Heuristics and Applications for Resource-Constrained Project Scheduling with Minimal and Maximal Time Lags

Klaus Neumann, University of Karlsruhe, Germany

1. Introduction

The paper deals with the problem of minimizing the duration of a project subject to time constraints (prescribed minimal and maximal time lags between the activities of the project) and resource constraints (limited availability of renewable resources). This problem is also called *RCPSP/max* (*R*esource *C*onstrained *P*roject *S*cheduling *P*roblem with minimal and *max*imal time lags).

Exact algorithms of the branch-and-bound type for RCPSP/max have been devised by Bartusch et al. (1988) and De Reyck & Herroelen (1996). Heuristic procedures for RCPSP/max represent either priority-rule methods or truncated branch-and-bound methods. Priority-rule methods for RCPSP/max have been proposed by Zhan (1994) and Neumann & Zhan (1995) and truncated branch-and-bound methods by Brinkmann & Neumann (1996).

In this paper, we give an overview of heuristic procedures for RCPSP/max, where we restrict ourselves to priority-rule methods, which generally outperform truncated branch-and-bound methods. We also report on some results from an experimental performance analysis.

2. Cyclic Activity-on-Node networks

Suppose that the project under consideration consists of n activities 1, ..., n, which are to be carried out without interruption. In addition, we introduce the fictitious activities 0 and n+1, which represent the beginning and completion of the project, respectively. Let $D_i \in \mathbb{Z}_+$ be the duration and $S_i \in \mathbb{Z}_+$ be the start time of activity i (i = 0, 1, ..., n+1) where $D_0 = D_{n+1} = 0$ and $S_0 := 0$. Then S_{n+1} represents the project duration. We assign the nodes 0, 1, ..., n+1 of a directed graph to the activities 0, 1, ..., n+1.

If there is a *minimal time lag* $T_{ij}^{min} \geq 0$ between the start of activities i and j, i.e. $S_j - S_i \geq T_{ij}^{min}$, we introduce an arc <i,j> with weight $b_{ij} := -T_{ij}^{min}$. If there is a *maximal time lag* $T_{ij}^{max} \geq 0$ between the start of activities i and j, i.e. $S_j - S_i \leq T_{ij}^{max}$, we introduce a *backward arc* <j,i> with weight $b_{ji} := -T_{ij}^{max}$.

The resulting weighted directed graph with node set V = {0, 1, ..., n+1}, arc set E, and weights b_{ij}, where $S_j - S_i \geq b_{ij}$ ($<i,j> \in E$), represents the *project network* N. In general, N contains cycles of nonpositive length. Note that arcs with negative weight may occur inside and outside of cycles. A detailed description of the construction and properties of project network N can be found in Neumann & Schwindt (1997).

The heuristic procedures for RCPSP/max require some *strict order* ρ in the node set V. A *cycle structure* in N is a strong component which contains at least two nodes. For $i, j \in V, i \neq j$, we then define $i \rho j$ exactly if (a) there is a path in N from i to j in case that i and j do not belong to one and the same cycle structure or (b) there is a path from i to j of positive length in case that i and j belong to one and the same cycle structure.

In literature, maximal time lags between activities are discussed extremely rarely although they occur in practice very often. Some examples are:

(i) Several activities have to be begun or have to be completed exactly at the same point in time.
(ii) Several activities have to be carried out one after another without any delay.
(iii) There are prescribed deadlines or time windows for certain activities.
(iv) Scheduling of make-to-order production where customer orders and precribed delivery dates have to be met and the overlapping of operations is permitted without interrupting any jobs (cf. Neumann & Schwindt, 1997).

3. Resource-Constrained Project Scheduling

Assume that the renewable resources 1, ..., K are required for carrying out the project. Let $R_\kappa > 0$ be the capacity of resource κ available and let $r_{i\kappa}$ be the amount of resource κ used by activity i ($\kappa = 1, ..., K$; $i \in V$) where $0 \leq r_{i\kappa} \leq R_\kappa$ and $r_{0,\kappa} = r_{n+1,\kappa} = 0$. For simplicity, R_κ and $r_{i\kappa}$ are assumed to be constant. Let

$$A(t) := \{i \in V | t - D_i < S_i \leq t\}$$

be the set of activities in progress at time t (or in time interval [t, t+1[, respectively), also called the *active set*. Moreover, let

$$r_\kappa(t) := \sum_{i \in A(t)} r_{i\kappa}$$

be the amount of resource κ used at time t. The RCPSP/max for project network N, also denoted by (N), can then be formulated as follows:

$$\begin{aligned}
&\text{Min.} && S_{n+1} \\
&\text{s.t.} && r_\kappa(t) \le R_\kappa \quad (\kappa = 1,\ldots,K;\ t = 0,1,\ldots,\bar{T}-1) && (1) \\
& && S_j - S_i \ge b_{ij} \quad (<i,j> \in E) \\
& && S_0 = 0 \\
& && S_i \in \mathbf{Z}_+ \quad (i \in V) && (2) \\
& && \text{Activity splitting is not allowed}
\end{aligned}$$

where

$$\bar{T} := \sum_{i \in V} \max\left(D_i, \max_{<i,j> \in E} b_{ij}\right)$$

is an upper bound on the project duration.

For each cycle structure C of N treated as a separate subproject and started at time 0, a project-scheduling problem (C) corresponding to (N) can be formulated. A sequence $(S_0, S_1, \ldots, S_{n+1})$ which satisfies (2) is termed a *schedule*. A schedule that satisfies (1) is called a *feasible schedule*. Analogously, the concept of a (*feasible*) *subschedule* for a subproject corresponding to a cycle structure C is defined. A sequence $(S_i)_{i \in V'}$, where $V' \subseteq V$ is not necessarily the node set of a cycle structure C is called a *partial schedule*.The next theorem (cf. Bartusch et al., 1988) is important for the heuristic procedures for solving (N):

Theorem 1. There is a feasible schedule for (N) exactly if, for each cycle structure C of N, there is a feasible subschedule for (C).

4. Overview of Heuristics for RCPSP/max

Three different sets of activities are used in a heuristic procedure for (approximately) solving problem (N). The activities completed up to the current schedule time form the *complete set* C. The activities in progress (i.e. started but not yet completed) constitute the *active set* A. $C \cup A$ is the set of the *scheduled activities*. The *decision set* D is the set of the unscheduled activities all of whose predecessors (with respect to the underlying strict order Ω) belong to C.

At the beginning of a heuristic, a temporal analysis for N is performed, that is, the earliest and latest start times ES_i and LS_i $(i \in V)$ are computed. The heuristic then constructs a sequence of partial schedules until a feasible schedule for (N) is attained. We distinguish between a serial and a parallel generation scheme for schedules.

A *serial generation scheme* consists of |V| stages. At each stage, one activity i is selected from D according to some priority rule and scheduled at the earliest time $\tau \geq ES_i$ such that the resource constraints are satisfied. If $\tau > LS_i$, a *backward scheduling process* is started which results in an appropriate right-shift of some activities already scheduled such that $\tau^{new} \leq LS_i^{new}$. For each successor j of i, ES_j has to be updated if $\tau > ES_i$, and LS_j has to be updated if $\tau < LS_i$. After that, activity i is deleted from D and inserted in C. The unscheduled activities all of whose predecessors now belong to C are added to D.

A *parallel generation scheme* consists of at most |V| stages. At each stage, a schedule time τ is determined, which equals the earliest completion time of any activity from A observing possible minimal time lags. Activities from A completed at time τ are deleted from A and inserted in C. Unscheduled activities all of whose predecessors now belong to C are again added to D. Moreover, the activities $i \in D$ are selected successively according to some priority rule and scheduled at time τ (i.e. removed from D and added to A) in case that the resource constraints are satisfied. The backward scheduling process and updating of the quantities ES_j and LS_j for the successors j of i are performed in the same way as in the serial procedure.

We note that if the project network is acyclic, the serial generation scheme constructs active schedules whereas the parallel procedure constructs non-delay schedules (cf. Kolisch 1995).

Two kinds of heuristic methods are proposed. The *sequential* or *direct method* processes the activities or respectively nodes of the project network one after another without considering the cycle structures separately. The *contraction method* uses a bottom-up technique exploiting Theorem 1. In Step 1, a feasible subschedule is determined for each cycle structure. In Step 2, each cycle structure is replaced by a single node and the resulting acyclic network is treated by the direct method. For the contraction method, the generation schemes (serial or parallel) used in the two steps may be different.

5. Experimental Performance Analysis

The heuristics have been tested using a test set of about 1500 project networks generated by the new network generator ProGen/max developed by Schwindt (1996). Each project has 100 activities and requires from 5 to 8 renewable resources. Different values of several network parameters (such as resource factor, resource strength, and restrictiveness) have been considered. The following main results have been obtained:

(a) The best priority rules are LST (Latest Start Time) and WCS (Worst Case Slack). The latter priority rule was proposed by Kolisch (1995).
(b) In general, the direct method provides better feasible schedules than the contraction method. However, the direct method requires much more computing time because a significantly larger number of backward scheduling steps are necessary.
(c) In the contraction method, the parallel or serial generation scheme can be used for evaluating the cycles structures. The resulting acyclic network should be evaluated using the serial scheme.

References

1. Bartusch, M., Möhring, R.H., Radermacher, F.J. (1988), Scheduling Project Networks with Resource Constraints and Time Windows. *Ann. of Op. Res. 16*, 201-240
2. Brinkmann, K., Neumann, K. (1996), Heuristic Procedures for Resource-Constrained Project Scheduling with Minimal and Maximal Time Lags: the Resource-Levelling and Minimum Project-Duration Problems. *J. of Decision Systems 5*, 129-155
3. De Reyck, B., Herroelen, W. (1996), A Branch-and-Bound Procedure for the Resource-Constrained Project Scheduling Problem with Generalized Precedence Constraints. *Research Report No. 9613*, Department of Applied Economics, K.U. Leuven
4. Kolisch, R. (1995), *Project Scheduling under Resource Constraints*. Physica, Heidelberg
5. Neumann, K., Schwindt, C. (1997), Activity-on-Node Networks with Minimal and Maximal Time Lags and Their Application to Make-to-Order Production. *OR Spektrum 19*, to appear
6. Neumann, K., Zhan, J. (1995), Heuristics for the Minimum Project-Duration Problem with Minimal and Maximal Time Lags under Fixed Resource Constraints. *J. of Intell. Manufact. 6*, 145-154
7. Schwindt, C. (1996), Generation of Resource-Constrained Project Scheduling Problems with Minimal and Maximal Time Lags. *Technical Report WIOR-489*, Institut für Wirtschaftstheorie und Operations Research, University of Karlsruhe

8. Zhan, J. (1994), Heuristics for Scheduling Resource-Constrained Projects in MPM Networks. *EJOR 76*, 192-205