Yugoslav Journal of Operations Research
2 (1992), Number 1, 13-32

# COMBINATORIAL METHODS FOR SOLVING PROBLEMS OF OPTIMAL PORTIONAL RESOURCE DISTRIBUTION

V. N. BURKOV, M. I. RUBINSTEIN

*Institute of Control Problems*
*Moscow, Profsoyuznaya 81*

## 1. INTRODUCTORY REMARKS AND FORMULATIONS

Problems of optimum resource distribution appear when there are several operations consuming a resource of the same kind and at the same time effectiveness of any operation fulfilment depends on the quantity of resources assigned to it while the total amount of resource is limited. The above situation occurs in a great number of applications. In many of them a distributed resource is quantified or we shall say, portional. One-type equipment units, production workers, standardized capacities, containers and even financial means can be considered as a portional resource if allotted sums are distributed with a preassigned accuracy.

Problems of portional resource optimum distribution require combinatorial methods in a formalized solution. This paper is dedicated to generalization of the known methods and development of new combinatorial methods of solving the problems of portional resource optimum distribution and the essential place in this work is given to the results of experimental testing of the proposed algorithms. Three mathematical models which are close to one another with respect to the applied solution methods are treated as representative applications.

The first out of the three mathematical models is considered as follows. Let $y_0$ be a total volume of a distributed portional resource, i.e. $y_0$ is integer. Let there be $M$ operations with the effectiveness of each $i$-th operation being determined by the effectiveness function $\Psi_i(y_i)$ where $y_i$ is the volume of resource assigned to the $i$-th operation ($y_i$ as well as $y_0$ are integers and $i = \overline{1, M}$). Then the problem, which will be further referred to as POPRD1, is formulated as follows:

$$\sum_{i=1}^{M} \Psi_i(y_i) \longrightarrow \mathrm{extr}_1 \tag{1.1}$$

subject to:

$$\sum_{i=1}^{M} y_i = y_0, \qquad y_0 - \text{integer} \tag{1.2}$$

$$y_{i1} \le y_i \le y_{i2}, \qquad y_i - \text{integer}, \quad i = \overline{1, M}. \tag{1.3}$$

It is easy to observe that a necessary and sufficient condition of problem (1.1)–(1.3) feasibility is

$$\sum_{i=1}^{M} y_{i1} \le y_0 \le \sum_{i=1}^{M} y_{i2}. \tag{1.4}$$

Note that the effectiveness functions $\Psi(y)$ can estimate the effect of operation fulfilment (for example, the cost of a produced item) as well as the expenses covering operations fulfilment (for example, the time spent). The objective function in (1.1) can be maximized ($\mathrm{extr}_1 = \max$) or minimized ($\mathrm{extr}_1 = \min$). As to the values $y_{i1}$ and $y_{i2}$, they can get minimal and maximal levels of resource consumption in separate operations.

Along with POPRD1 its minimax (maximin) variant is considered which is called POPRD2 and formulated as follows:

$$\underset{i=\overline{1,M}}{\mathrm{extr}_2}\{\Psi_i(y_i)\} \longrightarrow \mathrm{extr}_1 \tag{1.5}$$

with the constraints (1.2) and (1.3). In (1.5) $\mathrm{extr}_2$ and $\mathrm{extr}_1$ have opposite values: if $\mathrm{extr}_2 = \max$, then $\mathrm{extr}_1 = \min$ and vice versa. The objective function can be interpreted for example as a maximal duration of all operations fulfilment under the condition that these operations are being fulfilled in parallel. And the effectiveness function $\Psi_i(y_i)$ should be interpreted as the fulfilment time of the $i$-th operation with the resource $y_i$ assigned to it.

Incidentally, with the same interpretation of effectiveness function in POPRD1 it will represent a problem of minimization (with $\mathrm{extr}_1 = \min$) of the total time spent on fulfilment of the initial complex of operations, but under the condition of sequential operations fulfilment unlike those fulfilled in parallel in POPRD2. It should be noted that another "time" interpretation of POPRD1 can be considered. Let $y_0$ represent the time assigned for all operations fulfilment under the condition that they are sequentially fulfilled (in an arbitrary or the assigned order). Let the total time be distributed between operations on the assumption that it is quantized (this assumption is fully justified when considering the planning problems where the check periods include the entity of shifts, days, quarters, etc.). The effectiveness function in this case determines the cost of an operation, when it is given a time interval $y_i$ for its fulfilment. It is required to distribute times between operations so that the total cost of the complex of operations fulfilment will be minimized.

If we have in view the above POPRD1 interpretation then its generalization for the complex of operations which contains the constraints of the preceding, leads to the following problem which will be referred to as POPRD3 and formulated as follows.

Let the order of fulfilling the operations in the initial complex be assigned by the network $G_M^m$ consisting of $M$ arcs which correspond to operations and $m$ vertices separating the preceding operations from those directly following them. With no discrepancy in the system of the preceding constraints, in the network $G_M^m$ there are no circuits. Besides, without limiting the generality, we may consider that in the

initial network there exists one initial vertex with no input arcs and one final vertex with no output arcs. These vertices are given the indices 1 and $m$, respectively. Let us associate to the set of arcs $I_0 = \{1, 2, \ldots, M\}$ of the network $G_M^m$ the function $g(i)$ and $h(i)$. For each arc $i \in I_0$, $g(i)$ and $h(i)$, the initial and final vertices, are respectively indicated. Further consider the character of a resource distributed in POPRD3 (time is a distributed) and introduce the variables $z_j$ ($j \in J_m$) time of occurring events, connected with the vertex $j$. The time $z_j$ of any event occurred for $j > 1$ will be a period of time from the moment of complex operations start till the completion of all operations, represented in the complex network by arcs included in the vertex $j$ (hence, for example, $z_1 = 0$). Now POPRD3 can be formulated as follows in the variables $z_j$:

$$\sum_{i=1}^{M} \Psi_i[z_{h(i)} - z_{g(i)}] \longrightarrow \mathrm{extr}_1 \tag{1.6}$$

with the constraints

$$z_{h(i)} - z_{g(i)} \geq y_{i1}, \qquad i \in I_0, \tag{1.7}$$

$$z_m - z_1 = y_0, \tag{1.8}$$

$$z_j \quad \text{are integers,} \quad j = \overline{1, m}. \tag{1.9}$$

With the POPRD3 formulation in the form (1.6)–(1.9) it is assumed that for $y_i > y_{i2}$ the functions $\Psi_i(y_i)$ are additionally determined $\Psi_i(y_i) = \Psi_i(y_{i2})$ ($i = \overline{1, M}$). This assumption allows us not to take into account the upper constraint on the difference $[z_{h(i)} - z_{g(i)}]$ in (1.7) and realize transition from variables $z_j$ ($j \in J_m$) to variables $y_i$ ($i \in I_0$) by the formula

$$y_i = \min\{z_{h(i)} - z_{g(i)}, y_{i2}\}, \tag{1.10}$$

It is easy to see that POPRD1 in the last cost-time interpretation represents a particular case of POPRD3 under the condition that $G_M^m$ consists of a unique path connecting the initial and final vertices. A similar cost-time interpretation of POPRD2 also represents a particular case of POPRD3 under the condition that $G_M^m$ consists of parallel arcs, connecting the initial vertex with the final one.

The above problems of optimal distribution of a portional resource have been considered earlier. POPRD1 under a special condition (the convexity of effectiveness functions) was considered in [1]. In Section 2 of this paper the corresponding result will be generalized for some other conditions (conditions 1–4).

Consideration of this problem in Section 2 proves that under conditions of the monotonicity of efficiency functions, the algorithm of POPRD2 solution practically coincides with the algorithm of POPRD1 solution under conditions 1–4.

POPRD1 in a general case was considered in many papers (see, for example, monograph [2]). In papers [3, 4] solution algorithms of this problem were outlined and partially investigated. In Section 3 a complete investigation of these algorithms will be given.

Effectiveness of POPRD3 for convex functions is investigated in [5]. The continuous variant of POPRD3 for convex functions is considered in [6]. The algorithm of POPRD3 is not, in fact, a "projection" of the algorithm for solving of a continuous problem from [6]. It is constructed by using a different iterative scheme.

## 2. PROBLEM OF OPTIMAL RESOURCE DISTRIBUTION BETWEEN INDEPENDENT PROCESSES

The problem formulated in (1.1)–(1.3) will be considered, additional constraints (C1–C4) imposed on the functions $\Psi_i(y_i)$):

C1: $\mathrm{extr}_1 = \max$, $\Psi_i$ are nondecreasing and convex upwards;

C2: $\mathrm{extr}_1 = \min$, $\Psi_i$ are nonincreasing and convex downwards;

C3: $\mathrm{extr}_1 = \max$, $\Psi_i$ are nonincreasing and convex upwards;

C4: $\mathrm{extr}_1 = \min$, $\Psi_i$ are nondecreasing and convex downwards.

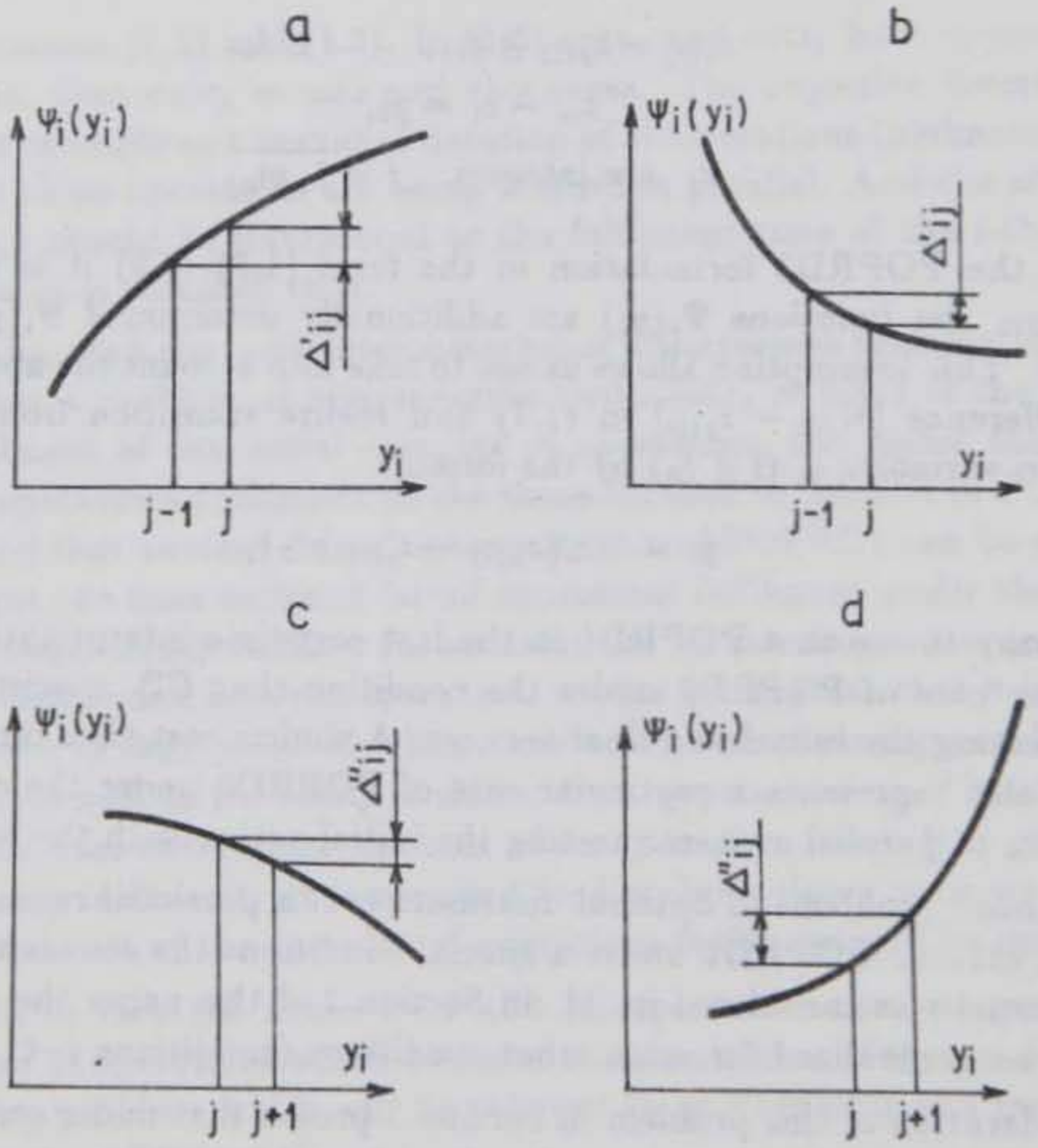



Fig. 1. Approximate form od the $\Psi_i(y_i)$ functions

Approximate form od the $\Psi_i(y_i)$ functions is shown in Fig. 1 where the variants a)–d) correspond to conditions C1–C4. POPRD1 under additional conditions C1–C4 will be repectively denoted as POPRD11–POPRD14.

Let us begin consideration with POPRD11. Denote by $\Delta'_{ij}$ the values to be determined in the following way (see Fig. 1.a):

$$\Delta'_{ij} = \Psi_i(j) - \Psi_i(j-1), \quad j = y_{i1}+1, \ldots, y_{i2}, \ i \in I_0. \tag{2.1}$$

Determine the value $M' = \left(\sum_{i=1}^{i=M} y_{i2}\right) - y_0$. Note that by virtue of (1.4) the total number of elements in the set of all values $\{\Delta'_{ij}\}$ determined in (2.1) is not smaller than $M'_0$. Now we can formulate two assertions.

ASSERTION 1. *Let $M'$ of minimal values $\Delta'_{i_l j_l}$ $(l = \overline{1, M})$ be chosen from the set of values $\{\Delta'_{ij}\}$ determined from* (2.1). *Then the chosen values can be distributed among the sequences of the following form*

$$\{\Delta'_{i'j'}, \ j' = y_{i'2} - k'_{i'} + 1, y_{i'2} - k'_{i'}, \ldots, y_{i'2}\}, \quad i' \in I', \tag{2.2}$$

*where $I'$ is some subset of the index set $I_0$ which includes such $i'$ that $k_{i'} > 0$ and $\sum_{i' \in I'} k_{i'} = M'$.*

PROOF follows directly from condition C1 (see Fig. 1.a) which provides monotone non-decreasing property of $\Delta'_{i'j'}$ in each of the sequences (2.2).

ASSERTION 2. *Optimal solution POPRD11 $\{y'_{i0}, \ i \in I_0\}$ is determined by values $k_{i'}$ and subset $I'$ appearing in Assertion* 1:

$$y'_{i0} = y'_{i2} - k_{i'}, \qquad i \in I';$$
$$y'_{i0} = y'_{i2}, \qquad i \in I_0 \setminus I'.$$

PROOF. It is easy to observe that the value of the objective function of POPRD11 for any solution $\{y'_{i0}, i \in I_0\}$ (or briefly, for solution $\{y'_{i0}\}$) differs from its value for the solution $\{y'(i2)\}$ in the sum of some $M'$ values from the sets $\{\Delta'_{ij}\}$. However, as it follows from Assertion 1, the solution $\{y'_{i0}\}$ differs from the solution $\{y'_{i2}\}$ in the sum $M'$ of minimal values from the set $\{\Delta'_{ij}\}$ which proves its optimality.

Assertions 1 and 2 for POPRD11 are directly transferred to POPRD12 if the values $\Delta'_{ij}$ are not determined in the way it was done in (2.1) but they are determined as follows: $\Delta'_{ij} = \Psi_i(j-1) - \Psi_i(j)$. Now consider POPRD13. Denote by $\Delta''_{ij}$ the values determined as follows (see Fig. 1.a);

$$\Delta''_{ij} = \Psi_i(j) - \Psi_i(j+1), \quad j = y_{i1}, y_{i1}+1, \ldots, y_{i2}-1, \quad i \in I_0. \tag{2.3}$$

Further determine the value $M'' = y_0 - \sum_{i=1}^{i=M} y_{i1}$. Note that by virtue of (1.4) the total number of elements in the set of values $\{\Delta''_{ij}\}$ from (2.1) is not smaller than $M''$. Now for POPRD13 two assertions similar to Assertions 1 and 2 can be formulated.

ASSERTION 3. *Let $M''_0$ of minimal values $\Delta''_{i_l j_l}$ $(l = \overline{1, M''})$ be chosen for the sets of values $\{\Delta''_{ij}\}$. Then the chosen values can be distributed among the sequences of the form*

$$\{\Delta_{i''j''}, \ j'' = y_{i''1}, y_{i''1}+1, \ldots, y_{i''1}+k''_{i''}-1\}, \quad i'' \in I'',$$

*where $I''$ is some subset of the index set $I_0$ including such $i''$, that $\sum_{i'' \in I''} k''_{i''} = M''$.*

ASSERTION 4. *Optimal solution of POPRD13 $\{y''_{i0}, i \in I_0\}$ is determined by the values $k''_i$ and subset $I''$ appearing in Assertion 3 as follows:*

$$y''_{i0} = y_{i2} - k''_{i''}, \qquad i \in I'';$$
$$y''_{i0} = y_{i2}, \qquad i \in I_0 \setminus I''.$$

Proof of Assertions 3 and 4 is presented similarly to the proof of Assertions 1 and 2.

Assertions 3 and 4 are transferred directly from POPRD13 to POPRD14. And at the same time $\Delta''_{ij} = \Psi_i(j+1) - \Psi_i(j)$ (compare with (2.3)).

On the basis of Assertions 1–4, the solution algorithm for POPRD11-POPRD14 can be substantiated. To simplify the description of this algorithm $A_1(y_0, M, Y_1, Y_2, \{\Psi_i\})$ ($Y_1$ and $Y_2$ are $M$-dimensional vectors with the components $y_{i1}$ and $y_{i2}$ respectively), we introduce some unified notations ($y$ is an integer scalar):

$$\overline{M} = \begin{cases} M', & \text{when solving POPRD11 and POPRD12,} \\ M'', & \text{when solving POPRD13 and POPRD14;} \end{cases}$$

$$\Delta(i, y) = \begin{cases} \Psi_i(y) - \Psi_i(y-1), & \text{in solving POPRD 11,} \\ \Psi_i(y-1) - \Psi_i(y), & \text{in solving POPRD 12,} \\ \Psi_i(y+1) - \Psi_i(y), & \text{in solving POPRD 13,} \\ \Psi_i(y) - \Psi_i(y+1), & \text{in solving POPRD 14.} \end{cases}$$

(Assignment of values $\Delta(i, y)$ in four above indicated cases is shown in Fig. 1). Now the algorithm $A_1$ can be represented as follows:

THE ALGORITHM $A_1(y_0, M, Y_1, Y_2, \{\Psi_i\})$

*Initial step*. Components $y_{i0}$ of a desired solution $y_0$ are given initial values $y_{i2}$ (in solving POPRD11 and POPRD12) or initial values $y_{i1}$ (in solving POPRD13 and POPRD14). For the formed solution we determine the set of values $\{\Delta(i, y_{i0}), i \in I_0\}$.

*Iterative step* (to be repeated $M_0$ times). Determine the index $i'$, such that

$$\Delta(i', y_{i'0}) = \min_{i \in I_0} \{\Delta(i, y_{i0})\}.$$

For the found index $i'$ we assume $y_{i'0} = y_{i'0} + \delta$, where $\delta = -1$ in solving POPRD11, POPRD12 and $\delta = +1$ in solving POPRD13, POPRD14. Then for the same index $i'$ we additionally calculate the value $\Delta(i', y_{i'0})$. In this connection, if $y_{i'0} = y_{i'1}$ (in case of POPRD11 and POPRD12 solution) or $y_{i'0} = y_{i'2}$ (in case of POPRD13 and POPRD14 solution) we assume $\Delta(i', y_{i'0}) = \text{const}_1$, where $\text{const}_1$ is a sufficiently large constant ($\text{const}_1$ can be given the value $\max_{i=\overline{1,M}}\{|\Psi_i(y_{i2}) - \Psi_i(y_{i1})|\}$).

Analyze the algorithm $A_1$ from the viewpoint of its complexity estimate. Here and in what follows there will be either asymptotical complexity estimates, determining the character of its value dependence on the main problem parameters with their sufficiently large values, or experimental complexity estimates which represent average (by the series of experiments with the algorithm) times of calculation of initial data randomly generated.

Regardless of the character of the complexity estimate $\nu_1$ of algorithm $A_1$ it can be represented as follows:

$$\nu_1 = \nu_1' + M\nu_1'', \tag{2.4}$$

where $\nu_1'$ and $\nu_1''$ are complexity estimates for the algorithm initial and iterative steps, respectively. Relation (2.4) will be applied for obtaining the complexity estimates of two modifications of the algorithm $A_1$ — algorithms $A_1^{(1)}$ and $A_1^{(2)}$, in which we realize in various ways one of the three mass operations, i.e. an operation of finding the minimal value out of the values $\{\Delta(i, y_{i0}), i \in I_0\}$ (other two mass operations are operations of additional calculation of the values $y_{i0}$ and $\Delta(i, y_{i0})$). Thus, consider algorithms $A_1^{(1)}$ and $A_1^{(2)}$.

In the algorithm $A_1^{(1)}$ the operation of finding the minimal out of a finite number of values is realized by means of the simplest algorithms of ordering $A_2(ISX, M)$, where $ISX$ is the array of minimal values and $M$ is its length.

ALGORITHM $A_2(ISX, M)$

*Step* 1. Set $i_0 = 1$, $i = 1$.

*Step* 2. Set $i = i + 1$ and compare values $ISX(i)$ and $ISX(i_0)$. If the first one happens to be strictly smaller than the second one, set $i_0 = i$.

*Step* 3. If $i = M$, finish the procedure. In the opposite case go back to step 2.

It is easy to observe that by the end of the algorithm $A_2$ operation the minimal value in the array $ISX$ coincides with $ISX(i_0)$. Asymptotical complexity estimate of algorithm $A_2$ is evidently determined by the value

$$\nu_2 = O(M). \tag{2.5}$$

(The value $\nu_2$ in (2.5) estimates the number of the simplest operations of adding and comparing type fulfilled by the algorithm $A_2$. The asymptotical complexity estimates of other algorithms, which will be described further in the text, have the same meaning).

In the algorithm $A_1^{(2)}$ the operation of finding the minimal value out of the finite number of values is realized with the use of a special structure $S_1$ which is formed before the first iteration of the algorithm and corrected in the course of its iterative procedure. Describe the structure $S_1$ and algorithms of its construction and correction. The considered structure $S_1$ will be characterized by two integral

arrays $II$ and $IK$, where $II$ is an array of the initial values indices, and $IK$ is an array of "internal coordinates" of the initial values indices in the structure $S_1$. The initial values array will be, as before, denoted by $ISX$, considering its length as equal to $M$. Lengths of the arrays $II$ and $IK$ are equal and they will be further determined. The structure $S_1$ assigned by the arrays $II$ and $IK$ can be schematically represented in the form of a tree $D_1$. The lower level of this tree corresponds to various values of the initial array $ISX$. Each new (higher) level of the tree is generated as follows: all vertices of the lower level are divided into pairs and the best vertex of each pair (the best vertex is the one which corresponds to the smallest value in the array $ISX$) is "transferred" to a generated level. The vertex which has not found a pair for itself also passes to the generated level. Construction of the tree $D_1$ ends when the level consisting of one (root) vertex is found. Vertices of the tree $D_1$ will be labeled by the indices of the corresponding array $ISX$ values, however in order to show the connection of this array with the tree $D_1$ it will be denoted by $D_1(ISX, M)$. Fig. 2.a shows the tree $D_1(ISX, M)$ for the array $ISX = (5, 2, 8, 1, 1.8, 4.3, 7.8, 8.0, 9.1, 10.2, 2.8)$.

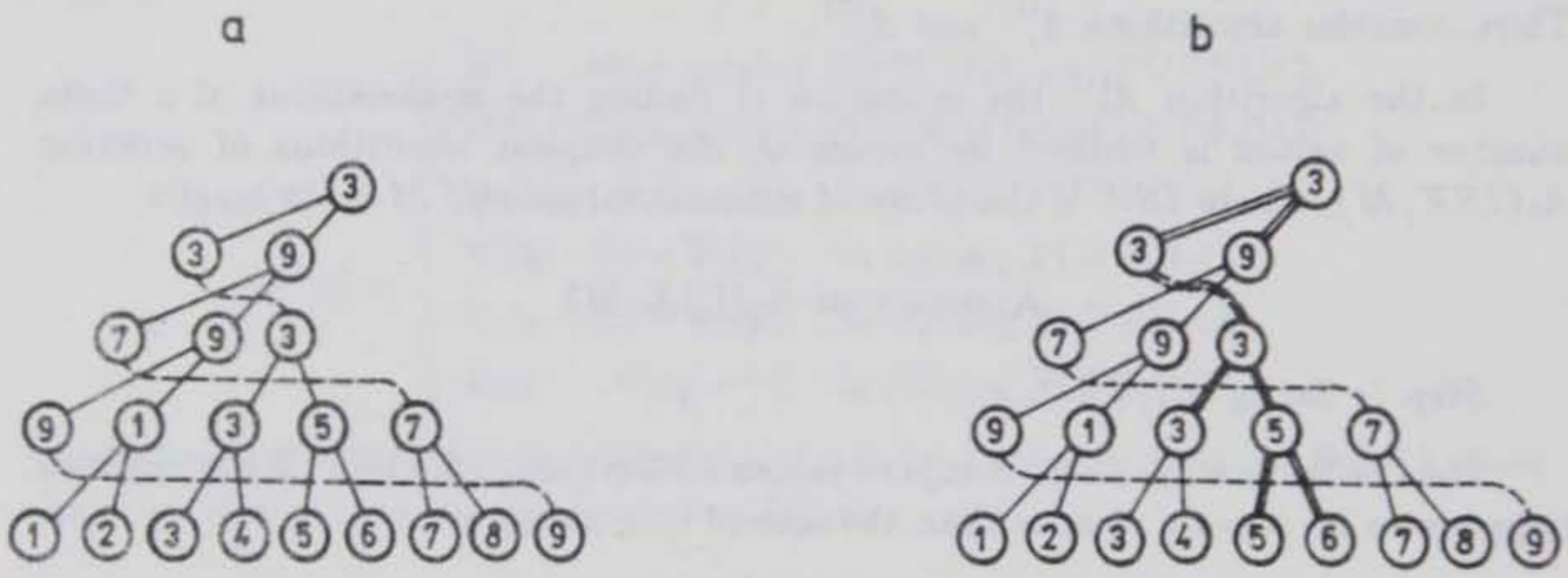



Fig. 2. The tree $D_1$ and the path of the algorithm

Between arrays $II$, $IK$ and the tree $D_1$ of the considered structure $S_1$ there is correspondence. In array $II$ the indices of the array $ISX$ values lie in the following order: first the indices corresponding to the vertices of the lower level, then the indices corresponding to the vertices of the next level, etc. The indices corresponding to vertices passing over from one level of the $D_1$ tree to another one (which is not the result of comparison with other vertices but because of the absence of their pair) are included only in that part of array $II$, which corresponds to a higher level of the tree $D_1$. As to the coordinate of array $IK$, zeros alternate with nonzero indices. For each pair of the vertices of the tree $D_1$ generated in the course of a higher level formation, 0 in the array $IK$ corresponds to the left vertex while the index which is in $IK$ to the right of it, indicates in the array $II$ of the internal coordinate the index of that vertex which passes over from a compared vertices pair to a generated level. It is easy to see that the number of levels $L_0$ in

the tree $D_1$ is determined only by the parameter $M$:

$$L = \begin{cases} \log_2 M + 1, & \text{if } M = 2^L, \\ [\log_2 M] + 2, & \text{otherwise,} \end{cases} \tag{2.6}$$

where $L$ is some integer and $[x]$, as usually, denotes an integral part $x$.

Now consider algorithms for construction and correction of the structure $S_1(ISX, M)$ which will be a composite part of the algorithm $A_1^{(2)}$. The above algorithms of construction and correction will be denoted by $A_3(ISX, M)$ and $A_4(ISX, M, i)$ respectively.

ALGORITHM $A_3(ISX, M)$

*Step* 1. Assume $m_0 = M$, $m' = -1$ as well as $II(j) = j$, $j = \overline{1, M}$.

*Step* 2. Assume $m' = m' + 2$, $m'' = m' + 1$. If $m'' > m_0$ the procedure should be finished; in the contrary case assume $m_0 = m_0 + 1$, $IK(m') = 0$, $IK(m'') = m_0$ and go to the next step.

*Step* 3. Compare two elements $j' = II(m')$ and $j'' = II(m'')$. If the element $j'$ is better than $j''$ (i.e. $ISX(J') < ISX(J'')$, assume $II(m_0) = J'$, otherwise $II(m_0) = J''$. Go back to step 2.

It is easy to show that the number of iterations of steps 2 and 3 in the algorithm $A_3$ is equal to $(M-1)$. This result follows from an obvious relation $2m = M + m - 1$ whose left part determines the total number of the tree $D_1$ vertices compared in the algorithm $A_3$, while the right part is the total number of the tree $D_1$ vertices (without the root vertex and vertices passing to a higher level with no "pairs" taken into account). Thus, asymptotical complexity estimate of the algorithm A9, has the form:

$$\nu_3 = O(m) - O(M). \tag{2.7}$$

Additionally it should be noted that, since $m = M - 1$, the length of arrays $II$ and $IK$ are $(2M - 1)$.

Now formulate the main feature of the structure $S_1$, generated by the algorithm $A_3$ in the form of assertion.

ASSERTION 5. *The last element in the array $II$, i.e. the element $II(2M - 1)$, indicates the minimal index of the values included in the array $ISX$.*

PROOF directly follows from the description of $A_3$. Passing over to the algorithm of structure $S_1$ correction, consider its two modifications — algorithm $A_4^{(1)}(ISX, M, i_0)$ and $A_4^{(2)}(ISX, M, i_0)$. Both algorithms introduce corrections in the structure $S_1$, connected with the change of the $i_0$ — the component in $ISX$. Algorithms $A_4(1)$ and $A_4(2)$ realize movement along the path connecting, in the tree $D_1$, the vertex $i_0$ with the root vertex. In the course of this movement the vertices included in this path are corrected. This correction is reduced to a possible

replacement of their indices. In the algorithm $A_4^{(1)}$ the above path goes to the the root vertex. In the algorithm $A_4^{(2)}$ the movement may end by the "arrival" to the vertex whose index is not changed in the correction process. Fig. 2.b shows how the tree $D_1(ISX, M)$, depicted in Fig. 2.a, is corrected when the value of $ISX(5)$ is changed from 7.8 to 3.9. The path of tree correction is shown in Fig. 2.b by double lines. The vertices participating in correction are connected to this path also by double lines. The algorithm $A_4^{(1)}$ "follows" the path (shown in Fig. 2.b) up to the root vertex. The algorithm $A_4^{(2)}$ "follows" this path only to the third (from below) level of the tree $D_1$ since the rule of the algorithm $A_4^{(2)}$ stop (simultaneous fulfilment of conditions $II(i'') = \tilde{j}$ and $j \neq i_0$) starts operating in the movement along the second level of the tree $D_1$.

It is easy to see that the number of Step 2–4 iterations in the algorithm $A_4$ coincides with the number of levels in the structure $S_1$ tree. Thus, the asymptotical complexity estimate of the algorithm $A_4^{(1)}$ by virtue of (2.6) gives the value

$$\nu_4 = O(\log_2 M). \tag{2.8}$$

Note, that the structure $S_1$, connected with the ranking of arrays of a finite length is described, for example, in [7] and in some earlier papers. However, such a detailed description of this structure and algorithms of its construction and correction are presented for the first time in this paper.

Now we are able to determine asymptotical complexity estimates of the algorithms $A_1^{(1)}$ and $A_1^{(2)}$. As to the algorithm $A_4^{(1)}$ the finding of minimal value among $\{\Delta(i, y_{i0}), i \in I_0\}$ is implemented by means of the algorithm $A_2$. Therefore, using (2.7) and (2.8), the asymptotical complexity estimate $\nu_1^{(1)}$ of the algorithm $A_1^{(1)}$ can be represented as

$$\nu_1^{(1)} = O(M + M\overline{M}) = O(M\overline{M}). \tag{2.9}$$

In the algorithm $A_1^{(2)}$ the finding of the minimal value among the values $\{\Delta(i, y_{i0}), i \in I_0\}$ is realized by means of the structure $S_1$ in one operation. However this structure in the algorithm $A_1^{(2)}$ should be, at first, initially constructed and, secondly, it should be corrected upon completion of each iteration. The first procedure is realized by means of the algorithm $A_3$, the second one — by means of the algorithm $A_4^{(1)}$. Using (2.4), (2,8) and (2.9) we determine the asymptotical complexity estimate $\nu_1^{(2)}$ of the algorithm $A_1^{(2)}$:

$$\nu_1^{(2)} = O(M + M\log_2 M). \tag{2.10}$$

Now consider POPRD2. The problem will be considered under each of the following additional conditions (C5, C6), imposed on $\Psi_i(y_i)$ $(i = \overline{1, M})$:

C5: $\Psi_i$ are non-decreasing;

C6: $\Psi_i$ are non-increasing.

POPRD2, where $\mathrm{extr}_1 = \max$, $\mathrm{extr}_2 = \min$ in fulfilling the condition 5 or $\mathrm{extr}_1 = \min$, $\mathrm{extr}_2 = \max$ under the condition 6, will be denoted by POPRD21 or POPRD22, respectively. POPRD2, where $\mathrm{extr}_1 = \min$, $\mathrm{extr}_2 = \max$ under the condition 5 or $\mathrm{extr}_1 = \max$, $\mathrm{extr}_2 = \min$ under the condition 6, will be denoted by POPRD23 or POPRD23, respectively.

The algorithm $A_6(y_0, M, Y_1, Y_2, \{\Psi_i\})$ similar to the algorithm $A_1$ can be applied to solve POPRD21–POPRD24. Before describing the algorithm $A_6$ we introduce the notation used in its description:

$$\overline{M} = \begin{cases} \left(\sum_{i=1}^{i=M} y_{i2}\right) - y_0, & \text{in solving POPRD21 and POPRD22,} \\ y_0 - \left(\sum_{i=1}^{i=M} y_{i1}\right), & \text{in solving POPRD3 and POPRD24;} \end{cases}$$

$$\Psi(i, y) = \begin{cases} \Psi_i(y-1), & \text{in solving POPRD21 and POPRD22,} \\ \Psi_i(y+1), & \text{in solving POPRD23 and POPRD24;} \end{cases}$$

$$\Psi(i, y_{i1}) = \mathrm{const}_2 = \min\{\Psi(i, y_{i1})\} - 1, \quad \text{in solving POPRD21;}$$

$$\Psi(i, y_{i1}) = \mathrm{const}_2 = \min\{\Psi(i, y_{i1})\} + 1, \quad \text{in solving POPRD22;}$$

$$\Psi(i, y_{i2}) = \mathrm{const}_4 = \min\{\Psi(i, y_{i2})\} + 1, \quad \text{in solving POPRD23;}$$

$$\Psi(i, y_{i2}) = \mathrm{const}_5 = \min\{\Psi(i, y_{i2})\} - 1, \quad \text{in solving POPRD24.}$$

ALGORITHM $A_6(y_0, M, Y_1, Y_2, \{\Psi_i\})$

*Initial Step*. Ascribe to the $y_{i0}$ of the desired solution $Y_0$ the initial $y_{i2}$ (in solving POPRD21 and POPRD22) or initial values $y_{i1}$ (in solving POPRD23 and POPRD24).

*Iteration step (to be repeated M times)*. Determine the index $i'$ such that $\Psi(i', y_{i'0}) = \mathrm{extr}_{i \in I_0}\{\Psi(i, y_{i'0})\}$. For the found index $i'$ assume $y_{i'0} = y_{i'0} + \delta$, where $\delta = -1$ in solving POPRD21, POPRD22 and $\delta = +1$ in solving POPRD23, POPRD24. Then for the same index $i'$ the value $\Psi(i', y_{i'0})$ should be recalculated.

The proof that the algorithm $A_6$ provides an accurate solution of POPRD21–POPRD24 is trivial.

## 3. PROBLEMS OF OPTIMAL RESOURCE DISTRIBUTION BETWEEN INDEPENDENT OPERATIONS WITH ARBITRARY RETURN FUNCTIONS

We start considering POPRD1. Now consideration will be fulfilled with no additional conditions limiting the kind of functions $\Psi_i(y_i)$. The corresponding problem will be referred to as POPRD15. For its solution we propose the algorithm constructed by the scheme of dynamic programming.

Description and investigation of the algorithm require introduction of some notation. Denote by $\{P(y, I)\}$, where $I$ is a fixed subset of the set $I_0 = \{1, 2, \ldots, M\}$,

the family of problems where each of them with an assigned integer parameter $y$ is formulated as follows:

$$\sum_{i\in I} \Psi_i(y_i) \longrightarrow \mathrm{extr}_1 \tag{3.1}$$

under constraints

$$\sum_{i\in I} y_i = y, \tag{3.2}$$

$$y_{i1} \le y_i \le y_{i2}, \qquad y_i \text{ — integer}, \quad i \in I. \tag{3.3}$$

To provide feasibility of problems from the family $\{P(y, I)\}$ it is necessary that the integral parameters $y$ satisfy the constraint

$$\sum_{i\in I} y_{i1} = y_1(I) \le y \le y_2(I) = \sum_{i\in I} y_{i2}. \tag{3.4}$$

Denote by $\Psi(y, I)$ the optimal value of the objective function of the problem $P(y, I)$. $\{\Psi(y, I)\}$ will denote the table of values $\Psi(y, I)$ that corresponds to various values of the integral parameter $y$, satisfying (3.4). It is evident that any table $\{\Psi(y, I)\}$ has a dimension (length) $n(I)$, where $n(I) = y_2(I) - y_1(I) + 1$ (values $y_1(I)$ and $y_2(I)$ are determined in (3.4)).

The basis of the considered algorithm of dynamic programming is the following easily verifiable relation which is true for any (complete) partitioning of the subset $I'$ of the set $I_0$ into the eigen subsets $I'_1$ and $I'_2$, and any integer $y$ from the interval $[y_1(I'), y_2(I')]$:

$$\Psi(y, I') = \operatorname*{extr}_{y_1(y,I'_1,I'_2)\le y'\le y_2(y,I'_1,I'_2)} \{\Psi(y', I') + \Psi(y - y', I'_2)\}, \tag{3.5}$$

where

$$\begin{aligned} y_1(y, I'_1, I'_2) &= \max\{y_1(i'_1), y - y_2(I'_2)\}, \\ y_2(y, I'_1, I'_2) &= \max\{y_2(i'_1), y - y_1(I'_2)\}. \end{aligned} \tag{3.6}$$

As a rule the algorithm for dynamic programming includes two subalgorithms: the first one which constructs the system of the objective function optimal values can be conditionally called a direct move while the second one is to be called a reverse move. For brevity the problems family $\{P(y, I')\}$ will be denoted by $P(I')$ and simply called — problems. Respectively, the table $\{\Psi(y, I)\}$ for the problem $P(I')$ is denoted by $\Psi(I')$. Consider first the algorithm of the "direct move" (the algorithm $A_7$). With its help the value of the objective function $\Psi(y_0, I_0)$ is constructed for the initial problem $P(y_0, I_0)$).

ALGORITHM $A_7[P(y_0, I_0)]$

*Initial Step*. Include in the current front all problems $P(i)$, where $i \in I_0$, and solve each of the front problems, having constructed the tables $\Psi(i)$ for them.

*Iterative Step* (to be repeated till the obtaining of a current front, which consists of two problems $P(I_1)$ and $P(I_2)$ with $I_1 \cup I_2 = I$). Choose two problems

$P(I_1')$ and $P(I_2')$ in the current front. Then, using their tables $\Psi(I_1)$ and $\Psi(I_2)$ and by means of (3.7), construct the table $\Psi(I_1' \cup I_2')$ for the problem $P(I_1' \cup I_2')$. Replace in the current front the problems $P(I_1')$ and $P(I_2')$ by the problem $P(I')$ where $I' = I_1' \cup I_2'$ (from the step description it is easy to conclude that for any current front composed from the problems $P(I_k)$, where $k = \overline{1, K}$, we always have $I_1 \cup I_2 \cup \ldots I_k = I_0$).

*Concluding Step.* Using two unique problems $P(I_1)$ and $P(I_2)$ of the current front and by means of (3.5) from tables $\Psi(I_1)$ and $\Psi(I_2)$ we obtain the optimal value $\Psi(y_0, I_0)$ of the objective function of the problem $P(y_0, I_0)$.

Note that with the arbitrary $\Psi_i(y_i)$ the table $\Psi(y')$ can be constructed by means of (3.5) with respect to tables $\Psi(I_1')$ and $\Psi(I_2')$ only by scanning all values of the parameter $y'$. An admissible region of interval parameter $y'$ changing is to be determined by the following relations (see 3.5), (3.6)):

$$y_1(y, I_1', I_2') \le y' \le y_2(y, I_1', I_2'), \qquad y_1(I') \le y \le y_2(I'). \tag{3.7}$$

The described region (3.7) is shown in Fig. 3.a and Fig. 3.a corresponds to the case $y_1(I_1') + y_2(I_2') \ge y_2(I_1') + y_1(I_2')$ $(n(I_2') \ge n(I_1'))$, while Fig. 3.b corresponds to the case, determined by a reverse inequality.

From Fig. 3 it is easy to conclude, that the volume of a direct scanning in construction of the table $\Psi(I')$ with the help of (3.5) amounts to the value

$$\nu(I') = O[n(I_1'), n(I_2')]. \tag{3.8}$$

From the same Fig. 3 it can be seen that obtaining of the optimal value of the objective function of the problem $P(y', I')$ (for some fixed value $y'$) is connected with scanning whose volume is estimated by the value $O[\max\{n(I_1'), n(I_2')\}]$.

From (3.8) the following can be directly obtained.

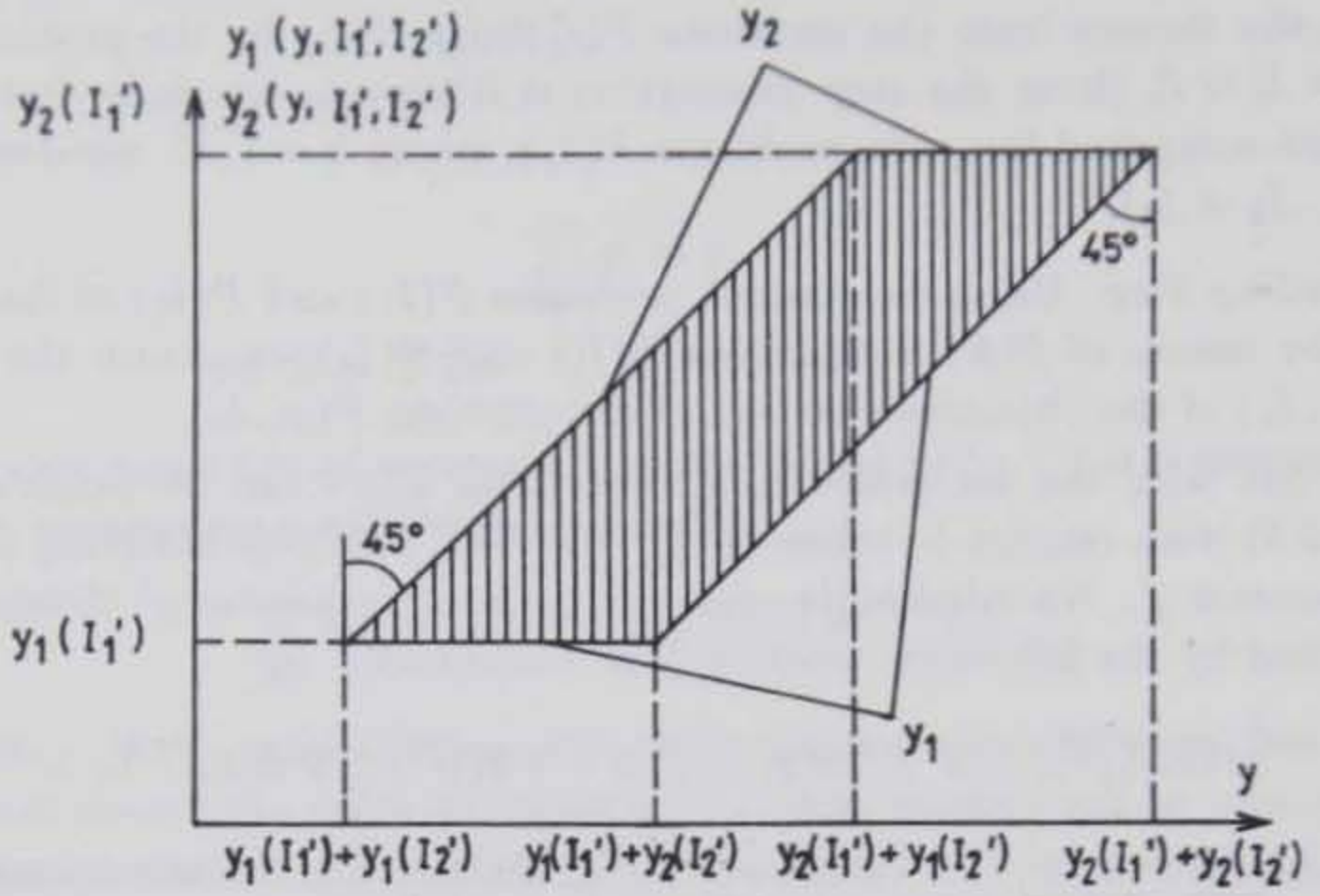
ASSERTION 6. *Asymptotical complexity estimate of table $\Psi(I_0)$ construction in the algorithm $A_7$ amounts to the value*

$$\nu(I_0) = O\Big( \sum_{i_1, i_2 \in I_0 \ (i_1 \ne i_2)} n(i_1) n(i_2) \Big). \tag{3.9}$$

It should be noted that the result formulated in Assertion 6 does not depend on the rule of choosing the pair of problems in the current front in the iterative step of the algorithm $A_7$. However, efficiency of a reverse move subalgorithm to be considered below essentially depends on the rule, applied. in the algorithm $A_7$, of choosing a pair of joined problems.

As to the rule of choice, we shall consider two such rules (in some sense, opposite ones). In both rules the ranks of problems $P(I')$ generated in the algorithm $A_7$ are used. These ranks, denoted by $r(I')$, are determined in the course of algorithm $A_7$. For the problems of the initial current front the ranks are assigned as zero

a




b

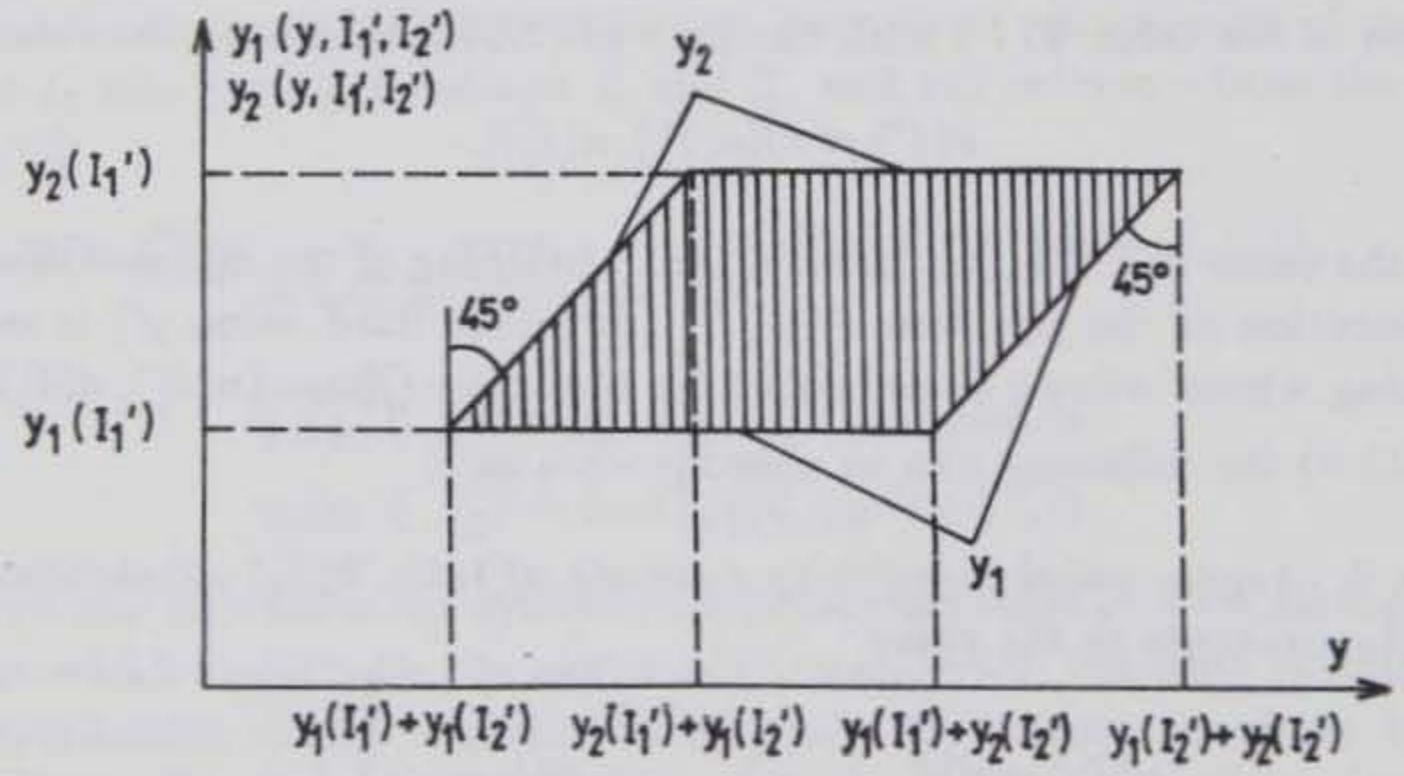



Fig. 3. Admissible regions of interval parameters

ones. With each correction of the front connected with the joining of the problems $P(I_1')$ and $P(I_2')$ in the problem $P(I')$ the rank of a new problem is determined by the formula

$$r(I') = \max\{r(I_1'), r(I_2')\} + 1. \tag{3.10}$$

*Rule* 1. One of the chosen problems in the current front has a minimal rank and the other has a maximal one.

*Rule* 2. Both chosen problems have minimal ranks with respect to other problems of the front.

Action of Rule 1 and 2 in the algorithm $A_7$ can be observed on some tree $D_2$. The tree $D_2$ vertices correspond to problems $P(I')$ where the algorithm $A_7$ operates while the arcs connect each pair of vertices corresponding to the joined problems $P(I'_1)$ and $P(I'_2)$ with the vertex of a corresponding problem $P(I'_1 \cup I'_2)$. The tree $D_2$ for the algorithm $A_7$ which uses Rules 1 and 2 is shown in Fig. 4.a and 4.b, respectively (in both cases $M = 9$).

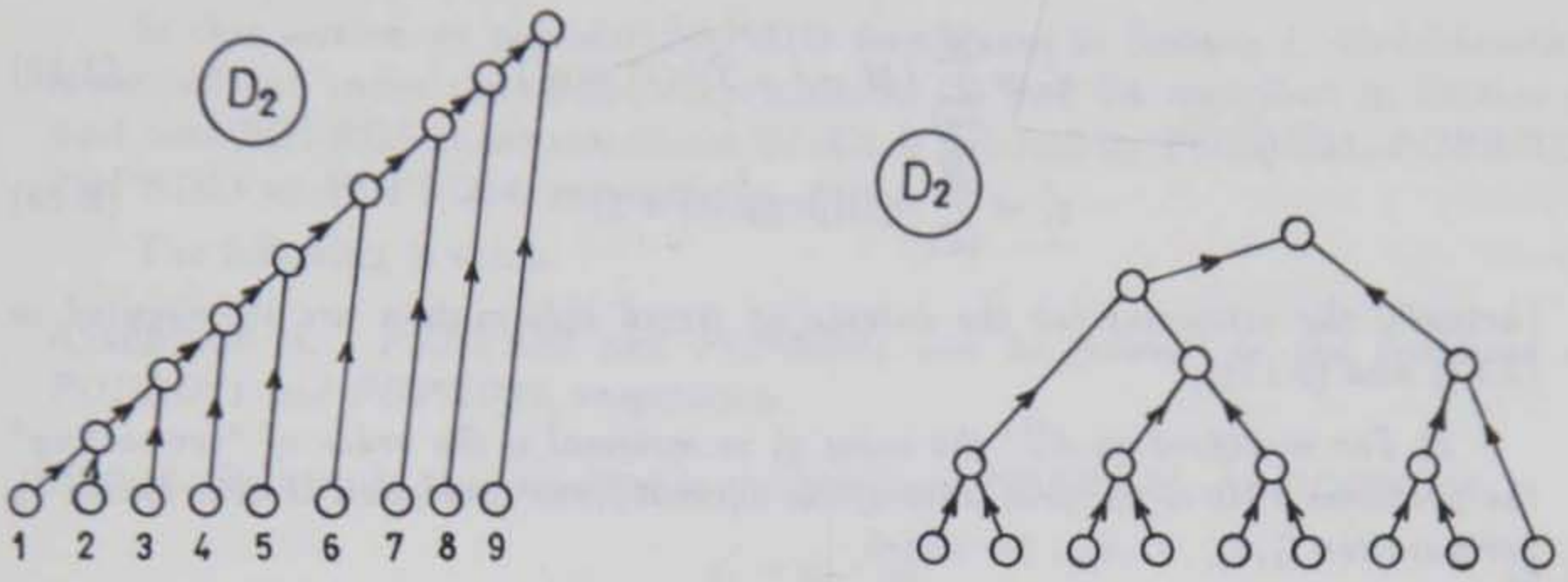



Fig. 4. The tree $D_2$ for the algorithm $A_7$

Efficiency of the reverse move algorithm is influenced not only by the rule of choosing the joined problems in the algorithm $A_7$ but also by the variants of generating and storage of the intermediate information.

A remark of a general kind should be made. Up to now we have considered only asymptotical complexity estimates. But in many cases estimates of a required memory volume are important too. These estimates are especially important for the algorithms of dynamic programming, where a necessity of creating and processing relatively large arrays of intermediate information occurs.

Consider two variants of storing intermediate information in the course of algorithm $A_7$ work.

*Variant* 1. The tables $\Psi(I')$ of all problems $P(I')$ generated by the algorithm $A_7$ are to be stored.

*Variant* 2. The tables of only those of problems $P(I'_k)$, which compose the current front, are to be stored. And the table of each new problem is located in that region of memory which was taken by the problems "joined by it".

Consider two modifications of the algorithm $A_7$. In the first one, which is referred to as the algorithm $A_7^{(1)}$, in the iterative step of the algorithm $A_7$ one uses Rule 1 of the joined problems choice. In the second modification, to be denoted by $A_7^{(2)}$, in the iterative step one uses Rule 2. In the algorithm of the direct and reverse moves of the considered solution of POPRD15 (problems $P(y_0, I_0)$ the main component in the total volume of intermediate information is the summarized volume of the stored tables $\Psi(I')$. This volume will be denoted by $q'_7$ for variant 1 and by $q''_7$ for variant 2 of storing intermediate information. The following is valid:

ASSERTION 7. 1) *For both modifications* $A_7^{(1)}$ *and* $A_7^{(2)}$ *of the algorithm* $A_7$

$$q_7'' = \sum_{i=1}^{M} n(i). \tag{3.11}$$

2) *For modifications* $A_7^{(1)}$ *and* $A_7^{(2)}$ *of the algorithm* $A_7$ *respectively*

$$q_7' = \sum_{i=1}^{M} (M - i + 2)n(i) - n(1), \tag{3.12}$$

$$q_7' = \sum_{i=1}^{M} n(i)([\log_2 M] + 1) \tag{3.13}$$

(*actually the estimates for the volume of stored information are represented in* (3.12) *and* (3.13)).

3) *For modification* $A_7^{(1)}$ *the value* $q_7'$ *is minimal if the order of "connecting" the problems* $P(i)$ *to the procedure of the current front correction is determined by permutation* $i_1', i_2', \ldots, i_M'$, *for which*

$$m(i_1') \le n(i_2') \le \ldots \le n(i_M'). \tag{3.14}$$

Further, consider two modifications of the reverse move algorithm. One of them, denoted by $A_8^{(1)}$, is oriented to variant 1 and the other, one denoted by $A_8^{(2)}$, to variant 2 of storing the intermediate information, generated in the algorithm $A_7$. Let us accept that in variant 1 (tables $\Psi(I')$) are stored) of all problems, generated in the algorithm $A_7$ after "removal" of each problem $P(I'')$ from the current front, the table $\Psi(I'')$ for this problem is replaced by the table $\{y_1(y, I'')\}$ (or briefly, by the table $\{y_1(y)\}$, where $y_1$ is the optimal magnitude of the values $y'$ from (3.15). Let us also accept that in variant 2 of storing the intermediate information together with the tables $\Psi(I_k')$ of the current front problems the tables $i_1(I')$ of the same problems are also stored. To realize both conditions, it is necessary to form the table $y_1(I')$ parallel to forming of each table $\Psi(I')$ with the help of (3.15) and it does not change asymptotical estimates of the algorithm $A_7$ memory.

Since in the algorithm $A_8^{(2)}$ the algorithm $A_7$ must be used as a component, one can consider two modifications of the algorithm $A_8^{(2)}$: algorithms $A_8^{(2,1)}$ and $A_8^{(2,2)}$. And in the algorithm $A_8^{(2,1)}$ one can consider a modification $A_7^{(1)}$, and in the algorithm $A_8^{(2,2)}$ — a modification $A_7^{(2)}$ of the algorithm $A_7$.

Now the four modifications of the algorithm $A_9$ of POPRD15 solution can be constructed. These modifications will be referred to as algorithm $A_9^{(l_1,l_2)}$ with $l_1, l_2 \in \{1, 2\}$, considering that $l_1 = 1$ and $l_1 = 2$ correspond to rules 1 and 2 of choosing the problems in the iterative step of the algorithm $A_7$, and $l_2 = 1$ and $l_2 = 2$ correspond to variants 1 and 2 of storing the intermediate information, generated by the algorithm $A_7$. With fixed $l_1$ and $l_2$, the algorithm $a_9^{(l_1,l_2)}$ includes the algorithm $A_7^{(l_1)}$ as the direct move algorithm and as a reverse move algorithm

the $A_s^{(l_1)}$ algorithm will be included with $l_2 = 1$ and algorithm $A_s^{(2,l_1)}$ — with $l_2 = 2$.

## 4. PROBLEMS OF OPTIMAL RESOURCE DISTRIBUTION BETWEEN DEPENDENT OPERATIONS IN CASE OF CONVEX RETURN FUNCTIONS

In this section we consider POPRD3 formulated in Section 1. Consideration is carried out under the additional conditions C1 and C4 described in Section 2. And here POPRD3 under conditions C1–C4 is denoted by POPRD31, POPRD32, POPRD33 an POPRD34, respectively.

The following is valid:

ASSERTION 8. *POPRD33 and POPRD34 can be reduced to the problems of POPRD31 and POPRD32, respectively.*

PROOF. Substitute the variables in the problems POPRD33 and POPRD34

$$y_i' = \bar{y}_i - y_i, \tag{4.1}$$

where $\bar{y}_i$ is any constant not smaller than $y_{i2}$ ($i = \overline{1, M}$). It can be verified that in the new variables the monotonically increasing function $\Psi_i(y_i)$ becomes monotonically decreasing and vice versa. And the character of convexity (convexity downwards or convexity upwards) of the function $\Psi_i(y_i)$ is not changing. This proves the considered assertion (see also Fig. 1).

It should be noted that in substituting the variables in (4.1) the boundaries of changes of new variables $y_i'$ are determined as follows:

$$y_{i1}' = \bar{y}_i - y_{i2}, \qquad y_{i2}' = \bar{y}_i - y_{i1}.$$

It is easy to see that for the pairs POPRD11 and POPRD12, POPRD13 and POPRD14 the assertion analogous to Assertion 8 is valid. However, in Section 2 we did not resort to a corresponding consideration, analyzing each of the above pairs of problems separately. As to the problems POPRD31–POPRD34 the further considered algorithm of their solution can be directly applied only to POPRD21 and POPRD22. According to the kind of the objective functions, POPRD31 and POPRD32 can be respectively interpreted as the problems of maximization of a total effect or of the problem of minimization of total expenses in fulfilment of complex operations within an assigned time $y$.

Consider one important result which will be essential in construction of the algorithm $A_{10}(G_M^m, y_0, Y_1, Y_2, \{\Psi_i\})$ intended for solution of POPRD31 and POPRD32. Denote by $Z(y) = \{z_j(y),\ j = \overline{1, m}\}$, the optimal solution of POPRD31 or POPRD32 when replacing $y_0$ in (1.8) by $y$ (parameter $y$ is integer). Considering $\text{const}_6$ sufficiently large (for example, $\text{const}_6 = \sum_{i \in I_0} |\Psi_i(y_{i2}) - \Psi_i(y_{i1})|$) determine for each arc $i$ of the network $G_M^m$ the values $\Delta z_i(y)$, $c_i^{(1)}$ and $c_i^{(2)}$ ($i = \overline{1, M}$):

$$\Delta z_i(y) = z_{h(i)}(y) - z_{g(i)}(y); \tag{4.2}$$

$$c^{(1)} = \begin{cases} 0, & \Delta z_i(y) \geq y_{i2}, \\ |\Psi_i[\Delta z_i(y)+1] - \Psi_i[\Delta z_i(y)]|, & y_{i1} \leq \Delta z_i(y) < y_{i2}; \end{cases} \tag{4.3}$$

$$c^{(2)} = \begin{cases} \text{const}_6, & \Delta z_i(y) \leq y_{i1}, \\ |\Psi_i[\Delta z_i(y)] - \Psi_i[\Delta z_i(y)-1]|, & y_{i1} < \Delta z_i(y). \end{cases} \tag{4.4}$$

Denote by $G(y)$ the network $G_M^m$ and its arcs by means of (4.2)–(4.4). The values $c_i^{(1)}(y)$ and $c_i^{(2)}(y)$ of upper and lower capacities ($i \in I_0$) are assigned. On the network $G(y)$ one should consider various cuts which will be characterized either by corresponding full partitioning of the set of vertices $J_m$ into two subsets $J_1(y)$ and $J_2(y)$, where $1 \in J_1(y)$, $m \in J_2(y)$, or by corresponding subsets of arcs $u(y)$, whose one part $u_1(y)$ includes straight arcs (i.e. arcs $(j_1, j_2)$ for which $j_1 \in J_1(y)$, $j_2 \in J_2(y)$), and the other part $u_2(y)$ are the inverse arcs $(j_1', j_2')$ (i.e. arcs for which $j_1' \in J_2(y)$, $j_2' \in J_1(y)$). The value of any cut $u(y)$ in the network $G(y)$ is to be determined as follows:

$$c(u) = \sum_{i \in u_1(y)} c_i^{(2)}(y) - \sum_{i \in u_2(y)} c_i^{(1)}(y). \tag{4.5}$$

Denote by $\bar{u}$ or $(\bar{J}_1(y), \bar{J}_2(y))$ the minimal cut in the network $G(y)$, i.e., the cut for which the value in (4.5.) is minimal. Now we can formulate a necessary result.

ASSERTION 9. *Let POPRD31 (POPRD32) with $y_0 = y$ be admissible and have the solution $Z(y)$. Let the minimal cut $(J_1(y), J_2(y))$ be determined for the network $G(y)$. If the value of this cut does not exceed the value* $\text{const}_6$ *then solution $Z(y-1)$ POPRD31 (POPRD32) with $y_0 = y-1$ can be obtained from the solution $Z(y)$ as follows:*

$$z_j(y-1) = \begin{cases} z_j(y), & j \in \bar{J}_1(y); \\ z_j(y) - 1, & j \in \bar{J}_2(y). \end{cases} \tag{4.6}$$

The proof can be realized by the known scheme used in the analysis of a classical problem referred to as the minimal cost network problem [8, Chapter 3].

From Assertion 9 follows the possibility of recurrent construction of solution $\{Z(y)\}$ sequence. Construction of this sequence begins with the value of the parameter $y$, coinciding withy $y^{\max}$: $Z(y^{\max})$. The construction of the sequence $\{Z(y)\}$ should be finished with the value of the parameter $y$, coinciding with $y_0$.

ASSERTION 10. *Let $z_j$ be the length of a minimal path, connecting vertex 1 with vertex $j$ ($j > 1$, $z_1^{\max} = 0$) in the network $G_M^m$ with the arc lengths coinciding with the values $y_{i2}$ ($i \in I_0$). Then*

$$\begin{aligned} y^{\max} &= z_m^{\max}, \\ z_j(y^{\max}) &= z_j^{\max}, \qquad j \in J_m. \end{aligned} \tag{4.7}$$

PROOF. Optimality of the solution $Z(y^{\max})$, determined in (4.7) for POPRD31 or POPRD32, follows from the fact, that for all operations $i$ ($i \in \bar{I}_0$)

$$z_{h(i)}(y^{\max}) - z_{g(i)}(y^{\max}) \geq y_{i2}.$$

By virtue of monotonicity of the efficiency functions, it provides maximal total efficiency (POPRD31) minimal total expenses (POPRD32) when fulfilling all operations within the time $y^{\max}$. For the values of the parameter $y$ there also exists a lower boundary $y^{\min}$ and its violation leads to admissibility of corresponding problems. Let $z_m^{\min}$ be the length of the shortest path connecting vertex 1 with vertex $m$ in the network $G_M^m$ with the arcs lengths determined by the values $y_{i1}$ $(i \in I)$. Then

$$y^{\min} = z_m^{\min}. \tag{4.8}$$

Thus, a correct statement of POPRD31 and POPRD32 assumes that

$$y^{\min} \le y_0 \le y^{\max}. \tag{4.9}$$

On the basis of Assertions 9 and 10 the solution algorithm for POPRD31 and POPRD32 can be represented as follows:

ALGORITHM $A_{10}(G_M^m, y_0, Y_1, Y_2, \{\Psi_i\})$

*Initial step*. Construct the solution $Z(y^{\max})$ with the help of (4.7). Generate the network $G(y^{\max})$, i.e. by means of (4.2)–(4.4) assign the values $c_i^{(1)}(y_i^{\max})$ and $c_i^{(2)}(y_i^{\max})$ for each arc $i$ $(i \in I_0)$ of the network $G_M^m$.

*Iterative step* (to be repeated $(y^{\max} - y_0)$ times).

1. Determine a minimal cut $\{\bar{u}_1(y), \bar{u}_2(y)\}$ in the network $\bar{G}_M^m$. With the help of (4.6) construct the solution $Z(y-1)$ with respect to a found cut and the known solution $Z(y)$.

2. Determine the values of lower and upper capacities of the arcs in the network $G(y-1)$ using the following "recalculation formulas"

$$c_i^{(1)}(y-1) = \begin{cases} c_i^{(1)}(y), & i \notin \bar{u}_1(y) \cup \bar{u}_2(y); \\ c_i^{(2)}(y), & i \in \bar{u}_1(y); \\ |\Psi_i[\Delta z_i(y)+2] - \Psi_i[\Delta z_i(y)+1]|, & i \in \bar{u}_2(y). \end{cases} \tag{4.10}$$

$$c_i^{(2)}(y-1) = \begin{cases} c_i^{(2)}(y), & i \notin \bar{u}_1(y) \cup \bar{u}_2(y); \\ c_i^{(1)}(y), & i \notin \bar{u}_2(y); \\ |\Psi_i[\Delta z_i(y)-1] - \Psi_i[\Delta z_i(y)-2]|, & i \in \bar{u}_1(y). \end{cases} \tag{4.11}$$

3. Assume that $y = y - 1$ and complete the iteration.

An important component of the algorithm $A_{10}$ is the algorithm of constructing the minimal network cut or the algorithm of finding the maximal flow which solves the same problem.

In the classical statements of the network the problem dealing with the maximal flow in a network, has unilateral constraints on the arcs capacities. In the presence of bilateral constraints there appears the difficulty connected with creation

of the initial feasible flow. Such difficulty does not exist for the above algorithm: a zero flow is feasible in the first iteration, while a maximal flow obtained in the previous iteration is feasible in the current iteration (it follows from (4.10), (4.11)).

Consider asymptotical estimates of a work time $\nu_{10}$ and of a required memory $q_{10}$ for the algorithm $A_{10}$. The asymptotical estimate of a work time of the algorithm $A_{10}$ iterative step is determined by the analogous estimate for the algorithm of finding a maximal flow in the network $G_M^m$. The best estimate among the known algorithms has the form $\nu' = O(m^3)$ [9], hence, taking into account that the number of iterations $(y_{\max} - y_0)$ does not exceed the values $q_0 = \sum_{i \in I_0}(y_{i2} - y_{i1} + 1)$ and an initial step asymptotical estimate (determined by the analogous estimate of finding an extreme path in the network) has the order of $O(M^2)$, we obtain

$$\nu_{10} = O(q_0 m^2). \tag{4.12}$$

The volume of intermediate information in the algorithm $A_{10}$ is maximal when the algorithm of finding a maximum flow is used within the general procedure. For the best algorithm recommended for inclusion in $A_{10}$, such estimate has the form $O(M)$ from which $q_{10} = O(M)$.

## REFERENCES

[1] W. Shih: *A new application of incremental analysis of resource allocations*, Op. Res. Quart. **25** (1974), 587–597.

[2] G. Hadley: *Nonlinear and Dynamic Programming*, Addison-Wesley Publishing Company, London, 1964.

[3] A. A. Golos: *Hierarchic scheme of dynamic programming algorithms for performance control of design solutions*, ITK BSSR Academy of Sciences, Minsk, 1979, 85–93.

[4] A. A. Golos: *Hierarchical approach to multistep processes of decision making*, (ditto), 101–106.

[5] M. I. Rubinstein and A. M. Cherkashin: *Discrete problem of optimal resource allocation activity networks* Automation and Remote Control **41** (1), Part 2 (1980), 116–122.

[6] M. I. Rubinstein: *The network cost-optimization problem* Automation and Remote Control 3 (1970), 460–468.

[7] E. A. Dinitz: *On solving two problems of assignment*, in: *Investigations in Discrete Optimization*, Nauka, Moscow, 1976, 333–347.

[8] L. Ford, D. Fulkerson: *Flows in Networks*, Princeton University Press, 1962.

[9] E. A. Dinitz, A. V. Karzanov: *Flow Algorithms*, Nauka, Moscow, 1975, 158 pp.