The RCPSP with Time Dependent Resource Availabilities and its Applications

Peter Brucker, Silvia Heitmann, Christian Strotmann

January 2004

The Resource Constraint Project Scheduling Problem (RCPSP) is a complex discrete optimization problem. It covers a wide range of applications from project planning to machine scheduling (Brucker et. al [1999], Hartmann [1999], Neumann et al. [2003]). In Brucker & Knust [2000] the RCPSP with time-dependent resource availabilities has been introduced and used to model timetabling problems. Based on these concepts we have implemented local search methods and genetic algorithms to solve specific school- and university timetabling problems as well as job-shop scheduling problems with limited machine availabilities. The purpose of this paper is to document these results. This documentation is organized as follows. Firstly, we present a mathematical model for the RCPSP with time-dependent resource availabilities. A characteristic of this model is that it contains strong and weak resource constraints. Based on this model local search methods for solving the problem are introduced. These methods use certain list representations of schedules. For a given list a corresponding schedule can be constructed and evaluated. Representations and evaluations are discussed in Section 2. The last sections are devoted to applications. We show how the methods described in the first two sections can be applied to a school timetabling problem, a university course scheduling problem, and a job-shop problem with limited machine availabilities. Corresponding computational results are presented.

Mathematical model

The RCPSP with time-dependent resource availabilities can be formulated as follows. We are given a time interval $[0, T[$, a set of activities $1, \dots, n$ and a set of resources $1, \dots, r$. The availability of the resources is limited. For a resource k there are $R_k(t)$ units available in the interval $[t, t+1[$ for $t = 0, \dots, T-1$. We distinguish between two kinds of resources. Resources with $R_k(t) \in \{0, 1\}$ are called *disjunctive*, whereas other resources with $R_k(t) \in \{0, 1, 2, \dots\}$ are called *cumulative*.

Activity j must be processed for p_j time units without interruption and requires r_{jk} units of resource k for all $k = 1, \ldots, r$ during its processing. Additionally, arbitrary precedence constraints with possibly positive time-lags may be given. A precedence constraint is given by a relation $i \to j$, where $i \to j$ means that activity j cannot be started before activity i has finished. If additionally a time-lag $d_{ij} > 0$ between i and j is given, the earliest starting time of j is d_{ij} time units after the completion of i. If there is not time-lag we set $d_{ij} = 0$. The time-lag d_{ij} may depend on the completion time of activity i. It is convenient to add a dummy start activity 0 with $p_0 = 0$, $0 \to j$ and $d_{ij} = 0$ for all activities j.

Let S_j denote the starting time of activity j. Then $C_j := S_j + p_j$ is the completion time of activity j. The problem is to determine intervals $[S_j, S_j + p_j[$ for the processing of each activity j such that the following conditions hold:

(1) in each time period $[t, t+1[$ the total resource demand is less or equal to the availability $R_k(t)$ of each resource k, and

(2) all precedence constraints and time-lags are fulfilled.

A solution is called *feasible*, if conditions (1) and (2) hold and all activities are completed before time T.

The problem of finding a feasible solution is $\mathcal{NP}$-complete, as this problem generalizes the classical RCPSP. We want to apply various local search methods to find feasible solutions if they exist. Therefore we have to modify the problem in order to have a representation of infeasible solutions. We introduce two modifications of the problem. Both modifications are based on the idea, that we are able to obtain infeasible solutions but every infeasibility is penalized by a positive cost term in an objective function. By trying to minimize the objective function we try to get rid of these penalties and therefore of infeasibility.

As a first modification we add a overflow period $[T, T + L[$ after the time period $[0, T[$, where L is the length of the longest path in the activity-on-node networks in which the arcs (i, j) between activities i and j are evaluated by $p_i + d_{ij}$. During this overflow period the availabilities of all resources are assumed to be sufficiently large to supply all activities at the same time. Therefore activities which cannot be scheduled in $[0, T[$ can be scheduled in the overflow period. If an activity j is scheduled in the overflow period a penalty cost w_j occurs in the objective function.

As a second modification we introduce weak resources. Requirements of weak resources may be relaxed to zero, whereas the requirements of the other (hard) resources have to be fulfilled in any timetable. If the request for a weak resource is not respected, we penalize this by adding a positive cost term to the objective function. If for activity j the requirement of the weak resource k is relaxed to zero, a penalty cost w_{jk} occurs in the objective function.

Additionally, we may introduce some more penalties in the objective function in order to model a variety of different constraints. The problem of finding a feasible

solution with respect to the described weak and additional constraints leads to the problem of finding a solution with minimal objective value.

Representation and Evaluation of Solutions

Solutions are presented by priority lists which are compatible with the given precedence constraints, i.e. if $i \to j$ then activity i appears earlier in the list than j.

Given a list a corresponding schedule is constructed by scheduling activities one after the other in the list order. Each activity is scheduled as early as possible respecting all resource availabilities. If an activity cannot be planned in the given time period $[0, T[$ (infeasible solution), we try to relax the problem by deleting requirements of weak resources for a set of activities in order to obtain feasibility.

Before defining the algorithm we introduce some notations. By $\mathcal{K}_j$ we denote the set of all resources which are required for the processing of activity j. This set is divided into the hard resources $\mathcal{H}_j$ and the weak resources $\mathcal{W}_j$. Starting with a given priority list for an instance of the RCPSP with time dependent resource profiles we apply the following procedure *Earliest Start Schedule*.

Procedure **Earliest Start Schedule**

1. **WHILE** an activity exists which is not scheduled **DO**
   BEGIN
2. Let j be the first activity in the list which is not scheduled;
3. $t := \max_{i \to j}\{S_i + p_i + d_{ij}\}$;
4. **WHILE** a resource $k \in \mathcal{K}_j$ exists with $r_{jk} > R_k(\tau)$ for a $\tau \in [t, t + p_j[$ **DO**
   BEGIN
5. Compute minimal time $t_k > t$, such that j can be scheduled in $[t_k, t_k + p_j[$, if only resource k is considered and set $t := t_k$;
6. **IF** $(t + p_j \geq T)$ **AND** $r_{jk} \leq R_k(\tau)$ for all $k \in \mathcal{K}_j$ and $\tau \in [t, t + p_j[$ **AND** $\mathcal{W}_j \neq \emptyset$ **THEN**
   BEGIN
7. Delete a set of weak resources from $\mathcal{W}_i$ for a set of activities i;
8. $t := \max_{i \to j}\{S_i + p_i + d_{ij}\}$;
   END
   END
9. Schedule j in $[t, t + p_j[$ and update the resource profiles $R_k(t)$ in $[t, t + p_j[$ for all resources $k \in K_j$
   END

For the dummy activity 0 we set $S_0 = 0$. Because $0 \to j$ for all activities j we always have $t \geq 0$ in steps 3 and 8.

We applied local search methods like simulated annealing, tabu search, and a genetic algorithm with the set of all lists compatible with the precedence constraints as

search space. Neighborhoods have been defined by interchange and shift operators. For crossover we used one-point crossover, order crossover, and linear crossover which are known to be compatible with the precedence constraints (Hartmann [1999]).

Applications

The method has been applied to the following problems:

- A school timetabling problem with data from a high-school in Osnabrueck.
- A university timetabling problem for the faculty of mathematics and informatics at the University of Osnabrueck.
- A job-shop problem with restricted machine availabilities.

A full description of these problems, implementation results of the corresponding solvers, and of the computational experiments with these solvers can be found in the full version of the paper.

References

1. P. Brucker, A. Drexl, R. Moehring, K. Neumann and E. Pesch. Resource-constrained project scheduling: Notation, classification, models and methods. European Journal of Operational Research 112 (1999), 3-14.
2. P. Brucker and S. Knust. Resource-constrained project scheduling and timetabling. In: Edmund Burke and Wilhelm Erben (editors), The Practice and Theory of Automated Timetabling III: Selected Papers from the 3rd International Conference on the Practice and Theory of Automated Timetabling, Konstanz, Germany, August 16th-18th 2000, Springer Lecture Notes in Computer Science Series 2001, Vol 2079
3. S. Hartmann. Project scheduling under limited resources. Lecture Notes in Economics and Mathematical Systems 478, Springer, Berlin, 1999.
4. K. Neumann, C. Schwindt, and J. Zimmermann. Project scheduling with time windows and scarce resources. Second Edition. Springer, Berlin, 2003.