The solution algorithms for problems on the minimal vertex cover in networks and the minimal cover in Boolean matrixes

Sergey Listrovoy[1], Sergii Minukhin[2]

[1] **Ukrainian State Academy of Railway Transport, 61050, Kharkov, Ukraine**

[2] **Kharkiv National University of Economics, 61166, Kharkov, Ukraine**

Abstract
The authors propose approximate algorithms for solving the problem of the minimal vertex cover of arbitrary graphs and the problem of minimal cover on the basis of their reduction, respectively, to the problems of quadratic and nonlinear Boolean programming, their specificity allowing to construct algorithms with time complexity not exceeding $\boldsymbol{O}$ (mn^2), where in the case of solving the problem of minimal vertex cover of arbitrary graphs n is the number of vertices in the graph, m is the number of edges in the graph, and in the case of solving the problem of minimal cover n is the number of columns in the Boolean matrix, m is the number of rows in Boolean matrix.
Keywords: vertex covers in graph, the minimal cover, Boolean matrix, time complexity.

1. Introduction

Problem on the minimal vertex cover (MVCP) and problem on the minimal cover (MCP) have considerable value and widely applied in the theory of complex systems, computing systems and networks building [1] and in development of their software and mathematical support, and also for planning of resources distribution in GRID. Besides, MCP has wide application in GRID systems for diagnostics of systems and networks [1], in arrangement of service stations [1, 3], in information retrieval systems, for assigning of crews in transportation industry [1, 3, 5, 6], in designing of chips [6] and conveyor lines etc.

Upon that, the basic requirement to solution algorithms for those problems is high efficiency of a solution and ensuring minimum possible errors of these solutions.

The problem of determination of independent maximum sets or vertex covers can be solved, for example, with consecutive search of independent sets with simultaneous check of each set on maximal (the last is executed by adding to investigated set of an additional vertex which does not belong to it, with subsequent clearing up whether the independence remains) and storing of maximum sets. However this mode becomes rather cumbersome with magnification of vertex number. Algorithms of Bron and Carbosh [1] are built on the basis of advancement of this procedure. As it is shown in study [3], the problem of a vertex cover is difficult to solve and effective algorithms of its solution for arbitrary networks are unknown. For bipartite graphs on the basis of Hopkroft and Carp algorithms (with postorder tree search) there are developed the methods [3] which allow finding a minimum vertex cover and maximum independent set of vertices in an arbitrary bipartite graph H = <X, Y, E> in time $\boldsymbol{O}((m+n)\sqrt{n})$, where n = |X∪Y| and m = |E|. Polynomial algorithms of stability number determination have been received for perfect graphs, i.e. those graphs for any generated subgraph of which the chromatic number is equal to clique number. The algorithm of stability number determination of a graph [5] is based on a method of ellipsoids and uses procedure of graph matrixes separating. However in terms of computing this algorithm has a number of the essential shortages which prevent its use in practice. As shown in study [5] it is practically impossible to receive a correct solution when number of vertices in a graph is more than 10. All known determined exact algorithms of MCP solution [1–4, 6, 7] have exponential complexity.

It is necessary to note that application of known greedy algorithms of Greedy-Set-Cover type [8], as shown in [8–11], can give an error on the order of $\boldsymbol{O}$ *(log n)*. P. Slavik [12] has shown that the error of greedy algorithm approximation for a problem solution on the minimal cover makes $\ln n - \ln \ln n + \boldsymbol{O}(1)$. Other algorithms of this type [13] are built on the harmonic series analysis, $H_k = \sum_{i=1}^{k} \frac{1}{i} + c$, where k is a potency of the set defining a cover, c is a some constant for which the estimations of approximation [13] specifying magnitude of a constant *c* are received. Thus, the most comprehensible for the comparative analysis of available theoretical outcomes of greedy algorithms approximation efficiency is estimation ***O*** (log *n*) which is proportional to magnitude of the logarithm of an amount of columns (vertices) in matrix ***B***. Thereat, these algorithms are hardly parallelized and therefore, their application as planning means in a distributed environment, for example, in GRID systems [2], is considered as inconvenient.

The purpose of the study is development of a method having whenever possible minimum complexity and error for MVCP (for arbitrary graphs) and MCP solution.

2. Problem statement and solution

At first let's consider formal characterization of the problem about a vertex cover in arbitrary graphs, for this purpose let's introduce cone timept of disassembly and assembly of the graph from base elements. Base elements $\{l_i\}$ of a graph G (X, E) with set of vertices X and set of edges E are referred to as vertices of the graph $\{i \in X\}$ together with edges incident with them. Let's say that base elements of the graph l_i and l_j can be united if they have common vertices or edges; then an association of two base components $l_i \cup l_j$ will be referred by us to as a subgraph of the graph G received by association of the edges with the same name and vertices in base elements l_i and l_j and junction of all vertices in an incorporated structure according to connections between vertices of the initial graph G. It is clear, that this operation is commutative, i.e. $l_i \cup l_j = l_j \cup l_i$. Association of components is probable if there are common edges or vertices in united components. Let's name procedure of removal of base components from a graph as disassembly of the graph G. If in the course of disassembly of the graph the base component l_i is deleted, let's designate the remained subgraph as G_i, and if two components l_i and l_j are deleted, let's designate such a subgraph as G_{ij}, etc. If a graph has n vertices, a number of base components of which the graph G consists is equal to n. For example, columns G shown on on fig. 1 consists of 5 vertices and has 5 base components.

Obviously, it is possible to consider graph G as association of all components l_i , i.e. $G = \bigcup_1^n l_i$.

Let's put in correspondence to each vertex in the graph a variable x_i then each component l_i can be described in the form of product $x_i \sum_{q \in m_{iq}} x_q$ where m_{iq} is a set of vertices adjacent to vertex i in a component l_i . If vertex i is included in a vertex cover, we will consider that a variable x_i =0. Multiplication by x_i = 0 in the product $x_i \sum_{q \in m_{iq}} x_q$ converts to zero all edges which are covered by the vertex i in a component l_i .

Let's consider the sum

$$x_1 \sum_{q \in m_{iq}} x_q + x_2 \sum_{q \in m_{iq}} x_q + \ldots + x_n \sum_{q \in m_{iq}} x_q. \qquad (1)$$

If the subset of vertices $\{x_i\}$ forms a cover, then the sum (1) equals 0. As in each pair x_i, x_j one of variables for cover formation should equal 0, then after multiplication in (1) equalities $x_i x_j + x_i x_j = x_i x_j$ will be correct equality and the relation (1) can be rewritten in the following form

$$x_1 \sum_{q \in m_{iq}} x_q + x_2 \sum_{q \in m_{iq}} x_q + \ldots + x_n \sum_{q \in m_{iq}} x_q = \sum_{i,j \in E} x_i x_j. \qquad (2)$$

Then the problem of minimum vertex cover definition can be formulated in the following form

$$\min \{x_i\} \qquad (3)$$

when the following condition is satisfied in the form of equality

$$\sum_{i,j \in E} x_i x_j = 0 \cdot \qquad (4)$$

The functional (3) means that we search for the minimum number of variables x_i =0 that will convert equality (4) into identity, i.e. we have come to a problem of square programming.

Taking into account (2), it is possible to rewrite problem (3) and (4) in the following form

$$\min \{x_i\} \qquad (5)$$

when following restrictions are satisfied:

$$x_1 \sum_{q \in m_{iq}} x_q = 0; x_2 \sum_{q \in m_{iq}} x_q = 0; \ldots, x_n \sum_{q \in m_{iq}} x_q = 0. \qquad (6)$$

The set of equations (6) can be easily made on the basis of base components of the graph, whereat the number of the equations will be equal to number of base components. For example, for the graph presented on fig. 1 the given set of equations will look like:

$$\begin{aligned} &x_1 x_2 + x_1 x_3 + x_1 x_4 = 0, \\ &x_2 x_3 + x_2 x_4 + x_2 x_1 = 0, \\ &x_4 x_1 + x_4 x_2 + x_4 x_3 = 0, \\ &x_5 x_4 = 0. \end{aligned} \qquad (7)$$

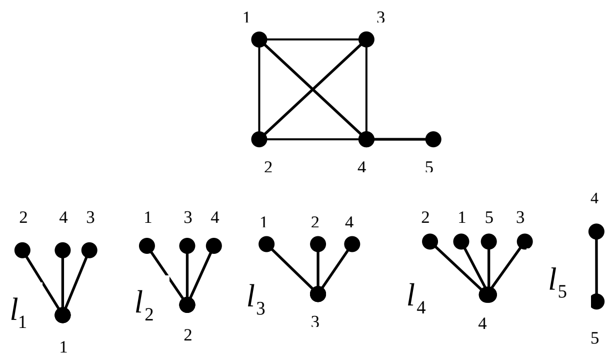

Fig. 1 Graph G and its base components.

For definition of the minimum vertex cover in the graph *G* on fig. 1, it is necessary to determine the minimal number of variables $x_i = 0$ converting to identity a set of equations (7).

In the study [2] it is shown that the following statement is correct.

The statement. If a graph *G* has a suspended vertex *i* formed by an edge (*i*, *j*) the vertex *j* belongs to one of the minimum vertex covers in the graph *G* if there are some vertex covers in graph *G*, and to the minimum vertex cover of the graph *G* if there is one vertex cover in the graph.

Presence of a suspended vertex in the graph means that we have at minimal one base component of the graph containing one edge, see fig. 1. The equation $x_5x_4=0$ corresponds to the suspended vertex in (7) and therefore we should in (7) set $x_4=0$, i.e. include 4-th vertex in a cover, thus the system will become:

$$x_1x_2 + x_1x_3 = 0$$
$$x_2x_3 + x_2x_1 = 0. \qquad (8)$$

Further it is clear that variables x_1, x_2 and x_3 occur equally often in the remained equations; therefore any of them could be included in a cover, for example, if we set $x_2=0$, we get $x_1\, x_3 = 0$, further we include either vertex 1, or 3 in a cover, i.e. we have received two minimum covers, these are vertices {4, 2, 1} or {4, 2, 3}. Correctness of the solution is easy for checking up having enumerated all vertex covers of the graph *G* on fig. 1 using representation of the graph in the form of a Boolean function

$$f = (x_1 \vee x_2)\,(x_1 \vee x_3)\,(x_1 \vee x_4)\,(x_2 \vee x_3)\,(x_2 \vee x_4)\,(x_3 \vee x_4)\,(x_4 \vee x_5). \qquad (9)$$

Multiplying the variables contained in brackets and collecting like terms with use of an absorption rule, we will get the list of vertex covers of the graph

$$f = x_1x_2x_4 \vee x_1x_3x_4 \vee x_2\, x_3\, x_4 \vee x_1x_2x_3x_5,$$

i.e. the graph *G*, fig. 1, has three minimum covers {4, 2, 1}; {4, 2, 3} {1, 3, 4} which are intersected by set of vertices {1, 2, 3} corresponding to set of variables $\{x_1, x_2, x_3\}$ that occur equally often in the equations on some step of search for maximum number of the variables which zeroes corresponding pairs of items in the equations (8), but it has built only covers {4, 2, 1} and {4, 2, 3}.

Let's introduce the following procedure of a problem (5, 6) solution.

Procedure A_1

Step 1. We check whether there are the variables occurring one time in a set of equations; if there are such variables, the variables coupled with them will be equated to zero, and corresponding vertices will be brought into a cover, then the following step is performed; if there are no such variables, we pass to step 3.

Step 2. We check in a set of equation whether all conjugate items are zeroed or not, if it is not, we go to step 1, otherwise the algorithm stops, and the generated cover is minimum.

Step 3. We find in a set of equations a variable which is more often encountered in conjugate products of a set of equations. If there are several such variables, then any of them is selected and equated to zero, and corresponding vertex is brought into a cover, and we go to step 2.

Let's consider the problem an optimality of A_1 procedure work. For this purpose let's consider three following cases:

1) Let's assume that on each step of procedure work there are variables occurring once, then with each step according to the statement 1 we connect to a cover the vertex which reliably belongs to the minimum vertex cover and therefore the cover obtained as a result of procedure A_1 will be minimum.

2) Let's assume that the initial graph *G* has one minimum cover, and on each step (3) of procedure A_1 work there is one variable which occurs most often in conjugate products of a set of equations. Let's show that in this case we will obtain the minimum vertex cover too. Let in this case the procedure build a cover with a potency *k*, and let's suppose that there is a cover with a potency *k*-1 in the graph *G*, but it is possible only in the event that there is a variable which occurs more often than the variables included in a cover with a potency *k* that contradicts the fact that at each step (3) of procedure A_1 a vertex with maximum frequency of occurrence in conjugate products of a set of equations (6) was included in the cover. Hence, the cover received with the procedure in this case will be minimal too.

3) Let's assume that there are more than one minimum covers in the initial graph *G*. It means that during process of procedure A_1 a situation occurs when at some step of its work it appears that a maximum number of variables which can be zeroed due to a choice of various variables $\{x_i\}$ is constant, i.e. it corresponds to that there are some minimum covers in the graph *G* which are either not intersected at all, or they are intersected on a subset of

the vertices corresponding to variables $\{x_i\}$. Unfortunately in a case when there are some nonintersecting covers, procedure ***A***$_1$ can lose an optimum solution, i.e. the procedure gives the approximate solution of a problem, and therefore it is of interest to estimate the error of its work. Thus, procedure ***A***$_1$ allows finding an approximate solution of the problem (5, 6) and consequently also of the problems (3, 4). The number of the equations in (5, 6) does not exceed n, and number of pairs of components in the equation does not exceed m. As in the course of procedure work we should review a set of equations no more than n times, time complexity of procedure work could not exceed $\boldsymbol{O}(mn^2)$.

Formalization and solution of MCP. The problem about the minimal cover can be formulated as a problem of linear Boolean programming [1, 7] which statement in a general view looks like:

$$L = \sum_{j=1}^{n} x_j \to \min, \tag{10}$$

at constraints

$$\sum_{j=1}^{n} \beta_{ij} x_j \geq 1,\ i = \overline{1,m}; \qquad \beta_{ij} \in \{0,1\};\ x_j \in \{0,1\}. \tag{11}$$

The problem (10, 11) can be considered as a problem of minimum graph number definition in a Boolean matrix *B* which covers with units all the lines of the given matrix [2]. Let's represent an arbitrary Boolean matrix in the form of a Boolean function *f* set in a conjunctive normal form (CNF) [2, 4] in which the disjunctive number is equal to number of lines in the matrix, and number of variables in every disjunction is equal to number of units in a line of matrix *B*. Then a problem on a minimal cover for an arbitrary matrix *B* set by some a Boolean function

$$f = (X_l \vee X_m \vee ... \vee X_k)(X_s \vee X_r \vee ... \vee X_t)...(X_q \vee X_d \vee ... \vee X_h) \tag{12}$$

could be considered as a problem of determination of the minimum set of variables $\{X_i=1\}$, at which Boolean function (12) is satisfiable. It is possible to write the statement in the following form

$$\min_i \{X_i = 1\}, \tag{13}$$

if the following conditions are satisfied:

$$(X_l \vee X_m \vee ... \vee X_k)(X_s \vee X_r \vee ... \vee X_t)...(X_q \vee X_d \vee ... \vee X_h) = 1. \tag{14}$$

If we consider a two-value Boolean function we will obtain

$$\min_i \{X_i = 0\} \tag{15}$$

$$X_l X_m ... X_k \vee X_s X_r ... X_t \vee ... \vee X_q X_d ... X_h = 0. \tag{16}$$

From (15, 16) it follows, that the problem about the minimal cover can be considered as a problem of nonlinear Boolean programming consisting in determination of the minimal number of variables $\{X_i = 0\}$ zeroing the left part of the condition (16). The following statement is correct: if the variable $X_{i=q}$ occurs once in items, it is possible to suppose the variable equal to 1 in (16), i.e. the variable $X_{i=q}$ is not included into the minimum cover. Correctness of this statement is clear from statement of a problem (13, 14). If variable $X_{i=q}$ in (14) occurs in disjuncts once, it means that all variables located in the same disjunction occur at minimal two or more times in others disjuncts, and if we suppose them equal to 1, they will also cover that disjunction where there is a variable $X_{i=q}$ and some more disjuncts where they are present, and therefore the variable cannot belong to the minimum cover, and it is possible to suppose that the variable is equal to zero in (14). Accordingly, it should equal 1 in a two-value Boolean function, quadrat demonstrandum. It is necessary to note that if as a result an item appears in (16) consisting of one variable, the given variable should be included into the minimum cover, because if not, then the equality (16) will have never been satisfied. Using the property it is possible to offer the following procedure ***A***$_2$ similar to procedure ***A***$_1$ for solution of the problem (15, 16).

Procedure *A*$_2$

Step 1. Check if there are variables occurring once in items; if there are such variables, they are equated to 1, then go on to the next step.

Step 2. Check if there are the items consisting of one variable; if there are such variables, they are equated to zero and brought into a cover; if there are no such variables, go on to the next step.

Step 3. Check whether all items are zeroed or not; if there are such variables procedure stops as the minimum cover is generated, otherwise we go on to the next step.

Step 4. We find a variable which is more often encountered among the remained items (if there are a number of them we select any), we suppose it equals zero and include it in a cover; then it is checked whether all items are zeroed or not. If there are such variables, procedure stops as the minimum cover is generated, otherwise we go to step 1.

For example, let's set the matrix *B*.

	1	2	3	4	5	6
1	1	1	0	0	0	1
2	0	0	1	0	1	1
3	0	1	0	0	1	0
4	1	1	1	0	0	0
5	0	0	1	1	0	0

Let's present *B* in the form of a Boolean function:

$$f=(X_1\vee X_2\vee X_6)(X_3\vee X_5\vee X_6)(X_2\vee X_5)(X_1\vee X_2\vee X_3)(X_3\vee X_4).$$

Two-value Boolean function will look like:

$$\overline{f}=X_1X_2X_6\vee X_3X_5X_6\vee X_2X_5\vee X_1X_2X_3\vee X_3X_4. \quad (17)$$

As the variable X_4 is encountered one time in items, we suppose $X_4=1$, upon that we have received an item with one variable X_3. Let's suppose $X_3=0$ and include it in a cover after that we still had items $X_1X_2X_6\vee X_2X_5=0$. We choose a variable which is most often encountered in items, it is X_2, let's suppose $X_2=0$ and include it in a cover. As upon that all items of a Boolean function (17) are zeroed, the cover generated of variables { X_2 X_3} is minimum.

Let's analyze an optimality of procedure $\boldsymbol{A}_2$ work. For this purpose let's consider three following cases:

1) Let's suppose that there are items containing one variable on each procedure step (such situation is possible when disjuncts of a Boolean function of matrix *B* contain two variables, that is equivalent to a problem solution about the minimum vertex cover in arbitrary graphs) then according to the proven statement, on each step we connect to a cover a variable which is reliably belonging to the minimum cover and therefore the cover obtained as a result of procedure $\boldsymbol{A}_2$ work will be minimum.

2) Let's suppose that there is one minimum cover for matrix *B* and on each step (4) of procedure $\boldsymbol{A}_2$ work there is one variable which is more often encountered in products of a set of equations. Let's show that in this case we will receive the minimum cover too. Let in this case the procedure has built a cover with potency k, and let suppose, that there is a cover with potency k-1 for a matrix *B*, but it is possible only in the event that there is a variable which is encountered more often than the variables included in a cover with a potency k that contradicts the fact that on each step (4) of procedure $\boldsymbol{A}_2$ a variable with maximum frequency of occurrence in items of a two-value Boolean function has been joined to the cover and, hence, the cover obtained through the procedure in this case will be minimum too.

3) Let suppose that there is more than one minimum cover in a matrix *B*. It means that there is a situation in process of procedure $\boldsymbol{A}_2$ when on some step of its work it occurs that a maximum number of variables which can be zeroed due to a choice of various variables $\{X_i\}$ is constant, i.e. it means that there are some minimum covers in *B*, and they either are not intersected at all, or intersected on a subset of variables $\{X_i\}$. As it is possible to execute a speculation presented in point 2 for each cover, it will mean that in this case as well the procedure $\boldsymbol{A}_2$ will build at minimal one of the minimum covers, but when covers are not intersected, as well as in a problem about a vertex cover, the optimum value loss is possible. Thus, procedure $\boldsymbol{A}_2$ allows determination of the approached solution of the problem (15, 16), and consequently the problem (13, 14) also. It is clear, that at number of variables not exceeding two in disjuncts of a Boolean function, the procedure $\boldsymbol{A}_2$ is actually degenerated to procedure $\boldsymbol{A}_{1.}$ The number of items in (15, 16) does not exceed *m*, the number of variables in the item does not exceed *n*, and as in the course of procedure we should review items no more than *n* times, time complexity of procedure performance cannot exceed $\boldsymbol{O}$ (mn^2).

3. Experimental research of procedures A_1 and A_2 performance

To check accuracy of solution of a problem about a vertex cover on the basis of procedure $\boldsymbol{A}_1$, the exact algorithm of exponential complexity built on the basis of concepts of the rank approach [7, 14] was used as the reference. Dimension of the graph varied from 10 to 30 vertices, graphs were generated with a various tightness of edges in the graph under the uniform law of distribution, not less than 100 realizations were generated for each dimension of the graph, and the error was measured. The average estimation of the time complexity of algorithm is carried out also; all values are received with a probability belief 0.95 for dimensions of graphs from 10 to 490 at various values of a tightness of edges in the graph. The diagrams of correspondence of average elementary operations number performed by the algorithm for different tightness of edges in the graph are shown on fig. 2.

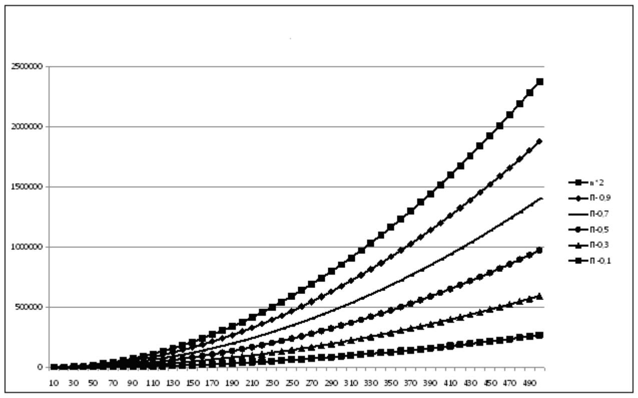

Fig. 2 Dependence of number of operations performed by the procedure $\boldsymbol{A}_1$ on the number of vertices for different tightness of edges in the graphs.

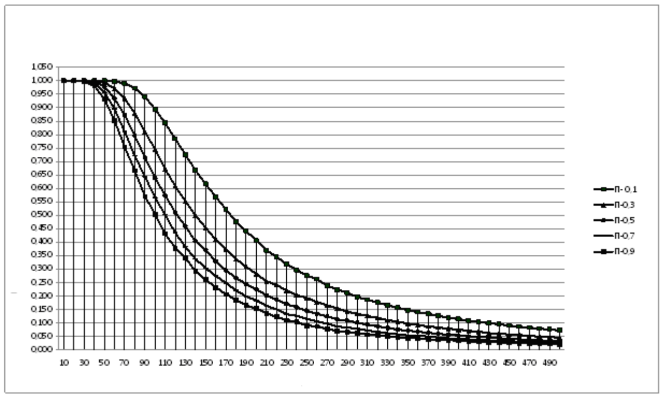

Fig. 3 Dependence of probability of a problem solution by the procedure $\boldsymbol{A}_1$ for allowed time 5 ms at various values of tightness of edges in graphs.

Fig. 3 shows correspondence of probability of a problem solution on the minimum vertex cover for allowed time 5 ms. It follows that the averaged time complexity of the algorithm does not exceed $\boldsymbol{O}(n^2)$ and in the cases of allowed time of the solution equal to 5 ms, problems for graphs with number of vertices not exceeding 100 can be effectively solved.

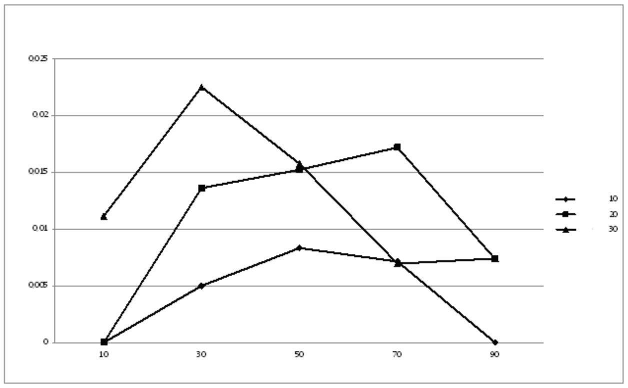

Fig. 4 Dependence of a relative error on number of edges in the graph at various numbers of vertices.

Fig. 4 shows dependence of an error of procedure $\boldsymbol{A}_1$ work on number of edges and vertices of the graph from which it is visible that magnitude of an error does not exceed 2.5 % on the average. Check on test examples of the big dimension has shown that there is a tendency to decreasing of magnitude of an error with magnification of dimension and a tightness of edges in the graph. On the diagrams mapping work of procedure $\boldsymbol{A}_2$ which is used for solving MCP (fig. 5 – fig. 8) the number of columns in matrix B is designated as n, and number of lines – as m. Fig. 5 shows dependence of number of elementary operations performed by procedure $\boldsymbol{A}_2$ with sizes of matrix B at various densities of filling by units of matrix B. Fig. 6 represents dependence of probability of solution MCP by procedure $\boldsymbol{A}_2$ for the allowed times of their solution equal to 50 ms and 100 ms. It is clear from the diagram that procedure $\boldsymbol{A}_2$ has the averaged time complexity which does not exceed $\boldsymbol{O}(mn)$. Test examples of a solution of problems at $n \geq 600$; m=400 have shown that the averaged time complexity did not exceed $\boldsymbol{O}(400mn)$. The analysis of an error of procedure $\boldsymbol{A}_2$ work (fig. 7, fig. 8) has shown that at densities of filling by units of matrix B more than 50 % the error does not exceed 5 %.

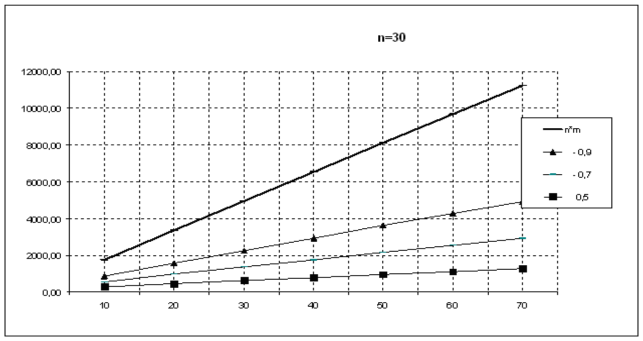

Fig. 5 Dependence of number of the operations performed by procedure $\boldsymbol{A}_2$ on dimension of matrix B.

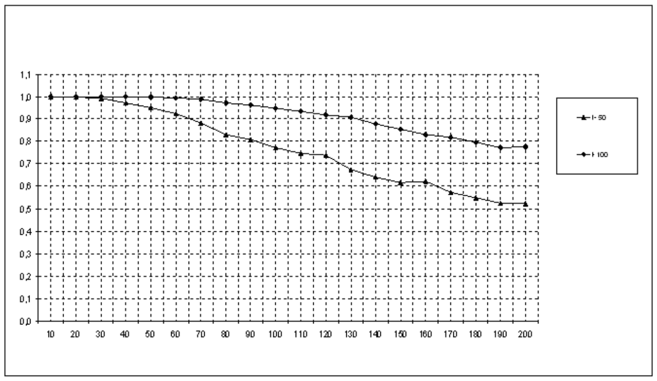

Fig. 6 Dependence of probability of MCP solution by procedure A_2 with an amount of columns in a matrix *B* at allowed times of MCP solution equal to 50 and 100 ms.

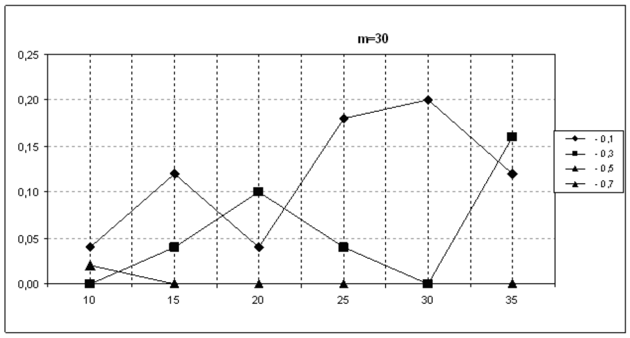

Fig. 7 Dependence of a relative error of procedure A_2 work on number of columns in a matrix *B*.

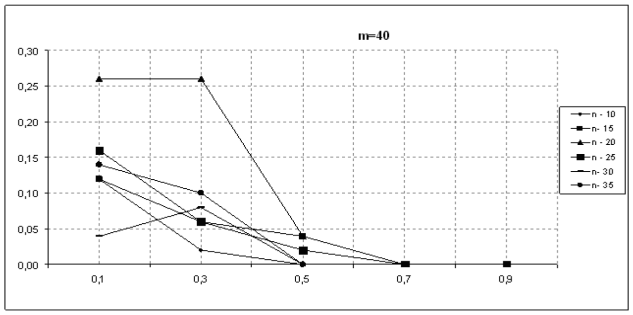

Fig. 8 Dependence of an error of procedure A_2 work on densities of units of matrix *B*.

4. Conclusions

Thus, the rough algorithms of MVCP solution are proposed in arbitrary graphs and MCP on the basis of their reduction to problems of square and nonlinear Boolean programming accordingly. Their specificity has allowed building algorithms with time complexity which does not exceed $\boldsymbol{O}$ (mn^2) where in case of MVCP solution n is an amount of vertices in the graph, m is a number of edges in the graph, and in case of MCP solution n is a number of graphs in a matrix *B*, and m is a number of lines in matrix *B*. Upon that, the error of solution of these problems by procedures A_1 and A_2 does not exceed 2.5–5 % at tightness more or equal to 0.5, and they work with time complexity $\boldsymbol{O}$ (n^2) for MVCP solution and $\boldsymbol{O}$ (mn) for solution MCP on the average that is very important at solution of resource distribution problems in modern GRID systems. It is clear from fig. 3 and fig. 6, that the proposed algorithms can be used for real time resource distribution planning [2, 4] if allowed time of scheduling lays in a range 5 – 100 ms.

References

[1] N. Christophides, Graph-Theory: an Algorithmic Approach, Moscow: MIR, 1978.
[2] V. S. Ponomarenko, and S.V. Listrovoy, "Problem-solving procedure for minimal cover as a GRID sceduling tool", Control problems, the Russian Academy of Sciences, No. 3, 2008, pp.78–84.
[3] V. Lipsky, Combination theory for programmers, Moscow: MIR, 1988.
[4] V. S. Ponomarenko, S.V. Listrovoy, S.V. Minukhin, and S.V. Znakhur, Methods and models of resource planning in GRID-systems. Monograph, Kharkov: Publishing House “ENGEC”, 2008.
[5] N.Z. Shor, and S.I. Stetsenko, Square extremum problems and nondifferentiable optimization, Kiev: Naukova dumka, 1989.
[6] K. Papadimitriu, and M. Stieglitz, Combinatorial optimization. Algorithms and complexity, Moscow: MIR, 1985.
[7] S.V. Listrovoy, and A. Yu. Gul, "Method of Minimum Covering Problem Solution on the Basis of Rank Approach", Engineering Simulation, No. 1, 1999, pp. 58 – 70.
[8] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, Introduction to Algorithms, MIT Press, 2009.
[9] R. Gandhi, S. Khuller, and A. Srinivasan, "Approximation algorithms for partial cover problems", In Proceedings of the 11th International Colloquium on Automata, Languages, and Programming (ICALP), Vol. 2076 of LNCS, 2001, pp. 225–236.
[10] N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, and J. Naor, "The online set cover problem", in Proceedings of the 35th Annual ACM Symposium on the Theory of Computation (San Diego, 2003), ACM, New York, 2003, pp. 100–105.
[11] V. Chvátal, "A greedy-heuristic for the set cover problem", Mathematics of Operations Research, Vol. 4, No. 3, 1979, pp. 233–235.

[12] P. Slavik, "A tight analysis of the greedy algorithm for set cover", Journal of Algorithms, Vol. 25, No. 2, 1997, pp. 237–254.
[13] R. Hassin, and A. Levin, "A better-than-greedy approximation algorithm for the minimum set cover problem", SIAM J. Computing, Vol. 35, No. 1, 2005, pp. 189–200.
[14] S.V. Listrovoy, and S.V. Minukhin, "General approach to solution of optimization problems in the distributed computing systems and the theory of building of intellectual systems", Journal of Automation and Information Sciences, No. 2, 2009, pp. 47–63.