Scholedge International Journal of Multidisciplinary & Applied Studies
ISSN 2394-336X, Vol.05, Issue 07 (2018)
Pg 82-88.
DOI: 10.19085/journal.sijmas050701

Published by: Scholedge Publishing
www.theSCHOLEDGE.org
Email: editorial@thescholedge.org

About One Task of Mixed Integer Programming of Large Dimension and its Solving Algorithm

A. Kh. Abdullayev
Department of Information Economy and Technologies, Azerbaijan State University of Economics UNEC

B.M.Aliyeva
Department of Economics and Management, Faculty of Turkish World Business Administration, Azerbaijan State Economic University Azerbaijan State University of Economics UNEC
.

ABSTRACT

One method of solving the mixed-integer task of a special type is offered in this paper, which emerges when solving the task of production placement by a modified method of Benders. The task is solved by the branch and boundary method. Calculative experiments are shown and optimal or close to optimal variants of production placement are gained. The scheme of branching is offered, which is economic if considering the required computer memory. Also, the rules of the rejection process of variants are offered.

KEYWORDS: mixed integer programming, branch and bound, Benders decomposition method, the task of placement.

Introduction

The sense of Bender's algorithm is about successive solving of mixed integer task with one continuous variable and the task of linear programming. It is known that the main difficulty about the realization of the decomposed system is about solving a mixed integer task with one continuous variable. Thereby, a modification scheme is offered for solving such kind of tasks, which idea is that they look not for a proper solution for a task on each step of an iterative process, but some permissible solution.

1. Problem statement

Many tasks of production placement amount to the task of mixed integer programming solving of the following type:

$$c`x + d`y \rightarrow \min$$
$$Ax + Dy \geq b \quad (1)$$
$$y \in \Omega,\ x \geq 0$$

Where,x–vector of continuous variables, y-vector of Boo variables, andΩ–some multiplicity. Sometimes the system of constraints is so, if$\in\Omega$multiplicity

$$\{x \mid Ax \geq b\text{-}Dy,\ x \geq 0\} \neq \emptyset \quad (2)$$

If this requirement is not fulfilled, we can add one more variable v≥0,and instead of task(1) examine the task

$$c`x + d`y + \lambda v \rightarrow \min$$
$$Ax + Dy + Ev \geq b, \quad (3)$$

http://dx.doi.org/10.19085/journal.sijmas050701

$y \in \Omega$, x≥ 0 , v≥ 0

Where λ≥0–some number, and E=(1,1,…1,1)–a vector of the corresponding dimension. In case of the value high enough, the tasks(1)and (3)solutions are equal. Such method was used, for example, in work [1]In such a way, we will consider the requirement (2) is being solved. In case of such assumption, a decomposed Bender's method for the task [1] solving comes down to the coherent solving of the two tasks. One of them is the task of linear programming, which is solved using fixed values:

$$U=\max(b-Dy)\ u$$
$$A`\ u\leq c, \quad (4)$$
$$u \geq 0$$

Another task is the task of mixed integer programming, but with one continuous variable z.

$$\min z,$$
$$d`y+(b-Dy)`u^i< z, \quad (5)$$
$$y \in \Omega$$

Hereu^i–are the solutions for the task (4) using different values of y .

The aim of Bender's method is to solve coherently tasks (4), (5). Task (4) is solved firstly with some value of$_0 \in \Omega$. With the help of the solution received, u^{i}the task (5) restrictions are formed, which is solved later. After the task (5) solution received, task (4) is solved, one restriction in task (5) is added and etc. till the moment, when during stepN the requirement of optimal task (1) solutions will be fulfilled:

$$\max [(b-Dy^N)`u|A`u\leq c, u \geq 0]=z^{N-1} -d`y^{N-1} \quad (6)$$

wherez^{N-1} - task (5) optimal solution during N step. The procedure of solution is so that the number of restrictions in the task (5) at every step increases by one. A modification of this procedure is offered in the work [2] ,the essence of which lies I the fact that task (5) is solved on each step of the procedure, and the only question is solved about the existence of of an acceptable solution of the system of restrictions of the type:

$$d`y+(b-Dy)`u^i<z_\varepsilon^t,\ i=1,2,3,\ldots t \quad (7)$$
$$y \in \Omega$$

where,z_ε^t - is record meaning, and$\varepsilon$is a relative acceptable error of the task (1) solution. The value of the record on step t

$$R^t = \min_{i=0,1,\ldots,t-1} [\ d`y^j+u^i\ ]$$

It is obvious, that if at some stepN the system of restrictions(7)does not have an acceptable solution than the current record isε–an optimal value of the task (1).

2. One particular type of task of placement

In the current work, we observe one particular type of task of production placement, for which the system of inequalities(7)looks like

$$\sum_{\mu=1}^{m} \sum_{k=1}^{k\ \mu} \sum_{j=1}^{n} a_{\mu jk}^i\ y_{\mu j}^k\ + \alpha_i\ < z_\varepsilon^t,\ i=\overline{1,t} \quad (8)$$
$$\underline{V}_\mu \leq \sum_{k=1}^{k_\mu} \sum_{j=1}^{n} R_{\mu k}\ .\ y_{\mu j}^k\ \leq \overline{V}_\mu\ ,\quad \mu = \overline{1,m} \quad (9)$$
$$\sum_{\mu=1}^{m} \sum_{k=1}^{k\mu} y_{\mu j}^k\ \leq 1\ ,\ j=\overline{1,n} \quad (10)$$
$$y_{\mu j}^k \in \{0,1\}\ ,\ \mu=\overline{1,m}\ ,\ k=\overline{1,k_\mu}\ ,\ j=\overline{1,n} \quad (11)$$

Now we will indicate the sense of restrictions (8)-(11), which define some class of the tasks of placement, for which the method of solution, which is offered in this work, is applicable. Restrictions (8) –are

restrictions of the general type, which appear in the Bender's subtask(5). Restrictions(9) create possible range [$\underline{V}_\mu$,$\overline{V_\mu}$]the changes of total power in each field, at that$R_{\mu k}$ k –я possible power in μ field. The variable of Booleany$^{k}_{\mu j}$ is equal to1, if in point jthe enterprise ofμ field is placed of $R_{\mu k}$power and is zero otherwise.

Let it be J={1,2,…,n}, M={1,2,…,m}, $K^\mu = \{1,2,\dots k_\mu\}, T = \{1,2,\dots,t\}$, $\overline{L = \{1,2,\dots,n\sum_{i=1}^{m} k_i}\}$ -an ordered multiplicity of numbers of all components of the vector y.Value$f_\mu = n\sum_{i=1}^{\mu} k_i$, μϵM is equal to the number of the last component of the vector y , which is connected to the branchμ. If to take f_0 =0 , then multiplicity L^μ ={l | l=$\overline{f_{\mu-1}+1, f_\mu}$-is a multiplicity of the numbers of all components of the vector y, which are connected to the branchμ, and multiplicity$L^\mu_k = \{f_{\mu-1}+n(k-1)+1,\dots,f_{\mu-1}+nk\}$-is a multiplicity of numbers of all components of vector y, connected toμ, and power k. Besides we will enter multiplicity$L_j = \{j, j+n,\dots,j+n(f_\mu-1)\}$, the multiplicity of numbers of the components of vector, which are related to the point of placement j. Between entered multiplicities of indexes the following correlations exist:

$$L^\mu = \cup_{k\in K^\mu} L^\mu_k,\ \overline{L} = \cup_{\mu\in M} L^\mu = \cup_{j\in J} L_j$$

Considering entering vectory and appropriate multiplicities, the task (8)-(11) can be rewritten in the following way:

$\sum_{l\in\overline{L}} a_{il}y_l + \alpha_i < z^t_\varepsilon$, i∈T (12)

$\underline{V}_\mu \le \sum_{l\in L^\mu} r_{\mu l}y_l \le \overline{V_\mu}$, μ ∈ M (13)

$\sum_{l\in L_j} y_l \le 1$, j∈ J (14)

$y_l \in \{0,1\}, l \in \overline{L}$ (15)

where,

$a_{i,f_{\mu-1}+n(k-1)+j} = a_{\mu kj}$, $r_{\mu l} = R^k_\mu$, , j∈ J, μ ∈ M, k ∈ K^μ , l∈ L^k_μ

Consequently,

$r_{\mu l_1} = r_{\mu l_2}, l_1, l_2 \in L^\mu_k$

Let's assume that the multiplicity of powers in any branch is ordered, i.e. for μ, $k_1 < k_2$: $R^{k_1}_\mu > R^{k_2}_\mu$ and consequently for$l_1 < l_2 : r_{\mu l_1} > r_{\mu l_2}$.

3. About some schemes of branch and bound methods of extreme tasks solution

To solve the task (8)-(11)further some scheme will be offered, which is based on the branch and bound method – a method of solution extreme tasks of mixed integer programming. We will emit a method of limited branches and bounds from a group of methods of the branch and bound, a method of Abadi [4], which owns a range of positive properties. We must admit that the ideas of theme thod of Abadi were used I the work for the solution of the task of finding the admissible solution of the system of linear Boolean inequalities. The ideas of the method of branches and bounds are well known. One of the best statements of this method is given in the review article. [5].

One of the difficulties in applying this method is that it is impossible to indicate the number of options that need to be memorized in the solution process. This lack is absent in the Abadi method. For example, in the Boolean programming task of dimension n, the number of memorized variants does not exceed 2 (n-1).

4. The method of finding an admissible solution of the system of linear inequalities of the Benders sub problem

It is required to find one admissible solution of the system of inequalities (9) - (12) or to establish the fact of its inconsistency. To solve this task, we will use the ideas of works [5, 7] in which the ideology of the Abadi method is used to solve a similar task. First, we will outline the general scheme of the method for solving the task for a fixed number t of constraints of type (9), then will give a detailed description of the method, and then indicate how the method differs as the number of restrictions (9) is increased by one.

LetL_0иL_1denote the multiplicities of numbers of fixed variables, which take values 0и1consequently, i.e.y_l=0, если$l \in L_0$ иy_l=1, если$l \in L_1$. The multiplicity of numbers of all fixed variables –L.

4.1. Building the tree of variants

We will describe the construction of the tree of the variants being viewed. Between the tree variants and all possible sets (L_0, L_1) there is a one-to-one correspondence$S \leftrightarrow (L_0, L_1)$that is why the vertex and the set can be denoted by S.

The root of the tree is vertex S_0 =(ø,ø). The tree of branching will be represented in a level structure. All vertices of S, located at a given level n (s), are characterized by the fact that n (s) = | L_1 | i.e. these vertices correspond to variants in which n (s) variables are equal to one. Each vertex of level n (s) is connected by arcs only to some vertices of the next level and to one vertex of its level n (s).

We will choose at the beginning of branching some branch. Without loss of generality, we can assume that the number of this branch is μ = 1. Variants for capacities we will consider in some given order, for example in the order of their decrease. Variants on the placement of variants will be selected in accordance with some algorithm. Let's begin the procedure for constructing the variants tree. Assuming (in accordance with the algorithm of choice of variants for placements points)$y^1_{1i_1}$ =1.Some vertex – 10 corresponds to this variant at the first level.

For all of the variants of the first level, which are connected by the way with vertex 10, the meaning is$y^1_{1i_1}$=0. All these vertices are connected with the options for placing of the first capacity, the first branch for different placement points. In general, in the tree of variants, all groups of vertices of the same level, connected in some way, correspond to the placement of some capacity of the known industry in the possible placements. We will observe the following variant: $y^1_{1i_1}$=1, $y^{k_{\mu_2}}_{\mu_2 j_2} = 1$(Choice$\mu_2$, j_2 , K_{μ_2} is made upon some algorithm.) Vertex 20 corresponds to this variant on the 2nd level. The vertex 20 is connected by the way with vertices of the 2nd lever, for which $y^1_{1i_1}$=1, $y^{k_{\mu_2}}_{\mu_2 j_2} = 1$ and for some numbers j

(the numbers of points of capacity placement of branchK_{μ_2}, branchμ_2)$y^{k_{\mu_2}}_{\mu_2 j_2} = 1$. Further from the top of the second level we will go down to the third level, etc. As a result, we will drop down the left branch of the tree, remembering only the vertex groups including the vertices of this branch. At each step, the vertices 10, 20, 30, ... are subjected to the test, i. E. to some verification, showing whether it makes sense to consider this variant, which lies on the leftmost branch and all the derived variants, descendants, lying on the lower levels. If it turns out that some vertex has not passed the test, then the next vertex found in this group of variants is considered. Let the top 30 fail the test. Then the next variant is considered 31. From this vertex 31 the, tree grows according to the principle described above. Return to the previous level occurs only when all the vertices of the group are viewed. So, the return to the top 20 will happen only when the third level options are viewed.

Let' point the properties of the tree of variants.

1. Each time, when moving to the lower level, the question of which branch to consider is algorithmically solved.
2. All the vertices in one group correspond to the variants in which μиk $\in K^{\mu}$are equal at the last fixed variable.
3. The maximum number of arcs connecting any vertex to the vertices of the next level is equal to the number of possible capacities of the selected branch. The smaller number of arcs can be when some capacity variants are unacceptable at this branching step.
4. The maximum number of vertices in a group is equal to the number of possible placement points. At the level n (s), this number is equal to n-n (s). A smaller number of vertices can be in case if at a given branching stage a certain capacity of a known branch cannot be placed at any one point.
5. Since the algorithm for selecting the branch in the transition to the next level (it is described below) does not depend on the location of the capacity, then all the vertices of the tree at one level, connected by arcs with the vertices of one group of the previous level, are characterized in that for them the power value of the last fixed variable$y^k_{\mu j}$ - is the same.
6. In the left vertex of the group, the value of the last fixed variable is $y^k_{\mu j} = 1$, and for the rest vertices of this group are$y^k_{\mu j} = 0$.
7. The maximum number of memorized variants does not exceed n (N -1) / 2-2.

We should note that the rule for constructing a tree implies that the maximum number of arcs originating from the root of the tree is equal to the number of possible powers of the branch chosen before branching.

We described the general scheme for solving the problem (9) - (12). According to this scheme, starting from the root of the tree, the values of Boolean variables are fixed consequently$y_{\mu j}^{k}$ untilla valid solution will not be found or information will be received that there is no such solution. Now we will clarify some of the details of this method.

4.2. The choice of the branch and branching

When describing the general scheme of the method, it was said that when you go to the next level, each time the industry is selected, the placement options for which will be considered. Now let us describe this procedure. For convenience, we rewrite inequality (13) in the form:

$$0 \le (\sum_{l \in L^{\mu}} r_{\mu l} y_l - \underline{V}_{\mu}) / (\overline{V_{\mu}} - \underline{V}_{\mu}) \le 1, \quad \mu \in M \quad (16)$$

Let us stay at some vertex S=(L_0, L_1).Then for vertex Sthe level of plan performing on the branch μthe following value will characterize

$$\aleph_{\mu}(s) = (\sum_{l \in L^{\mu}} r_{\mu l} y_l - \underline{V}_{\mu}) / (\overline{V_{\mu}} - \underline{V}_{\mu}) \quad (17)$$

If

$$0 \le \aleph_{\mu}(s) \le 1$$

Then the plan of power placement of μ branch is performed. We shall admit that in the root of the tree:

$$\aleph_{\mu}(S_0) = -\underline{V}_{\mu}) / (\overline{V_{\mu}} - \underline{V}_{\mu})$$

When going to the next level while choosing the next branch we will follow the principle of equable of fulfilling the conditions (18). At the same time, we will consider introduced powers of the branches. For vertex Sthe number of maximum possible power of the branch is equal to $K_{\mu}(S)$and in the general case$K_{\mu}(S)$≠1, though we sorted the powers by their decreasing. The thing is that before watching the vertex S it can turn out that for the tree branch, on which the vertex S is placed, it was found out that some powers with numbers $K_{\mu} < K_{\mu}(S)$– are unacceptable. The condition for the uniform implementation of the branch input plan leads to the following criterion for selecting the branch. The branch μ_0 is chosen, satisfying the following condition:

$$\aleph_{\mu_0} = \min_{\mu \in M} [\aleph_{\mu}(s) + R_{\mu}^{K_{\mu}(S)}] / (\overline{V_{\mu}} - \underline{V}_{\mu}) \quad (18)$$

4.3. Inadmissibility check of k power μ branch

Inadmissibility check of k power μ branch is made in case, when returning to the vertex S, a branch of variants starts to be constructed, corresponding to the next power in order (the industry number for μ (S) of the vertex S is fixed to this moment). Recall that the powers of each industry are ordered by their descending, and from the vertex come branches of variants that correspond to an ordered sequence of powers of the industry μ (S) starting from power$K_{\mu}(S) := K_{\mu}(S) + 1$.

The inadmissibility check of power$K_{\mu}(S)$is essential whether the points of placement at a fixed capacity $K_{\mu_0}(S)$ enough at the branchμ_0(S) to perform the plan of the placement of all branches. At the same time, it is taken into account that only one branch can be placed in a given branch. We will now state the criterion for this verification.

Let

$$\min \{p \mid 0 \le \aleph_{\mu}(s) \le pR_{\mu}^{K_{\mu}(S)}] / (\overline{V_{\mu}} - \underline{V}_{\mu})\} \quad (19)$$

-the minimal number of points, which is necessary to perform the plan of branch μplacement. As at the level n(S),at which the vertex S=(L_0 , L_1)is situated, the number of busy points of placement is equal to $|L_1|$=n(S) , andn–the general quantity of points of placement, then the criterion of inadmissibility of power K placement will look like:

$$n_{\mu_0}(S) > n - n(S) - \sum_{\mu \in M \setminus \{\mu_0\}} n_{\mu}(S) \quad (20)$$

If this criterion is satisfied, then from the consideration is excluded not only the branch corresponding to this power, but also all other branches corresponding to smaller powers. And since by the time of return to the top S all branches that are located to the left of the branch with power $K_{\mu}(S)$, then consequently the vertexSitself must be excluded from the observation.

4.4. Inadmissibility check of power placement

Each vertex S located at the level n (S) is connected by an arc to the vertex group at the level n (S) +1. All the vertices of this group correspond to the variants of the location of known power at various locations. In the process of constructing a tree of options, it is checked that this capacity can be placed in unoccupied locations. The number of such points at the time of formation of the group at the level n (S) +1 is equal to n-n (S) and their numbers form the set J (S). The test is as follows. We denote by

$z_j(S)=\max_{i\in T}(\alpha_i + \sum_{l\in L_1} a_{il} + a^i_{\mu jk})$

the maximum from the meaning of left parts of the inequality system (12) for vertexS provided that $y^k_{\mu j} = 1. Then$

$$\Omega(S)=\{j \mid j\in J(S),\ z_j(S)<z^t_{\varepsilon}\} \qquad (21)$$

-is a multiplicity of numbers of admissible placement points and $J(S)\setminus\Omega(S)$–is a multiplicity of culled points.

Verticesj∈ Ω(S), which for a group of vertices, are ordered not by the value $z_j(S)$ decline. For the vertex of this group, which corresponds to the power placement variant $R_{\mu k}$in point j, multiplicity L_1grows up by one element, and multiplicityL_0changes in the following way:

-for all these vertices$y^k_{\mu j} = 0$ для j∈J(S)\ Ω (S) and for indexesμ, k which figure in the observed capacity $R_{\mu k}$;
-$y^k_{\mu j} = 0$ for the restμ, kиj∈ Ω.

4.5. Vertices' check

In the process of branching, all vertices are checked for the possibility of their rejection. For this purpose, an evaluation task is used. It is constructed in the following way. After the system of constraints (12) is formed in the process of solving the initial allocation problem by a modified Benders method, the following auxiliary problem is solved

$$\min z \qquad (22)$$

$$\sum_{l\in\bar{L}} a_{il} y_l + \alpha_i < z^t_{\varepsilon},\ i\in T \qquad (23)$$

$$\underline{V}_{\mu} \le \sum_{l\in L^{\mu}} r_{\mu l} y_l \le \overline{V}_{\mu},\quad \mu\in M \qquad (24)$$

$$\sum_{l\in L_j} y_l \le 1,\ j\in J \qquad (25)$$

$$y_l \ge 0,\ l\in\bar{L} \qquad (26)$$

It is obvious that an obligatory requirement for the task (12)-(15) to have admissible solution, is the requirement $z^* < z^t_{\varepsilon}$ где,z* -an optimal task (22)-(26) solution. Let's $\bar{\lambda}=(\overline{\lambda^1_i}, i\in T;\ \overline{\lambda^2_{\mu}}, \mu\in M;\ \overline{\lambda^3_{\mu}}, \mu\in M;\ \overline{\lambda^4_j}, j\in J)$is a vector of dual variables on the optimal task (22)-(26) solution. With the help of these dual variables we will built evaluative task for the vertices of the branching tree.

As already described above, in the process of branching, we drop from the last vertex considered to the next level, select the industry number, form an ordered group of vertices, and then check the leftmost vertex from this group. For this vertex S=(L_0,L_1) the nu, ber of fixed variables are known, which take the value 0 and1. Evaluative task for vertex S=(L_0,L_1) looks like, calculate:

$$F(S)= b(s)+\min_{y\ge 0}[\sum_{l\in\bar{L}} a_l y_l \mid \sum_{l\in\bar{L}} r_l y_l \ge v(S), \sum_{l\in L_j} y_l \le 1,\ j\in J] \qquad (27)$$

где, $b(S)=\sum_{i\in T}\alpha_i \overline{\lambda^1_i} + \sum_{l\in L1} a_l,\ a_l = \sum_{i\in T}\overline{\lambda^1_i} a_{il},\ r_l = \sum_{\mu\in M}(\overline{\lambda^2_i} - \overline{\lambda^3_i}) r_{\mu l}$

$$v(S)=\sum_{\mu\in M}\overline{\lambda_i^2}\underline{V_\mu}-\sum_{\mu\in M}\overline{\lambda_\mu^3}\,\overline{V_\mu}-\sum_{l\in L1}\overline{r_l},$$

For the solution of this task we can offer effective methods, considering its specificity.

It is obvious that if F(S) $>z_\varepsilon^t$, then the vertex Sand all the descendants must be excluded from the following observation.

5. The general scheme for solving the subtask of Benders.

The problem of finding an admissible solution of the system of inequalities must be solved many times, and each time the number of inequalities t in (9) increases by one, and the quantity $z_\varepsilon^{t+1}\leq z_\varepsilon^t$.If some variants are inadmissible with the number of inequalities t and meaningz_ε^t, then they will be inadmissible on the next step too. Thus, in step (t + 1), it is not necessary to review the already rejected variants, but you should start browsing the branch formed in step t. This branch consists of vertex groups (one at each level), and the vertex groups at two neighboring levels are connected by an arc.

In any case at the beginning of step (t+1)a continuous task(22)-(26) is solved. If the solution of this task is equal to or more thanz_ε^tthen the task(22)-(26) does not have any solution. Otherwise, starting from the root of the tree, the branch of the variants remaining from step t is checked for the admissibility. Two cases should be distinguished: in the case, only (t + 1) -th restriction is checked, in the case $z_\varepsilon^{t+1}\leq z_\varepsilon^t$all the restrictions of inequality system are checked(12).

Computational experiments are given and optimal production and closer to optimal variants of branch placement are obtained. The main results of computational experiments showed:

- The main computer time for solving the problem is spent not on finding the optimal solution, but on confirming the optimality of the last solution found;
- The solution time grows exponentially with an increase in the number of discrete variables;
- The time of solving the task essentially depends on the basic^{0}admissible variant;
- The decision method allows you to quickly find solutions that are close to optimal, the accuracy of which in most cases is quite sufficient for those who make practical decisions.

References

[1]. McDaniel D., Devine M. A Modified Bender's Partitioning Algorithm for Mixed Integer Programming, Manag. Sci., 1977, 24, N3, pp. 312-319.

[2]. 2.Buryan S.B._ Serov S.S._ Dinamicheskaya zadacha razmescheniya predpriyatii otrasli i chislennii metod ee reshe¬niya. II. Avtomatika i telemehanika_ 1976_ № 5_ s.112 _ 121.

[3]. 3.A.Kh.Abdullayev, B.M.Aliyeva, The Problem of locating agricultural production,Turan Center for Strategic Researches, International Scientific Peer-Reviewed and Refereed Journal; ISSN: 1308-8041, e-ISSN: 1309-4033; Year: 2017; Volume: 9.

[4]. 4.Abadie J. Une method arborescente pour les programmes non lineairs partiellement discrets. Revue francaise d'informatique et de Recherche Operation nelle, v.3, A.3, pp. 25-49.

[5]. 5.Uzdemir A.P. Metod resheniya kombinatornoi zadachi opredeleniya momentov vvoda predpriyatii. Avtomatika i telemehanika_ 1978_ № 10_ s.142_152.

[6]. 6.Geoffrion A.M., Marsten R.E., Integer Programming Algorithms: frameworx and state-of-the-art survey. Manag. Sci., 1972,18,N9, pp. 465-491.

[7]. 7.Uzdemir A.P. Dekompoziciya pri reshenii kombinatornoi zadachi opredeleniya momentov vvoda predpriyatii. Avtomatika i telemehanika_1977_I 10_ s.110_ 121.

[8]. Hachaturov V.R._ Veselovskii V.E._ Zlotov A.V._ Kaldibaev S.U._ Kaliev E.J._ Kovalenko A.G._ Montlevich V.M._ Sigal I.H._ Hachaturov R.V. Kombinatornie metodi i algoritm resheniya zadach diskretnoi optimizacii bolshoi razmernosti. Moskva_ «Nauka»_ 2000.