Extraction of a Common up to the Names of Arguments Sub-Formula of Two Elementary Conjunctions and Some AI Problems

Tatiana, Kosovskaya
St. Petersburg State University
St. Petersburg, Russia
e-mail: kosovtm1@gmail.com

ABSTRACT

The paper is devoted to the use of the notion of a common up to the names of arguments sub-formula of two elementary conjunctions of predicate atomic formulas. It allows to construct a level description of a set of goal formulas and to decrease the number of proof steps for one of NP-complete problems. Logic-predicate networks which may change its configuration (the number of layers and the number of cells in the layer) during the process of training are described. The problem of multi-agent description when every agent can describe only a part of an object (these parts are intersected) but every agent gives its own names to the elements of the whole object may be solved with the use of the notion of such a common sub-formula.

Keywords

Predicate calculus, NP-completeness, level description, predicate network, multi-agent description.

1. INTRODUCTION

In the 70-th of the XX century many authors (see for example [2]) offered to use predicate calculus and automatic proof of a theorem for AI problems solving. Until the notion of NP-complete problem (in particular, described in [1]) was not widely adopted, such an approach seemed to be very convenient.

In 2007 the author proved NP-completeness of a series of AI problems formalized with the help of predicate calculus formulas [3], proved upper bounds for number of steps of algorithms solving these problems [6], and offered a level description of goal formulas for decreasing the number of proof steps [4]. Such a level description is based on the extraction of a common up to the names of its arguments sub-formula of the set of elementary conjunctions of atomic predicate formulas. These sub-formulas define generalized characteristics of an object.

Extraction of such sub-formulas allows to construct logic-predicate networks [7] which may change its configuration (the number of layers and the number of cells in the layer) during the process of training.

Also extraction of these sub-formulas may be an instrument for solving a multi-agent description problem solving when every agent can describe only a part of an object (these parts are intersected) but every agent gives its own names to the elements of the whole object [8].

2. LOGIC-PREDICATE APPROACH TO AI PROBLEMS

Let an investigated object be represented as a set of its elements $\omega = \{\omega_1, \ldots, \omega_t\}$ and be characterized by predicates $p_1, \ldots, p_n$ which define some properties of the elements or relations between them. The description $S(\omega_1, \ldots, \omega_t)$ of the object ω is a set of all constant literals with predicates $p_1, \ldots, p_n$ which are valid on ω. There is a set of goal formulas $A_1(x_1, \ldots, x_{m_1}), \ldots, A_K(x_1, \ldots, x_{m_K})$ in the form of elementary conjunctions.

The solution of many Artificial Intelligence problems may be reduced to the proof of a series of formulas (for $k = 1, \ldots, K$) in the form

$$S(\omega) \Rightarrow \exists(x_1, \ldots, x_{m_k})_{\neq}\ A_k(x_1, \ldots, x_{m_k}).^1 \qquad (1)$$

The verification problem of such a formula is NP-complete [3]. The upper bound of number of steps for an exhaustive algorithm [6] is

$$O(t^{m_k}).$$

The upper bound of the number of steps for an algorithm based on the derivation in sequential calculus or on the use of resolution method [6] is

$$O({s_k}^{a_k}),$$

where s_k is the maximal number of atomic formulas in $S(\omega)$ with the same predicate symbol having occurrences in $A_k(x_1, \ldots, x_m)$, a_k is the number of atomic formulas in the elementary conjunction $A_k(x_1, \ldots, x_m)$ [6].

Note that both a logical algorithm and an exhaustive algorithm allow not only to prove that there exist values for variables satisfying the formula $A_k(x_1, \ldots, x_m)$ but to find these values. So an algorithm verifying the formula (1) allows to solve the problem "what are the different values of $x_1, \ldots, x_m$ from ω that satisfy the formula $A_k(x_1, \ldots, x_m)$

$$S(\omega) \Rightarrow ?(x_1, \ldots, x_m)_{\neq} A_k(x_1, \ldots, x_m). \qquad (2)$$

This problem is NP-hard and its solving algorithms have the same upper bounds as for the problem (1).

[1]The notation $\exists(x_1, \ldots, x_m)_{\neq}\ P$ is used for the formula $\exists x_1, \ldots, x_m(\&_{i=1}^{m-1} \&_{j=i+1}^{m}\ x_i \neq x_j\ \&\ P)$.

3. COMMON UP TO THE NAMES OF ARGUMENT SUB-FORMULA

Definition 1. *Elementary conjunctions P and Q are called isomorphic if there is an elementary conjunction R and substitutions $\lambda_{R,P}$ and $\lambda_{R,Q}$ of the arguments of P and Q, respectively, instead of the variables in R such that the results of these substitutions coincide up to the order of literals.*

The substitutions $\lambda_{R,P}$ and $\lambda_{R,Q}$ are called unifiers of R with P and Q, respectively.

Definition 2. *Elementary conjunction C is called a common up to the names of arguments sub-formula of two elementary conjunctions A and B if it is isomorphic to some sub-formulas A' and B' of A and B, respectively.*

For example, let $A(x,y,z) = p_1(x)\&p_1(y)\& p_1(z)\&p_2(x,y)\&p_3(x,z)$, $B(x,y,z) = p_1(x)\&p_1(y)\& p_1(z)\&p_2(x,z)\&p_3(x,z)$. If the formula $P(u,v) = p_1(u)\&p_1(v)\&p_2(u,v)$ is their common sub-formula? Is the formula $P(u,v)$ their common up to the names of variables sub-formula with the unifiers $\lambda_{P,A} = \left|{}^{u\ v}_{x\ y}\right.$ and $\lambda_{P,B} = \left|{}^{u\ v}_{x\ z}\right.$ because $P(x,y) = p_1(x)\&p_1(y)\&p_2(x,y)$ is a sub-formula of $A(x,y,z)$ and $P(x,z) = p_1(x)\&p_1(z)\&p_2(x,z)$ is a sub-formula of $B(x,y,z)$.

An algorithm of extraction of a maximal (having a maximal number of literals) common up to the names of arguments sub-formula C of two elementary conjunctions A and B and determining the unifiers $\lambda_{C,A'}$ and $\lambda_{C,B'}$ is described in [9]. The number of steps of this algorithm is $O(N_A^{N_A} N_B^{N_B})$, where N_A and N_B are the numbers of literals in A and B respectively. The minimal number of steps of this algorithm is $O((N_A N_B)^2)$, the middle estimate is $O((N_A N_B)^{1/2\log(N_A N_B)})$.

4. LEVEL DESCRIPTION

Level description of goal formulas allows essentially to decrease the number of steps for an algorithm solving the problems (1) and (2). This notion is based on the extraction of common up to the names of arguments sub-formulas $P_i^1(\overline{y}_i^1)$ $(i = 1,\dots,n_1)$ of goal formulas $A_k(x_1,\dots,x_m)$ $(k = 1,\dots,K)$ with "small complexity". Simultaneously we find unifiers of $P_i^1(\overline{y}_i^1)$ and sub-formulas of $A_k(x_1,\dots,x_m)$.

$$\left\{\begin{array}{ccc} & A_k^L(\overline{x}^L) & \\ p_1^1(y_1^1) & \Leftrightarrow & P_1^1(\overline{y}_1^1) \\ & \vdots & \\ p_{n_1}^1(y_{n_1}^1) & \Leftrightarrow & P_{n_1}^1(\overline{y}_{n_1}^1) \\ & \vdots & \\ p_i^l(y_i^l) & \Leftrightarrow & P_i^l(\overline{y}_i^l) \\ & \vdots & \\ p_{n_L}^L(y_{n_L}^L) & \Leftrightarrow & P_{n_L}^L(\overline{y}_{n_L}^L) \end{array}\right. . \qquad (3)$$

Introduce new first-level predicates p_i^1 with new first-level arguments y_i^1 for lists of initial variables defined by the equivalences $p_i^1(y_i^1) \Leftrightarrow P_i^1(\overline{y}_i^1)$. Change in $A_k(x_1,\dots,x_m)$ every occurrence of sub-formula isomorphic to $P_i^1(\overline{y}_i^1)$ by $p_i^1(y_i^1)$. The obtained formulas denote by $A_1^1(\overline{x}_1^1),\dots,A_K^1(\overline{x}_K^1)$, where $\overline{x}_k^1$ is the list of initial variables (may be not all) and the first-level variables. Repeat this procedure with $A_1^l(\overline{x}_1^l),\dots,A_K^l(\overline{x}_K^l)$ for $l = 1,\dots,L$ and obtain an L-level description (3).

The solution of the problem in the form (1) with the use of a level description is decomposed on the sequential $(l = 1,\dots,L)$ implementation of the items 1 and 2.

1. For every i $(i = 1,\dots,n_l)$ check[2] $S^{l-1}(\omega) \Rightarrow \exists\overline{y}_{j\,\neq}^1 P_j^1(\overline{y}_j^1)$ and find all lists $\overline{\tau}_j^l$ of previous levels constants for the values of the variable list $\overline{y}_j^1$ such that $S^{l-1}(\omega) \Rightarrow P_j^1(\overline{\tau}_j^1)$;

2. Add all constant atomic l-level formulas in the form $p_j^l(\overline{\tau}_j^l)$ ($\overline{\tau}_j^l$ were received in the first item) to $S^{l-1}(\omega)$ and obtain $S^l(\omega)$.

At last check $S^L(a_1,\dots,a_t) \Rightarrow \exists\overline{x}_{k\,\neq}^L A_k^L(\overline{y}_k^L)$.

Such a description allows essentially to decrease the number of steps needed for solving the problem (1) or (2) even with the use of only first-level predicates.

Number of steps for an exhaustive algorithm decreases from $O(t^{m_k})$ to $O(n_1 \cdot t^r + t^{\delta_k^1 + n_1})$, where r is a maximal number of arguments in the 1-st level predicates, n_1 is the number of such predicates, δ_k^1 is the number of initial variables which are presented in $A_k(\overline{x}_k)$ and are absent in $A_k^1(\overline{x}_k^1)$. Number of steps for an algorithm based on the derivation in sequential calculus or on the use of resolution method decreases from $O({s_k}^{a_k})$ to $O({s^1}^{a_k^1} + \sum_{j=1}^{n_1} s^{\rho_j^1})$, where a_k and a_k^1 are maximal numbers of literals in $A_k(\overline{x}_k)$ and $A_k^1(\overline{x}_k^1)$, respectively, s s^1 are the numbers of literals in $S(\omega)$ and $S^1(\omega)$, respectively, ρ_j^1 is the numbers of literals in $P_i^1(\overline{y}_i^1)$.

4.1 Construction of a level description

Algorithm for extraction of a maximal common up to the names of arguments sub-formula C of two elementary conjunctions A and B and determining the unifiers $\lambda_{C,A'}$ and $\lambda_{C,B'}$ allows to construct a level description for a set of goal elementary conjunctions. Essential difference between maximal common sub-formulas and sub-formulas in the level description consists in the fact that in the level description it is needed to extract sub-formulas with "small complexity" but not a maximal one.

1. For every pair of elementary conjunctions $A_i(\overline{x_i})$ and $A_j(\overline{x_j})$ extract their maximal common up to the names of arguments sub-formula $Q_{ij}^1(\overline{x_{ij}^1})$ and find their unifiers.

l $(l = 2,\dots,L')$.[3] For every pair of not isomorphic elementary conjunctions $Q_{i_1\dots i_{2^{l-1}}}^{l-1}(\overline{x_{i_1\dots i_{2^{l-1}}}})$ and $Q_{j_1\dots j_{2^{l-1}}}^{l-1}(\overline{x_{j_1\dots j_{2^{l-1}}}})$ extract their maximal common up to the names of arguments sub-formula $Q_{i_1\dots i_{2^{l-1}} j_1\dots j_{2^{l-1}}}^{l}(\overline{x_{i_1\dots i_{2^{l-1}} j_1\dots j_{2^{l-1}}}})$ and find their unifiers.

$L' + l$ $(l = 1,\dots,L-1)$. Among all the remained (among

[2] $S^0(\omega) = S(\omega)$

[3] The number L' is finite as for every l the length of the extracted sub-formula decreases.

all for $l = 1$) extracted sub-formulas find such that has no extracted sub-formulas and denote them by means of $P_i^l(\overline{y_i^l})$ $(i = 1, \ldots, n_l)$. Substitute the l-level atomic formulas $p_i^l(y_i^l)$ defined by the equalities $p_i^l(y_i^l) \Leftrightarrow P_i^l(\overline{y}_i^l)$ instead of $P_i^l(\overline{y_i^l})$ into the extracted sub-formulas and into $A_k^{l-1}(\overline{x}^{l-1})$ and rewrite all unifiers for the l-level variables. Delete formulas $P_i^l(\overline{y_i^l})$ from the set of the extracted formulas.

Let N be the maximal number of literals in $A_k(\overline{x}_k)$ $(k = 1, \ldots, K)$. The upper bound of this algorithm number of steps is the following.

The extraction maximal common up to the names of arguments sub-formula for one pair of formulas and finding their unifiers in the item 1 requires $O(N^N N^N) = O(N^{2N})$ steps. Item 1 requires $O(\frac{K(K+1)}{2} N^{2N})$ steps.

Item l $(l = 2, \ldots, L')$ requires (in practice essentially less) $O(\frac{n_l(n_l+1)}{2}(N-l+1)^{2(N-l+1)})$ steps. The sum of these numbers is $O(\frac{K(K+1)}{2} N^{2N+1})$.

Items $L' + l$ $(l = 1, \ldots, L-1)$ require more less number of steps.

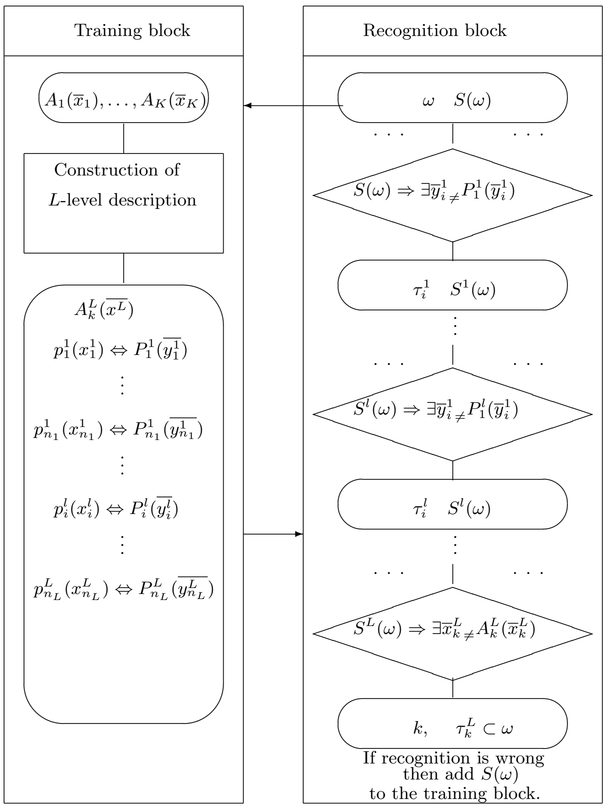
5. LOGIC-PREDICATE NETWORK

A logic-predicate network consists of two blocks: a training block and a recognition block. Let a training set of objects $\omega^1, \ldots, \omega^K$ be given to form an initial variant of the network training block. Replace every constant ω_j^k in $S(\omega^k)$ by a variable x_j^k $(k = 1, \ldots, K,\ j = 1, \ldots, t^k)$ and substitute the sign & between the atomic formulas. Initial goal formulas $A_1(\overline{x}_1), \ldots, A_K(\overline{x}_K)$ are obtained. Construct a level description for these goal formulas. The first approximation to the recognition block is formed.

If after the recognition block run an object is not recognized nor has wrong identification then it is possible to train anew the network. The description of the wrong object must be added to the input set of the training block. The training block extracts common sub-formulas of this description and previously received formulas forming the recognition block. Some sub-formulas in the level description would be changed. Then the recognition block is reconstructed. The scheme of the network is presented in Figure 1.

6. MULTI-AGENT DESCRIPTION OF AN OBJECT

Let an investigated object be represented as a set of its elements $\omega = \{\omega_1, \ldots, \omega_t\}$ and is characterized by predicates $p_1, \ldots, p_n$, each of which is defined on the elements of ω and gives properties of these elements and relations between them. Information (description) $I(\omega_1, \ldots, \omega_t)$ of an object ω is an elementary conjunction of atomic formulas with predicates $p_1, \ldots, p_n$ and arguments from ω. There are m agents $a_1, \ldots, a_m$ which can detect some values for some predicates of some elements of ω. The agent a_j does not know the true number of the elements in ω and suppose that it deals with the object $\omega^j = \{\omega_1^j, \ldots, \omega_{tj}^j\}$. That is the agent a_j has the information $I^j(\omega_1^j, \ldots, \omega_{tj}^j)$ in the form of elementary conjunction of atomic formulas. It is required to construct the description $I(\omega_1, \ldots, \omega_t)$ of ω.



Training block
Recognition block
$A_1(\overline{x}_1), \ldots, A_K(\overline{x}_K)$
Construction of L-level description
$A_k^L(\overline{x^L})$
$p_1^1(x_1^1) \Leftrightarrow P_1^1(\overline{y_1^1})$
$p_{n_1}^1(x_{n_1}^1) \Leftrightarrow P_{n_1}^1(\overline{y_{n_1}^1})$
$p_i^l(x_i^l) \Leftrightarrow P_i^l(\overline{y_i^l})$
$p_{n_L}^L(x_{n_L}^L) \Leftrightarrow P_{n_L}^L(\overline{y_{n_L}^L})$
$\omega \quad S(\omega)$
$S(\omega) \Rightarrow \exists \overline{y}_{i\neq}^1 P_1^1(\overline{y}_i^1)$
$\tau_i^1 \quad S^1(\omega)$
$S^l(\omega) \Rightarrow \exists \overline{y}_{i\neq}^1 P_1^l(\overline{y}_i^1)$
$\tau_i^l \quad S^l(\omega)$
$S^L(\omega) \Rightarrow \exists \overline{x}_{k\neq}^L A_k^L(\overline{x}_k^L)$
$k, \quad \tau_k^L \subset \omega$
If recognition is wrong then add $S(\omega)$ to the training block.

Figure 1. The scheme of logic-predicate network.

As every agent uses its own notifications for the names of the object elements, it is needed to find all common up to the names of arguments sub-formulas C_{ij} of the informations $I^i(\omega_1^i, \ldots, \omega_{ti}^j)$ and $I^j(\omega_1^j, \ldots, \omega_{tj}^j)$ $(i \neq j)$ and their unifiers.

6.1 Algorithm of multi-agent description

In this subsection the arguments of informations will be omitted. Let every agent a_j have information I^j about the described object ω $(j = 1, \ldots, m)$. To construct a description of ω the following algorithm is offered.

1. Change all constants in $I^1, \ldots, I^m$ by variables in such a way that different constants are changed by different variables and the names of variables in I^i and I^j $(i \neq j)$ do not coincide. Obtain $I'_1, \ldots, I'_m$.

2. For every pair of elementary conjunctions I'_i and I'_j $(i = 1, \ldots, m-1,\ j = i+1, \ldots, m)$ find a maximal common up to the names of variables of I'_i and I'_j sub-formula C_{ij} and unifiers $\lambda_{i,ij}$ and $\lambda_{j,ij}$. Every argument of C_{ij} has a unique name.

3. For every pair i and j $(i > j)$ check if I'_i and I'_j contain a contradictory pair of atomic formulas or two sub-formulas which cannot be satisfiable simultaneously. If such a contradiction is established then delete from C_{ij} atomic formulas containing the variables which are in the contradictory sub-formulas. Change the unifiers by means of elimination of these variables.

4. For every i identify the variables in C_{ij} $(i \neq j)$ which are substituted in I'_i and I'_j instead of the same variable. The names of the identified variables are changed in unifiers by the same name.
5. With the use of the unifiers obtained in items 2 – 4 change the names of variables in $I'_1, \ldots, I'_m$. Obtain $I''_1, \ldots, I''_m$.

6. Write down the conjunction $I_1'' \& \ldots \& I_m''$ and delete the repeating atomic formulas.

6.2 Upper bound of the number of steps

To estimate the number of the algorithm run steps we estimate every item of the algorithm.

1. Item 1 requires not more than $\sum_{j=1}^{m} ||I_j||$ "steps".

2. Item 2 requires $O({t_i}^{t_j} \cdot 2^{||I_i||})$ "steps" for an exhaustive algorithm and $O({s_i}^{||I_j||} \cdot ||I_i||^3)$ "steps" for an algorithm based on the derivation in the predicate calculus[4]. It is needed to summarize the above estimates for $i = 1, ..., m1,\ j = i+1, ..., m$. So we have $O(t^t \cdot ||I|| \cdot m^2)$ "steps" for an exhaustive algorithm and $O(s^{||I||} \cdot ||I||^3 \cdot m^2)$ "steps" for an algorithm based on the derivation in the predicate calculus. Here t and $||I||$ are respectively the maximal numbers of variables and atomic formulas in I^j $(j = 1, ..., m)$.

3. Consistency checking of the formula I_i' requires $||I_i||$ "steps". This item of the algorithm requires not more than $\sum_{i=1}^{m}(m-i)||I_i||$ "steps" that is $O(m^2||I||)$ "steps".

4. For every i identifying of the variables in C_{ij} $(j > i)$ consists in the comparison of the replaced parts of the unifiers. It requires not more than $(m-i)t_i$ "steps". Summarizing it for $i = 1, , m$ we have not more than $\sum_{i=1}^{m}(m-i)t_i = O(m^2 t)$ "steps".

5. The number of "steps" required for changing of the names of variables in $I_1, \cdots, I_m$ is linear under $\sum_{j=1}^{m} ||I_j|| = O(m||I||)$.

6. The number of "steps" required for deleting of the repeated conjunctive terms is not more than $\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} ||I_i|| \cdot ||I_j|| O(m^2||I||^2)$.

The number of the algorithm run "steps" is $O(t^t \cdot ||I|| \cdot m^2)$ for an exhaustive algorithm and $O(s^{||I||} \cdot ||I||^3 \cdot m^2)$ for an algorithm based on the derivation in the predicate calculus. The analysis of the received estimation shows that the main contribution to it is made by the summarized number of maximal common up to the names of variables sub-formulas extractions (item 2).

7. CONCLUSION

Applications of the notion of a common up to the names of arguments sub-formula of two elementary conjunctions of predicate formulas are described in the paper. This notion allows to construct algorithms for solving a series of AI problems. In particular, it is possible essentially to decrease the number of steps of the NP-complete problem (1) by means of a level description construction if we deal with the fixed set of goal formulas.

The level description gives the possibility to create a self-training logic-predicate network with predicate formulas in the cells. Very interesting is the possibility of a weighted predicates and variables use. Probably these weights must vary in dependence of "valid" or "wrong" recognition as it is usual for traditional neuron networks.

[4] i and j may be chosen in such a way that $t_j \leq t_i$ and $||I_j|| \leq s_i$

An algorithm solving a rather complicated problem of multi-agent description of a complex object in terms of predicate calculus language with the condition that different agents may give different names to the same elements of the object is presented in the paper.

For every algorithm upper bound of the step numbers is received. The analysis of these estimations allows to formulate restrictions upon the initial predicates for decreasing the practical time of the algorithm run. For example, if we deal with a great number of initial predicates each of which has very amount of occurrences in the object description then the practical time of the algorithm run decreases.

REFERENCES

[1] M. Garey, D. Johnson, "Computers and Intractability: A Guide to the Theory of NP-Completeness." Freeman Press. San Francisco, 1979, 340 p.

[2] N. Nilson, "Problem-solving methods in artificial intelligence." New York, McGRAW-HILL BOOK COMPANY Press. Publ., 1971, 280 p.

[3] T. Kossovskaya, "Proofs of the number of steps bounds for solving of some pattern recognition problems with logical description", *Vestnik of Saint-Petersburg University. Series 1. Mathematics, mechanics, astronomy*, pp. 82-90, 2007, issue 4. (In Russian)

[4] T. Kossovskaya, "Level descriptions of classes for decreasing step number of pattern recognition problem solving described by predicate calculus formulas", *Vestnik of Saint-Petersburg University. Series 10. Applied mathematics. Computer science. Control processes.*, pp. 64-72, 2008, issue 1. (In Russian)

[5] T. Kossovskaya, "Partial deduction of predicate formula as an instrument for recognition of an object with incomplete description", *Vestnik of Saint-Petersburg University. Series 10. Applied mathematics. Computer science. Control processes.*, pp. 45-55, 2009, issue 3. (In Russian)

[6] T. Kosovskaya, "Some artificial intelligence problems permitting formalization by means of predicate calculus language and upper bounds of their solution steps", *SPIIRAS Proceedings*, pp. 58-75, 2010. no. 14. (In Russian)

[7] T. Kosovskaya, "Self-modificated predicate networks", *International Journal on Information Theory and Applications*, pp. 245-257, 2015, Vol. 22, No 3.

[8] T. Kosovskaya. "Multi-agent description of an object by means of a predicate calculus language", *International Journal on Information Theories & Applications*, Vol. 23, No 4. 2016. P. 338 346.

[9] D.A. Petrov, "Algorithms of extraction of a maximal common up to the names of variables predicate formulas and their implementation", *Proc. of the 9-th Conference "Information Technology in Management". St.Petersburg*, pp. 97-102, 2016. (In Russian)