ZASTOSOWANIA MATEMATYKI
APPLICATIONES MATHEMATICAE
XVII, 2 (1982)

J. GRABOWSKI and E. SKUBALSKA (Wrocław)

MAXIMAL FLOW PROBLEM IN A NETWORK WITH A VARIABLE STRUCTURE

1. INTRODUCTION

In many transportation systems there arise problems of maximization of transport system efficiency by optimal allocation of the transportation facilities. In such problems there exist a fixed transportation network and a certain set of additional transportation facilities which may be utilized for the transportation of materials between any pair of nodes in the transportation network. An example of such a problem is the transportation system in a mine, where there exist a fixed conveyer network and reverse conveyers. In this case the problem arises to allocate the reverse conveyers so that the total amount of materials which is to be carried is maximal. Similar problems arise in public transportation, railway systems, freight transportation [3] and so on. The additional transportation facilities can be cars, drivers, crews, engines, ferries etc.

The mathematical model of the problem can be formulated by using disjunctive graphs. Disjunctive graphs have been used so far to solve sequencing problems (see [1], [2], [6] and [7]). A number of new notions are introduced and some properties are proved which allow us to construct a new implicit enumeration algorithm. Finally, computation results are presented.

2. MATHEMATICAL FORMULATION OF THE PROBLEM

Let $G = (N, U)$ denote a network (multigraph), where N is the set of vertices and U is the set of arcs. The set N is of the form

$$N = \{s\} \cup R \cup \{t\},$$

where s and t are the source and the sink of the network, respectively. Let $c\colon U \to R_+ \cup \{0\}$ be a mapping of U into non-negative numbers. The value $c(u)$ is called the *capacity of the arc* $u \in U$. Let $f\colon U \to R_+ \cup \{0\}$ be a mapping of U into non-negative numbers. The value $f(u)$ is called the *arc flow* in the network G if the following constraints are satisfied:

$$\sum_{u\in A_x} f(u) - \sum_{u\in B_x} f(u) = \begin{cases} v, & x = s, \\ 0, & x \in R, \\ -v, & x = t, \end{cases}$$

$$0 \leqslant f(u) \leqslant c(u), \quad u \in U,$$

where A_x (B_x) denotes the set of all arcs leading from the vertex x (to the vertex x) in G, and v is a non-negative number called the *flow value.*

Let $Q = \{1, 2, \ldots, q\}$ be the set of transportation routes which can be added between pairs of vertices of G. It is clear that adding the transportation route $k \in Q$ is equivalent to adding an arc to G. Let $V^k = \{w_1^k, w_2^k, \ldots, w_{r_k}^k\}$ be the set of arcs where the transportation route k can be added to G. Let

$$V = \bigcup_{k\in Q} V^k.$$

Further, we assume that the mapping c is extended over the set V. The value of the mapping $c(w_p^k)$ is called the *capacity of the arc w_p^k of the transportation route k.*

Therefore, the problem arises to find such an arc from the set V^k for each transportation route $k \in Q$ that the flow value in G is maximal.

Let

$$x_{kp} = \begin{cases} 1 & \text{if } w_p^k \text{ is to be added to } G, \\ 0 & \text{otherwise.} \end{cases}$$

We can formulate the problem as follows:

(1) $$\text{maximize } v$$

subject to

(2) $$\sum_{u\in A'_x} f(u) - \sum_{u\in B'_x} f(u) = \begin{cases} v, & x = s, \\ 0, & x \in R, \\ -v, & x = t, \end{cases}$$

(3) $$0 \leqslant f(u) \leqslant c(u), \quad u \in U,$$

(4) $$0 \leqslant f(w_p^k) \leqslant x_{kp} c(w_p^k), \quad w_p^k \in V^k,\ k \in Q,$$

(5) $$\sum_{p=1}^{r_k} x_{kp} = 1, \quad k \in Q,$$

(6) $$x_{kp} \in \{0, 1\}, \quad w_p^k \in V^k,\ k \in Q,$$

where A'_x (B'_x) denotes the set of all arcs leading from the vertex x (to the vertex x) in $G' = (N, U \cup V)$.

Conditions (1)-(6) constitute the problem which is called *Problem P*. The problem (1)-(3) is known as the linear programming problem of finding a maximum flow value from s to t in the network G (see [4]). If an arc w_p^k of the set V^k is added to G, then the flow through this arc is not greater than the capacity of this arc, whereas the flows through other arcs of V^k are equal to zero. This is expressed by constraints (4). Constraint (5) ensures that exactly one arc for each transportation route can be added to G.

Problem P is known as a mixed integer programming problem. Let $(\bar{f}, \bar{x})$ be a feasible solution to P, where $\bar{f}$ is the vector of variables $f(u)$, $u \in U \cup V$, and $\bar{x}$ is the vector of variables x_{kp}, $w_p^k \in V^k$, $k \in Q$.

It follows from (6) that constraint (5) can be replaced by

$$(7) \qquad (x_{k1} = 1) \dot{\vee} (x_{k2} = 1) \dot{\vee} \ldots \dot{\vee} (x_{kr_k} = 1), \qquad k \in Q,$$

where $\dot{\vee}$ denotes a disjunction. Hence for each variable x_{kp} which is equal to one the variables x_{ks} $(s = 1, 2, \ldots, r_k, s \neq p)$ are equal to zero and constraints (4) take the form

$$(8a) \qquad 0 \leqslant f(w_p^k) \leqslant c(w_p^k),$$

$$(8b) \qquad 0 \leqslant f(w_s^k) \leqslant 0, \quad w_s^k \in V^k, \ s \neq p, \ k \in Q.$$

Therefore, to each variable which is equal to one we can assign exactly one arc $w_p^k \in V^k$. Moreover, to each constraint (7) we can assign the following set of disjunctive arcs in G:

$$(9) \qquad (w_1^k, w_2^k, \ldots, w_{r_k}^k), \quad w_p^k \in V^k, \ k \in Q,$$

where w_p^k has the capacity $c(w_p^k)$. Then the disjunctive network of Problem P can be determined as $\bar{G} = (N, U; V)$.

An example of the disjunctive network for

$$V^1 = \{w_1^1, w_2^1, w_3^1\}, \qquad V^2 = \{w_1^2, w_2^2, w_3^2\}, \qquad V^3 = \{w_1^3, w_2^3, w_3^3\}$$

is shown in Fig. 1, where the disjunctive arcs are plotted by dashed lines.

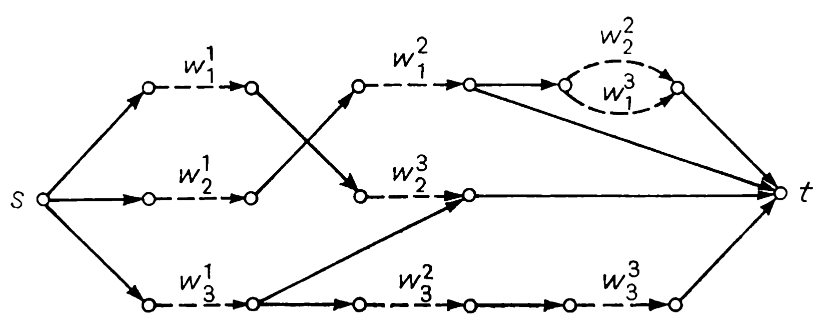

Fig. 1

The subset of V containing exactly one disjunctive arc from each set V^k is called a *selection*. Let

$$\mathscr{R} = \{S_1, S_2, \ldots, S_m\}$$

be the family of all selections. Any arc from the set V^k is called a *cousin* of every other arc of this set. Each arc in V^k has $r_k - 1$ cousins. A replacement of any arc by its cousin is called *complementing*.

Each selection $S_r \in \mathscr{R}$ generates the network

$$G_r = (N, U \cup S_r). \tag{10}$$

Let

$$\mathscr{G} = \{G_r = (N, U \cup S_r)\}$$

be the family of graphs of the form (10). It can be easily seen that

$$L = |\mathscr{G}| = \prod_{k \in Q} r_k. \tag{11}$$

The maximum flow value v_0 in the network $G_0 \in \mathscr{G}$ is called *bimaximal flow* in the disjunctive network $\bar{G}$, and the associated selection S_0 is *optimal* if

$$v_0 = \max_{G_r \in \mathscr{G}} v_r, \tag{12}$$

where v_r is the maximum flow value in G_r.

THEOREM 1. *Problem* P *is equivalent to the problem of finding the value* (12) *of a bimaximal flow v_0 and an optimal selection S_0 in the disjunctive network $\bar{G} = (N, U; V)$. The maximum flow value v_0 in G_0 is the optimal value of v.*

Proof. Let $(\bar{x}_r)$ be a feasible solution of Problem P. Let us denote by Z_r the problem P which is obtained by putting $(\bar{x}) = (\bar{x}_r)$. Let $Z = \{Z_1, Z_2, \ldots, Z_m\}$ be the set of all problems Z_r for every feasible solution of the vector $(\bar{x})$ which can be obtained from Problem P. The problem $Z_r \in Z$ corresponds to the network G_r so that for each variable x_{kp} which is equal to one we have constraint (8a) in problem Z_r and $w_p^k \in S_r$. The relation $w_s^k \notin S_r$ corresponds to each variable x_{ks} which is equal to zero. The condition set by the theorem for the network G_0 equals, therefore, the requirement of finding among all feasible problems $Z_r \in Z$ a problem Z_0 such that its optimal solution is maximal over the set of solutions of all problems $Z_r \in Z$. But this is exactly what solving Problem P amounts to.

It follows from the above that the algorithm of finding the bimaximal flow in $\bar{G}$ solves also Problem P. In this paper we present an implicit enumeration algorithm which finds the bimaximal flow by generating a sequence of networks $G_r \in \mathscr{G}$ and solving the maximum flow value problem for each G_r in the sequence. Each network G_s is obtained from a certain network G_r by complementing one disjunctive arc.

Let $C_r = (X_r, \bar{X}_r)$ denote a cut of the network $G_r \in \mathcal{G}$, where $X \subset N$, $s \in X$ and $\bar{X} = N - X$, $t \in \bar{X}$. The cut $(X_r, \bar{X}_r)$ consists of all arcs $u = (x, y)$ such that $x \in X$ and $y \in \bar{X}$. The capacity of the cut $(X_r, \bar{X}_r)$, denoted by $c(X_r, \bar{X}_r)$ or $c(C_r)$, equals

$$c(X_r, \bar{X}_r) = \sum_{u \in C_r} c(u).$$

As is known [4], the maximum flow value is equal to the minimum capacity of a cut of G_r. This cut is called the *minimum capacity cut* and denoted by C_r^0 or $(X_r^0, \bar{X}_r^0)$. We say that any arc $w_p^k \in V$ is *adjacent* to the cut $C_r = (X_r, \bar{X}_r)$ in G_r and write $w_p^k \perp C_r$ if there exist $x \in X_r$ and $y \in \bar{X}_r$ such that $w_p^k = (x, y)$.

THEOREM 2. *Let $C_r^0 = (X_r^0, \bar{X}_r^0)$ be the minimal cut in $G_r \in \mathcal{G}$. If there exists a network $G_s \in \mathcal{G}$ with the maximum flow value greater than $c(X_r^0, \bar{X}_r^0)$, then the selection S_s contains a cousin $w_z^k \perp C_r^0$ of at least one arc $w_p^k \in S_r$, $z \neq p$, $k \in Q$. Moreover, if $w_p^k \in C_r^0$, then $c(w_z^k) > c(w_p^k)$.*

Proof. If S_s does not contain the cousin $w_z^k \perp C_r^0$ of any $w_p^k \in S_r$, then there exists a set $W \subset C_r^0$ which is a cut in G_s. The set W equals either C_r^0 or $C_r^0 - \{w_p^k\}$. Since the maximum flow value in G_s cannot be greater than the capacity of W, we have $v_s \leqslant c(W) \leqslant c(C_r^0) = v_r$, which contradicts the assumption. If $w_p^k \in C_r^0$, then the set $[C_r^0 - \{w_p^k\}] \cup \{w_z^k\}$ is a cut in G_s. Therefore, we have $v_s \leqslant v_r + c(w_z^k) - c(w_p^k)$. From the assumption it follows that $v_s > v_r$, thus we obtain $c(w_z^k) > c(w_p^k)$.

THEOREM 3. *Let $G_r \in \mathcal{G}$ and let $G_s \in \mathcal{G}$ be the network obtained from G_r by complementing an arc $w_p^k \in S_r$ by a cousin $w_z^k \perp C_r^0$, where C_r^0 is the minimum capacity cut in G_r. Then*

$$v_s \leqslant v_r + c(w_z^k).$$

Moreover, if $w_p^k \in C_r^0$, then

$$v_s \leqslant v_r + c(w_z^k) - c(w_p^k). \tag{13}$$

Proof. In order to prove the theorem note that there exists a set $W \subset C_r^0$ such that $W \cup \{w_z^k\}$ is a cut in G_s (the set W equals either C_r^0 or $C_r - \{w_p^k\}$). The capacity of this cut is not greater than $c(C_r) + c(w_z^k)$. Since the maximal flow in G_s cannot be greater than the capacity of any cut, we have

$$v_s \leqslant c(C_r^0) + c(w_z^k) = v_r + c(w_z^k).$$

If $w_p^k \in C_r^0$, then the set $[C_r^0 \cup \{w_z^k\}] - \{w_p^k\}$ is a cut in G_s. Thus we obtain inequality (13).

Therefore, by complementing the arc w_p^k by a cousin $w_z^k \perp C_r^0$, the lower bound to the value v_s of a maximal flow in G_s is $v_r + c(w_z^k)$ or $v_r + c(w_z^k) - c(w_p^k)$ if $w_p^k \in C_r^0$.

3. ALGORITHM

The bimaximal flow of the disjunctive network $\bar{G}$ is obtained by generating a sequence of networks $G_r \in \mathcal{G}$ and finding the maximum flow value for each G_r in the sequence.

Let $S_1 \in \mathcal{R}$ be an initial selection. The disjunctive arc $w_p^k \in V$ is called *normal* if $w_p^k \in S_1$, and the disjunctive arcs $w_z^k \in V - S_1$ are called *reverse*. Each arc $w_p^k \in S_1$ has $r_k - 1$ reverse arcs in the set $V - S_1$.

Starting with the network $G_1 = (N, U \cup S_1)$, we generate a sequence of networks $G_r \in \mathcal{G}$. Each network G_s is obtained from a certain network G_r of the sequence by complementing one normal arc from the selection S_r. Each arc from S_r is complemented by $r_k - 1$ reverse arcs. The process of generating can be presented in the form of the solution tree H. Each node in H corresponds to a network G_r. Each arc in H represents a pair of networks (G_r, G_s) such that G_s is obtained from G_r. Since G_s differs from G_r by exactly one disjunctive arc $w_p^k \in S_r$, the arc (G_r, G_s) in H is associated with the reverse arc $w_z^k \in S_s$ of w_p^k. We say that G_r is the *predecessor* of G_s (and G_s is the *successor* of G_r) if there is a path in H between G_r and G_s. The initial selection G_1 is the *root* in the solution tree H. The generation of a new branch in H is connected with choosing a certain normal arc for complementing from S_r. This operation is called the *operation of choice*. For each network G_r from the sequence we perform an operation of testing to check the maximum flow value and the possibility of generating a network $G_s \in \mathcal{G}$ with a maximum flow value greater than that already found. If the testing is negative, we abandon the considered network G_s and backtrack the tree H to the predecessor G_r from which the network G_s was generated. If a new network G_s is obtained from the network G_r by complementing a normal arc $w_p^k \in S_r$ by a reverse arc $w_z^k \in S_s$, we constantly fix this reverse arc. This arc cannot be complemented by any other successor of G_r in H. However, if we backtrack for the n-th time (where $n < r'_k = r_k - 1$) by a reverse arc of a certain normal arc to G_r, we momentarily fix this reverse arc. If we backtrack the r'_k-th time, we constantly fix this normal arc. The normal arc such that its reverse arc is momentarily fixed can be complemented if we need to perform another operation of complementing which has not been performed yet for this normal arc. So, for each network G_r we constantly fix a subset $F_r \subset S_r$ and momentarily fix a set F_r^t of disjunctive arcs. The reverse arcs in F_r are constantly fixed and represent the path from the root to

G_r in H. Each normal arc in F_r is constantly fixed and represents r'_k reverse arcs which have been abandoned during the backtracking process. Each momentarily fixed arc in F_r^t is a reverse arc which has been abandoned during the backtracking process. The set F_r^t contains all arcs belonging to the path from the root to G_r in H. No arc from the set F_r can be complemented by any successor G_t of G_r.

3.1. Operation of choice. The purpose of the operation of choice is to find a normal arc for complementing and to generate a successor of G_r in H. The arcs $E_r = S_r - F_r$ are said to be *free*. Let K_r be the set of all reverse arcs the normal arcs of which belong to the set E_r and let $K'_r \subset K_r$ be the set of reverse arcs which are adjacent to the minimum capacity cut of G_r and such that if $w_p^k \in C_r^0$, then $c(w_z^k) > c(w_p^k)$, where w_p^k is a normal arc of the arc w_z^k. Let $E'_r \subset E_r$ be the set of normal arcs at least one reverse arc of which belongs to the set K'_r. It follows from Theorem 2 that we may complement only arcs belonging to the set E'_r, called the *set of candidates*. We want to choose a normal arc the complementing of which generates a successor with the possibly greatest maximum flow value. It follows from Theorem 3 that the choice criterion of a reverse arc of K'_r is either the value $c(w_z^k)$ if $w_p^k \notin C_r^0$ or the value $c(w_z^k) - c(w_p^k)$ if $w_p^k \in C_r^0$.

3.2. Operation of testing. The basic task of the operation of testing is to compute an upper bound to the maximum flow value for every possible successor $G_s \in \mathcal{G}$ which can be generated from G_r.

(a) Test 1. Let

$$G(F_r) = (N,\ U \cup F_r)$$

be the network generated from the set F_r of constantly fixed disjunctive arcs and let

$$\bar{G}(F_r) = (N,\ U \cup F_r;\ V_r), \qquad \text{where } V_r = V - [F_r \cup F'_r],$$

be the disjunctive network of the network $G(F_r)$, F'_r being the set of disjunctive arcs the cousins of which belong to F_r. Let $v(F_r)$ and $C(F_r)$ be the maximum flow value and any cut of $G(F_r)$, respectively.

As we have proved in Theorem 1, the process of generating successors of G_r in H is equivalent to the problem of finding the value of the bimaximal flow of the disjunctive network $\bar{G}(F_r)$. The initial selection is the set $E_r = S_r - F_r$. Let $V_r^k = V^k \cap V_r$, and let $V_r^{k'} \subset V_r^k$ be the set of disjunctive arcs which are adjacent to the cut $C(F_r)$. Let $Q' \subset Q$ be the set of transportation routes k for which $V_r^{k'} = \emptyset$ and let

$$c(w_m^k) = \max_{w_p^k \in V_r^{k'}} c(w_p^k), \qquad k \in Q'.$$

It can be easily seen that

$$v_0(F_r) \leqslant c[C(F_r)] + \sum_{k\in Q'} c(w_m^k), \tag{14}$$

where $v_0(F_r)$ is the value of the bimaximal flow in $\bar{G}(F_r)$.

Inequality (14) holds for any cut of $G(F_r)$; in particular, for the minimum capacity cut $C^0(F_r)$ we obtain

$$v_0(F_r) \leqslant \bar{v}^1(F_r) = v(F_r) + \sum_{k\in Q'} c(w_m^k). \tag{15}$$

The value $\bar{v}^1(F_r)$ is the upper bound of $v_0(F_r)$, i.e. the upper bound of all successors of G_r.

Let v^* be the value of the greatest maximal flow value found so far. Then, if $v^* \geqslant \bar{v}^1(F_r)$, we can reject the network G_r and all its successors.

(b) Test 2. By the definition of E_r we have $S_r = F_r \cup E_r$ and

$$C_r^0 = (U\cup S_r)\cap C_r^0 = [(U\cup F_r)\cap C_r^0]\cup(E_r\cap C_r^0).$$

Since $S_r\cap U = \emptyset$, the sets $(U\cup F_r)\cap C_r^0$ and $E_r\cap C_r^0$ are disjoint and we obtain

$$v_r = c(C_r^0) = \sum_{u\in(U\cup F_r)\cap C_r^0} c(u) + \sum_{u\in E_r\cap C_r^0} c(u). \tag{16}$$

Since $G(F_r)$ is a subgraph of G_r, the set $C(F_r) = (U\cup F_r)\cap C_r^0$ is a cut of $G(F_r)$. Therefore, combining (15) and (16) we obtain

$$v_0(F_r) \leqslant \bar{v}^2(F_r) = v_r - \sum_{u\in E_r\cap C_r^0} c(u) + \sum_{k\in Q'} c(w_m^k). \tag{17}$$

Thus, if $v^* \geqslant \bar{v}^2(F_r)$, we can reject the network G_r and all its successors.

3.3. Algorithm. We start with $G_1 = (N, U\cup S_1)$, $F_1 = \emptyset$, $F_1^t = \emptyset$, $v^* = 0$. The network G_1 corresponds to the root of the solution tree H.

Let $G_r = (N, U\cup S_r)$ be the current network and let F_r, F_r^t be the current sets of constantly and momentarily fixed disjunctive arcs in the r-th iteration of the algorithm.

Step 1 (test step 1). Compute the upper bound $\bar{v}^1(F_r)$ defined by (15). If $\bar{v}^1(F_r) \leqslant v^*$, then go to step 5. Otherwise, go to step 2.

Step 2 (evaluation step). Compute v_r in the network G_r. If $v_r > v^*$, then put $v^* = v_r$. Identify the set of candidates E_r'. If $E_r' = \emptyset$, then go to step 5. Otherwise, identify the set K_r'. Next perform $K_r' = K_r' - F_r^t$ and go to step 3.

Step 3 (test step 2). If $K'_r = \emptyset$, then go to step 5. Otherwise, compute $\bar{v}^2(F_r)$ defined by (17). If $\bar{v}^2(F_r) \leqslant v^*$, then go to step 5. Otherwise, go to step 4.

Step 4 (forward step). Choose the normal arc $w_p^k \in E'_r$ the reverse arc $w_z^k \in K'_r$ of which is such that

$$c(w_z^k) - \delta(w_p^k)c(w_p^k) = \max_{w_a^k \in K'_r} c(w_a^k) - \delta(w_p^k)c(w_p^k),$$

where

$$\delta(w_a^k) = \begin{cases} 1 & \text{if } w_a^k \perp C_r^0, \\ 0 & \text{otheiwise.} \end{cases}$$

Next generate the network G_s by complementing the normal arc w_p^k and constantly fixing the reverse arc w_z^k, $z \neq p$, i.e. by letting

$$S_s = [S_r - \{w_p^k\}] \cup \{w_z^k\}, \qquad F_s = F_r \cup \{w_z^k\}, \qquad F_s^t = F_r^t.$$

Simultaneously, add to the solution tree H the node G_s and the new arc (G_r, G_s) associated with the reverse arc w_z^k. Then go to step 1.

Step 5 (backtracking step). Backtrack to the predecessor G_l of G_r in H. If G_r has no predecessor, then the algorithm terminates, the selection S_* associated with the current v^* is optimal and the maximal flow in G is bimaximal in $\bar{G}$. Otherwise, drop the data for G_r and update the data for G_l as follows.

If the network G_r is generated by the reverse arc $w_z^k \in K'_r$, then perform $K'_l = K'_l - \{w_z^k\}$ and $F_l^t = F_l^t \cup \{w_z^k\}$.

If the backtracking is performed for the r'_k-th time by a reverse arc of the normal arc w_p^k, perform $F_l = F_l \cup \{w_p^k\}$ and go to step 3.

4. EXAMPLE

This part of the paper contains the solution of the example with the disjunctive network shown in Fig. 2.

We have

$$V^1 = \{w_1^1, w_2^1, w_3^1\}, \qquad V^2 = \{w_1^2, w_2^2, w_3^2, w_4^2\},$$
$$V^3 = \{w_1^3, w_2^3, w_3^3\}, \qquad V^4 = \{w_1^4, w_2^4\}.$$

The networks $G_r \in \mathcal{G}$ obtained in the sequence of iterations are shown in Figs. 3-9. Dashed lines represent the minimum capacity cut. The first numbers at the arrows denote the capacity of the arcs, and the second ones denote the arc flows. The solution tree is shown in Fig. 10.

We start from the network $G_1 = (N, U \cup S_1)$, where $S_1 = \{w_1^1, w_1^2, w_1^3, w_1^4\}$.

Iteration 1. $F_1 = \emptyset, F_1^t = \emptyset, v^* = 0$ (Fig. 3).

(a) Test step 1: $\bar{v}^1 = 39 > 0 = v^*$.

(b) Evaluation step: $v_1 = 10, v^* = 10, K_1' = \{w_3^2, w_4^2, w_2^3, w_3^3\}$.

(c) Test step 2: $\bar{v}^2 = 27 > 10$.

(d) Forward step: choose w_1^2 and generate G_2 fixing w_3^2.

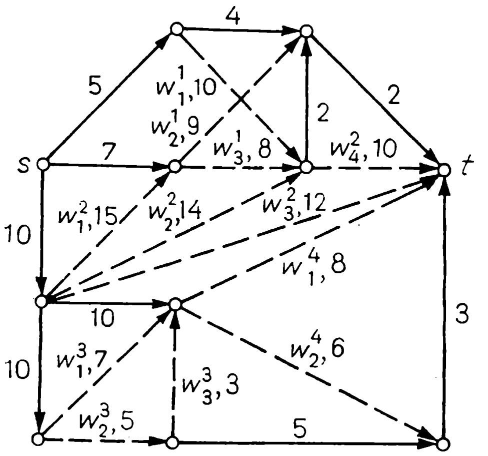

Fig. 2

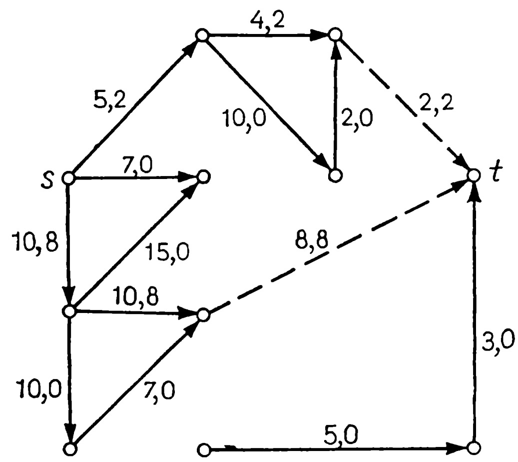

Fig. 3

Iteration 2. $F_2 = \{w_3^2\}, F_2^t = \emptyset$ (Fig. 4).

(a) Test step 1: $\bar{v}^2 = 22 > 10 = v^*$.

(b) Evaluation step: $v_2 = 12, v^* = 12, K_2' = \emptyset$.

(c) Backtracking step: backtrack to G_1, $F_1 = \emptyset$, $F_1^t = \{w_3^2\}$, $K_1' = \{w_4^2, w_2^3, w_3^3\}$.

(d) Test step 2: $\bar{v}^2 = 25 > 12 = v^*$.

(e) Forward step: choose w_1^2 and generate G_3 fixing w_4^2.

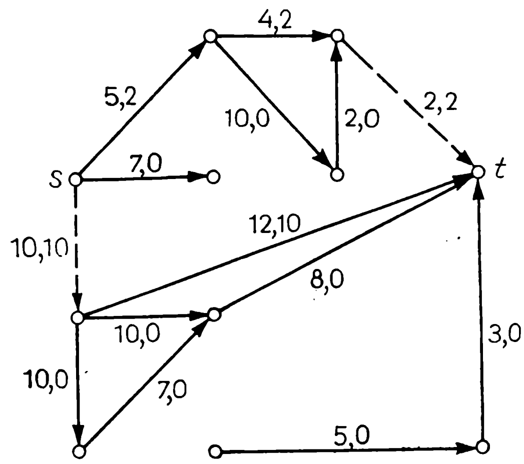

Fig. 4

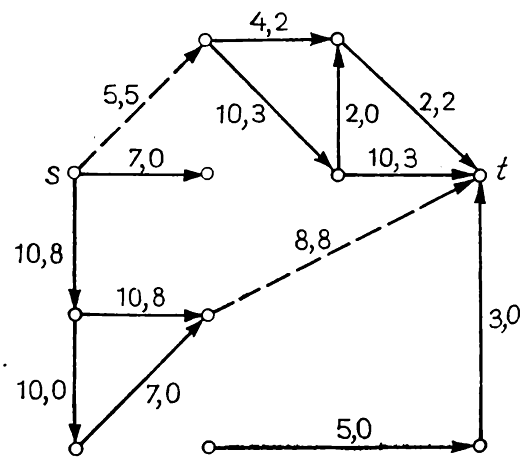

Fig. 5

Iteration 3. $F_3 = \{w_4^2\}, F_3^t = \{w_3^2\}$ (Fig. 5).
(a) Test step 1: $\bar{v}^1 = 25 > 12 = v^*$.
(b) Evaluation step: $v_3 = 13, v^* = 13, K_3' = \{w_2^1, w_3^1, w_2^3, w_3^3\}$.
(c) Test step 2: $\bar{v}^2 = 27 > 13 = v^*$.
(d) Forward step: choose w_1^1 and generate G_4 fixing w_2^1.
Iteration 4. $F_4 = \{w_4^2, w_2^1\}, F_4^t = \{w_3^2\}$ (Fig. 6).
(a) Test step 1: $\bar{v}^1 = 15 > 13 = v^*$.
(b) Evaluation step: $v_4 = 10, K_4' = \{w_2^3, w_3^3\}$.
(c) Test step 2: $\bar{v}^2 = 15 > 13 = v^*$.
(d) Forward step: choose w_1^3 and generate G_5 fixing w_2^3.

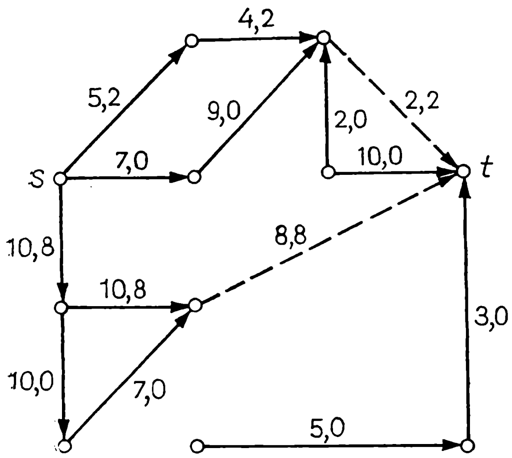

Fig. 6

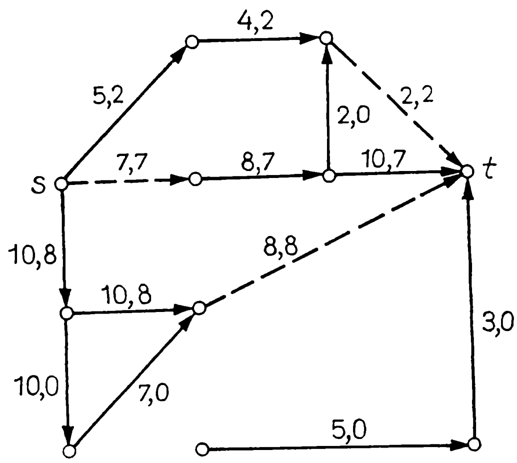

Fig. 7

Iteration 5. $F_5 = \{w_4^2, w_2^1, w_2^3\}, F_5^t = \{w_3^2\}$.
(a) Test step 1: $\bar{v}^1 = 13 \leq 13 = v^*$.
(b) Backtracking step: backtrack to G_4, $F_4 = \{w_4^2, w_2^1\}$, $F_4^t = \{w_3^2, w_2^3\}$.
(c) Test step 2: $\bar{v}^2 = 13 \leq 13 = v^*$.
(d) Backtracking step: backtrack to G_3, $F_3 = \{w_4^2\}$, $F_3^t = \{w_3^2, w_2^1\}$.
(e) Test step 2: $\bar{v}^2 = 26 > 13 = v^*$.
(f) Forward step: choose w_1^1 and generate G_6 fixing w_3^1.
Iteration 6. $F_6 = \{w_4^2, w_3^1\}, F_6^t = \{w_3^2, w_2^1\}$ (Fig. 7).
(a) Test step 1: $\bar{v}^1 = 22 > 13 = v^*$.
(b) Evaluation step: $v_6 = 17, v^* = 17, K_6' = \{w_2^3, w_3^3\}$.
(c) Test step 2: $\bar{v}^2 = 22 > 17 = v^*$.
(d) Forward step: choose w_1^3 and generate G_7 fixing w_2^3.
Iteration 7. $F_7 = \{w_4^2, w_3^1, w_2^3\}, F_7^t = \{w_3^2, w_2^1\}$ (Fig. 8).
(a) Test step 1: $\bar{v}^1 = 20 > 17 = v^*$.
(b) Evaluation step: $v_7 = 19, v^* = 19, K_7' = \emptyset$.
(c) Backtracking step: backtrack to G_6, $F_6 = \{w_4^2, w_3^1\}$, $F_6^t = \{w_3^2, w_2^1, w_2^3\}, K_6' = \{w_3^3\}$.

(d) Test step 2: $\bar{v}^2 = 20 > 19 = v^*$.

(e) Forward step: choose w_1^3 and generate G_8 fixing w_3^3.

Iteration 8. $F_8 = \{w_4^2, w_3^1, w_3^2\}$, $F_8^t = \{w_3^2, w_2^1, w_2^3\}$ (Fig. 9).

(a) Test step 1: $\bar{v}^1 = 20 > 19 = v^*$.

(b) Evaluation step: $v_8 = 19$, $K_8' = \emptyset$.

(c) Backtracking step: backtrack to G_6, $F_6 = \{w_4^2, w_3^1, w_1^1\}$, $F_6^t = \{w_3^2, w_2^1, w_2^3, w_3^3\}$, $K_6' = \emptyset$; hence again backtrack to G_3, $F_3 = \{w_4^2\}$, $F_3^t = \{w_3^2, w_2^1, w_3^1\}$.

(d) Test step 2: $\bar{v}^2 = 18 \leqslant 19 = v^*$.

(e) Backtracking step: backtrack to G_1, $F_1 = \emptyset$, $F_1^t = \{w_3^2, w_4^2\}$.

(f) Test step: $\bar{v}^2 = 15 \leqslant 19 = v^*$.

(g) Backtracking step: backtrack from the root of the tree H. End.

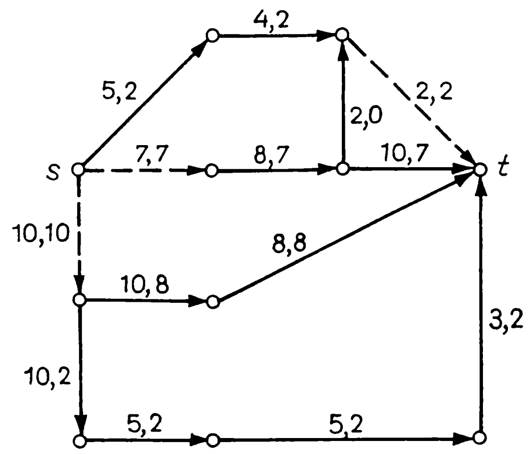

Fig. 8

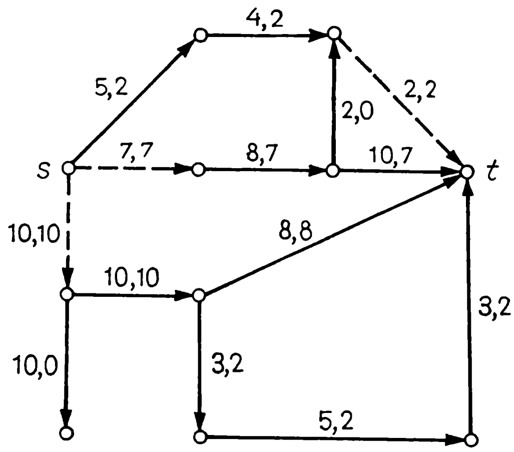

Fig. 9

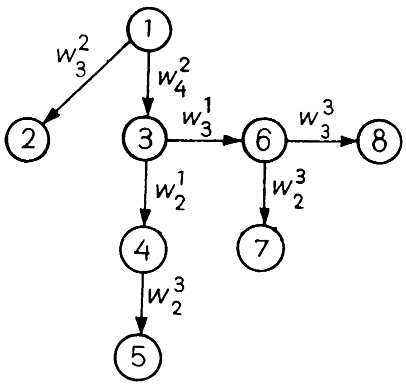

Fig. 10

5. COMPUTATIONAL RESULTS

The algorithm was implemented in ALGOL 60 and was used to solve several problems on the computer ODRA 1325.

The basic task of the algorithm is to compute the maximum flow value in the network G_r. In order to attain this, Dinic's method (see [4] and [5]) was used.

TABLE

		Case	Problems 1			2			3		
$\|N\|$		I and II	11			20			25		
$\|U\|$			10			25			50		
$\|V\|$			13			43			100		
Number of G_r for complete enumeration			96			$1.62 \cdot 10^6$			$11.25 \cdot 10^{10}$		
			for capacities								
			1-20	11-30	91-110	1-20	11-30	91-110	1-20	11-30	91-110
Number of iterations to optimum	A	I	5	6	6	66	171	227	6	4	4
	M		4	5	5	35	22	80	4	4	4
	A	II	4	6	6	136	226	192	17	7	5
	M		2	5	5	84	273	134	53	7	5
Number of iterations (step 1)	A	I	9	10	10	120	298	505	16	6	8
	M		7	8	8	85	181	375	8	4	6
	A	II	5	7	8	708	733	784	58	171	82
	M		5	5	6	519	744	1101	225	59	18
Time (s)	A	I	1	1	1	55	126	216	18	5	7
	M		1	1	1	36	86	181	7	4	5
	A	II	1	1	1	239	257	290	58	132	73
	M		1	1	1	192	255	385	236	61	18
Number of unsolved examples		I	0	0	0	0	0	1	1	0	0
		II	0	0	0	0	2	2	5	3	1

A — average, M — median.

Three problems have been solved. Each problem was solved 30 times (10 times for each range of the capacities), and the capacities were generated randomly from a uniform distribution with lower and upper limits shown in the Table. The results of the computations which were always started with an initial selection

$$S_1 = \bigcup_{k\in Q} \{w_m^k\}, \qquad \text{where } c(w_m^k) = \max_{w_p^k \in V^k} c(w_p^k),$$

are shown in the Table (case I). The same problems have been solved in the case where the capacities of the arcs belonging to V^k, $k \in Q$, are the same, i.e. $c(w_p^k) = c_k$, $w_p^k \in V^k$, $k \in Q$, and the computations were started with a pseudorandom initial selection S_1 (for the results see case II in the Table). The examples have been solved with the time limit of 600 seconds.

References

[1] E. Balas, *Discrete programming by the filter method,* Operations Res. 15 (1967), p. 915-967.

[2] — *Machine sequencing via disjunctive graphs — an implicit enumeration algorithm,* ibidem 17 (1969), p. 941-957.

[3] C. Colding-Jorgensen, O. H. Jensen and P. Stig-Nielsen, *Scheduling of trains — an optimization approach,* Proc. 8-th IFIP Conf. Optimization Techniques, Würzburg, September 5-9, 1977, Part 2 (1978), p. 422-433.

[4] E. A. Dinic (Е. А. Диниц), *Алгоритм решения задачи о максимальном потоке в сети со степенной оценкой,* Докл. Акад. наук СССР 194 (1970), p. 754-757.

[5] S. Even and R. E. Tarjan, *Network flow and testing graph connectivity,* SIAM J. Comput. 4 (1975), p. 507-518.

[6] J. Grabowski, *A new formulation and solution of the sequencing problem: mathematical model,* Zastos. Mat. 15 (1976), p. 325-343.

[7] — *Formulation and solution of the sequencing problem with parallel machines,* Proc. 8-th IFIP Conf. Optimization Techniques, Würzburg, September 5-9, 1977, Part 2 (1978), p. 400-410.

INSTITUTE OF TECHNICAL CYBERNETICS
TECHNICAL UNIVERSITY OF WROCŁAW
50-370 WROCŁAW

Received on 29. 3. 1979

J. GRABOWSKI i E. SKUBALSKA (Wrocław)

PRZEPŁYW O MAKSYMALNEJ WARTOŚCI W SIECI O ZMIENNEJ STRUKTURZE

STRESZCZENIE

W pracy omówiono zagadnienie transportowe maksymalnego przepływu w sieci o zmiennej strukturze. Problem został sformułowany przy użyciu pojęć teorii grafów dysjunktywnych. Zdefiniowano pojęcie przepływu bimaksymalnego w sieci dysjunktywnej i udowodniono pewne jego własności. Na tej podstawie sformułowano algorytm rozwiązania zagadnienia, stosując metodę podziału i ograniczeń (strategia podziału z kolejnego węzła).

Do pracy dołączony jest przykład ilustrujący działanie algorytmu i wyniki obliczeniowe na m.c.