INSTITUT FÜR WIRTSCHAFTSTHEORIE
UND OPERATIONS RESEARCH
UNIVERSITÄT KARLSRUHE

# Generation of Resource–Constrained Project Scheduling Problems Subject to Temporal Constraints

**Christoph Schwindt**

**Report WIOR–543**

**TECHNICAL REPORT**

**Universität Karlsruhe**

**Kaiserstraße 12, D–76128 Karlsruhe, Germany**

# Generation of Resource–Constrained Project Scheduling Problems Subject to Temporal Constraints

**Christoph Schwindt**

**Report WIOR–543**

**November 1998**

This research has been supported by the Deutsche Forschungsgemeinschaft (Grant Ne 137/4).

**Abstract.** We describe a problem generator for three different types of multi–mode resource–constrained project scheduling problems subject to general temporal constraints: the project duration problem, the resource leveling problem, and the net present value problem. The generation of problem instances decomposes into two steps: the construction of the activity–on–node network and the definition of the resource data. After the construction of an acyclic skeleton of the project network, strong components including at least two nodes are generated. Subsequently, appropriate arc weights are determined. The resource constraints are given by the requirements of the activities and the limited resource capacities. Full factorial design experiments described in literature show that both the network and the resource parameters have a strong impact on the problem hardness.

**Key words:** Project management, resource–constrained project scheduling, problem generation, temporal constraints

# Contents

# 1 Introduction

Several problem generators for project scheduling problems are known from literature (cf. Demeulemeester et al. 1993, Kolisch et al. 1995, and Agrawal et al. 1996). The most popular problem in the field of resource–constrained project scheduling is the Resource–Constrained Project Scheduling Problem (RCPSP) which consists of the minimization of the project duration subject to resource constraints and precedence relationships between activities. Until 1992, an inhomogeneous testset by Patterson (1984) was used to benchmark algorithms for RCPSP. This testset includes problems published in Davis (1969), Patterson and Huber (1974), Davis and Patterson (1975), Talbot and Patterson (1978), and Patterson (1984). Later, the development of efficient branch–and–bound procedures for RCPSP revealed that all problems of the Patterson testset belong to a class of 'easy' problems, that is, they can be solved to optimality within a very short amount of time. Kolisch et al. (1995) showed that there are lower–sized problems which are much harder to solve. Therefore, the experimental performance analysis of algorithms should be based on problem instances which have been generated systematically by a problem generator. The performance of the tested algorithms can then be evaluated depending on different problem parameters.

The network generator by Demeulemeester et al. (1993) generates acyclic weakly connected digraphs where each digraph (with given number of nodes and arcs) is generated with the same probability. Due to the specific approach required for the uniform distribution over the set of all networks, other network measures controlling the shape of the network cannot be observed. The problem generator ProGen by Kolisch et al. (1995) constructs problem instances of RCPSP and the corresponding multi–mode problem MRCPSP. Several network measures as well as parameters determining the tightness of the resource constraints can be specified. Control parameters of the generator by Agrawal et al. (1996) are the number of nodes, the number of arcs, and the CI–index of reduction complexity (the minimum number of node reductions sufficient to reduce a series–parallel digraph to a single edge, cf. Bein et al. 1992). Whereas the networks generated by Kolisch et al. (1995) are so-called activity–on–node networks (that is, the activities are identified with the nodes of the project network, whereas the arcs define precedence constraints between activities), the generators by Demeulemeester et al. (1993) and Agrawal et al. (1994) construct activity–on–arc networks for which the arcs correspond to the activities of a project.

An important generalization of RCPSP is problem RCPSP/max where arbitrary minimum and maximum time lags between the start of activities define general temporal constraints. Maximum time lags can, for instance, be used to model activity deadlines, time–varying resource demands of activities, time–varying resource capacities, or time windows due to technological or organizational restrictions. For applications we refer to Neumann and Schwindt (1997). Project networks including general temporal constraints are no longer acyclic such that the need for the parameter–driven generation of cycle structures arises. The problem generator ProGen/max proposed in this paper extends the features of ProGen to the case of general temporal constraints and to different types of objective functions. The main emphasis is on methods for an efficient parameter–driven construction of cyclic networks. The generator as well as testsets for several types of resource–constrained project scheduling problems are available via world–wide–web (cf. Kolisch et al. 1998).

The remainder of this paper is organized as follows. Section 2 is concerned with three different types of resource–constrained project scheduling problems. The algorithm for the generation of the activity–on–node project network is described in Section 3. Section 4 is devoted to the generation of resource constraints.

# 2 Models and Notation

The project scheduling problems to be dealt with throughout this paper can be stated as follows. A set $V = \{0, 1, \ldots, n, n+1\}$ of *activities* has to be executed where the fictitious activities 0 and $n+1$ represent the beginning and the termination of the project, respectively. The processing of activities uses *renewable resources* and consumes *nonrenewable resources*. The sets of renewable and nonrenewable resources resources are denoted by $\mathcal{R}^\rho$ and $\mathcal{R}^\nu$, respectively. From a renewable resource $k \in \mathcal{R}^\rho$, $R_k$ units are available at any point in time. From a nonrenewable resource $k \in \mathcal{R}^\nu$, $R_k$ units are available in total. $R_k$ is referred to as the *capacity* of resource $k$.

Each activity $i \in V$ has to be performed in one out of several execution modes $m_i \in \mathcal{M}_i$. If all activities $i \in V$ possess exactly one execution mode, the scheduling problem is a *single–mode problem*. Otherwise, we speak of *multi–mode problem*. If carried out in mode $m_i$, activity $i$ uses $r_{im_ik}$ units of renewable resources $k \in \mathcal{R}^\rho$ during its execution and consumes $r_{im_ik}$ units of nonrenewable resources $k \in \mathcal{R}^\nu$. As for the resource requirements, the *durations* or *processing times* $p_{im_i}$ of activities $i \in V$ depend on the execution mode $m_i \in \mathcal{M}_i$. While being processed, activities must not be interrupted.

Between the starts of two activities minimum and maximum time lags may be prescribed. In general, these temporal constraints depend on the execution modes $m_i$ and $m_j$ of both activities. Let $i$ and $j$ be two distinct activities executed in modes $m_i$ and $m_j$, respectively. A *minimum time lag* $d^{min}_{im_ijm_j} \geq 0$ states that $j$ can start $d^{min}_{im_ijm_j}$ units of time after the start of $i$ at the earliest. A *maximum time lag* $d^{max}_{im_ijm_j} \geq 0$ implies that $j$ has to start $d^{max}_{im_ijm_j}$ units of time after the start of $i$ at the latest. A minimum time lag between activities $i$ and $j$ which equals the duration of $i$ is termed *precedence constraint* between $i$ and $j$.

In the following, all input data are supposed to be integer–valued.

Activities $i \in V$ as well as the temporal constraints can be represented by an *activity–on–node network* $N = \langle V, E; \delta \rangle$ with node set $V$, arc set $E$, and arc weights $\delta$. If between two activities $i$ and $j$ a minimum time lag is prescribed, we introduce an arc $\langle i, j \rangle$ from node $i$ to node $j$ weighted by matrix $(d^{min}_{im_ijm_j})_{m_i \in \mathcal{M}_i, m_j \in \mathcal{M}_j}$. Maximum time lags between activities $i$ and $j$ are represented by arcs $\langle j, i \rangle$ from node $j$ to node $i$ weighted by matrix $(-d^{max}_{im_ijm_j})_{m_i \in \mathcal{M}_i, m_j \in \mathcal{M}_j}$. In general, network $N$ may be cyclic. There is no feasible solution to the corresponding project scheduling instance if the activity modes cannot be chosen such that $N$ does not contain cycles of positive length.

For each activity $i \in V$ a start time $S_i$ and an execution mode $m_i$ have to be determined such that the usage of the renewable resources does not exceed the respective resource capacities at any point in time, the consumption of nonrenewable resources is restricted to the corresponding availabilities, the minimum and maximum time lags are met, and a given objective function $f$ in the start times and mode assignments is minimized.

A solution to the above problem can be specified by a *schedule*

$$S = (S_0, \ldots, S_{n+1})$$

and a *vector of mode assignments*

$$x = (x_{01}, x_{11}, \ldots, x_{1|\mathcal{M}_1|}, \ldots, x_{n1}, \ldots, x_{n|\mathcal{M}_n|}, x_{n+1,1}).$$

The binary decision variable $x_{im_i}$ equals one exactly if activity $i$ is performed in mode $m_i$. Given a solution $(S, x)$, the set of activities which are in progress at time $t$ is

$$\mathcal{A}(S, x, t) = \{j \in V \mid S_j \leq t \leq \sum_{m_i \in \mathcal{M}_i} x_{im_i} p_{im_i}\}.$$

The corresponding usage of renewable resource $k \in \mathcal{R}^\rho$ at time $t$ is then given by

$$r_k(S, x, t) = \sum_{i \in \mathcal{A}(S,x,t)} \sum_{m_i \in \mathcal{M}_i} r_{im_ik} x_{im_i}$$

and the consumption of resource $k \in \mathcal{R}^\nu$ is determined by

$$r_k(x) = \sum_{i \in V} \sum_{m_i \in \mathcal{M}_i} r_{im_ik} x_{im_i}.$$

The resource–constrained project scheduling problem (PSP) can now be stated as follows:

$$(\mathrm{PSP}) \begin{cases} \min & f(S,x) & & (1) \\ \text{s.t.} & S_0 = 0 & & (2) \\ & S_j - S_i \geq \sum_{m_i, m_j \in \mathcal{M}_i} \delta_{im_ijm_j} x_{im_ijm_j} & (\langle i,j \rangle \in E) & (3) \\ & \sum_{m_i \in \mathcal{M}_i} x_{im_i} = 1 & (i \in V) & (4) \\ & r_k(S,x,t) \leq R_k & (k \in \mathcal{R}^\rho, t \geq 0) & (5) \\ & r_k(x) \leq R_k & (k \in \mathcal{R}^\nu) & (6) \\ & x_{im_i} \in \{0,1\} & (i \in V, m_i \in \mathcal{M}_i) & (7) \end{cases}$$

For the objective function

$$f(S,x) = S_{n+1}$$

(PSP) corresponds to a *project duration problem.* If $f(S,x)$ is a function in the resource profiles $r_k(S,x,t)$ of the renewable resources $k \in \mathcal{R}^\rho$ and in the resource consumption $r_k(S,x,t)$ of the nonrenewable resources $k \in \mathcal{R}^\nu$, (PSP) is a *resource–levelling problem.* Two levelling objective functions which have been treated in literature for the single–mode case are

$$f(S,x) = \sum_{k \in \mathcal{R}^\rho} \max_{t \geq 0} c_k r_k(S,x,t) + \sum_{k \in \mathcal{R}^\nu} c_k r_k(x)$$

and

$$f(S,x) = \sum_{k \in \mathcal{R}^\rho} c_k \int_{t=0}^{\infty} (r_k(S,x,t) - y_k)^2 dt + \sum_{k \in \mathcal{R}^\nu} c_k (y_k - r_k(x))^2$$

where $c_k$ are the unit costs of resource $k$ and $y_k$ is a threshold value for the resource requirements. The objective function value of a *net present value problem* is

$$f(S,x) = -\sum_{i \in V} \sum_{m_i \in \mathcal{M}_i} c^F_{im_i} x_{im_i} e^{-\alpha(S_i + p_i)}$$

with $c^F_{im_i}$ as the cash flow associated with activity $i$ if it is executed in mode $m_i$. For a comprehensive review of algorithms for project duration problems, resource–levelling problems, and net present value problems we refer to Brucker et al. (1998).

We now briefly consider the hardness of the feasibility problems of (PSP) and of relaxations of (PSP). Due to the cyclicity of the underlying project network, the feasibility problem of (PSP) is strongly $\mathcal{NP}$–complete even for the single–mode case (Bartusch et al. 1988 provide a polynomial reduction to CLIQUE). For the temporal relaxation of (PSP) (i.e. problem (1) s.t. (4), (5), (6), (7)) the existence of feasible solution can be verified in polynomial time provided that there is at most one nonrenewable resource. For the general case with arbitrary number of nonrenewable resources the problem has been shown to be $\mathcal{NP}$–complete by Kolisch (1995, polynomial reduction to KNAPSACK). The single–mode version of the resource relaxation of (PSP) (i.e. problem (1) s.t. (2), (3), (6), (7)) with objective function $S_{n+1}$ can be solved in polynomial time by network flow algorithms. For the multi–mode case the corresponding feasibility problem turns out to be strongly $\mathcal{NP}$–complete.

**Theorem 1.**

The decision problem whether the resource–relaxation of (PSP) possesses a feasible solution is strongly $\mathcal{NP}$–complete.

**Proof.**

We construct a polynomial reduction to PARTIALLY ORDERED KNAPSACK (cf. e.g. Garey and Johnson 1979). For a given set of objects $i$ integer benefits $u_i > 0$ and integer weights $w_i > 0$ are given. The knapsack's capacity is limited by $W$. Moreover, a partial order on the set of objects is given whose elements $(i, j)$ imply that $j$ has to be selected if $i$ is put into the knapsack. The objective is to maximize the benefit $U$ of all selected objects such that the partial order and the capacity of the knapsack are respected.

For each object $i$, we introduce two activities $i'$ and $i''$ with two distinct execution modes each. Object $i$ is selected exactly if the processing of activities $i'$ and $i''$ is performed in the first mode.

We model a partial order $\preceq$ in the set of activities establishing that an activity $j'$ has to be performed in the first mode if an activity $i'$ is carried out in the first mode. To this end we introduce two arcs $\langle i', j'\rangle$ and $\langle j', i'\rangle$ weighted by

$$(\delta_{i'j'}) = \begin{pmatrix} -\infty & 1 \\ -\infty & -\infty \end{pmatrix} \text{ and } (\delta_{j'i'}) = \begin{pmatrix} -\infty & -\infty \\ 1 & -\infty \end{pmatrix}$$

such that the combination $m_{i'} = 1$, $m_{j'} = 2$ results in time–infeasibility. *Mode identity constraints* $i' \sim j'$ between two activities $i'$ and $j'$ can now be expressed by $i' \preceq j'$ and $j' \preceq i'$. For any pair $(i', i'')$ of activities belonging to an object $i$ we define $i' \sim i''$. The partial order in the set of objects is taken into account by corresponding relationships $i' \preceq j'$. Let $i_1, \ldots, i_{n/2}$ be a consecutive numbering of the objects. Between any two contiguous activities $i'_q$ and $i'_{q+1}$ we introduce an arc $\langle i'_q, i'_{q+1} \rangle$ weighted by

$$(\delta_{i'_q i'_{q+1}}) = \begin{pmatrix} -u_{i_q} & -u_{i_q} \\ 0 & 0 \end{pmatrix}.$$

Analogously, two contiguous activities $i''_q$ and $i''_{q+1}$ are joined by arc $\langle i''_q, i''_{q+1} \rangle$ weighted by

$$(\delta_{i''_q i''_{q+1}}) = \begin{pmatrix} -w_{i_q} & -w_{i_q} \\ 0 & 0 \end{pmatrix}.$$

Next, we add arcs $\langle i'_{n/2}, i'_1 \rangle$ and $\langle i''_{n/2}, i''_1 \rangle$ weighted by

$$(\delta_{i'_{n/2} i'_1}) = \begin{pmatrix} U - u_{i_{n/2}} & U - u_{i_{n/2}} \\ U - u_{i_{n/2}} & U - u_{i_{n/2}} \end{pmatrix}$$

and

$$(\delta_{i''_{n/2} i''_1}) = \begin{pmatrix} -W + w_{i_{n/2}} & -W + w_{i_{n/2}} \\ -W + w_{i_{n/2}} & -W + w_{i_{n/2}} \end{pmatrix},$$

respectively. Finally the dummies 0 and $n+1$ are linked up with the initial activities $i'_1$ and $i''_1$ and the terminal activities $i'_{n/2}$ and $i''_{n/2}$, respectively. The corresponding arc weights are chosen to $(\sum_{q=1}^{n/2-1} u_{i_q}, \sum_{q=1}^{n/2-1} u_{i_q})$, $(0, 0)$, $(0, 0)^\top$, and $(0, 0)^\top$.

The resulting (PSP) instance possesses a time–feasible solution exactly if there is feasible solution to the knapsack problem with objective function greater than or equal to $U$. □

# 3 Activities and Temporal Constraints

In this section we consider the parameter–driven generation of the project network $N = \langle V, E; \delta \rangle$ which reflects the relationships between the activities of the project given by minimum and maximum time lags. The generation can be decomposed into three steps. First, we randomly generate the execution modes $m_i$ of the activities $i \in V$ with corresponding durations $p_{im_i}$ and construct an acyclic weakly connected activity–on–node digraph $G = \langle V, E \rangle$. Subsequently, we create cycles in $G$ by an appropriate introduction of additional arcs. The resulting digraph $G$ represents the network structure of $N$.

Finally, the arcs of $G$ are weighted such that a necessary condition for the existence of a time–feasible solution to (PSP) is secured.

Throughout this section we use the following notation. Let $P = (\rho_{ij})_{i,j \in V}$ be the reachability matrix of $G$. $\rho_{ij}$ equals 1 exactly if there is a directed path in $G$ from $i$ to $j$, and 0, otherwise. In particular, $\rho_{ii}$ equals 1 for all nodes $i \in V$. By $\rho_{ij}^{(2)} = \sum_{h \in V} \rho_{ih}\rho_{hj}$ we denote the $(i, j)$–element of the squared reachability matrix.

## 3.1 Acyclic Skeleton

We now discuss the construction of a weakly connected acyclic digraph $G = \langle V, E \rangle$ with node set $V$ and arc set $E$. After the generation of the node set we select those nodes which will be sources and sinks of $G$. Then, we add non–redundant arcs such that $G$ becomes weakly connected. Further non–redundant arcs are added until a measure controlling the number of precedence relationships defined by $G$ has been reached. Finally, we introduce redundant arcs which do not affect the precedences among the activities.

In the following, we provide some results on redundant arcs in acyclic digraphs. We call an arc $\langle i, j \rangle$ *redundant* in an acyclic digraph, if there is path from $i$ to $j$ which contains at least two arcs.

**Proposition 1.**

An arc $\langle i, j \rangle$ is redundant exactly if $\rho_{ij}^{(2)} > 2$.

The proof relies on the property that the number of nodes on all directed paths in $G$ from a node $i$ to a node $j$ equals $\rho_{ij}^{(2)}$.

The addition of an arc $\langle i, j \rangle$ to $G$ may generate redundancy even if $\langle i, j \rangle$ is not redundant itself. The following definition refers to arcs whose addition increases the number of redundant arcs in $G$.

**Definition 1.** *Redundancy–generating arc*

An arc $\langle i, j \rangle$ is called redundancy–generating, if $\langle i, j \rangle$ is redundant or if there is a distinct arc $\langle g, h \rangle \in E$ such that $\langle i, j \rangle$ belongs to a path from $g$ to $h$.

The next proposition provides a condition for redundancy–generating arcs which will be used for the construction of digraphs without redundant arcs.

**Proposition 2.**

An arc $\langle i, j \rangle \notin E$ is redundancy–generating exactly if

$$\rho_{ij}^{(2)} + \sum_{\langle g,h \rangle \in E} \rho_{gi}\rho_{jh} > 0.$$

The structure of the underlying project network generally has a strong impact on the time which enumeration algorithms require for the solution of combinatorial optimization problems as well as on the gap between the optimum and the objective function values of solutions which have been obtained by heuristics. In literature, a large number of network measures can be found which characterize the logic and and the shape of networks (cf. Kaiman 1974, Davis 1975, Patterson 1976, Thesen 1977, Elmaghraby and Herroelen 1980, or Kurtulus and Davis 1982). Since project scheduling is concerned with the definition of precedence relationships between activities, the number of feasible linear orderings of $V$ may serve as a measure for the computational effort of enumeration procedures. A corresponding $[0,1]$–normalized network parameter is the restrictiveness of the strict order $\{(i,j) \in V^2 \mid i \neq j, \rho_{ij} = 1\}$.

**Definition 2.** *Restrictiveness*

Let $\prec_G$ be the strict order in node set $V$ given by digraph $G$ and let $l(\prec_G)$ denote the number of linear extensions of $\prec_G$. The restrictiveness $\sigma$ of $\prec_G$ is defined as

$$\sigma = 1 - \log \frac{l(\prec_G)}{|V|!}$$

where $|V|!$ represents the maximum number of linear extensions of a partial order in set $V$.

The restrictiveness measures the degree to which the precedences between activities given by the arcs of $G$ restrict the number of feasible execution sequences of the activities. For parallel digraphs $G$, the restrictiveness of $\prec_G$ is minimum and equals 0, for series digraphs the corresponding restrictiveness is maximum and equals 1. The higher the restrictiveness of $\prec_G$, the less different alternatives exist to resolve resource conflicts between activities by the definition of additional precedence constraints. For this reason, $\sigma$ constitutes a problem parameter which directly influences the hardness of many resource–constrained project scheduling instances both with respect to the feasibility problem and the optimization problem. Unfortunately, the determination of the restrictiveness of a strict order is $\#\mathcal{P}$ complete. For the generation of the acyclic skeleton we use the order strength, a control parameter which is closely related to the restrictiveness.

**Definition 3.** *Order strength*

Let $\prec_G$ be the strict order in node set $V$ given by digraph $G$. The order strength $OS$ of $\prec_G$ is defined as

$$OS = \frac{|\prec_G|}{|V|(|V|-1)/2}$$

where $|V|(|V|-1)/2$ represents the maximum number of elements of a partial order in set $V$.

As for the restrictiveness, the order strength is a $[0,1]$–normalized measure which is minimum for parallel and maximum for series digraphs. Moreover, the addition of a non–redundant arc to $G$ increases $OS$ whereas $OS$ remains unchanged if redundant arcs are added. In order to assess the quality of $OS$ as an approximation of $\sigma$ we generated 330 acyclic digraphs including 10 nodes each where $OS$ was varied from 0.2 to 1.0 and determined the respective values for the restrictiveness and the order strength. Based on this set of digraphs a linear regression relating $OS$ to $\sigma$ provided a squared correlation coefficient of 0.923, that is, $OS$ accounts for more than 92% of $\sigma$'s variance. This confirms the results obtained by Thesen (1977) identifying $OS$ as the best approximation for $\sigma$ among over 40 tested network parameters. Moreover, De Reyck (1995) showed that regarding the correlation to the hardness of instances $OS$ clearly outperforms the network measures which have been used by Kolisch et al. (1995) and Agrawal et al. (1996).

As stated by the following Proposition, $OS$ can be determined very efficiently.

**Proposition 3.**

The order strength $OS$ of $\prec_G$ satisfies

$$OS = \frac{\sum_{i,j \in V} \rho_{ij} - |V|}{|V|(|V|-1)/2}.$$

Arcs $\langle i, j \rangle$ which are redundant w.r.t. to the structure $G$ of network $N$ may be associated with time lags which cannot be omitted without modification of the feasible region. This holds to be true if the length of a longest path from $i$ to $j$ consisting of at least two arcs is less than the weight of arc $\langle i, j \rangle$. That is why the acyclic skeleton of the project network $N$ generally also includes redundant arcs. The number of redundant arcs to be generated is controlled by the degree of redundancy which refers to the maximum number of redundant arcs which can be added to $G$. This number depends on the structure of $G$ and can be determined as follows.

**Proposition 4.**

The maximum number $m_{red}^{max}$ of redundant arcs which can be added to an acyclic digraph $G$ without redundant arcs is

$$m_{red}^{max} = \sum_{i,j \in V} \rho_{ij} - |V| - |E|.$$

Table 1 lists the control parameters for the generation of the acyclic skeleton of the project network.

Table 1: Parameters acyclic skeleton

| Symbol | Parameter |
|---|---|
| $n^{min}, n^{max}$ | min. and max. number of real activities |
| $\mu^{min}, \mu^{max}$ | min. and max. number of modes per real activity |
| $p^{min}, p^{max}$ | min. and max. duration of modes |
| $cf^{min}, cf^{max}$ | min. and max. cash flows of modes |
| $r^{min}, r^{max}$ | min. and max. number of sources |
| $s^{min}, p^{max}$ | min. and max. number of sinks |
| $pred^{max}$ | max. number of predecessors |
| $succ^{max}$ | max. number of successors |
| $os^{min}$ | min. order strength |
| $\rho$ | degree of redundancy |

The algorithm for the generation of the acyclic skeleton of project network $N$ is given by Fig. 1.

Determine number of activities $n \in \{n^{min}, \ldots, n^{max}\}$
and set $V := \{1, \ldots, n\}$.
**FOR** $i \in V$ **DO**
  Select number $\mu_i$ of modes in $\{\mu^{min}, \ldots, \mu^{max}\}$
  and set $\mathcal{M}_i := \{1, \ldots, \mu_i\}$
  **FOR** $m_i \in \mathcal{M}_i$ **DO**
    Determine duration $p_{im_i} \in \{p^{min}, \ldots, p^{max}\}$.
    Determine cash flow $c^F_{im_i} \in \{cf^{min}, \ldots, cf^{max}\}$.
Determine number $r \in \{r^{min}, \ldots, r^{max}\}$ of sources and number $s \in \{s^{min}, \ldots, s^{max}\}$ of sinks.
Generate predecessors $i$ of nodes $j \in \{r+1, \ldots, n\}$ and successors $j$ of nodes $i \in \{1, \ldots, n-s\}$. Secure that resulting digraph $G$ is acyclic, does not contain redundant arcs, and observes the limitation w.r.t. maximum indegree and outdegree of nodes.
**WHILE** the order strength of $\prec_G$ is less than $os^{min}$ **DO**
  Add additional arc without generating redundancy or cycles. Observe the limitation w.r.t. maximum indegree and outdegree of nodes.
Add $\lfloor \rho m^{max}_{red} \rfloor$ redundant arcs such that $G$ remains acyclic.
Add dummy activities 0 and $n+1$ to $V$.
Introduce arcs $\langle 0, i \rangle$ for all $i \in \{1, \ldots, r\}$ and arcs $\langle i, n+1 \rangle$ for all $i \in \{n-s+1, \ldots, n\}$.

Figure 1: Generation of acyclic skeleton

## 3.2 Cycle Structures

Neumann and Zhan (1994) have shown that an instance of (PSP) with at most one nonrenewable resource is solvable exactly if a feasible schedule can be assigned to all partial projects corresponding to the execution of the activities belonging to one and the same strong component of $N$. That is why the size and the structure of strong components of the project network play a crucial role w.r.t. to the hardness of project scheduling instances. A strong component including at least two nodes is termed a *cycle structure*. We consider three different characteristics of cycle structures: their number, their size (i.e. the number of included nodes), and their density (i.e. the number of arcs joining two nodes which belong to the same cycle structure). The generation of cycle structures in $G$ decomposes into three phases. First, we create the desired number of cycle structures. Then, they are extended to the prescribed size, and finally additional redundant arcs are added in order to obtain the specified density.

**Definition 4.** *Creation, extension, and densification of cycle structures*

Let $G'$ be the digraph which results from $G$ by addition of arc $\langle j, i \rangle$ with $\rho_{ij} = 1$. The set of cycle structures of $G$ and $G'$ are denoted by $\mathcal{C}$ and $\mathcal{C}'$, respectively. The operation transforming $G$ into $G'$ is called

1. creation of a cycle structure if $|\mathcal{C}'| > |\mathcal{C}|$,
2. extension of a cycle structure if $|\mathcal{C}'| = |\mathcal{C}|$ and $\sum_{C \in \mathcal{C}'} |C| > \sum_{C \in \mathcal{C}} |C|$,
3. densification of a cycle structure if $|\mathcal{C}'| = |\mathcal{C}|$ and $\sum_{C \in \mathcal{C}'} |C| = \sum_{C \in \mathcal{C}} |C|$.

An arc $\langle j, i \rangle$ which is added to $G$ is called *forward arc* if $\rho_{ij} = 0$ and *backward arc*, otherwise. All arcs of the acyclic skeleton are forward arcs whereas the arcs which are added for the generation of cycle structures are backward arcs. The size of the cycle structures is influenced by the creation and the extension of cycle structures. The sufficient and necessary conditions of the next proposition allow for an efficient generation of cycle structures according to the above three–phase approach. It can be shown (cf. Schwindt, 1996) that any digraph which results from addition of backward arcs to the acyclic skeleton can be generated by the consecutive creation, extension, and densification of cycle structures.

**Proposition 5.**

The addition of a backward arc $\langle j, i \rangle \notin E$

1. creates a cycle structure, exactly if $\sum_{h \in V : \rho_{hh}^{(2)} > 1} \rho_{ih} \rho_{hj} = 0$,
2. extends a cycle structure, exactly if $\sum_{h \in V : \rho_{hh}^{(2)} > 1} \rho_{ih} \rho_{hj} = 1$,
3. densifies a cycle structure, exactly if $\rho_{ji} = 1$.

As for the redundancy of arcs, the number of nodes belonging to a strong component can be controlled by means of the squared reachability matrix $P^2$.

**Proposition 6.**

The addition of a backward arc $\langle j, i \rangle$ leads to a cycle structure including $\rho_{ij}^{(2)}$ nodes.

Table 2: Parameters cycle structures

| Symbol | Parameter |
|---|---|
| $p_{back}^{min}, p_{back}^{max}$ | min. and max. percentage of backward arcs |
| $cs^{min}, cs^{max}$ | min. and max. number of cycle structures |
| $n_c^{min}, n_c^{max}$ | min. and max. size of cycle structure |
| $p_{dens}$ | percentage of arcs employed for densification after creation of cycle structures |

Table 2 shows the network parameters referring to the cycle structures of $G$. Fig. 2 summarizes the algorithm for the generation of the cycle structures of project network $N$.

Determine number of backw. arcs $m_{back} \in \{\lceil |E| p_{back}^{min} \rceil, \dots, \lfloor |E| p_{back}^{max} \rfloor\}$.
Determine number of cycle struct. $cs \in \{cs^{min}, \dots, \min\{m_{back}, cs^{max}\}\}$.
**REPEAT** $cs$ times
  Create a new cycle structure.
**REPEAT** $\lfloor (1 - p_{dens})(m_{back} - cs) \rfloor$ times
  Extend a cycle structure.
**REPEAT** $\lceil p_{dens}(m_{back} - cs) \rceil$ times
  Densify a cycle structure.

Figure 2: Generation of cycle structures

## 3.3 Arc Weights

The determination of an appropriate arc weight differs whether the arc belongs to the acyclic skeleton or if it has been added during the generation of cycle structures. Let $E_{forw}$ and $E_{back}$ denote the sets of forward and backward arcs, respectively. For $\langle i, j \rangle \in E_{forw}$ the arc weights are chosen such that a maximum relative deviation from the respective durations of activity $i$ is not exceeded. For the generation of weights for $\langle j, i \rangle \in E_{back}$ a lower bound $\underline{d}_{ij}$ and an upper bound $\overline{d}_{ij}$ are calculated on the time which may necessarily elapse between the starts of activities $i$ and $j$. $\underline{d}_{ij}$ is set to the $(i, j)$–element of the *distance matrix* $\underline{D}$. $\underline{D}$ corresponds to the matrix of longest path lengths in the network with node set $V$, arc set $E_{forw}$, and arc weights $\delta_{ij} = \min_{i \in \mathcal{M}_i, j \in \mathcal{M}_j} \delta_{im_i jm_j}$. The upper bound $\overline{d}_{ij}$ is chosen such that activity $i$ and all activities $g$ which due to temporal constraints can neither be started arbitrarily earlier than $i$ nor arbitrarily later than $j$ can be executed before $j$ observing the renewable resource constraints:

$$\overline{d}_{ij} = \sum_{\substack{g \in V \setminus \{j\} \\ \rho_{ig}\rho_{gj}=1}} \max_{m_g \in \mathcal{M}_g} \max\{p_{gm_g}, \max_{\substack{\langle g,h \rangle \in E_{forw} \\ \rho_{hj}=1}} \max_{m_h \in \mathcal{M}_h} \delta_{gm_g hm_h}\}$$

The weights of backward arcs are now randomly drawn from an interval whose boundary points depend on a parameter $\tau_c$ called cycle structure tightness (cf. Fig. 3). For $\tau_c = 0$ the arc weight coincides with the negative upper bound $\overline{d}_{ij}$ times a slack factor $\sigma_c$. For $\tau_c = 1$ the arc weight coincides with the negative lower bound $\underline{d}_{ij}$ times $\sigma_c$. A positive slack factor avoids that activities are firmly tied by temporal constraints.

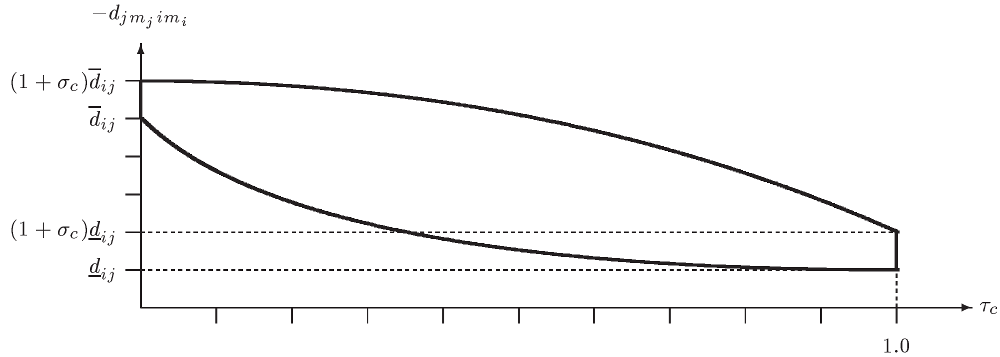


Figure 3: Arc weight depending on cycle structure tightness

The control parameters for the generation of the arc weights are given by Table 3.

Table 3: Parameters arc weights

| Symbol | Parameter |
|---|---|
| $\epsilon_p^{max}$ | max. relative deviation of forward arc weight from duration |
| $\tau_c$ | tightness of cycle structures |
| $\sigma_c$ | slack factor of cycle structures |
| $\sigma_{\overline{d}}$ | slack factor of max. project duration |

Fig. 4 shows the algorithm for the generation of minimum and maximum time lags between activities. After the determination of arc weights it is secured that no activity starts before the beginning of the project or ends after the termination of the project. Eventually, additional arcs emanating from node 0 or leading into node $n+1$, respectively, have to be introduced. Finally, the maximum project duration $\overline{d}$ is determined on the basis of the lower bound $\underline{d}_{0,n+1}$ and the corresponding slack factor $\sigma_{\overline{d}}$.

**FOR** $\langle i,j\rangle \in E_{forw}$ **DO**
  **FOR** $m_i \in \mathcal{M}_i, m_j \in \mathcal{M}_j$ **DO**
    **IF** $i = 0$ **THEN**
      Set weight $\delta_{0,1,jm_j} := 0$.
    **ELSE IF** $j = n+1$ **THEN**
      Set weight $\delta_{im_i,1,n+1} := p_{im_i}$.
    **ELSE**
      Determine weight
      $\delta_{im_ijm_j} \in \{\lfloor(1-\epsilon_p^{max})p_{im}\rfloor, \ldots, \lceil(1+\epsilon_p^{max})p_{im}\rceil\}$.
Compute distance matrix $\underline{D}$.
**FOR** $\langle j,i\rangle \in E_{back}$ **DO**
  Determine upper bound $\overline{d}_{ij}$ and set $\Delta d_{ij} := \overline{d}_{ij} - \underline{d}_{ij}$.
  **FOR** $m_i \in \mathcal{M}_i, m_j \in \mathcal{M}_j$ **DO**
    Determine weight $\delta_{jm_jim_i} \in$
    $\{-\lceil\overline{d}_{ij} - \Delta d_{ij}\tau_c^2\rceil(1+\sigma_c), \ldots, -\lfloor(\overline{d}_{ij} - 2\Delta d_{ij}\tau_c + \Delta d_{ji}\tau_c^2)\rfloor\}$.
  Update matrix $\underline{D}$ by $\underline{d}_{ji} := \max\{\underline{d}_{ji}, \min_{m_i\in\mathcal{M}_i, m_j\in\mathcal{M}_j} \delta_{jm_jim_i}\}$ and restore transitivity of $\underline{D}$.
**FOR** $i \in V$ **DO**
  **IF** $\underline{d}_{0i} < 0$ **THEN** add arc $\langle 0,i\rangle$ weighted by $(0)_{m_i\in\mathcal{M}_i}$.
  **IF** $\overline{d}_{i,n+1} < \max_{m_i\in\mathcal{M}_i} p_{im_i}$ **THEN** add arc $\langle i,n+1\rangle$ weighted by $(p_{im_i})^\top_{m_i\in\mathcal{M}_i}$.
Set maximum project duration $\overline{d} := \underline{d}_{0,n+1}(1+\sigma_{\overline{d}})$.

Figure 4: Generation of arc weights

# 4 Resource Constraints

The resource constraints of problem (PSP) are given by the resource requirements of the activities on the one hand and the limited resource capacities on the other hand. Numerous resource characteristics for resource-constrained scheduling problems can be found in literature, for example in Kurtulus and

Davis (1982), Patterson (1976), Davis (1982), Kurtulus and Narula (1985), and Kolisch et al. (1995). By generalizing and normalizing measures which had been used in literature, Kolisch et al. (1995) developed a new set of control parameters which have a strong impact on the hardness of problem instances. ProGen/max employs the same set of resource measures for the problem generation. In the following, we describe the generation of resource requirements and resource availabilities.

## 4.1 Resource Requirements

The processing of an activity uses or consumes a certain amount of renewable or nonrenewable resources, respectively. After the determination of the numbers of resources, the generation of resource usage and consumption is performed in two steps. First, for any given activity-mode combination $(i, m_i)$ we select a set of resources required for the processing of activity $i$ in mode $m_i$ (generation of *requests*). This process is controlled by two types of parameters: the minimum and the maximum number of resources which may be requested for the processing of activities and the so–called resource factors which indicate the mean percentage of resources which are affected by the execution of an activity:

$$rf_\rho = 1/(|\mathcal{R}^\rho|n) \sum_{k\in\mathcal{R}^\rho} \sum_{i\in V} 1/|\mathcal{M}_i| \sum_{m_i\in\mathcal{M}} 1(r_{im_ik} > 0)$$

and

$$rf_\nu = 1/(|\mathcal{R}^\nu|n) \sum_{k\in\mathcal{R}^\nu} \sum_{i\in V} 1/|\mathcal{M}_i| \sum_{m_i\in\mathcal{M}} 1(r_{im_ik} > 0).$$

Then, for all activity modes we fix the number of units of requested resources which will be used or consumed for the processing of the activity in the corresponding mode. In contrast to the approach by Kolisch et al. (1995), for a given activity $i$ and a given resource $k$, the resource requirements may vary with modes $m_i \in \mathcal{M}_i$, if the respective option (mode–varying resource requirements) has been selected. In addition, we do not avoid the generation of so–called *inefficient modes* (that is, modes $m_i$ which are dominated w.r.t. resource requirements and duration) since for the case of general temporal constraints these modes may constitute the unique optimal assignment (recall that the temporal constraints depend on the execution modes of both activities). The parameters used for the generation of resource requirements are listed in Table 4. The corresponding algorithm is depicted in Fig. 5.

Table 4: Parameters resource requirements

| Symbol | Parameter |
|---|---|
| $k_\rho^{min}, k_\rho^{max}$ | min. and max. number of renewable resources |
| $k_\nu^{min}, k_\nu^{max}$ | min. and max. number of nonrenewable resources |
| $q_\rho^{min}, q_\rho^{max}$ | min. and max. request of renewable resources |
| $q_\nu^{min}, q_\nu^{max}$ | min. and max. request of nonrenewable resources |
| $rf_\rho^{min}, rf_\rho^{max}$ | min. and max. resource factor of renewable resources |
| $rf_\nu^{min}, rf_\nu^{max}$ | min. and max. resource factor of nonrenewable resources |
| $u_\rho^{min}, u_\rho^{max}$ | min. and max. usage of renewable resources |
| $u_\nu^{min}, u_\nu^{max}$ | min. and max. consumption of nonrenewable resources |

Determine the number of renewable resources $k_\rho \in \{k_\rho^{min}, \dots, k_\rho^{max}\}$ and set $\mathcal{R}^\rho := \{1, \dots, k_\rho\}$.
Determine the number of nonrenewable resources $k_\nu \in \{k_\nu^{min}, \dots, k_\nu^{max}\}$ and set $\mathcal{R}^\nu := \{1, \dots, k_\nu\}$.
**FOR** $i \in V$ **DO**
  **FOR** $m_i \in \mathcal{M}_i$ **DO**
    Randomly determine $q_\rho^{min}$ renewable and $q_\nu^{min}$ nonrenewable resource requests of mode $m_i$.
Determine resource factor of ren. resources $rf_\rho \in [rf_\rho^{min}, rf_\rho^{max}]$.
**WHILE** $rf_\rho$ has not been attained
  Select randomly an activity–mode–resource combination $(i, m_i, k)$ with $k \in \mathcal{R}^\rho$ corresponding to a request of $k$ by $m_i$. Observe limitations w.r.t. maximum number $q_\rho^{max}$ of requested resources.
Determine resource factor of nonren. resources $rf_\nu \in [rf_\nu^{min}, rf_\nu^{max}]$.
**WHILE** $rf_\nu$ has not been attained
  Select randomly an activity–mode–resource combination $(i, m_i, k)$ with $k \in \mathcal{R}^\nu$ corresponding to a request of $k$ by $m_i$. Observe limitations w.r.t. maximum number $q_\nu^{max}$ of requested resources.
**FOR** requests of resources $k \in \mathcal{R}^\rho$ by modes $m_i$ of activities $i$ **DO**
  Determine resource usage $r_{im_ik} \in \{u_\rho^{min}, \dots, u_\rho^{max}\}$.
**FOR** requests of resources $k \in \mathcal{R}^\nu$ by modes $m_i$ of activities $i$ **DO**
  Determine resource consumption $r_{im_ik} \in \{u_\nu^{min}, \dots, u_\nu^{max}\}$.

Figure 5: Generation of resource requirements

## 4.2 Resource Availability

The generation of resource capacities is performed on the basis of the resource strength parameter introduced by Kolisch et al. (1995). The resource strength measures the scarcity of the resource availability w.r.t. to the respective requirements. According to the resource strength the resource capacities are chosen between a lower bound and an upper bound on the required number of resource units. For the renewable resources $k \in \mathcal{R}^\rho$ the lower bound is given by $\underline{R}_k = \max_{i \in V} \min_{m_i \in \mathcal{M}} r_{im_ik}$. The upper bound $\overline{R}_k$ is calculated as follows. For any activity $i$, the execution mode $m_i$ is set to a mode with maximum requirements $r_{im_ik}$. $\overline{R}_k$ then equals the peak $\max_{t \geq 0} r_k(ES, x, t)$ of the corresponding resource profile, where $ES$ denotes the earliest start schedule. For a nonrenewable resource $k \in \mathcal{R}^\nu$ the lower and the upper bound are given by $\underline{R}_k = \sum_{i \in V} \min_{m_i \in \mathcal{M}_i} r_{im_ik}$ and $\overline{R}_k = \sum_{i \in V} \max_{m_i \in \mathcal{M}_i} r_{im_ik}$, respectively. Notice that w.r.t. both the renewable and the nonrenewable resources the existence of a resource–feasible schedule is only secured if the capacities coincide with the upper bounds. The $[0, 1]$–normalized resource strength represents the ratio of $R_k - \underline{R}_k$ and $\overline{R}_k - \underline{R}_k$. That is, for a resource strength of 0 the capacities equal the lower bounds, for a resource strength of 1 the capacities correspond to the upper bounds. A full factorial design experiment performed by Schwindt (1996) for the single–mode case of the project duration problem shows that the resource strength strongly influences the hardness of this problem. Interestingly, the relationship between resource strength and problem hardness seems not to be monotone. On the one hand, for instances with tight resource constraints the number of alternatives for the resolution of resource conflicts is relatively small. On the other hand, the number of resource conflicts decreases with increasing resource strength.

After the generation of the resource capacities the unit costs and the thresholds for usage and consumption, respectively, have to determined. The corresponding control parameters are listed in Table 5. Fig. 6 summarizes the resource availability data generation.

Table 5: Parameters resource capacities

| Symbol | Parameter |
|---|---|
| $rs_\rho^{min}, rs_\rho^{max}$ | min. and max. resource strength of renewable resources |
| $rs_\nu^{min}, rs_\nu^{max}$ | min. and max. resource strength of nonrenewable resources |
| $c_\rho^{min}, c_\rho^{max}$ | min. and max. unit costs of renewable resources |
| $c_\nu^{min}, c_\nu^{max}$ | min. and max. unit costs of nonrenewable resources |
| $y_\rho^{min}, y_\rho^{max}$ | min. and max. threshold for usage of renewable resources |
| $y_\nu^{min}, y_\nu^{max}$ | min. and max. threshold for consumption of nonrenewable resources |

**FOR** resources $k \in \mathcal{R}^\rho$ **DO**
    Determine lower bound $\underline{R}_k$ and upper bound $\overline{R}_k$ on required resource capacity.
**FOR** resources $k \in \mathcal{R}^\nu$ **DO**
    Determine lower bound $\underline{R}_k$ and upper bound $\overline{R}_k$ on required resource capacity.
Determine resource strength of ren. resources $rs_\rho \in [rs_\rho^{min}, rs_\rho^{max}]$.
**FOR** resources $k \in \mathcal{R}^\rho$ **DO**
    Set capacity $R_k := \underline{R}_k + \lceil rs_\rho(\overline{R}_k - \underline{R}_k) \rceil$.
    Determine resource unit costs $c_k \in \{c_\rho^{min}, \ldots, c_\rho^{max}\}$.
    Determine usage threshold $y_k \in \{y_\rho^{min}, \ldots, y_\rho^{max}\}$.
Determine resource strength of nonren. resources $rs_\nu \in [rs_\nu^{min}, rs_\nu^{max}]$.
**FOR** resources $k \in \mathcal{R}^\nu$ **DO**
    Set capacity $R_k := \underline{R}_k + \lceil rs_\nu(\overline{R}_k - \underline{R}_k) \rceil$.
    Determine resource unit costs $c_k \in \{c_\nu^{min}, \ldots, c_\nu^{max}\}$.
    Determine consumption threshold $y_k \in \{y_\nu^{min}, \ldots, y_\nu^{max}\}$.

Figure 6: Generation of resource capacities, unit costs, and thresholds

# 5 Conclusions

ProGen/max, a generator for a general class of resource–constrained project scheduling problems subject to temporal constraints, has been presented. The generation of easy and hard instances can be controlled by a large number of network and resource measures. In addition to the algorithm proposed in this paper, a second procedure for the generation of cyclic network structures is available in ProGen/max constructing individual strong components of the project network which are then linked by forward arcs. Full factorial design experiments which have been performed in literature using ProGen/max indicate that both, network and resource parameters have a strong impact on the hardness of the feasibility and the optimization problems. In particular, the number of activities, the order strength of the network's acyclic skeleton, and the resource strength are decisive for the tractability of the scheduling problems. The generator and several benchmark testsets are available via world–wide–web.

## References


[1] Agrawal, M., Elmaghraby, S., and Herroelen, W. (1996), "DAGEN: A generator of testsets for project activity nets", *European Journal of Operational Research 90*, 376–382.

[2] Bartusch, M., Möhring, R.H., Radermacher, F.J. (1988), "Scheduling project networks with resource constraints and time windows", *Annals of Operations Research 16*, 201–240.

[3] Bein, W., Kamburowski, J., Stallmann, M. (1992), "Optimal reduction of two–terminal directed acyclic graphs", *SIAM Journal on Computing 21*, 1112–1129.

[4] Brucker, P., Drexl, A., Möhring, R.H., Neumann, K., Pesch, E. (1998), "Resource–constrained project scheduling: notation, classification, and methods", *European Journal of Operational Research*, to appear.

[5] Davis, E.W. (1975), "Project network summary measures constrained–resource scheduling", *AIIE Transactions 7*, 132–142.

[6] Demeulemeester, E.L., Dodin, B., Herroelen, W. (1993), "A random activity network generator", *Operations Research 41*, 972–980.

[7] De Reyck, B. (1995), "On the use of the restrictiveness as a measure of complexity for resource–constrained project scheduling", *Onderzoeksrapport Nr. 9535*, Department of Applied Economics, Katholieke Universiteit Leuven.

[8] Elmaghraby, E., Herroelen, W. (1980), "On the measurement of complexity in activity networks" *European Journal of Operational Research 5*, 223–234.

[9] Garey, M.R., Johnson, D.S. (1979), *Computers and Intractability*, Freeman and Co. San Francisco.

[10] Kaimann, R.A. (1974), "Coefficient of network complexity", *Management Science 21*, 172–177.

[11] Kolisch, R. (1995), *Project Scheduling under Resource Constraints — Efficient Heuristics for Several Problem Classes*, Physica, Heidelberg.

[12] Kolisch, R., Schwindt, C., Sprecher, A. (1998), "Benchmark instances for project scheduling problems", in: J. Węglarz (ed.), *Handbook on Recent Advances in Project Scheduling*, Kluwer, to appear.

[13] Kolisch, R., Sprecher, A., Drexl, A. (1995), "Characterization and generation of a general class of resource–constrained project scheduling problems", *Management Science 41*, 1693–1703.

[14] Kurtulus, I., Davis, E.W. (1982), "Multi–project scheduling: categorization of heuristic rules performance", *Management Science 28*, 161–172.

[15] Kurtulus, I.S., Narula, S. (1985), "Multi–project scheduling: analysis of project performance", *IIE Transactions 17*, 58–65.

[16] Neumann, K., Schwindt, C. (1997), "Activity–on–node networks with minimal and maximal time lags and their application to make–to–order production", *Operations Research Spektrum 19*, 205–217.

[17] Neumann, K., Zhan, J. (1994), "Heuristics for the minimum project–duration problem with minimal and maximal time lags under fixed resource constraints", *Journal of Intelligent Manufacturing 6*, 145–154.

[18] Patterson, J.H. (1976), "Project scheduling: the effects of problem structure on heuristic performance", *Naval Research Logistics Quarterly 23*, 95–123.

[19] Patterson, J.H. (1984), "A comparison of exact approaches for solving the multiple constrained resource, project scheduling problem", *Management Science 30*, 854–867.

[20] Schwindt, C. (1996), "Generation of resource–constrained project scheduling problem subject to minimal and maximum time lags", *Report WIOR–489*, University of Karlsruhe.

[21] Thesen, A. (1977), "Measures of the restrictiveness of project networks", *Networks 7*, 193–208.