Cybernetics and Systems Analysis, Vol. 40, No. 3, 2004

SOLUTION OF OPTIMIZATION PROBLEMS WITH FRACTIONAL-LINEAR OBJECTIVE FUNCTIONS AND ADDITIONAL LINEAR CONSTRAINTS ON PERMUTATIONS

O. A. Emets[a] **and L. N. Kolechkina**[b]

UDC 519.854

The statement of a problem of Euclidean combinatorial optimization with a fractional-linear objective function on a common set of permutations and with additional linear constraints is formulated. A problem with a fractional-linear objective function is transformed into that with a linear objective function. An approach is proposed to the solution of such problems, and a method of combinatorial truncation of solutions of problems of combinatorial type with fractional-linear objective functions on permutations is developed.

Keywords: *fractional-linear function, combinatorial set, permutation polyhedron, set of permutations, method of combinatorial truncation.*

STATEMENT OF THE PROBLEM

Let it be required to determine a pair $<F(t^*), t^*>$ such that

$$F(t^*) = \underset{t \in R^m}{extr} \frac{\sum_{j=1}^{m} c_j t_j + c_0}{\sum_{j=1}^{m} d_j t_j + d_0}, \quad t^* = \arg \underset{t \in R^m}{extr} \frac{\sum_{j=1}^{m} c_j t_j + c_0}{\sum_{j=1}^{m} d_j t_j + d_0}, \tag{1}$$

under a condition

$$x = (x_1, x_2, ..., x_k) \in E_{kn}(G) \subset R^k \tag{2}$$

and additional linear constraints

$$\sum_{j=1}^{m} \alpha_{ij} t_j \le b_i \quad \forall i \in J_r, \tag{3}$$

where $J_r = \{1, 2, \ldots, r\}$, $t = (t_1, t_2, ..., t_k, t_{k+1}, ..., t_m) \in R^m$, $x_j = t_j \; \forall j \in J_k$, m, k, n, and r are natural numbers, α_{ij}, b_i, c_j, d_j, c_0, and d_0 are real numbers $\forall j \in J_m$, $\forall i \in J_r$, $E_{kn}(G)$ is the total set of permutations [1], $m \ge k$, and G is a multiset consisting of k real numbers $g_1, g_2, \ldots, g_k$ [1] arranged in nondecreasing order. We call $F(t^*)$ the extremum and t^* the extremal of the problem. Assume that the condition $\sum_{j=1}^{m} d_j t_j + d_0 > 0$ is satisfied for any point $t = (t_1, \ldots, t_m)$ that belongs to the domain of problem (1)–(3).

[a]Yu. Kondratyuk University, Poltava, Ukraine, *slemets@e-mail.pl.ua* and *pmimm@pntu.poltava.ua.* [b]University of Consumer Cooperation, Poltava, Ukraine, *luda@ukr.net* Translated from Kibernetika i Sistemnyi Analiz, No. 3, pp. 30–42, May-June 2004. Original article submitted May 28, 2003.

To solve problem (1)–(3), we pass to a problem with a linear objective function and additional linear constraints by using the mappings φ

$$y_0 = \frac{1}{\sum_{j=1}^{m} d_j t_j + d_0}, \; z_j = t_j \cdot y_0 \quad \forall j \in J_m . \tag{4}$$

We denote $\varphi(t) = z$, $t = (t_1, t_2, ..., t_m)$, $z = (y_0, z_1, ..., z_m)$, and $\varphi(E_{kn}(G)) = E'$, $z_j = y_j \; \forall j \in J_k$. As a result of transformations (4), problem (1)-(3) is reduced to the determination of a pair $<\Phi(z^*), z^*>$ such that we have

$$\Phi(z^*) = \underset{z \in R^{m+1}}{extr} \left(\sum_{j=1}^{m} c_j z_j + c_0 y_0 \right); \; z^* = \arg \underset{z \in R^{m+1}}{extr} \sum_{j=1}^{m} (c_j z_j + c_0 y_0) \tag{5}$$

under a condition

$$y = (y_0, y_1, ..., y_k) \in E' \subset R^{k+1} \tag{6}$$

and additional linear constraints

$$\sum_{j=1}^{m} \alpha_{ij} z_j - b_i y_0 \le 0 \quad \forall i \in J_r, \tag{7}$$

where $z = (y_0, z_1, ..., z_k, z_{k+1}, ..., z_m) \in R^{m+1}$, $y_j = z_j \quad \forall j \in J_k$.

METHODS AND ALGORITHMS OF SOLVING PROBLEMS WITH FRACTIONAL-LINEAR OBJECTIVE FUNCTIONS ON PERMUTATIONS

In [2, 3], truncation methods are described for problems of Euclidean combinatorial optimization with linear objective functions and additional linear constraints on combinatorial sets E (permutable and polypermutable sets, sets of arrangements and polyarrangements, etc.) with the property

$$E = \text{vert conv } E . \tag{8}$$

For a problem of the form (5)–(7), this property is also true, i.e., we have

$$E' = \text{vert conv } E' = \text{vert} \, Q'_{kn}(G), \tag{9}$$

where the polyhedron $Q'_{kn}(G)$ is the convex envelope $\text{conv} \, E'$ of the set E'. This convex envelope is obtained in [4] as the solution of the system of linear constraints

$$\begin{cases} \sum_{j=1}^{k} y_j \le \sum_{j=1}^{k} g_j y_0, \\ \sum_{j=1}^{i} y_{a_j} \ge \sum_{j=1}^{i} g_j y_0, \\ \sum_{j=0}^{k} d_j y_j + \sum_{j=k+1}^{m} d_j z_j = 1, \end{cases} \tag{10}$$

$$y_0 > 0, \; y_j \ge 0 \; \forall j \in J_k, \; \alpha_j \in J_k, \; \alpha_j \ne \alpha_t \quad \forall j \ne t, \forall j, t \in J_i, \; \forall i \in J_k .$$

We call the *i*-th union of system (10) the totality of inequalities (10) in which *i* has the same value. Thus, to solve problem (1)–(3), its form (5)–(7) can be used for which it makes sense to use the combinatorial truncation method [2, 3].

A program realization of combinatorial truncation methods for problems with linear objective functions and results of numerical experiments are described in [5]. Based on the analysis of data obtained as a result of investigations pursued, the algorithms constructed on the basis of these methods [2, 3] can lead to the accumulation of a large amount of truncation constraints. Therefore, in the truncation method proposed to solve partially combinatorial problems with homographic objective functions on a common set of permutations, the method of truncation of insignificant constraints is used together with the method of sequential addition of constraints (MSAC) [1]. The former method decreases the dimension of the problem being solved at each step, and the latter substantially influences the number of constraints in the initial system in solving auxiliary problems of linear programming.

The truncation method for a partially combinatorial problem with a homographic objective function on a set of permutations is described below.

1. Pass from problem (1)–(3) with fractional-linear objective function to the problem with linear objective function (5)–(7) with the help of transformations (4).

2. Relax problem (5)–(7), namely, replace condition (6) by the condition

$$y = (y_0, y_1, \dots, y_k) \in \operatorname{conv} E', \tag{11}$$

where conv E' is specified by system (10).

3. Write problem (5), (7), (11) by the MSAC in the following form: determine the pair

$$\Phi^\tau(z^\tau) = \underset{z \in D_\tau}{extr} \sum_{j=1}^{m} c_j z_j + c_0 y_0, \; z^\tau = \arg \underset{z \in D_\tau}{extr} \sum_{j=1}^{m} c_j z_j + c_0 y_0 \tag{12}$$

for $\tau = 0$, where $z = (y_0^\tau, z_1^\tau, \dots, z_m^\tau) \in R^{m+1}$, $y_j^\tau = z_j^\tau \;\; \forall j \in J_k$, and a domain $D_0 \subset R^{m+1}$ is specified by the system S that contains constraints (7) and (10); when $\tau = 1$, we form a system S_1 of linear constraints ($S_1 \subset S$) that contains a substantially smaller number of constraints than S and specifies a domain D_1 such that $D_1 \supset D_0$. The system S_1 includes constraints (7) and also, for example, the following constraints taken from system (10):

$$\begin{cases} \sum\limits_{j=1}^{k} y_j = \sum\limits_{j=1}^{k} g_j y_0, \\ \sum\limits_{j=0}^{k} d_j y_j + \sum\limits_{j=k+1}^{m} d_j z_j = 1, \\ y_j \ge g_1 y_0 \quad \forall j \in J_k, \\ y_j \le g_k y_0 \quad \forall j \in J_k. \end{cases} \tag{13}$$

We note that the arbitrariness in choosing the system S_1 allows one to form it in each specific case so that the solution of its auxiliary problem be closest to the solution of the initial one. As is obvious, the number of variables in the left sides of some constraints of the system of constraints (10) varies from one to k. We add to the system S_1 the inequalities of the first and (k-1)th unions of (10) and two equalities, one of which is formed by inequalities of the zeroth and kth unions and the other equality is given in (10); moreover, additional linear constraints (7) are also added.

4. Solve linear programming problem (LPP) (12) on the domain D_τ for the current value of the index τ ($\tau \ge 1$) by the simplex method (modified or dual).

5. We determine a point $y^\tau = (y_0^\tau, y_1^\tau, \dots, y_k^\tau)$ with the help of the obtained point $z^\tau = (y_0^\tau, z_1^\tau, \dots, z_k^\tau, \dots, z_m^\tau)$, where $y_j^\tau = z_j^\tau \;\; \forall j \in J_k$, and check the satisfaction of all the constraints of system (10) in it, i.e., the satisfaction of the condition $y^\tau \in D_0$. If the point satisfies all the constraints, then go to item 8.

6. Form the system $S_{\tau+1}$ by addition of the inequality of system (10) that is not true at the point y^τ to the system of constraints S_τ.

7. Increase τ by unity: replace the domain D_τ described by the system S_τ by the domain $D_{\tau+1}$: $S_{\tau+1}$, $D_\tau \supset D_{\tau+1}$, and go to item 4.

8. Check whether the solution $y^\tau = (y_0^\tau, y_1^\tau, ..., y_k^\tau)$ satisfies condition (6). If the result is positive, then problem (5)–(7) is solved (problem (1)–(3) is also solved); STOP. Otherwise, go to item 9.

9. Find the nodes adjacent to $z^\tau = (y_0^\tau, z_1^\tau, ..., z_k^\tau, ..., z_m^\tau)$ for the polyhedron described by the system S_τ.

10. Determine the half-space whose boundary passes through these nodes adjacent to z^τ and such that the point z^τ does not belong to it, i.e., we construct the truncation that does not contain the found point in the form of the linear inequality

$$\sum_{j=1}^{s} \alpha_{m+1,j} z_j \le b_{i+1}, \quad i \in J_p. \tag{14}$$

11. Form the system $S_{\tau+1}$ that describes the domain $D_{\tau+1}$ by adding truncation (14) to the system S_τ.

12. Increase τ by unity.

13. Check the constraints of the system S_τ and find inactive constraints that can be truncated.

14. Form the system $S_{\tau+1}$ by truncating inactive constraints from S_τ; go to item 4.

To efficiently check the condition of item 5 of the combinatorial truncation method, we will prove the theorem given below.

THEOREM 1. Let we have $z = (y_0, z_1, ..., z_k, z_{k+1}, ..., z_m) \in R^{m+1}$,

$$y = (y_0, y_1, ..., y_k) \in E' \subset R^{k+1}, \; y_j = z_j \quad \forall j \in J_k,$$

$$y_j \le y_{j+1} \quad \forall j \in J_{k-1}^0, \; J_k^0 = \{0\} \cup J_k. \tag{15}$$

Then the satisfaction of the constraint

$$y_1 + ... + y_i \ge (g_1 + ... + g_i) y_0 \tag{16}$$

that belongs to a union i of inequalities of system (10) implies the satisfaction of all the other constraints of the union i of inequalities of this system at the point $y = (y_0, y_1, ..., y_k)$, where $i \in J_{k-1}$.

Proof. If condition (15) is true, then, for all the inequalities

$$y_{\alpha_1} + ... + y_{a_i} \ge (g_1 + ... + g_i) y_0, \tag{17}$$

$\alpha_j \in J_k \; \forall j \in J_i$ that form the union under an arbitrary number i, the following inequalities are true at the point z:

$$y_{\alpha_1} + ... + y_{\alpha_i} \ge y_1 + ... + y_i. \tag{18}$$

Conditions (16) and (18) imply the fulfillment of condition (17) at the point z, which is what had to be proved.

Thus, it follows from the theorem that, to check condition (11) by the combinatorial truncation method, it suffices to check the satisfaction for (k–1) constraints, i.e., for one constraint from each union of system (10).

Let us consider the process of finding an adjacent node by the combinatorial truncation method.

According to the linear programming theory, based on the simplex-table that determines some node z^* of the polyhedron of solutions, to obtain a node adjacent to it, one should take a nonbasic variable z_j (in the corresponding linear programming problem) whose vector P_j has at least one positive component and choose the tth row of the simplex-table from the condition

$$\frac{b_t}{\alpha_{tj}} = \min_{i:\, \alpha_{ij} > 0} \frac{b_i}{\alpha_{ij}} = \Theta_j, \tag{19}$$

where α_{ij} are the coefficients of unknowns z_j in a row *i* and b_i is the free term in the corresponding constraint of linear programming problem (5), (7), (10) (see, for example, [6]), add the vector P_j instead of P_t to the basis, and, as a result, to obtain the simplex-table for some node adjacent to $z*$.

We now construct truncations [3]. We denote by *J* the totality of numbers of nonbasic variables for which we can specify relations (19) and by *I* the set of numbers of basic variables. Based on the latter simplex-table of the LPP, we write the constraint determined by a basic variable (by a number *i*) as follows:

$$z_i + \alpha_{i,(\beta+1)} z_{j_1} + \ldots + \alpha_{i,(\beta+\gamma)} z_{j_\gamma} = b_i,$$

where $i \in I$, $\beta = |I|$, $\gamma = |J|$, $I \cup J = J_{m+1}$, $\beta + \gamma = m+1$, $j_\tau \in J \ \ \forall \tau \in J_\gamma$.

Let $z^* = (z_0^*, \ldots, z_k^*, \ldots, z_m^*)$ be an optimal solution of linear programming problem (5), (7), (10). Let us use the well-known theorem [2] that assumes the following form in the notations introduced above: if $j_\tau (j_\tau \in J \ \forall \tau \in J_\gamma, \gamma = |J|)$ are the numbers of nonbasic variables in the solution $z^* = (z_0^*, \ldots, z_k^*, \ldots, z_m^*)$ of the problem of linear programming (5), (7), (10) and the value of Θ_{j_τ} is computed $\forall \tau \in J_\gamma$ by formula (19), then the inequality

$$\frac{z_{j_1}}{\Theta_{j_1}} + \frac{z_{j_2}}{\Theta_{j_2}} + \ldots + \frac{z_{j_\gamma}}{\Theta_{j_\gamma}} \geq 1 \qquad (20)$$

is satisfied as the equality by all the nodes of the allowable domain that are adjacent to z^* and the point z^* does not satisfy inequality (20). According to this theorem, we construct truncation (14) in the form (20).

Let us consider the algorithm of the truncation method and its substantiation.

The algorithm A1 of the described combinatorial truncation method for solution of problem (1)–(3) uses the algorithm A2, which realizes the MSAC for an auxiliary problem, and a procedure of truncation of insignificant constraints, which is represented in the form of the algorithm A3.

At the input of the algorithm A1, the following items must be specified: the space dimension *m* that is determined by the number of unknowns in a problem, the number *k* of elements of the multiset *G*, the number *r* of additional constraints, the elements of the multiset *G*, and coefficients of the objective function and constraints (which are introduced or randomly generated). At the output of this algorithm, the following items must be specified: a point *t** that is the solution of the problem being solved, the value $F^*(t)$ of the objective function of the problem at this point, and also auxiliary parameters that characterize the functioning of the algorithm on the whole, namely, the number *q* of added constraints, number *p* of truncated constraints, number *s* of truncations, and operating time *T* of the algorithm.

Algorithm A1. We first put *s* = 0, i.e., assign zero to the integer variable *s* that specifies the number of truncations in the problem being solved.

Step 1. Specify the parameters for formation of problem data as follows: introduce *m*, *k*, *r,* and the following data for problem (1)–(3):

(1) the multiset G whose elements form the total set of permutations $E_{kn}(G)$ that appear in the problem;

(2) the coefficients of the objective function c_j and d_j $\forall j \in J_m^0$ and coefficients of additional constraints α_{ij} and b_i $\forall i \in J_r$.

Step 2. Form (specify) the objective function

$$\frac{\sum_{j=1}^{m} c_j t_j + c_0}{\sum_{j=1}^{m} d_j t_j + d_0} \to \underset{t \in R^m}{\text{extr}} = F(t^*) \qquad (21)$$

and additional linear constraints

$$\sum_{j=1}^{m} \alpha_{ij} t_j \le b_i \quad \forall i \in J_r, \tag{22}$$

$$t = (t_1, t_2, ..., t_k, t_{k+1}, ..., t_m) \in R^m, \quad x_j = t_j \ \forall j \in J_k.$$

Step 3. Using the mapping $\wp$ that is specified by relation (4), pass from problem (21), (22) on the set $E_{kn}(G)$ to the linear programming problem with the objective function

$$\operatorname*{extr}_{z \in D_\tau} \left(\sum_{j=1}^{m} c_j z_j + c_0 y_0 \right) \to \operatorname*{extr}_{z \in D_z} = \Phi(z^*), \tag{23}$$

where τ =1 at this step.

The constraints

$$\begin{cases} \sum_{j=1}^{k} z_j = \sum_{j=1}^{k} g_j y_0, \\ z_j \ge g_1 y_0 \ \forall j \in J_k; \\ z_j \le g_k y_0 \ \forall j \in J_k; \\ \sum_{j=0}^{k} d_j y_j + \sum_{j=k+1}^{m} d_j z_j = 1 \end{cases} \tag{24}$$

and additional linear constraints

$$\sum_{j=1}^{m} \alpha_{ij} z_j - b_i y_0 \le 0 \quad \forall i \in J_r \tag{25}$$

are added to the system S_1 that determines the domain D_1, where $z = (y_0, z_1, ..., z_k, z_{k+1}, ..., z_m) \in R^{m+1}$, $y_j = z_j \quad \forall j \in J_k$.

Step 4. Call the algorithm A2 that realizes the MSAC for auxiliary linear programming problems. At the input of the algorithm, we have linear programming problem (23) on the domain D_τ. At the output of the algorithm we have (1) the solution in the form of a point $z^\tau = (y_0^\tau, z_1^\tau, ..., z_k^\tau, ..., z_m^\tau)$, where $y_j^\tau = z_j^\tau \quad \forall j \in J_k$, from which a point $y^\tau = (y_0^\tau, y_{\alpha_1}^\tau, ..., y_{\alpha_k}^\tau)$ is determined; here, the coordinates are arranged in nondecreasing order; (2) the value of the parameter q that determines the number of constraints added according to the MSAC; (3) the varying domain D_τ specified by the system of constraints S_τ.

Step 5. Check condition (6) for the point $y^\tau = (y_0^\tau, y_{\alpha_1}^\tau, ..., y_{\alpha_k}^\tau)$. If condition (6) is true, then problem (5)–(7) is solved and initial problem (1)–(3), where $\Phi^*(z^*) = F^*(t^*)$, is also solved. The transformation $t_j^* = \dfrac{z_j^*}{y_0^*}$, $t_j = x_j \ \forall j \in J_k$, is performed that determines the solution of problem (1)–(3). The algorithm comes to an end.

Step 6. Increase s by unity.

Step 7. Find the nodes of the polyhedron D_τ specified by the system S_τ that are adjacent to the node z^τ.

Step 8. Determine truncation inequalities (20) for the point z^τ in the form

$$p^{(s)}(z) = \frac{z_{j_1}}{\Theta_{j_1}} + \frac{z_{j_2}}{\Theta_{j_2}} + ... + \frac{z_{j_\gamma}}{\Theta_{j_\gamma}} - 1 \ge 0, \tag{26}$$

where the quantity Θ_{j_τ} is specified by relations of the form (19).

Step 9. Form the system $S_{\tau+1}$ that specifies the domain $D_{\tau+1}$ as follows: add inequality (26) to the system S_τ. Increase τ by unity (replace $D_{\tau+1}$ by D_τ).

Step 10. Call the algorithm A3 that realizes the procedure of truncation of constraints. At the input of the algorithm, we have (1) the current system of constraints S_τ that specifies the domain of allowable solutions D_τ of problem (23); (2) the solution of the problem at different stages in the form of a point, namely, $z^{\tau-1}$ is the solution at the $(\tau-1)$th step of the algorithm or $\tilde{z}^i$ is the solution obtained after addition of the constraint with a number *i*. At the output of the algorithm, we have the system of constraints S_τ.

Step 11. Go to step 4 of the algorithm.

Algorithm A2. The method of sequential addition of constraints is used. At the input of the algorithm, the following items are specified: linear programming problem (23) that is specified on the domain D_τ by the system S_τ, and the integer variable $q = 0$ that specifies the number of added constraints at the output. At the output of the algorithm, we have the solution in the form of a point $z^\tau = (y_0^\tau, z_1^\tau, ..., z_k^\tau, ..., z_m^\tau)$ and a new system of constraints S_τ that specifies a new domain D_τ.

Step 1. Solve linear programming problem (23) on the domain D_τ (for $\tau \geq 1$) by the (modified or dual) simplex-method, where $z^\tau \in D_\tau$ is the sought-for solution.

Step 2. Form the point $y^\tau = (y_0^\tau, y_1^\tau, ..., y_k^\tau)$ from the first (k+1) coordinates of the point $z^\tau = (y_0^\tau, z_1^\tau, ..., z_k^\tau, ..., z_m^\tau)$, where $y_j^\tau = z_j^\tau \quad \forall j \in J_k$. Arrange the coordinates of the point $y^\tau = (y_0^\tau, y_1^\tau, ..., y_k^\tau)$ in nondecreasing order and, as a result, obtain

$$\tilde{y}^\tau = (y_0^\tau, y_{\alpha_1}^\tau, ..., y_{\alpha_k}^\tau).$$

Step 3. At the point $\tilde{y}^\tau = (y_0^\tau, y_{\alpha_1}^\tau, ..., y_{\alpha_k}^\tau)$, check the fulfillment of $(k-3)$ constraints of system (10) (by Theorem 1, one constraint is taken from each union). If all the constraints are satisfied at this point, then come to an end or, otherwise, go to the next step.

Step 4. Increase q by unity.

Step 5. Form the system $S_{\tau+1}$ that specifies the domain $D_{\tau+1}$, where $D_\tau \supset D_{\tau+1}$. To this end, add one inequality that belongs to system (10) and is not true at the point $\tilde{y}^\tau = (y_0^\tau, y_{\alpha_1}^\tau, ..., y_{\alpha_k}^\tau)$, namely,

$$p^q(z^\tau) = \sum_{j=1}^{i} y_{a_j} - \sum_{j=1}^{i} g_j y_0 \geq 0 \qquad (27)$$

to system (24), where $y_0 > 0$, $y_j \geq 0 \ \forall j \in J_k$, $\alpha_j \in J_k$, $\alpha_j \neq \alpha_t \quad \forall j \neq t, \ \forall j, t \in J_i, \ \forall i \in J_k$.

Step 6. Increase τ by unity, i.e., replace $D_{\tau+1}$ by D_τ ($S_{\tau+1}$ by S_τ) and go to step 1.

Note that the domain D_τ always contains the constraints of systems (24) and (25) and an inequality of the form (27) that is not true at the point that is the solution at the previous step of the algorithm A2.

Algorithm A3. The procedure of truncation of inactive constraints is realized.

At the input of the algorithm, we have (1) a system of constraints S_τ that specify the domain D_τ; (2) the solution of the problem at different steps of the algorithm, namely, the last solution z^τ and the solution z^i that is obtained after addition of the constraint with a number $(i-1)$.

At the output of the algorithm, we have a modified system of constraints S_τ that specifies the domain D_τ in which inactive constraints are absent according to the conditions of the algorithm A3.

After starting the algorithm, the integer variable that specifies the number of truncated constraints is initially equal to zero, $p = 0$. The variables q and s declared in the algorithms A1 and A2 determine, respectively, the number of the constraints added and the number of the truncations made.

Step 1. Compute $\omega = q + s$.

Step 2. Check the condition

$$p^{(i)}\left(z^{q+1}\right) > \delta \quad \forall i \in J_{\omega} \tag{28}$$

for all the constraints of the system S_{τ} that are added to the initial system S_1. If the condition is not fulfilled for any constraint that is added earlier, then come to an end. Otherwise, go to the next step.

Step 3. Form the set D of the indices of the constraints for which condition (28) is true.

Step 4. For all the constraints whose numbers belong to the set D, check the condition

$$F'\left(z^{\tau}\right) \geq F'\left(\tilde{z}^{i}\right) + p^{i}\left(\tilde{z}^{i}\right) \quad \forall i \in J_{\omega}. \tag{29}$$

If the condition is true for at least one constraint whose number belongs to the set D, then go to the next step or, otherwise, set $p = 0$ and come to an end.

Step 5. Eliminate the indices from the set D that specify the constraints for which condition (29) is not true.

Step 6. Form the system of constraints S_{τ} that contains no constraints whose indices do not belong to the set D.

Step 7. Determine the number s of elements of the set D. STOP.

We note that the parameter $\delta > 0$ is used in the proposed algorithm A3, and the number of the truncated constraints depends on the value of this parameter. The experience of solution of examples of using a similar procedure [7, 8] shows that δ should be chosen over the range $10^{-3} \div 10^{-2}$. As is noted in [7, 8], the increase in δ leads to the accumulation of many constraints at every step of the algorithm. The decrease in the quantity δ can actually lead to the preservation of only active constraints, which also slows down the functioning of the algorithm.

The algorithm A1 is general in the sense that it can be used for solution of partially combinatorial and completely combinatorial problems and also for conditional and unconditional optimization problems with homographic objective functions. Based on the algorithm A1, algorithms were constructed, their program realizations were developed, and numerical experiments were performed with an unconditional problem with a homographic objective function, with a completely combinatorial problem with additional linear constraints, and with problems of various kinds with linear objective functions and additional linear constraints. All the algorithms are realized in the Pascal language in the Delphi environment.

NUMERICAL EXPERIMENTS AND EXAMPLES OF THE TRUNCATION METHOD

Numerical experiments were performed to estimate the efficiency of the algorithms constructed on the basis of the described truncation method. The computations were performed on a PC with a Pentium II processor whose clock frequency equaled 233 MHZ.

The numerical experiments were performed for problems of Euclidean combinatorial optimization on permutations with homographic objective functions, namely, (1) for an unconditional optimization problem; (2) for a completely combinatorial problem with additional linear constraints; (3) for partially combinatorial problems with additional linear constraints.

To perform numerical experiments, elements of the multiset G that determine the total set of permutations $E_{kn}(G)$ were chosen as uniformly distributed random numbers on the intervals [1; 1000]: [1; 200], [1; 100], [1; 50], and [1; 10]. In the same manner, the coefficients of the objective function and additional constraints were chosen. In the numerical experiments, the number of elements of the multiset G varied from 10 to 23. In Tables 1-3, the following notations are used: m is the dimension of the space in which the problem is considered (for partially combinatorial problems), G is the multiset that specifies the total set of permutations $E_{kn}(G)$, k is the number of elements in the multiset G, n is the number of different elements in the multiset G, b is an interval from which the elements of the multiset G are chosen, F is the objective function of the problem being solved, r is the number of additional constraints of the problem, q is the number of constraints added according to the MSAC, p is the number of constraints truncated according to the method of truncation of inactive constraints, s is the number of truncations, x^* is the point (extremal) at which an extremum is reached, $F(x^*)$ is the value of the objective function at the point x^* (extremum), T is the computation time, δ is the parameter with the help of which a truncated constraint is determined, and d is the number of constraints in the system at the moment of calling the procedure of truncation of inactive constraints.

TABLE 1

Experiment Number	Results of Numerical Experiments for Completely Combinatorial Problems with the Parameters Enumerated									
	k	n	b	r	q	p	s	d	δ	T
1	10	7	[1;10]	6	7	0	0	42	0.001	46 sec.
2	10	10	[1;10]	1	37	32	1	46	0.01	3 min 59 sec
3	10	9	[1;100]	2	35	16	1	46	0.001	3 min 55 sec
4	11	8	[1;1000]	1	29	14	3	46	0.01	3 min 29 sec
5	12	10	[1;50]	11	17	0	7	46	0.001	1 min 8 sec
6	12	8	[1;10]	2	24	14	2	46	0.01	5 min 23 sec
7	12	8	[1;100]	7	6	0	5	46	0.001	50 sec
8	13	8	[1;1000]	3	12	0	3	46	0.01	1 min 34 sec
9	13	12	[1;100]	3	31	11	3	46	0.01	5 min 23 sec
10	14	6	[1;10]	8	22	0	9	46	0.001	3 min 24 sec
11	14	13	[1;50]	2	47	19	2	42	0.01	7 min 39 sec
12	15	9	[1;100]	3	170	167	13	42	0.001	22 min 20 sec
13	15	9	[1;10]	2	137	136	11	46	0.001	16 min 31 sec
14	16	8	[1;50]	6	4	3	1	46	0.001	50 sec
15	17	11	[1;1000]	12	31	17	11	46	0.001	7 min 11 sec
16	17	10	[1;50]	2	184	184	13	52	0.01	24 min 49 sec
17	18	15	[1;50]	1	6	5	13	58	0.01	54 sec
18	19	6	[1;100]	22	46	38	7	62	0.001	42 min 57 sec
19	19	12	[1;100]	2	128	127	16	42	0.001	18 min 48 sec
20	19	15	[1;1000]	4	70	68	15	68	0.01	10 min 08 sec
21	20	19	[1;100]	2	179	177	18	46	0.01	26 min 08 sec
22	20	15	[1;50]	37	174	170	12	72	0.01	36 min 46 sec
23	21	18	[1;100]	1	246	240	16	45	0.001	53 min 47 sec

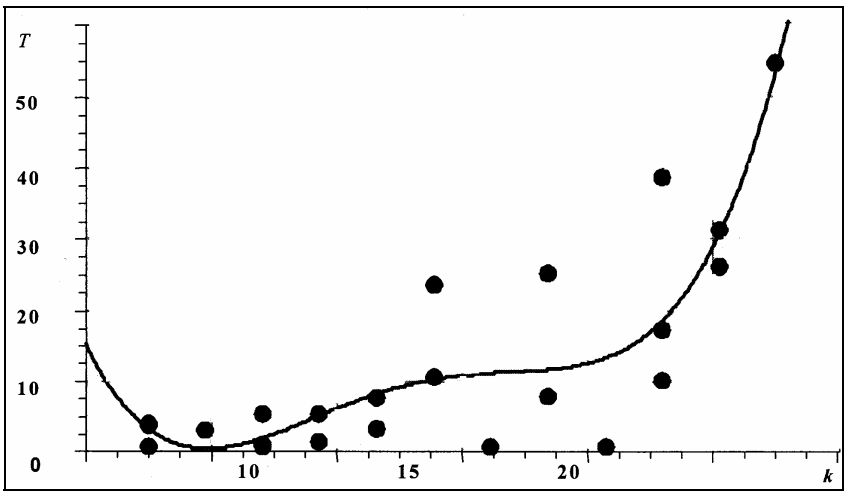

Fig. 1

In Table 1, the results of computations are given for completely combinatorial problems with homographic objective functions on a set of permutations with additional linear constraints. The elements of the multiset G that form the total set of permutations were chosen as uniformly distributed random numbers on one of the intervals [1; 10], [1; 50], [1; 100], and [1; 1000]. The numerical experiments were performed for the values of the parameter δ equal to 0.1, 0.01, 0.001, 0.0001, and 0.00001. Note that, for the smallest value of $\delta = 0.0001$, the constraints that must be truncated were practically absent and, for the largest value of $\delta = 0.1$, practically all the added constraints were truncated.

TABLE 2

Experiment Number	Results of Numerical Experiments for Partially Combinatorial Problems with the Parameters Enumerated										
	m	*k*	*n*	*b*	*r*	*q*	*p*	*s*	*d*	δ	*T*
1	21	16	8	[1;100]	6	202	200	6	46	0.01	27 min 05 sec
2	20	19	7	[1;50]	4	182	178	9	46	0.01	23 min 10 sec
3	20	17	10	[1;10]	5	146	140	8	42	0.01	20 min 34 sec
4	19	13	11	[1;10]	6	98	94	8	42	0.001	15 min 33 sec
5	19	18	15	[1;10]	4	124	120	9	40	0.01	21 min 22 sec
6	18	15	10	[1;50]	6	120	116	8	44	0.001	18 min 12 sec
7	18	16	11	[1;10]	4	178	170	7	46	0.01	22 min 10 sec
8	17	11	10	[1;100]	7	106	102	5	40	0.01	19 min 25 sec
9	16	10	10	[1;50]	2	98	87	6	42	0.001	17 min 33 sec
10	15	11	11	[1;50]	4	9	0	9	40	0.01	1 min 22 sec
11	14	10	10	[1;10]	2	10	0	8	40	0.01	1 min 20 sec
12	14	10	10	[1;10]	2	67	62	8	42	0.001	8 min 02 sec

TABLE 3

Experiment Number	Results of Numerical Experiments for a Partially Combinatorial Problem, $m = 21$ and $k = 16$								
	n	*b*	*r*	*q*	*p*	*s*	*d*	δ	*T*
1	8	[1;10]	6	202	200	6	46	0.01	27 min 05 sec
2	16	[1;50]	8	198	194	10	42	0.001	26 min 42 sec
3	12	[1;10]	4	188	180	8	44	0.001	20 min 40 sec
4	11	[1;10]	5	168	160	8	42	0.01	19 min 24 sec
5	10	[1;10]	4	160	156	9	44	0.01	15 min 18 sec
6	12	[1;100]	1	164	160	12	46	0.01	18 min 35 sec
7	14	[1;50]	8	88	84	10	42	0.001	26 min 34 sec
8	12	[1;10]	4	128	120	8	44	0.001	10 min 40 sec
9	11	[1;100]	2	156	152	5	42	0.01	12 min 24 sec
10	10	[1;50]	2	67	62	6	42	0.001	8 min 02 sec

To analyze tabular data that contain the results of numerical experiments, the applied program "Expert of Curves" was used [9] that makes it possible to represent data with the help of various regression models.

Analyzing Table 1, we can draw the conclusion that, for completely combinatorial problems with additional linear constraints, the minimum computation time equal to 46 seconds (for the set G with the number of elements equal to ten) and the maximum computation time equal to one hour was fixed. As is easily seen, the number of additional linear constraints and their arrangement is an important factor that influences the computation time and obtaining of solutions. The presence of additional constraints decreases the domain of allowable solutions of a problem. Therefore, in some problems with a homographic objective function and additional linear constraints, the computation time is smaller than in problems of the same dimension without additional constraints. It is relevant to note that the choice of the interval b of elements in the multiset G also influences the computation time. We can note that, for the examples with the same value of the parameters k, n, and r, the computation time increased with increasing the interval b.

In Fig. 1, the regression dependence of the computation time T on the number of elements k of the multiset G, i.e., on the problem dimension, is shown. Here, we have $T = a + bk + ck^2 + dk^3 + ek^4$, where $a = 85041.003$, $b = -24550.031$, $c = 2606.136$, $d = -120.582$, $e = 2.062$, and $r = 0.858$. The program "Expert of Curves" was used to obtain the plot.

In Tables 2 and 3, the results of numerical experiments for partially combinatorial problems with fractional-linear objective functions and additional linear constraints on permutations are presented. The computations were performed for the

elements of the multiset G that were chosen as uniformly distributed random numbers on the interval [1; 10], [1; 50], or [1; 100] and the coefficients of additional constraints were chosen in the interval [1; 5] or [1; 10].

The results of ten experiments for partially combinatorial problems with homographic objective functions and additional linear constraints on permutations are presented in Table 3. These computations were performed for elements of the multiset G that were chosen as uniformly distributed random numbers in the intervals [1; 10], [1; 50], and [1; 100].

Analyzing the results of numerical experiments presented in Table 2, we can draw the conclusion that, for partially combinatorial problems, the computation time is in the interval from 1 minute to 27 minutes. If the parameters k and n are in the intervals $k \in [10;19]$ and $n \in [7;18]$, then the minimum computation time is fixed for the smallest value of the parameter $m = 14$ that specifies the space dimension and the maximum computation time is fixed for the largest value of the parameter $m = 21$.

We note that, when $20 \le k \le 30$, some examples (that are randomly formed) were not solved because of the scantiness of program resources used for solving the LPP. For $k \ge 30$, the part of such problems was sizable. For such values of k, problems were not solved since the algorithm was forced to come to an end when the time of its operation exceeds one hour 30 minutes.

The procedure of truncation of inactive constraints was used in problems with $k \ge 15$. In Tables 1–3, the results of numerical experiments are given in which the value of the parameter δ, i.e., the candidate constraint determined for truncation, was equal to 0.1, 0.01, 0.001, 0.0001, and 0.00001. Note that, for the largest value of the parameter $\delta = 0.1$, practically all the candidate constraints for truncation were determined that were added at the preceding steps and, for the smallest value of $\delta = 0.0001$, such constraints were absent. In the first and second cases, this adversely affected the computation time. Therefore, in the majority of examples, the value 0.01 or 0.001 was assigned to δ.

Thus, the algorithms based on the combinatorial truncation method for problems with homographic objective functions are efficient in the proposed program realization for the problems of the considered class when $k < 30$. In the particular case where the denominator is equal to unity, an algorithm can be used to solve linear problems on permutations.

The program constructed on the basis of algorithms for the combinatorial truncation method considered above is universal for combinatorial problems on permutations with homographic and linear objective functions and additional linear constraints.

REFERENCES

1. Yu. G. Stoyan and O. O. Emets, Theory and Methods of Euclidean Combinatorial Optimization [in Ukrainian], Institute of Systems Investigations, Kyiv (1993).
2. O. A. Emets, “A truncation method for problems of combinatorial optimization,” Economics and Math. Methods, **33**, No. 4, 120–129 (1997).
3. O. O. Emets and Ye. M. Emets, “Truncations in linear partially combinatorial problems of Euclidean combinatorial optimization,” Dop. NAN Ukr., No. 9, 105–109 (2000).
4. O. O. Emets and L. M. Kolechkina, “An optimization problem with a fractional-linear objective function on permutations: Properties of the set of allowable solutions,” Ukr. Mat. Zh., **52**, No. 12, 1630-1640 (2000).
5. O. A. Emets, Ye. M. Emets, and L. N. Kolechkina, “Using the truncation method to solve the trim problem,” Radioelectronics and Informatics, No. 3, 114–117 (1998).
6. Yu. M. Yermol'yev, I. I. Lyashko, V. S. Mikhalevich, and V. I. Tyuptya, Mathematical Methods of Operations Research [in Russian], Vyshcha Shkola, Kiev (1979).
7. B. N. Pshenichnyi, E. I. Nenakhov, and V. N. Kuz'menko, “Mixed method for solving the general convex programming problem,” Kibern. Sist. Anal., No. 4, 121–134 (1998).
8. B. N. Pshenichnyi, E. I. Nenakhov, and V. N. Kuz'menko, “Modified cutting plane method for minimization of a convex function,” Kibern. Sist. Anal., No. 6, 142–149 (1997).
9. http://www2.msstate.edu/~dgh2/cvxpt.htm.