ZASTOSOWANIA MATEMATYKI
APPLICATIONES MATHEMATICAE
21,2 (1991), pp. 273–282

G. SCHEITHAUER and J. TERNO (Dresden)

A NEW BRANCH AND BOUND ALGORITHM FOR PROJECT SCHEDULING WITH RESOURCE CONSTRAINTS

Abstract. A new branch and bound algorithm for project scheduling with resource constraints is described. Only the preemptive case is considered. In this treatment the front model for scheduling of project networks is used. Branching is realized in analogy to subtour elimination algorithms for the traveling salesman problem. The main advantage of this algorithm is the use of a lower bound which is based on the resource constraints where the precedence relations are relaxed. The corresponding relaxation problems are solved using the revised simplex method and column generation technique. In order to generate one column (i.e. one subset of jobs which can be processed simultaneously) a special knapsack problem has to be solved.

Introduction. Project scheduling with resource constraints consists of determining a set of starting times for the jobs of the project in such a way that the precedence constraints between them and the resource constraints, limiting the total amount of available resources at each time of the schedule, are satisfied and the total completion time is minimized. In the preemptive case job splitting is allowed, i.e. the processing of any job may be interrupted and resumed at a later time without any additional costs. Preemptive scheduling with resource constraints is an NP-hard problem [15]. Problems of realistic size without resource constraints can be solved by critical path and network flow techniques. Therefore such problems are often considered as relaxation problems for resource constraint problems as well as for the non-preemptive case [5], [15]. In [15] the updating technique in the bound computation is used, and in [5] some improved relaxations are in-

1991 *Mathematics Subject Classification*: 90B35, 65K05.
Key words and phrases: project scheduling, resource constraints, branch and bound algorithm.

vestigated. There, in the underlying integer programming model the linear programming relaxation is used for the calculation of the bounds. In order to improve the quality of the bounds special cutting planes are introduced which avoid the violation of some integer constraints. The computational results in [5], [15] show that the developed branch and bound algorithms solve problems with up to 25 jobs and 3 resources in a reasonable time.

The non-preemptive scheduling problem is the more important practical case, and the knowledge of sharp lower bounds is of greatest interest for the construction of effective branch and bound algorithms. In this paper we consider the preemptive scheduling problem which may be possibly used as a lower bound for the non-preemptive case. The underlying model (see Section 2) is based on the concept of a so-called front. A front is a subset of jobs which may be processed at the same time (i.e. a front is an antichain (independent set) with respect to the precedence relation). Every front is represented in the model by one variable which is the duration of the front. The crucial restriction in the model is the so-called compatibility condition. With respect to the violation of this condition a branching process is proposed in Section 3.2. The branching is similar to that in subtour elimination algorithms for the traveling salesman problem [8]. The bounds (see Section 3) are based upon a linear optimization relaxation containing the resource constraints. Because of the exponential number of variables (fronts) the relaxation problems are solved by the column generation technique. Section 3.3 presents the whole algorithm. In Section 4 the power of the algorithm is demonstrated for a representative scheduling example with one resource constraint.

2. Description of the model. In the following we consider a project scheduling problem with resource constraints and with job splitting (preemptive case). There is given a finite set of jobs $V = \{v_1, \ldots, v_m\}$ and a set I of different types of non-storable resources with total availabilities $R_k, k \in I$. We assume that $R_k, k \in I$, are constant over the whole duration of the project.

A job v_j, $j = 1, \ldots, m$, has a duration l_j and requires an amount r_{jk}, $r_{jk} \leq R_k$, of resource k, $k \in I$. The precedence constraints induce an $m \times m$-adjacency matrix $W = (w_{ij})$ with

$$w_{ij} = \begin{cases} 1 & \text{if } v_i < v_j, \\ -1 & \text{if } v_j < v_i, \\ 0 & \text{otherwise} \end{cases}$$

(i.e. W represents the transitive closure of the precedence relation). $v_i < v_j$ means that v_i is completed before the starting of v_j. A nonempty set F of jobs, $F \subseteq V$, can be processed parallel if there is no pair $u, v \in F$ with $u < v$, i.e. F is an independent set with respect to the given precedence relation.

We call F a *front*. A front F is called *feasible* if all resource constraints are fulfilled, i.e. $\sum_{v_j \in F} r_{jk} \leq R_k$ for all $k \in I$. Every front can be represented in a unique manner by an m-dimensional 0-1 vector $a^F = (a_1^F, \ldots, a_m^F)^T$ with

$$a_i^F = \begin{cases} 1 & \text{if } v_i \in F, \\ 0 & \text{otherwise.} \end{cases}$$

The precedence relation "$<$" between jobs induces a precedence relation "$<$" between fronts. Let a and b represent two fronts. We write $a < b$ if there exist two indices i, j with $a_i * b_j = 1$ and $w_{ij} = 1$. Two fronts a and b are called *incompatible* if both $a < b$ and $b < a$. Let $A = \{F^1, \ldots, F^n\}$ with $n \leq 2^m$ be the set of all feasible fronts. We identify A with an $m \times n$ 0-1 matrix A where the column a^j represents the front F^j.

With the introduced notations the problem of project scheduling with resource constraints can be formulated as follows:

(1) $$z = e^T x = \min \quad \text{(completion time)}$$

subject to

(2) $$Ax = l \quad \text{(duration condition)},$$

(3) there exists no front set $\{a^{p_1}, \ldots, a^{p_q}\}$ with $J = \{p_1, \ldots, p_q\} \subseteq \{1, \ldots, n\}$ and $\prod_{j \in J} x_j > 0$ and $a^{p_1} < a^{p_2} < \ldots < a^{p_q} < a^{p_1}$ (compatibility condition),

(4) $$x \geq 0$$

where

$x = (x_1, \ldots, x_n)^T$ and x_i denotes the duration of front a^i,

$i = 1, \ldots, n$,

$e = (1, \ldots, 1)^T \in \mathbf{R}^n$,

$l = (l_1, \ldots, l_m)^T$.

Our solution strategy is as follows: We construct a branch and bound algorithm relaxing the condition (3). If the solution of the relaxation problem (1), (2), (4) does not fulfill the compatibility condition then branching is necessary. Subproblems are defined using additional precedence conditions.

3. The branch and bound algorithm

3.1. *Computation of the lower bounds.* Given a current subproblem P_0 where at the beginning P_0 is the original problem, then P_0 is characterized by a subset of columns of A fulfilling the current precedence relations of P_0. In order to obtain a lower bound $b(P_0)$ of P_0 the relaxation problem (1), (2), (4) is used. We solve this linear programming problem with the revised simplex method and column generation. First, we choose m linear independent

column vectors in A. These vectors form the basis B, the corresponding variables are denoted by x_B (basis variables), and the remaining column vectors and variables define the matrix A_N and the vector x_N respectively. Now we can write (1), (2), (4) as follows:

$$z = e_B^T x_B + e_N^T x_N = \min \tag{5}$$

subject to

$$B x_B + A_N x_N = l, \tag{6}$$

$$x_B \geq 0, \quad x_N \geq 0. \tag{7}$$

From this we get

$$x_B = -B^{-1} A_N x_N + B^{-1} l, \tag{8}$$

$$z = (e_N^T - e_B B^{-1} A_N) x_N + e_B^T B^{-1} l. \tag{9}$$

If $B^{-1}l \geq 0$ and $e_N^T - e_B^T B^{-1} A_N \geq 0$, then it follows that $x_B = B^{-1}l, x_N = 0$ is an optimal solution of (1), (2), (4). Therefore, to check the optimality of x_B $(= B^{-1}l)$ in the primal simplex algorithm (here $B^{-1}l \geq 0$) we have to determine a feasible front with minimal transformed objective function coefficient. That means we need to solve the problem

$$1 - d^T a^j = \min \tag{10}$$

subject to

$$a^j \text{ is a column of } A_N \tag{11}$$

where $d^T = e_B^T B^{-1}$ denotes the vector of the simplex multipliers. The problem (10), (11) can be replaced by

$$d^T a = \max \tag{12}$$

subject to

$$\sum_{i=1}^{m} r_{ik} a_i \leq R_k, \quad k \in J, \quad \sum_{i=1}^{m} \sum_{j=i+1}^{m} w_{ij} a_i a_j = 0, \quad a_i = 0 \text{ or } 1$$

(i.e. $a = (a_1, \ldots, a_m)^T$ represents a feasible front).

Let a^j be an optimal solution of (12). If $1 - d^T a^j \geq 0$ then $x_B = B^{-1}l, x_N = 0$ is an optimal solution of the relaxation problem (1), (2), (4). Otherwise, a new basis matrix B' containing a^j is to be computed and we have to return to the optimality test in the simplex algorithm.

Because of the linearity of the objective function and the knapsack-like restriction $\sum_{i=1}^{m} r_{ik} a_i \leq R_k$ for all $k \in I$ we call (12) a *special knapsack problem*. In summary we see that the relaxation problem (1), (2), (4) can be solved with the revised simplex method with column generation, where special knapsack problems as subproblems have to be considered in each simplex step. In order to solve the column generation problems (12) one

can use a modification of the best bound search algorithm of Kolesar [9] or an appropriate algorithm (e.g. of Martello–Toth [11]).

3.2. *Branching—Definition of subproblems.* For the calculation of lower bounds we relaxed the compatibility condition (3). Now, let F be the set of fronts in the optimal solution of the relaxation problem corresponding to the current subproblem P_0. If the calculated lower bound $b(P_0)$ is less than the current value of the best known solution, then we need to check whether F fulfills the condition (3) or not. For this we define a directed graph $G = (F, E)$, where the fronts of F are the nodes of G, and for all pairs $F_1 \in F$, $F_2 \in F$ with $F_1 < F_2$ we introduce an arc $(F_1, F_2) \in E$. Then the following statement holds: F fulfills (3) iff G contains no directed circuit (cycle) (see [7]). Therefore, if G contains no cycle, F is an optimal solution of the subproblem considered. Otherwise G contains at least one cycle.

Let $\overline{F} \subseteq F$ be a subset forming a cycle with minimal length (i.e. with minimal number of arcs) in G, and without loss of generality, numbered so that

$$\overline{F} = \{F_1, \ldots, F_r\} \quad (r \geq 2), \quad F_1 < F_2 < \ldots < F_r < F_1 .$$

This implies the existence of jobs $u_i \in F_i$ and $w_i \in F_i$, $i = 1, \ldots, r$, inducing the precedence relation between the fronts by

$$w_i < u_{i+1}, \quad i = 1, \ldots, r-1, \quad w_r < u_1 , \tag{13}$$

where $u_i \neq w_i, i = 1, \ldots, r$, since $\overline{F}$ is a cycle of minimal length.

Remark. The assumption that $\overline{F}$ represents a cycle with minimal length can be replaced by the weaker condition $u_i \neq w_i$, $i = 1, \ldots, r$, but then the number of subproblems of P_0 increases.

In order to avoid the violation of (3) a partition of the considered problem P_0 into subproblems P_i is realized. Therefore, we destroy the cycle of G by prohibition of the nodes (i.e. prohibition of fronts) in analogy to the subtour elimination algorithms (see [4, 8]). The subproblems of P_0 are defined as follows:

$$P_0 \Rightarrow \begin{cases} P_1 = P_0 \wedge (u_1 \nparallel w_1), \\ P_2 = P_0 \wedge (u_1 \| w_1) \wedge (u_2 \nparallel w_2), \\ \vdots \\ P_r = P_0 \wedge (u_1 \| w_1) \wedge \ldots \wedge (u_{r-1} \| w_{r-1}) \wedge (u_r \nparallel w_r). \end{cases} \tag{14}$$

The realization of $u_1 \nparallel w_1$ ("u_1 nonparallel to w_1"), which means the prohibition of the front F_1 and of all fronts containing u_1 and w_1 can be achieved by the condition "$u_1 < w_1$ or $u_1 > w_1$". This corresponds to the partition of P_1 into two disjunctive subproblems.

In the case $u_1 \| w_1$ ("u_1 parallel to w_1"), which means the prohibition of all fronts incompatible with the front $\{u_1, w_1\}$, we consider the additional

conditions

(15) $u < w$ for all $u \in V$ with $u < u_1$ and $w \in V$ with $w_1 < w$ and $w < u$ for all $w \in V$ with $w < w_1$ and $u \in V$ with $u_1 < u$.

By using the corresponding network with W as adjacency matrix, (15) is equivalent to the conditions

(16) $u < w$ for all immediate predecessors $u \in V$ of u_1 resp. w_1 and all immediate sucessors $w \in V$ of w_1 resp. u_1.

For $r = 2$, by (15) the front F_2 with $\{u_2, w_2\} \subseteq F_2$ is excluded since $w_1 < u_2$, $w_2 < u_1$, and it follows that $w_2 < u_2$. In this case a partition of P_2 into two subproblems is not necessary since the subproblem

$$P_0 \wedge (u_1 \| w_1) \wedge (u_2 < w_2) \tag{17}$$

implies the condition $w_1 < u_1$ and therefore (17) is a subproblem of $P_0 \wedge (w_1 < u_1)$. In other words, when $r = 2$ we get three subproblems:

$$P_0 \Rightarrow \begin{cases} P_0 \wedge (w_1 < u_1), \\ P_0 \wedge (u_1 < w_1), \\ P_0 \wedge (u_1 \| w_1). \end{cases} \tag{18}$$

In summary, the branching rule yields subproblems having the same structure as P_0, and only additional precedence conditions are to be taken into consideration.

3.3. *The algorithm.* In this section we describe the whole branch and bound algorithm based on the lower bounds and branching defined in Sections 3.1 and 3.2. Here K denotes the set of subproblems which still have to be investigated. The procedure is summarized as follows:

S0: START
The original problem is used to be the current subproblem P_0. Set $K := \emptyset$ and $z := \infty$.

S1: BOUND
Solve the relaxation problem (1), (2), (4) for P_0 with the revised simplex method and column generation. Let $b(P_0)$ denote the corresponding optimal value (i.e. the bound) and F the set of fronts in the optimal solution of the relaxation problem. If $b(P_0) \geq z$, then go to S3 (i.e. P_0 does not contain a better solution). Find a cycle in the graph $G = (F, E)$ induced by F and the corresponding precedence conditions. If no cycle exists, then set $z := b(P_0)$, save F and go to S3 (i.e. a better feasible solution of (1)–(4) is found).

S2: BRANCH
Select a subset $\overline{F} \subseteq F$ representing a cycle of minimal length in G. Add the subproblems, defined according to (14), to the set K.

S3: BACKTRACK
If $K = \emptyset$, then stop since no better solution exists (z is the optimal value of problem (1)–(4)). Otherwise select a new subproblem P_0 from K in accordance with a suitable branching strategy. Set $K := K \setminus \{P_0\}$ and go to S1.

In Step 3 we may use several branching strategies, e.g. best bound search or depth first search (LIFO). The processing order of the subproblems defined by (14) is variable and should be specified in a suitable manner.

4. An example. We consider the resource constrained network with 20 jobs v_i, $i = 1, \ldots, 20$, as represented in Fig. 1.

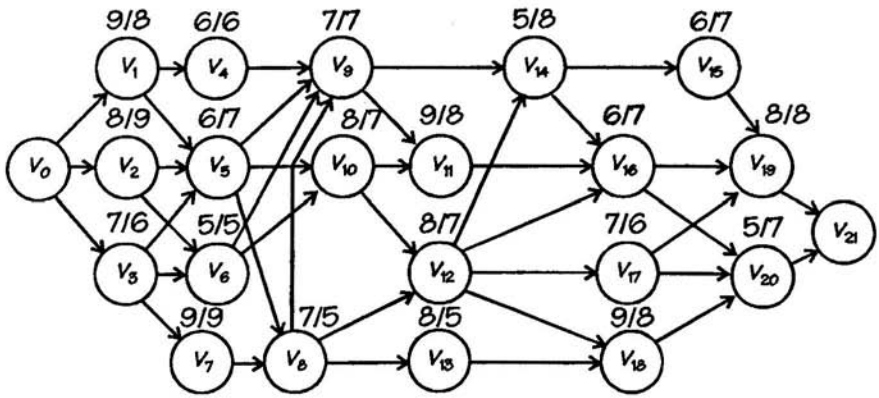

Fig. 1. Precedence relations; the nodes v_i are labelled with l_i/r_i; $R = 18$

The numbers next to the node v_i are the duration l_i and the amount r_i of one resource. The resource capacity is given with $R = 18$. The longest path in the network has length 53 (CPM-bound). The resource constraint gives at least a length

$$\Big(\sum_{i=1}^{20} l_i r_i\Big)/R = 56.389 \quad \text{(resource bound)}.$$

Solving the relaxation (1), (2), (4) of the given problem, we get the lower bound $b(P_0) = 65$. The corresponding optimal solution contains among others the incompatible fronts $F_1 = \{v_{11}, v_{12}\}$ and $F_2 = \{v_9, v_{13}, v_{17}\}$. According to (14) two subproblems

$$P_0 \wedge (v_{11} \nparallel v_{12}) \quad \text{and} \quad P_0 \wedge (v_{11} \| v_{12}) \wedge (v_9 \nparallel v_{17})$$

are defined. In order to realize the second subproblem we add the precedence conditions $v_9 < v_{17}$ and $v_9 < v_{18}$ to the network. According to (18) a partition of P_0 into three subproblems is realized:

$$P_0 \Rightarrow \begin{cases} P_{1,1} = P_0 \wedge (v_{12} < v_{11}), \\ P_{1,2} = P_0 \wedge (v_{11} < v_{12}), \\ P_{1,3} = P_0 \wedge (v_9 < v_{17}) \wedge (v_9 < v_{18}). \end{cases}$$

The complete branching tree for this example using the best bound search is shown in Fig. 2.

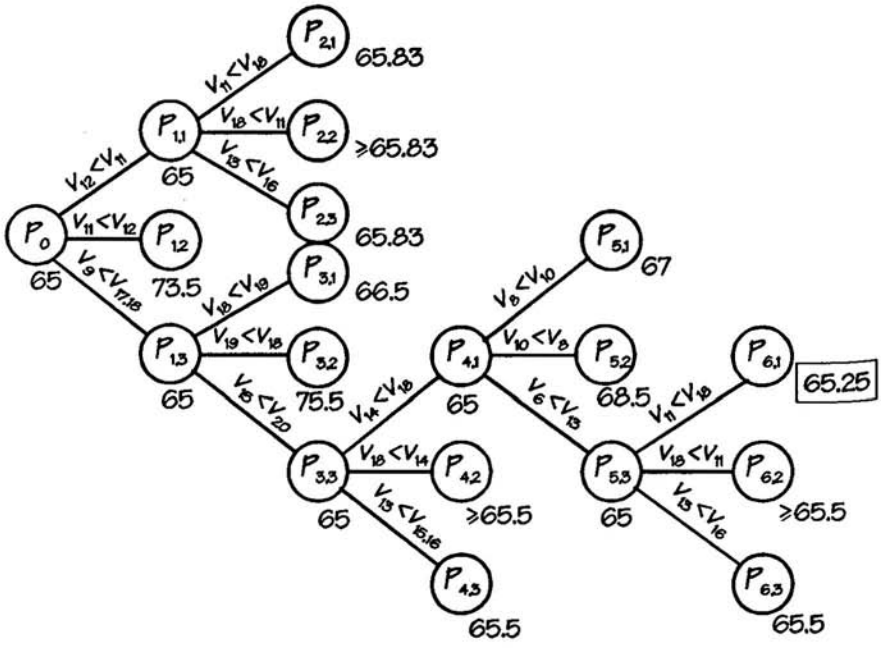

Fig. 2. Branching tree using best bound search

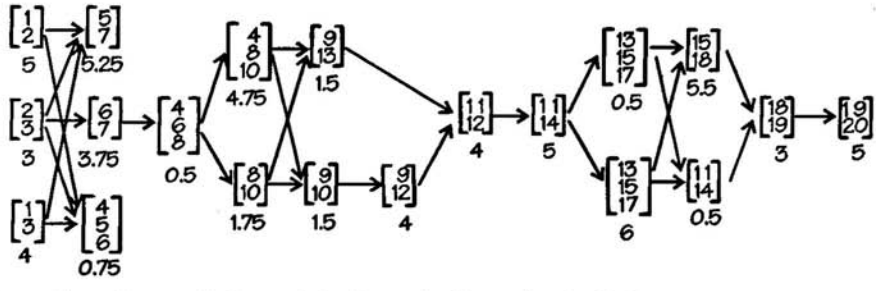

Fig. 3. Precedence relations of the fronts in the optimal solution

The lower bounds of the subproblems, i.e. the optimal values of the corresponding relaxation problems are shown below the nodes. The precedence

conditions defining the subproblems are given next to the arcs. The solution of the relaxation problem of $P_{6,1}$ is the optimal solution of the given problem (1)–(4). This solution $G = (F, E)$ is shown in Fig. 3. Here every front is represented by the indices of its jobs, and the processing duration of a front is denoted beneath it.

In this example a large number of optimal solutions exist since every sequence of fronts of F which does not violate the precedence condition yields an optimal one.

The CPM-bound for $P_{6,1}$ is 53, and it has not increased in comparison with the bound of P_0. Since $P_{2,2}$ is a subproblem of $P_{2,3}$, $P_{4,2}$ of $P_{4,3}$ and $P_{6,2}$ of $P_{6,3}$ it is not necessary to solve the corresponding relaxation problems.

5. Concluding remarks. The branch and bound algorithm for resource constraint project scheduling developed in this paper is based on a relaxation which takes the resource constraints into account. We refer e.g. to [5], where the importance of bound improvements is shown. The column generation technique is used to overcome the difficulties with the exponential-size feasible fronts, but it should not be forgotten that the generation of a column itself is an NP-hard problem. For an efficient implementation of the simplex method we refer to [3]. If the resource constraints are relaxed, the relaxation problems can be solved by critical path techniques. This relaxation technique is used in [5], [7], [15].

References

[1] E. Balas and E. Zemel, *An algorithm for the large zero-one knapsack problem*, Oper. Res. 28 (1980), 1130–1145.

[2] M. Bartusch, *Optimierung von Netzplänen mit Anordnungsbeziehungen bei knappen Betriebsmitteln*, Dissertation, Rheinisch-Westfälische Technische Hochschule Aachen, 1983.

[3] M. Bastian, *Lineare Optimierung großer Systeme*, Verlagsgruppe Athenäum/Hain /Scriptor/Haustein, 1980.

[4] M. Bellmore and J. Malone, *Pathology of traveling salesman subtour elimination algorithm*, Oper. Res. 19 (1971), 278–307.

[5] N. Christofides, R. Alvarez-Valdez and J. M. Tamarit, *Project scheduling with resource constraints: a branch and bound approach*, European J. Oper. Res. 29 (1987), 262–273.

[6] G. Deweß, *Ablaufplanung als verallgemeinertes Matrixspiel; μ-kritische Vorgänge*, Wiss. Z. Karl-Marx-Univ. Leipzig Math.-Natur. Reihe 27 (1978), 485–499.

[7] —, *Die Methode der markierten Verschiebung—ein neues Verfahren zur Optimierung ressourcenbeschränkter Netzplanabläufe*, Math. Operationsforsch. Statist. Ser. Optim. 14 (1983), 217–235.

[8] R. Garfinkel, *On partitioning the feasible set in a branch and bound algorithm for the asymmetric traveling salesman problem*, Oper. Res. 21 (1973), 340–342.

[9] P. S. Kolesar, *A branch and bound algorithm for the knapsack problem*, Management Sci. 13 (1967), 723–735.

[10] E. L. Lawler, J. K. Lenstra and A. H. G. Rinnooy-Kan, *Recent developments in deterministic sequencing and scheduling: A survey*, in: *Deterministic and Stochastic Scheduling*, M. A. H. Dempster, J. K. Lenstra and A. H. G. Rinnooy-Kan (eds.), Reidel, Dordrecht 1982.

[11] S. Martello and P. Toth, *The* 0-1 *Knapsack problem*, in: *Combinatorial Optimization*, N. Christofides, A. Mingozzi, P. Toth and C. Sandi (eds.), Wiley, New York 1979, 237–279.

[12] R. H. Moehring, *Minimizing costs of resource requirements subject to a fixed completion time in project networks*, Oper. Res. 32 (1984), 89–120.

[13] K. Neumann, *Operations Research Verfahren*, Band III: *Graphentheorie, Netzplantechnik*, Carl Hanser Verlag, München 1975.

[14] J. H. Patterson, *A comparison of exact algorithms for the multiple resource constraint project scheduling problem*, Management Sci. 30 (1984).

[15] F. J. Radermacher, *Scheduling of project networks*, Ann. of Oper. Res. 4 (1985/86), 227–251.

[16] F. B. Talbot and J. H. Patterson, *An efficient integer programming algorithm with network cuts for solving resource-constraint scheduling problems*, Management Sci. 24 (1978), 1163–1174.

[17] J. Węglarz, W. Błażewicz, W. Cellary and R. Słowiński, *ARSME—an automatic revised simplex method for resource constrained network schedulings*, ACM Trans. Math. Software 3 (1977), 295–300.

GUNTRAM SCHEITHAUER
JOHANNES TERNO
DEPARTMENT OF MATHEMATICS
DRESDEN TECHNICAL UNIVERSITY
MOMMSENSTR. 13
8027 DRESDEN, GERMANY

Received on 30.1.1990