations is taken for one **step**:

- assignment of a variable value (the solution of an equation of the form $x = a$);
- verifying the graphic coincidence of atomic formulas;
- substitution of a variable value into a formula.

Introduce the following notations:

l is the maximal number of arguments in the atomic formula;

s *is* the number of atomic formulas in $S\ (\omega)$;

γ_l *is* the number of i-ary predicates in $S\ (\omega)$;

γ_i^0 *is* the number of i-ary anomic formulas in $S\ (\omega)$ without negation;

$\gamma_i^{\neg}$ *is* the number of i-ary anomic formulas in $S\ (\omega)$ with negation.

Theorem 1. (The lower bound of the algorithm run.) The number of steps of the proof of the formula $\forall a_1,...,a_k \exists (x_1,...,x_n)_{\neq} \left(\overset{\delta}{\underset{i=1}{\&}} D_i(a_1,...,a_k,x_1,...,x_n) \right)$ with the use of an algorithm based on the tactics of the inverse method is not less then sl.

Proof. The lower bound is achieved when the number of variables is equal to the maximal number of arguments in the atomic formula and the answer is obtained by solving a system of l equations. Every elementary disjunction contains s constant formulas. Hence we have not less then sl steps. ■

Let $\tilde{s}$ be the maximal number of occurrences of the same predicate (only without the negations, or only with the negations) in the class description.

Theorem 2. (The upper bound of the algorithm run.) The number of steps of the proof of the formula $\forall a_1,...,a_k \exists (x_1,...,x_n)_{\neq} \left(\&_{i=1}^{\delta} D_i(a_1,...,a_k,x_1,...,x_n) \right)$ with the use of an algorithm based on the tactics of the inverse method is $O(\delta\, \tilde{s}^{\,\delta} l)$.

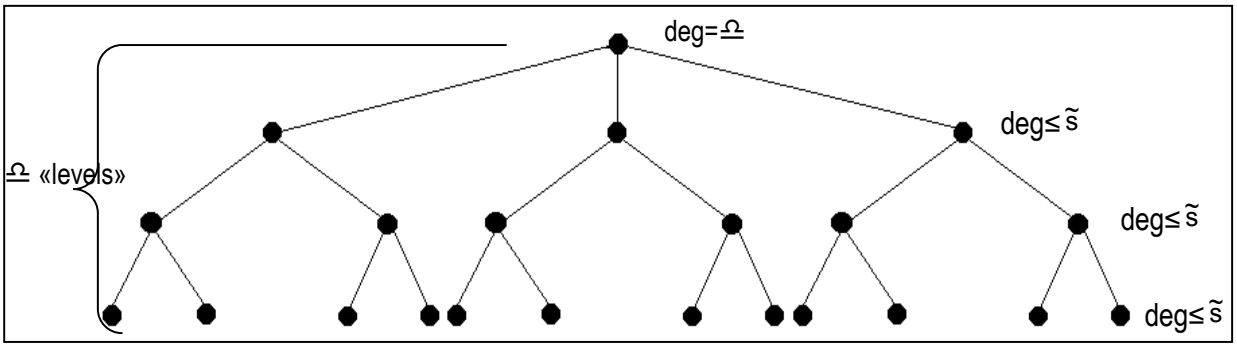
Proof. The first F-set under consideration is a list of the form $D_1(a_1,...,a_k, x_1,...,x_n)$, ..., $D_\alpha(a_1,...,a_k, x_1,...,x_n)$, where every $D_j(a_1,...,a_k,x_1,...,x_n)$ is an elementary disjunction of the type $\vee \neg S(a_1,...,a_k) \vee P_{k_i}(\bar{x})$ with the same constant formulas.

Let the atomic formulas in $S(a_1,...,a_k)$ are ordered by groups with the same predicate according decreasing of number of arguments. In every group predicates without negation precedes ones with negation.

If $p(l)$ denotes some atomic formula with constant arguments, $p(n)$ denotes some atomic formula with variable arguments then the list has the following structure:

$$\left.\begin{array}{l} \vee_{i_1=1}^{\gamma_l^0} p_{i_1}(\bar{a}_{i_1}) \vee \vee_{i_2=1}^{\gamma_l^-} \neg p_{i_2}(\bar{a}_{i_2}) \vee \vee_{i_3=1}^{\gamma_{l-1}^0} p_{i_3}(\bar{a}_{i_3}) \vee \vee_{i_4=1}^{\gamma_{l-1}^-} \neg p_{i_4}(\bar{a}_{i_4}) ... \vee \vee_{i_{2l-1}=1}^{\gamma_1^0} p_{i_{2l-1}}(\bar{a}_{i_{2l-1}}) \vee \vee_{i_{2l}=1}^{\gamma_1^-} \neg p_{i_{2l}}(\bar{a}_{i_{2l}}) \vee p_1(\bar{x}_1) \\ \ldots \\ \vee_{i_1=1}^{\gamma_l^0} p_{i_1}(\bar{a}_{i_1}) \vee \vee_{i_2=1}^{\gamma_l^-} \neg p_{i_2}(\bar{a}_{i_2}) \vee \vee_{i_3=1}^{\gamma_{l-1}^0} p_{i_3}(\bar{a}_{i_3}) \vee \vee_{i_4=1}^{\gamma_{l-1}^-} \neg p_{i_4}(\bar{a}_{i_4}) ... \vee \vee_{i_{2l-1}=1}^{\gamma_1^0} p_{i_{2l-1}}(\bar{a}_{i_{2l-1}}) \vee \vee_{i_{2l}=1}^{\gamma_1^-} \neg p_{i_{2l}}(\bar{a}_{i_{2l}}) \vee p_\delta(\bar{x}_\delta) \end{array}\right\}^{\delta} \text{ items}$$

The upper bound is reached, if the answer whether a formula is derivable is received at the last step of the algorithm. That is if you are to avoid all the branches of a tree the following form:



The number of levels in the tree equals to δ as it is necessary to assign values to variables in δ formulas. δ edges leaves from the upper node. Not more than $\tilde{s}$ edges leave from each node of the next levels. The number of nodes in the tree is not more than $\delta(\tilde{s}^{\,\delta}-1)$. Every node corresponds to a system of not more than l equations every of which may be verified not more than in l steps. That is the upper bound is $O(\delta\, \tilde{s}^{\,\delta} l)$, where δ is the number of disjunctive terms in the original formula, $\tilde{s}+1$ is the total number of atomic formulas in every disjunct, l is the largest number of arguments in the atomic formulas. ■

Conclusion

Thus, the asymptotic estimation of the number of steps of the described algorithm *Alg* using the tactics of the inverse method are as follows

$$O(sl) \le T(Alg) \le O(\delta\, \tilde{s}^{\,\delta} l),$$

where *s+1* is the total number of atomic formulas in disjuncts, δ is the number of disjunctive terms in the original formula, *l* - the largest number of arguments in the atomic formulas and $\tilde{s}$ is the maximal number of occurrences of the same predicate (only without the negations, or only with the negations) in the class description.

In [Kosovskaya, 2007] the following upper bound for solving the pattern recognition algorithm that uses the tactics of the method of resolution

$$O(D\,\tilde{s}^{\tilde{a}})$$

where D is the number of disjuncts in the descriptions of the classes used in problem solving,

$\tilde{s}$ is the maximum number of occurrences of the same predicate (only without the negations, or only with the negations) in the class description, $\tilde{a}$ is the maximum number of occurrences of atomic formulas of the elementary conjunctions that make up the class definitions was obtained.

As we can see, these estimates coincide to within a constant factor.

Bibliography

[Kosovskaya, Timofeev, 1985] T.M. Kosovskaya, A.B. Timofeev. About one new approach to the formation of logical decision rules in pattern recognition problems // Vestnik LGU, 1985, №8, pp. 22-29. (In Russian)

[Kosovskaya, 2007] Kosovskaya T.M. Proofs of the number of steps bounds for solving of some pattern recognition problems with logical description // Vestn. S- Petersburg University, Ser. 1, 2007. Ed. (2), pp. 82-90. (In Russian)

[Kosovskaya, 2011] Kosovskaya T. Discrete Artificial Intelligence Problems and Number of Steps of their Solution // International Journal "Information Theories and Applications", Vol. 18, Number 1, 2011. P. 93 – 99.

[Orevkov, 2003] Orevkov V.P. The inverse method of logical derivation // In Adamenko A. Kuchukov A. Programming Logic and Visual Prolog. St. Petersburg, "`BHV-Petersburg"'. (2003), pp.. 952-965. (in Russian)

Authors' Information

Tatiana Kosovskaya *– Dr., Senior researcher, St.Petersburg Institute of Informatics and Automation of Russian Academy of Science, 14 line, 39, St.Petersburg, 199178, Russia; Professor of St.Petersburg State Marine Technical University, Lotsmanskaya ul., 3, St.Petersburg, 190008, Russia; Professor of St.Petersburg State University, University av., 28, Stary Petergof, St.Petersburg, 198504, Russia, e-mail: kosov@NK1022.spb.edu*

Major Fields of Scientific Research: Logical approach to artificial intelligence problems, theory of complexity of algorithms.

Nina Petukhova *– PHD student, St.Petersburg State Marine Technical University, Lotsmanskaya ul., 3, St.Petersburg, 190008, Russia, e-mail: ninka_mat@mail.ru*

Major Fields of Scientific Research: Logical approach to Artificial Intelligence problems.

The paper is published with financial support of the project ITHEA XXI of the Institute of Information Theories and Applications FOI ITHEA (www.ithea.org) and the Association of Developers and Users of Intelligent Systems ADUIS Ukraine (www.aduis.com.ua).