17th International Conference on the Application of Computer
Science and Mathematics in Architecture and Civil Engineering
K. Gürlebeck and C. Könke (eds.)
Weimar, Germany, 12–14 July 2006

TREE-BASED METHODS FOR RESOURCE INVESTMENT AND RESOURCE LEVELLING PROBLEMS

J. Zimmermann* **and J.-H. Bartels**

**Clausthal University of Technology*
Julius-Albert-Straße 2, 38678 Clausthal-Zellerfeld, Germany
E-mail: juergen.zimmermann@tu-clausthal.de

Keywords: Project Scheduling, Resource Investment Problem, Resource Levelling Problem, Tree Based Enumeration Scheme.

Abstract. *The execution of project activities generally requires the use of (renewable) resources like machines, equipment or manpower. The resource allocation problem consists in assigning time intervals to the execution of the project activities while taking into account temporal constraints between activities emanating from technological or organizational requirements and costs incurred by the resource allocation. If the total procurement cost of the different renewable resources has to be minimized, we speak of a resource investment problem. If the cost depends on the smoothness of the resource utilization over time, the underlying problem is called a resource levelling problem.*

In this paper we consider a new tree-based enumeration method for solving resource investment and resource levelling problems exploiting some fundamental properties of spanning trees. The enumeration scheme is embedded in a branch-and-bound procedure using a workload-based lower bound and a depth first search. Preliminary computational results show that the proposed procedure is promising for instances with up to 30 activities.

1 INTRODUCTION

We consider project scheduling problems with schedule-dependent time windows. The project under consideration is given by an activity-on-node network N with activity set $V := \{0, 1, \ldots, n, n+1\}$, arc set $E \subset V \times V$, and arc weights δ_{ij}. Activities 0 and $n+1$ represent the beginning and completion, respectively, of the underlying project. Let $p_i \in \mathbb{Z}_{\geq 0}$ be the duration of activity $i \in V$, which is assumed to be carried out without interruption. Moreover, let $S_i \geq 0$ be the start time of activity $i \in V$ and $\overline{d} \in \mathbb{Z}_{>0}$ be a prescribed maximum project duration. Given $S_0 := 0$ (i.e., the project always begins at time zero), $S_{n+1} \leq \overline{d}$ represents the project duration. A vector $S = (S_0, S_1, \ldots, S_{n+1})$ with $S_i \geq 0$ $(i \in V)$ and $S_0 = 0$ is called a *schedule*.

Temporal constraints between the activities are specified by so-called minimum and maximum time lags. For each minimum time lag, claiming that an activity *j* has to start at least d_{ij}^{min} time units after the start of activity *i*, there exists a forward arc $\langle i, j \rangle$ with weight $\delta_{ij} := d_{ij}^{min}$ between nodes *i* and *j* in project network N. Likewise, for each maximum time lag, claiming that *j* starts at most d_{ij}^{max} time units after the start of *i*, N contains a backward arc $\langle j, i \rangle$ with weight $\delta_{ji} := -d_{ij}^{max}$. Minimum as well as maximum time lags lead to restrictions $S_j - S_i \geq \delta_{ij}$. To ensure that the project is terminated after $\overline{d}$ time units at the latest, we introduce a maximum time lag $d_{0,n+1}^{max} := \overline{d}$. The set of schedules satisfying the temporal constraints $S_j - S_i \geq \delta_{ij}$ for all arcs $\langle i, j \rangle \in E$, given by the underlying minimum and maximum time lags, is denoted by $\mathcal{S}_T$ and called the feasible region of the underlying network.

Assume that a set $\mathcal{R}$ of *renewable resources* (e.g., machines, manpower, or equipment) are required for carrying out the activities of the underlying project. Resource types different from renewable ones are not considered in this work. Let $R_k \in \mathbb{Z}_{\geq 0}$ be the capacity of renewable resource k available and $r_{ik} \in \{0, 1, \ldots, R_k\}$ be the amount of resource k used by activity i. Given a schedule $S = (S_i)_{i \in V}$,

$$\mathcal{A}(S, t) := \{i \in V \mid S_i \leq t < S_i + p_i\}$$

is the set of activities in progress, also called the *active set*, at time $t \in [0, \overline{d}]$.

$$r_k(S, t) := \sum_{i \in \mathcal{A}(S,t)} r_{ik}$$

is the amount of resource $k \in \mathcal{R}$ used at time $t \in [0, \overline{d}]$ given schedule S. Figure 1 depicts an activity-on-node network of a project with five real activities and a single resource.

2 RESOURCE INVESTMENT AND RESOURCE LEVELLING PROBLEMS

In practice, the overall result of a project often depends on how the set of scarce resources $k \in R$ necessary to carry out the activities is utilized. If resources have to be

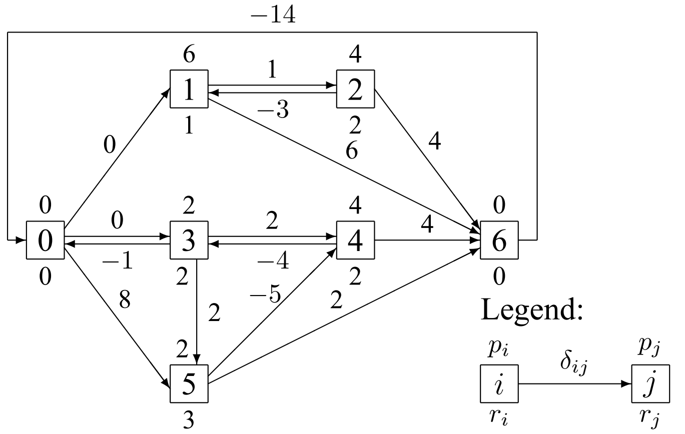

Figure 1: AoN project network with a single resource

purchased (e.g., expensive machinery) and we want to minimize the total procurement cost, we obtain a *resource investment problem*. Let $c_k \geq 0$ be the procurement cost per unit of resource $k \in R$. Then we minimize the objective function

$$f(S) := \sum_{k \in R} c_k \max_{0 \leq t \leq \overline{d}} r_k(S, t) \tag{RI}$$

Often some measure of the variation of resource utilization is to be minimized if the resources $k \in R$ should be used evenly over time. In this case, we consider a *resource levelling problem*. Let $c_k \geq 0$ be a cost incurred per utilized unit of resource $k \in R$ and per time unit, then we obtain the objective function

$$f(S) := \sum_{k \in R} c_k \int_0^{\overline{d}} r_k^2(S, t)\, dt \tag{RL}$$

which represents the total squared utilization cost given schedule S. Since workload $\int_0^{\overline{d}} r_k(S, t)\, dt$ does not depend on schedule S, objective function RL equals the weighted sum of the variances of the loading profiles $r_k(S, \cdot)$ plus a constant.

Our *project scheduling problem* now consists of minimizing objective function RI or RL over the set of all time-feasible schedules, i.e.

$$\left.\begin{array}{ll} \text{Minimize} & f(S) \\ \text{subject to} & S_j - S_i \geq \delta_{ij} \quad \forall \langle i, j \rangle \in E \\ & S_0 = 0 \end{array}\right\} \tag{P}$$

3 STRUCTURAL PROPERTIES

Let $O \subset V \times V$ be a *strict order* (i.e., an asymmetric and transitive binary relation) in activity set V. Then

$$\mathcal{S}_T(O) := \{S \in \mathcal{S}_T \mid S_j \geq S_i + p_i \text{ for all } (i, j) \in O\}$$

is called the *order polytope* of O. As a matter of course, for the empty strict order $O = \emptyset$ we have $\mathcal{S}_T(\emptyset) = \mathcal{S}_T$, and if $\mathcal{O}$ is the (finite) set of all inclusion-minimal feasible strict orders in activity set V, we have $\mathcal{S}_T = \bigcup_{O \in \mathcal{O}} \mathcal{S}_T(O)$. For problem P with objective function RI (RL), there is always a minimal point (or extreme point, respectively) S of some order polytope $\mathcal{S}_T(O)$ which is a minimizer of f on $\mathcal{S}_T \neq \emptyset$ (cf. Neumann/Schwindt/Zimmermann [3], Subsections 3.3.7 and 3.3.8).

Moreover, consider Network $N(O)$, which results from the underlying project network N by adding arc $\langle i, j \rangle$ with weight p_i for each pair $(i, j) \in O$. If N already contains an arc $\langle i, j \rangle$, its weight δ_{ij} is replaced by $\max(\delta_{ij}, p_i)$. Then each minimal point (extreme point) S of some order polytope $\mathcal{S}_T(O)$ corresponds to a spanning outtree of $N(O)$ with root 0 (to a spanning tree), where the $n+1$ arcs of such a spanning tree T, say arcs $\langle i, j \rangle \in E^T$ with weights δ^T_{ij}, correspond to $n+1$ linearly independent binding temporal constraints $S_j - S_i = \delta^T_{ij}$ ($\langle i, j \rangle \in E^T$). Together with $S_0 = 0$, this linear system of equations has a unique solution, namely the vertex in question.

4 SOLUTION PROCEDURE

An optimal solution to problem P with objective function RI or RL can now be determined as follows. We consecutively fix start times of activities such that, step by step, temporal constraints $S_j - S_i \geq \delta^T_{ij}$ become binding. For objective function RI we have to ensure that the corresponding arcs constitute an outtree rooted at node 0 and for RL an arbitrary spanning tree of some network $N(O)$. Using the bridge concept of Gabow/Myers [2], it is possible to construct non-redundant spanning trees. Moreover, to ensure that one and the same schedule (minimal point or extreme pint of some order polytope) is only constructed once, we use some redundance rules and the concept of t-minimal spanning trees proposed by Nübel [4].

The sketched enumeration scheme is embedded in a branch-and-bound procedure. Given two partial schedules $S^{\mathcal{C}} = (S_i)_{i \in \mathcal{C}}$ with $\mathcal{C} \subseteq V$, $r_k(S^{\mathcal{C}^1}, t) \leq r_k(S^{\mathcal{C}^2}, t)$ for all $k \in \mathcal{R}$ and all $t \in [0, \overline{d}]$ implies $f(S^{\mathcal{C}^1}) \leq f(S^{\mathcal{C}^2})$ for both objective functions RI and RL. Thus, $LB0(S^{\mathcal{C}}) := f(S^{\mathcal{C}})$ is a lower bound on objective function value $f(S)$ for each schedule $S \in \mathcal{S}_T$ which can be obtained by extending current partial schedule $S^{\mathcal{C}}$. For objective functions RI and RL, the least increase in the objective function value $f(S^{\mathcal{C}})$ is obtained if we "schedule" some additional workload represented by interruptible subactivities with resource demand of one at points in time t^* where $r_k(S^{\mathcal{C}}, t^*)$ is minimum.

Let $\overline{\mathcal{C}} := V \setminus \mathcal{C}$ be the set of activities which have not been considered thus far and let $w_k(\overline{\mathcal{C}}) := \sum\limits_{j \in \overline{\mathcal{C}}} r_{jk}\, p_j$ be the workload of those activities on resource $k \in \mathcal{R}$. We take workload $w_k(\overline{\mathcal{C}})$ into account by splitting up the activities $j \in \overline{\mathcal{C}}$ into interruptible subactivities with resource demand one. For each $k \in \mathcal{R}$, these subactivities are then scheduled at points in time $t^* \in \{0, 1, \ldots, \overline{d}\}$ for which $r_k(S^{\mathcal{C}}, t^*)$ is minimum. For

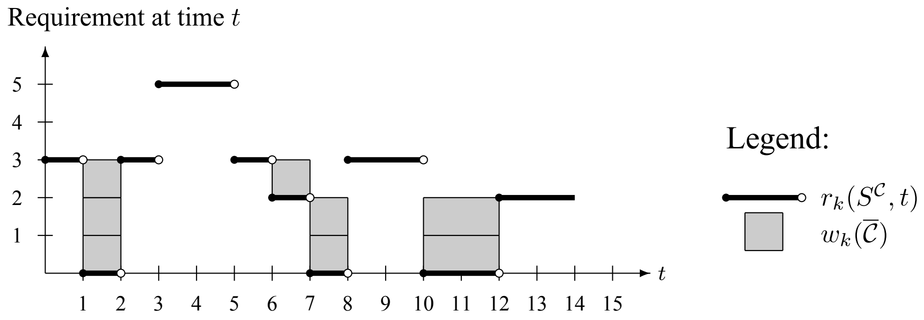

Figure 2: Resource profile $r_k(S^{\mathcal{C}}, \cdot)$ and workload $w_k(\overline{\mathcal{C}})$

illustration, we may think of the "ravines" of resource profiles $r_k(S^{\mathcal{C}}, \cdot)$, $k \in \mathcal{R}$, as containers which are filled "unit by unit" with the workload of the subactivities, see Figure 2. The resulting resource profiles are denoted by $r_k^W(S^{\mathcal{C}}, \cdot)$. $LBW(S^{\mathcal{C}}) := f(r_k^W(S^{\mathcal{C}}, \cdot)$ then represents a lower bound on $f(S)$ for each time-feasible extension S of $S^{\mathcal{C}}$ for both objective functions RI and RL.

Let d_{ij} be the longest path length from node i to node j in project network N which can be determined by a label correcting algorithm (cf. Ahuja/Magnanti/Orlin [1], Sect. 5.4). Then a basic version of our proposed branch-and-bound procedure reads as follows.

```
**Branch-and-bound procedure**
Set $S_0 := 0$ and $\mathcal{C} := \{0\}$
**For** all $j \in V \setminus \mathcal{C}$ **do**
    $ES_j := d_{0j}$ and $LS_j := -d_{j0}$
    **If** $ES_j = LS_j$ **then** set $S_j := ES_j$ and $\mathcal{C} := \mathcal{C} \cup \{j\}$
**End** $(*$ for $*)$
Initialize stack $\Omega := \{(\mathcal{C}, S^{\mathcal{C}})\}$
Initialize best solution $S^* := (ES_j)_{j \in V}$ and upper bound $UB := f(S^*)$
**Repeat**
    Pop pair $(\mathcal{C}, S^{\mathcal{C}})$ off stack $\Omega$
    **If** $\mathcal{C} = V$ **then** $S^* := S^{\mathcal{C}}$ and $UB := f(S^*)$
    **Else if** $LBW(S^{\mathcal{C}}) < UB$ **then**
        Initialize list $\Lambda := \emptyset$
        Determine all feasible extensions $(\mathcal{C}', S^{\mathcal{C}'})$ of current partial schedule $(\mathcal{C}, S^{\mathcal{C}})$
        using some redundance rules and add them to list $\Lambda$
        Push partial schedules $(\mathcal{C}', S^{\mathcal{C}'})$ from list $\Lambda$ to stack $\Omega$ in order of nonincreasing
        values $LB0(S^{\mathcal{C}'})$
    **End** $(*$ if $*)$
**Until** $\Omega = \emptyset$
**Return** $S^*$
```

Preliminary computational results on the "ubo" test instances generated with ProGen/max (cf. Neumann/Schwindt/Zimmermann [3], Subsections 2.8.1 and 2.8.2), show that the proposed procedure is promising for instances with up to 30 activities.

REFERENCES

[1] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin, *Network Flows*. Prentice Hall, Englewood Cliffs, 1993.

[2] H.N. Gabow and E.W. Myers, Finding all spanning trees of directed and undirected graphs. *Siam Journal on Computing 7*, 280–287, 1978.

[3] K. Neumann, C. Schwindt, and J. Zimmermann, *Project Scheduling with Time Windows and Scarce Resources*. Springer, Berlin, 2003.

[4] H. Nübel, The resource renting problem subject to temporal constraints. *OR Spektrum 23*, 359–381, 2001.

[5] J. Zimmermann, C. Stark, and J. Rieck, *Projektplanung: Modelle, Methoden, Management*. Springer, Berlin, 2006.