Kimms, Alf

Lagrangean relaxation for scheduling projects under resource constraints to maximize the net present value

Manuskripte aus den Instituten für Betriebswirtschaftslehre der Universität Kiel, No. 504

Provided in Cooperation with:
Christian-Albrechts-University of Kiel, Institute of Business Administration

Suggested Citation: Kimms, Alf (1999) : Lagrangean relaxation for scheduling projects under resource constraints to maximize the net present value, Manuskripte aus den Instituten für Betriebswirtschaftslehre der Universität Kiel, No. 504, Universität Kiel, Institut für Betriebswirtschaftslehre, Kiel

This Version is available at:
https://hdl.handle.net/10419/147592

Manuskripte
aus den
Instituten für Betriebswirtschaftslehre
der Universität Kiel

No. 504

Lagrangean Relaxation for Scheduling Projects under Resource Constraints to Maximize the Net Present Value

A. Kimms

Abstract

Resource–constrained project scheduling under a net present value objective attracts growing interest. Because this is an $\mathcal{NP}$–hard problem, it is unlikely that optimum solutions can be computed for large instances. Thus, heuristics have become a popular research field. Up to now, however, tight upper bounds have not been proposed. Therefore, most researchers evaluate their heuristics on the basis of a best known lower bound, but it is unclear how good the performance really is. With this contribution we close this gap and derive tight upper bounds on the basis of a Lagrangean relaxation of the resource constraints. We also use this approach as a basis for a heuristic and show that our heuristic as well as the cash flow weight heuristic proposed by Baroum and Patterson yield solutions very close to the optimum result. Furthermore, we discuss the proper choice of a test–bed, and, by reviewing the literature, emphasize that discount rates must be carefully chosen to give realistic instances.

Keywords: Resource–constrained project scheduling, net present value, Lagrangean relaxation, heuristics

1 Introduction

As a matter of fact, one of the key success factors in managing projects is scheduling. By having applications in diverse industries (see [22]), this area is known to be a challenging field of research (see, e.g., [5] for a recent survey of different research activities). While in terms of the number of publications makespan minimization seems to be the primary focus, financial objectives have become popular as well. Among them, maximizing the net present value of a project has attracted special attention which is plausible, because the net present value is the dominant selection criteria.

To be more specific about what is considered here, we assume that a project consists of a set $V = \{0, 1, \ldots, n, n+1\}$ of activities. For each activity j, a processing time $p_j \in \mathbb{N}_0$ is given. Preemption is not allowed, i.e. once an activity is started, it must run p_j time periods until it is completed. Associated with each activity j is a cash flow $c_j \in \mathbb{R}$. Without loss of generality, this cash flow occurs at the end of the particular activity. Note, if c_j is positive, we have a cash inflow, but if c_j is negative, we have a cash outflow. So, roughly speaking, we like to schedule the activities in such a way that cash inflows occur early and cash outflows occur late in time. Since, getting one dollar today is better than getting one dollar in the future (the reverse is true for paying a dollar), cash flows are discounted with a discount rate of $\alpha \in \mathbb{R}_{\geq 0}$ per time period.

The activities must be scheduled with respect to precedence constraints among them. Let $E \subset V \times V$ be the set of precedence constraints, i.e. $(i, j) \in E$ means that activity j must not be started before activity i is completed. Furthermore, we assume that the activity–on–node network $G = (V, E)$ contains no cycles. Without loss of generality, we can assume that 0 is the unique source and $n+1$ is the unique sink in the network. Given the precedence constraints, one can easily compute, for each activity j, the set of immediate predecessors $\mathcal{P}_j$ and the set of all predecessors $\bar{\mathcal{P}}_j$ using the definition

$$\bar{\mathcal{P}}_j = \mathcal{P}_j \cup \bigcup_{i \in \mathcal{P}_j} \bar{\mathcal{P}}_i \text{ with } \mathcal{P}_j = \{i \in V \mid (i, j) \in E\}.$$

Analogously, the set of immediate successors $\mathcal{S}_j$ of activity j is defined by $\mathcal{S}_j = \{i \in V \mid j \in \mathcal{P}_i\}$. It is now easy to compute the earliest completion time EC_j for each activity j by $EC_0 = p_0$ and $EC_j = \max\{EC_i + p_j \mid i \in \mathcal{P}_j\}$ for $j \geq 1$. In addition to that, we assume that a deadline $T \in \mathbb{N}$ must be met, i.e. activities must not be completed later than time T. Of course, $T \geq EC_{n+1}$ must hold. For each activity, the latest completion time LC_j^T for meeting the deadline T can then be determined by $LC_{n+1}^T = T$ and $LC_j^T = \min\{LC_i^T - p_i \mid i \in \mathcal{S}_j\}$ for $j \leq n$. In the sequel we will write LC_j instead of LC_j^T whenever it is clear from the context to which deadline T we refer to.

Finally, resource constraints must be respected, too. We consider a set K of so–called renewable resources, where $R_{kt} \in \mathbb{N}_0$ units of resource k are available in period t. Each activity j requires $r_{jk} \in \mathbb{N}_0$ units of resource k in each period it is processed. The activities must be scheduled in such a way that the sum of resource requirements does not exceed the resource availability in any period and for any resource.

It is worth to highlight that all data, especially the cash flows, are assumed to be deterministic. A straightforward binary linear programming model can now be formulated using decision variables x_{jt}. Such a variable is one, if activity j is completed at time t and zero, otherwise.

$$\max \sum_{j \in V} \sum_{t=EC_j}^{LC_j} \frac{c_j}{(1+\alpha)^t} x_{jt} \tag{1}$$

s.t.

$$\sum_{t=EC_j}^{LC_j} x_{jt} = 1 \qquad j \in V \tag{2}$$

$$\sum_{t=EC_j}^{LC_j} t \cdot x_{jt} - \sum_{t=EC_i}^{LC_i} t \cdot x_{it} \geq p_j \qquad (i,j) \in E \tag{3}$$

$$\sum_{j \in V} \sum_{\tau=\max\{t,EC_j\}}^{\min\{t+p_j-1,LC_j\}} r_{jk} x_{j\tau} \leq R_{kt} \qquad k \in K; t \in \{1,\ldots,T\} \tag{4}$$

$$x_{jt} \in \{0,1\} \qquad j \in V; t \in \{EC_j,\ldots,LC_j\} \tag{5}$$

The objective (1) is to maximize the net present value. Each activity must be scheduled (2). A feasible schedule must take the precedence constraints (3) as well as the resource constraints (4) into account. The decision variables are defined by (5). Apparently, the completion time C_j of activity j is determined by $\sum_{EC_j}^{LC_j} t \cdot x_{jt}$.

This problem is known to be an $\mathcal{NP}$–hard optimization problem which covers the makespan minimization problem as a special case ($\alpha = 1$, $c_{n+1} = 1$, and $c_j = 0$ for $j \leq n$) which is $\mathcal{NP}$–hard already [4]. Without resource constraints, the problem can be solved in polynomial time. To the best of our knowledge, A.H. Russell [33] was the first who considered this case. Later, Grinold [15] developed an efficient solution procedure. More recently, De Reyck and Herroelen [10] presented another optimum algorithm.

For the resource–constrained problem, optimum solution methods have been developed by Yang et al. [41], Icmeli and Erenguc [19], De Reyck and Herroelen [10], and Neumann and Zimmermann [28]. Baroum and Patterson [2] present an optimum algorithm which is confined to the case that all cash flows are non–negative. A bulk of publications is devoted to heuristics. See, e.g., Baroum and Patterson [1], Icmeli and Erenguc [18], Padman et

al. [31], Pinder and Marucheck [32], R.A. Russel [34], Smith–Daniels and Aquilano [36], Smith–Daniels et al. [37], and Yang et al. [42]. A more comprehensive survey on project scheduling with maximum net present value objective can be found in Brucker et al. [5], Herroelen et al. [17], Icmeli et al. [21], Kolisch and Padman [24], and Özdamar and Ulusoy [29].

Sophisticated upper bounds have not been developed so far. Icmeli and Erenguc [18] have used the LP–relaxation of the model formulation to evaluate their heuristics, but theses bounds were not tight. Padman and Smith–Daniels [30] as well as R.A. Russel [34] have relaxed the resource constraints. But, this bound was weak, too. As a result, it is common standard to evaluate heuristics on the basis of the best known lower bound. This paper is devoted to close this gap and discusses the computation of tight upper bounds.

The text is organized as follows: In Section 2 it is shown how upper bounds can efficiently be computed by a Lagrangean relaxation of the resource constraints. A heuristic is then presented in Section 3 which tries to derive a feasible solution from the Lagrangean relaxation result. Finally, a computational study is performed in Section 4. Concluding remarks finish the paper.

2 Lagrangean Relaxation of Resource Constraints

An upper bound for the optimum objective function value of model (1) thru (5) can be computed by considering a Lagrangean relaxation which is a well–known technique [14]. In our case, we consider a Lagrangean relaxation of the resource constraints (4). That is, for a given set of Lagrangean multipliers $\lambda_{kt} \geq 0$, to get an upper bound, we seek an optimum solution for the model:

$$\max \sum_{j \in V} \sum_{t=EC_j}^{LC_j} \frac{c_j}{(1+\alpha)^t} x_{jt} + \sum_{k \in K} \sum_{t=1}^{T} \lambda_{kt} \left(R_{kt} - \sum_{j \in V} \sum_{\tau=\max\{t,EC_j\}}^{\min\{t+p_j-1,LC_j\}} r_{jk} x_{j\tau} \right) \tag{6}$$

s.t. (2), (3), and (5)

By rearranging the terms in the objective function (6), we get

$$\max \sum_{j \in V} \sum_{t=EC_j}^{LC_j} w_{jt} x_{jt} \left(+ \sum_{k \in K} \sum_{t=1}^{T} \lambda_{kt} R_{kt} \right) \tag{7}$$

where

$$w_{jt} = \frac{c_j}{(1+\alpha)^t} - \sum_{\tau=t-p_j+1}^{t} \sum_{k \in K} \lambda_{k\tau} r_{jk}. \tag{8}$$

Since, in (7), the second term is a constant, we can ignore it for the purpose of optimization. Thus, given a set of Lagrangean multipliers $\lambda_{kt} \geq 0$, this is a maximum total weighted completion time objective. It is important to be able to solve this problem efficiently, because the basic working principle of the Lagrangean relaxation approach is to solve this problem repeatedly with different Lagrangean multipliers in each iteration.

It should be remarked that in the context of resource–constrained project scheduling, the Lagrangean relaxation of the resource constraints has previously been studied by Christofides et al. [7], who considered makespan minimization. They derived a problem

with an objective function similar to (7) and proposed a branch–and–bound procedure to solve it. Meanwhile, it is clear that this kind of problem can be solved in polynomial time, because Möhring et al. [27] showed how it can be transformed into a minimum cut problem. These ideas do also apply in our case, and we have implemented a maximum flow algorithm according to Cherkassky and Goldberg [6].

The remaining question to be discussed is how to choose the Lagrangean multipliers. As already mentioned above, the basic idea of the Lagrangean relaxation approach is to solve the problem repeatedly with a new set of Lagrangean multipliers in each iteration. As a result of each iteration, we get an upper bound for the maximum net present value, and the best (i.e. the lowest) bound obtained defines the result of the Lagrangean relaxation. Initially, we start with all multipliers being zero. Thus, iteration one actually gives the upper bound obtained by ignoring the resource constraints completely. Subgradient optimization [16] is then applied to update the Lagrangean multipliers after each iteration by

$$\lambda_{kt} = \max\left\{0, \lambda_{kt} + \delta \frac{(UB^* - LB^*)\Delta_{kt}}{\sum_{\kappa \in K} \sum_{\tau=1}^{T} \Delta_{\kappa\tau}^2}\right\} \tag{9}$$

where

$$\Delta_{kt} = \sum_{j \in V} \sum_{\tau=\max\{t, EC_j\}}^{\min\{t+p_j-1, LC_j\}} r_{jk} x_{j\tau} - R_{kt}, \tag{10}$$

UB^* is the best known upper bound, LB^* is the best known lower bound, and δ is some scaling factor. Initially, δ is chosen to be two, and it is reduced to its half whenever UB^* has not been improved within five consecutive iterations. At the beginning, $UB^* = \infty$, and, therefore, the value of UB^* is updated after iteration one for the first time. As we will discuss in the next section, we try to derive a feasible solution in each iteration, and, hence, the value of LB^* is updated as well. Initially, one could choose $LB^* = -\infty$, but we will return to this point when we report about the computational study.

To logic behind (9) is that, if some resource constraint is violated, the corresponding Δ_{kt} value is positive (see (10)), thus, λ_{kt} increases (see (9)), and, thus, the values w_{jt} decrease due to (8). Hence, there is a tendency that, in order to maximize (7), the next iteration yields a schedule that resolves the resource conflict.

3 Lagrangean Relaxation Based Heuristic

In each iteration of the Lagrangean relaxation procedure, the computed schedule contains some useful information which can be used to derive a feasible schedule for the resource–constrained problem. Hence, we describe a heuristic which tries to construct a feasible schedule on the basis of the result of the Lagrangean relaxation of the resource constraints.

Roughly speaking, we try to construct a feasible schedule by scheduling one activity after the other. If there is a choice which activity should be scheduled next, we use a permutation $\pi : V \to \{0, \ldots, n+1\}$ to make that decision where activities with a low value are favored. This permutation is defined by making use of the completion times C_j^{LR} determined with the Lagrangean relaxation procedure. Given the completion times, we also know the start times $S_j^{LR} = C_j^{LR} - p_j$ of that schedule. Formally, π is the unique function which fulfills the following four conditions for $i \neq j \in V$:

(i) $i \in \bar{\mathcal{P}}_j \Rightarrow \pi(i) < \pi(j)$

(ii) $S_i^{LR} < S_j^{LR} \Rightarrow \pi(i) < \pi(j)$

(iii) $S_i^{LR} = S_j^{LR} \wedge i||j \wedge \frac{c_i}{(1+\alpha)^{p_i}} > \frac{c_j}{(1+\alpha)^{p_j}} \Rightarrow \pi(i) < \pi(j)$

(iv) $S_i^{LR} = S_j^{LR} \wedge i||j \wedge \frac{c_i}{(1+\alpha)^{p_i}} = \frac{c_j}{(1+\alpha)^{p_j}} \wedge i < j \Rightarrow \pi(i) < \pi(j)$

We want to schedule an activity j only if all its preceding activities are scheduled already. This comes in via (i). If there is a choice to schedule the next activity, we want to select the one that starts earliest in the schedule computed with Lagrangean relaxation. This is stated by (ii). In case two activities start at the same time and there is no precedence relation among them, which is formally defined by

$$i||j \Leftrightarrow i \notin \bar{\mathcal{P}}_j \wedge j \notin \bar{\mathcal{P}}_i,$$

we schedule that activity first which, loosely speaking, incurs the highest cash flow. That is covered by (iii). Finally, if ties remain, we simply choose the lowest numbered activity as stated in (iv).

As an example, consider the network with $n = 6$ given in Figure 1 (boxes represent activities where the width of a box j illustrates p_j and the height illustrates r_{j1}), the parameter values in Table 1, and $|K| = 1$, $T = 16$, and $R_{kt} = 3$ for all t. For the sake of simplicity, we do not care about the concrete cash flow values c_j, but only about the sign of the cash flows as indicated in Figure 1, too. Suppose now that the Lagrangean relaxation procedure yielded the schedule depicted in Figure 2(a). The resulting function π can be seen in Table 2.

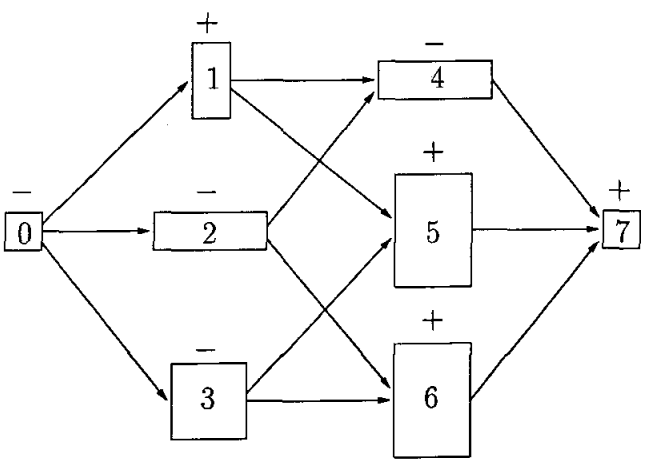

Figure 1: A Network with Eight Activities

The heuristic goes thru four phases in order to construct a suboptimal schedule for the resource–constrained problem. Phase one tries to construct a feasible solution by means of a so–called parallel scheduling scheme. If feasibility can be achieved, phases two to four try to improve that schedule without giving up feasibility. If phase one fails to find a feasible schedule, the procedure stops and further phases are not entered.

j	0	1	2	3	4	5	6	7
p_j	1	1	3	2	3	2	2	1
r_{j1}	1	2	1	2	1	3	3	1
$sign(c_j)$	−	+	−	−	−	+	+	+

Table 1: The Data of the Example

$\pi(j)$	0	1	2	3	4	5	6	7
j	0	1	3	5	2	6	4	7

Table 2: π Derived from the Schedule in Figure 2(a)

In more detail, phase one is specified in Table 3. It maintains a set U of activities which have not been scheduled so far. It begins at time $t = 0$ and determines the set D of activities which may be started at this time without violating the precedence constraints, the resource constraints, and the deadline. Formally, the precedence constraints are tested by a function

$$\Phi(j,t) = \begin{cases} 1, & \text{if } \forall i \in \mathcal{P}_j : C_i \leq t \\ 0, & \text{otherwise} \end{cases}$$

which yields 1 for activity j at time t, if all predecessors of j are scheduled in such a way that their completion times are not later than t. The resource constraints are tested by a function

$$\Psi(j,t) = \begin{cases} 1, & \text{if } \forall k \in K : \forall \tau \in \{t+1,\ldots,\min\{t+p_j,T\}\} : \\ & \qquad r_{jk} \leq R_{k\tau} - \sum_{\substack{i \in V \setminus \{j\} \\ \tau \leq C_i \leq \min\{\tau+p_i-1,T\}}} r_{ik} \\ 0, & \text{otherwise} \end{cases}$$

which gives 1 for activity j at time t, if the remaining resource availability in periods $\{t+1,\ldots,\min\{t+p_j,T\}\}$ suffices to schedule activity j. If more than one activity is contained in D, π is used to select the activity to be scheduled. At time t, activities are scheduled one after the other until no further activities can be started at t. Then, the procedure moves on to the very next period. This time spot becomes the new focus of attention. For the example, the result of phase one can be seen in Figure 2(b).

As an alternative to the parallel construction scheme one could also use a so–called serial construction scheme during phase one. The details for such scheme are given in Table 4. The underlying idea behind the serial construction scheme is to schedule an activity j only if all its preceding activities have already been scheduled. If ties exist, π is used to make a decision. Once an activity is selected, it is scheduled as early as possible where the earliest possible completion time C'_j is defined by $C'_j = p_j + \min\{t \mid \Phi(j,t) = 1 \wedge \Psi(j,t-p_j) = 1\}$. For the example, the result can be seen in Figure 2(b); it is identical to the result of the parallel construction scheme.

In case phase one succeeds in finding a feasible solution, phase two takes over to modify the schedule. Details are given in Table 5. The basic idea of phase two is to shift the activities to the right where the activities are tested for possible shifts in the order of decreasing π values. When shifting an activity, resource constraints are respected. Also,

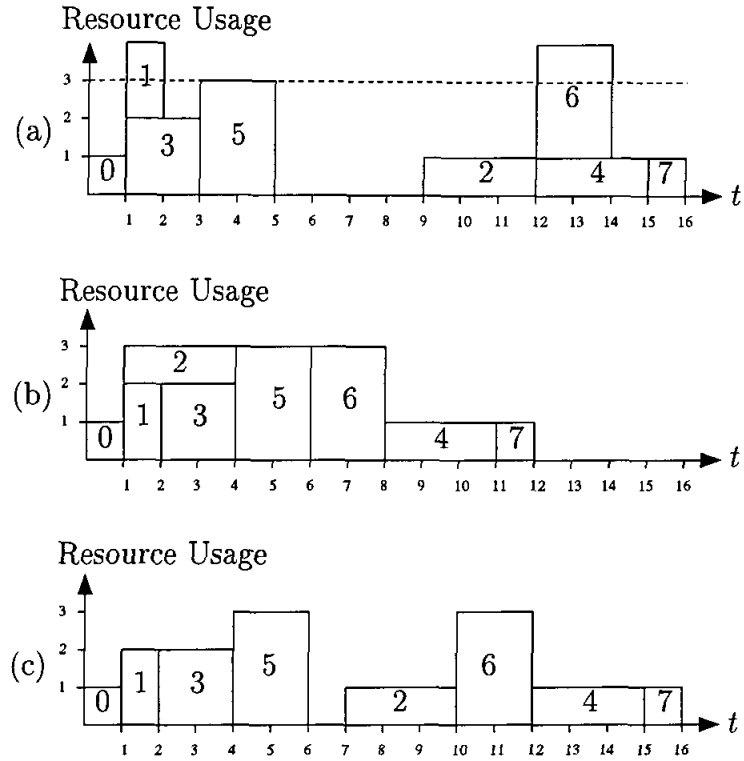

Figure 2: Heuristic Construction and Improvement Scheme

precedence constraints must not be violated, i.e. an activity j must not be completed later than $\tilde{C}_j = \min\{C_i - p_i \mid i \in \mathcal{S}_j\}$. Furthermore, we use the information contained in the schedule derived by the Lagrangean relaxation of the resource constraints and do not allow shifts beyond the completion time C_j^{LR}. The idea behind considering C_j^{LR} is that although it may be useful to shift activities with a positive cash flow to the right, we do not want to shift such activities too far. Our hope is that the schedule derived by Lagrangean relaxation contains some senseful information in this respect. Thus, $\hat{C}_j = \min\{\tilde{C}_j, \max\{C_j, C_j^{LR}\}\}$ is the latest possible completion time disregarding the resource constraints. Phase two is repeated until no further shifts are performed. Figure 2(c) shows the result of phase two for the example.

After termination of phase two, phase three performs additional right shifts. Table 6 provides the details. This phase improves the current schedule by shifting activities with a negative cash flow as far as possible to the right. Of course, resource constraints, precedence constraints as well as the deadline must be taken into account. Again, the activities are considered in the order of decreasing π values. Phase three is repeated until no further right shifts can be performed. For the example, no improvements are possible during phase three.

Finally, phase four is reached. See Table 7 for the details. Similar to phase three, phase four improves the current schedule by shifting activities. This time, activities with a positive cash flow are shifted as far as possible to the left. Besides the resource constraints and the earliest completion time, precedences have to be taken into account, too, thus $\bar{C}_j = \max\{C_i + p_j \mid i \in \mathcal{P}_j\}$ gives the earliest possible completion time of activity j respecting the schedule of preceding activities. For the example, no improvements are

$U = V$
$C_j = T + 1$ for all $j \in V$
$t = 0$
repeat
{ repeat
{ $D = \{j \in U \mid \Phi(j,t) = 1 \wedge \Psi(j,t) = 1 \wedge t + p_j \leq T\}$
if $(D \neq \emptyset)$
{ $j = \arg\min\{\pi(i) \mid i \in D\}$
$C_j = t + p_j$
$U = U \backslash \{j\}$
} } until $(D = \emptyset)$
$t = t + 1$
if $(t > T \wedge U \neq \emptyset)$ STOP
} until $(U = \emptyset)$

Table 3: The Parallel Construction Scheme (Phase 1)

possible during phase four. If at least one activity is shifted during phase four, the heuristic loops back to phase three to test for further improvements.

4 Computational Study

To test the performance of the proposed approaches, we have implemented them in GNU C on a Pentium II computer with 300 MHz and 64 MB RAM running a Linux operating system.

4.1 Test–Bed

We have used the ProGen instances from the project scheduling problem library (PSPLIB) [25] to create a test–bed. This library contains systematically generated instances for resource–constrained project scheduling. We have used the instances with $n \in \{30, 60, 90\}$. Each of theses instances has four resources. The instances have different network complexities $NC \in \{1.5, 1.8, 2.1\}$, different resource factors $RF \in \{0.25, 0.5, 0.75, 1.0\}$, and different resource strengths $RS \in \{0.2, 0.5, 0.7, 1.0\}$ — parameters which are known to have a strong impact on the performance of resource–constrained project scheduling procedures [26]. The network complexity reflects the average number of immediate successors of an activity, the resource factor is a measure of the average number of resources requested per activity divided by the total number of resources, and the resource strength measures the scarceness of the resources where a low resource strength indicates that the resource constraints are tight. For each parameter level combination, ten instances are provided which gives a total of $3 \cdot 3 \cdot 4 \cdot 4 \cdot 10 = 1440$ instances.

Cash flows are not specified in the library, so we followed Icmeli and Erenguc [19] and draw, for each instance, a different set of random numbers $c_j \in [-500; 1000]$ with uniform distribution.

$U = V \backslash \{0\}$
$C_0 = p_0$
$C_j = T + 1$ for all $j \in V \backslash \{0\}$
repeat
{ $D = \{j \in U \mid C'_j \leq T\}$
if $(D \neq \emptyset)$
{ $j = \arg\min\{\pi(i) \mid i \in D\}$
$C_j = C'_j$
$U = U \backslash \{j\}$
} } until $(D = \emptyset)$
if $(U \neq \emptyset)$ STOP

Table 4: The Serial Construction Scheme (Phase 1)

do
{ $flag = false$
$U = V$
repeat
{ $j = \arg\max\{\pi(i) \mid i \in U\}$
$t = \max\{\tau \in \{C_j, \ldots, \hat{C}_j\} \mid \Psi(j, \tau - p_j) = 1\}$
if $(t > C_j)$ $flag = true$
$C_j = t$
$U = U \backslash \{j\}$
} until $(U = \emptyset)$
} while $(flag = true)$

Table 5: Modification of the Schedule (Phase 2)

In order to specify a deadline, we wanted to make sure that each instance has a feasible solution. Hence, we applied a simple heuristic for minimizing the makespan. In our tests, we used a deterministic parallel priority rule procedure, i.e. a procedure that works similar to our phase one using a deadline that is not restrictive and a priority rule π that is defined on the basis of the latest start time $LS_j^{EC_{n+1}} = LC_j^{EC_{n+1}} - p_j$. Ties are broken favoring the lowest numbered activity. This way, we have computed a deadline T^{MinLS} for which a feasible solution exists and derived the deadline T used in our studies by $T = \theta \cdot T^{MinLS}$. In our tests, we have used $\theta \in \{1.0, 1.25, 1.5, 1.75, 2.0\}$. Moreover, the net present value of the initial schedule gives a lower bound which is used to initialize LB^* used for subgradient optimization.

To complete the specification of the instances, we need to define α, the discount rate per period. In practice, it is common to specify β, the discount rate per year, rather than the discount rate per period. Given that one period corresponds to $\frac{1}{T_0}$ years, the α value can easily be computed by

$$\alpha = \sqrt[T_0]{1 + \beta} - 1.$$

For instance, $T_0 = 52$ means that one period corresponds to one week, and $T_0 = 12$ means

do
{ $flag = false$
$U = \{j \in V \mid c_j < 0\}$
while $(U \neq \emptyset)$
{ $j = \arg\max\{\pi(i) \mid i \in U\}$
$t = \max\{\tau \in \{C_j, \ldots, \min\{\tilde{C}_j, T\}\} \mid \Psi(j, \tau - p_j) = 1\}$
if $(t > C_j)$ $flag = true$
$C_j = t$
$U = U \backslash \{j\}$
} } while $(flag = true)$

Table 6: Improvement of the Schedule (Phase 3)

that one period corresponds to one month. Unfortunately, it has become usual practice in the literature to specify α without making a definite statement about T_0. Table 8 provides some overview over what α has been used and what would be the β depending on which T_0 we assume (if β has been given and/or T_0 has been specified, we leave the remaining entries empty). The list of references in this table is not comprehensive, mainly because some authors do not report about their α values at all.

We like to point out that the discussion of α depending on T_0 is extremely crucial, and that specifying α without T_0 is meaningless. The problem setting treated here and in the references listed in Table 8 assumes deterministic data. Hence, a realistic value for β is a value that corresponds to the discount rate of risk–free assets on the capital market. Nowadays, real–life discount rates for risk–free assets are low, usually below 10%. These data are confirmed by a real–world case studies provided by Tavares [39, 40]. Thus, to make sure that the computational study does not use artificial data and that it makes some sense with respect to practice, we must use β values in that range, i.e. we must specify α and T_0 accordingly. By having a look at what is used in the literature so far (compare Table 8), we see that some authors use very unrealistic data. We also find that the understanding of what a period stands for is quite different. Some authors consider days while others must have meant quarter years. This, however, is not surprising, because the application area for project scheduling is wide. Nevertheless, if the discount rates indicate that months or even longer periods are meant, we must carefully check how long the investigated projects run. Especially when several hundreds of activities are considered, the makespan of the projects in the test–bed is likely to be several decades which would not make much sense either.

In our tests, we have used $T_0 = 52$ and $\beta \in \{5\%, 10\%\}$. In total, we thus have $1440 \cdot 5 \cdot 2 = 14400$ instances in our test–bed.

4.2 Results

To present the results, we have aggregated the data and do not show how the resource factor RF and the network complexity NC affect the performance. For each parameter level combination of the number of activities n, the resource strength RS, the deadline factor θ, and the discount rate β we thus have $4 \cdot 3 \cdot 10 = 120$ instances.

```
do
{   flag_phase3 = false
    do
    {   flag = false
        U = {j ∈ V | c_j > 0}
        while (U ≠ ∅)
        {   j = arg min{π(i) | i ∈ U}
            t = min{τ ∈ {max{C̄_j, EC_j}, ..., C_j} | Ψ(j, τ − p_j) = 1}
            if (t < C_j)
            {   flag = true
                flag_phase3 = true
            }
            C_j = t
            U = U\{j}
    }   } while (flag = true)
    if (flag_phase3 = true) Phase 3
} while (flag_phase3 = true)
```

Table 7: Improvement of the Schedule (Phase 4)

For studying the results, we have tested the Lagrangean relaxation based approach with a parallel construction scheme in phase one, *LRp* for short, and the Lagrangean relaxation based approach with a serial construction scheme in phase one, *LRs* for short. In either case 100 iterations were performed. To see, if our results are competitive, we have also implemented the so–called BIAS–CFW cash flow weight heuristic of Baroum and Patterson [1], *BP* for short, because it is a recent procedure and it gave amazing results as reported in [1]. This is an iterative procedure, too, and we performed 100 iterations as well.

Table 9 shows the number of instances for which a feasible solution could have been found. Recall that a feasible solution exists for every instance. Not surprisingly, we find that, if the deadline is not tight, i.e. $\theta \geq 1.25$, every instance is feasibly solved by all heuristics. The same is true, if resource constraints do not exist, i.e. $RS = 1.0$. For a tight deadline, i.e. $\theta = 1.00$, and scarce resources, i.e. $RS < 1.0$, we have instances for which no feasible solution can be found. As we see, *BP* is the best performer when the number of activities is small, i.e. $n = 30$. If the number of activities grows, *LRp* seems to be best, if resources are really scarce, i.e. $RS \leq 0.5$. For $RS = 0.7$, *BP* remains to be the top performer even for instances with many activities. *BP* shows weak results if resource constraints are very tight, i.e. $RS \leq 0.2$, especially when the number of activities is high. For $n = 90$ and $RS = 0.2$, for instance, only 34 out of 120 instances could have been feasibly solved with *BP*. The discount rate has almost no effect on the ability to find feasible solutions.

If we consider the run–time as a measure of performance (see Table 10 for CPU–seconds), it becomes apparent that *BP* is extremely fast even for larger instances. The computational burden of the Lagrangean relaxation based approach, *LR* for short, lies in solving the total weighted completion time subproblem. While the discount rate β has

α	$T_0 = 360$? β	$T_0 = 52$? β	$T_0 = 12$? β	$T_0 = 4$? β	References
0.00002	0.7%	0.1%	0.0%	0.0%	[37]
0.00006	2.2%	0.3%	0.1%	0.0%	[36]
—	4.2%	—	—	—	[12]
0.0008	33.4%	—	—	—	[35]
—	—	6.5%	—	—	[40]
0.00167	82.3%	—	—	—	[23], [35]
0.0018	—	9.8%	—	—	[39]
0.00333	231.0%	—	—	—	[35]
—		20.0%	—	—	[9]
0.008	1661.1%	—	—	—	[23]
0.01	—	—	12.7%	—	[15], [34]
0.01	3495.0%	67.8%	12.7%	4.1%	[18], [19], [20], [33], [42]
$0.\overline{01}$	3626.7%	68.6%	12.8%	4.1%	[28]
0.02	—	—	26.8%	—	[11], [38]
0.02	$1.2 \cdot 10^5$%	180.0%	26.8%	8.2%	[1], [3], [13], [18], [20]
0.03	$4.2 \cdot 10^6$%	365.1%	42.6%	12.6%	[13]
0.05	$4.2 \cdot 10^9$%	—	—	—	[23]
0.05	$4.2 \cdot 10^9$%	1164.3%	79.6%	21.6%	[19], [20], [32]
0.1	$8.0 \cdot 10^{16}$%	$1.4 \cdot 10^4$%	213.8%	46.4%	[19]
0.12	$5.2 \cdot 10^{19}$%	$3.6 \cdot 10^4$%	289.6%	57.4%	[32]
0.2	—	$1.3 \cdot 10^6$%	—	—	[8]
0.2	$3.2 \cdot 10^{30}$%	$1.3 \cdot 10^6$%	791.6%	107.4%	[32]

Table 8: Some Discount Rates from the Literature

no effect on the run–time performance, increasing RS values reduce the run–time effort for both BP and LR. The deadline significantly affects the run–time of the Lagrangean relaxation based procedure, but its effect on the BP performance is beyond what we can measure. The figures with $\theta = 1.00$ are small, because it is very likely that many iterations do not yield a feasible solution and the construction part of the heuristics terminates as soon as infeasibility is detected.

The most important performance measure is the deviation of the objective function value of a feasible solution from the upper bound. Table 11 lists these figures for the three heuristics BP, LRp, and LRs as well as for the heuristic, abbreviated with LST, that is used to compute the deadline T^{MinLS}. For each instance which was feasibly solved, the deviation in percent is defined as

$$100 \cdot \frac{NPV^{UB} - NPV^{LB}}{NPV^{UB}}$$

where NPV^{UB} is the upper bound and NPV^{LB} is the objective function value of the feasible solution computed with the heuristic. As we see, the Lagrangean relaxation based approach gives tight bounds. The average result for LRp is below 0.8% for every parameter level combination. The serial construction scheme LRs does not pay off and, given the results in Table 9, it is clear that the Lagrangean relaxation based approach

$n = 30$ β						0.05					
RS	0.2			0.5			0.7			1.0		
θ	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs
1.00	107	102	89	105	88	80	114	86	83	120	120	120
1.25	120	120	120	120	120	120	120	120	120	120	120	120
1.50	120	120	120	120	120	120	120	120	120	120	120	120
1.75	120	120	120	120	120	120	120	120	120	120	120	120
2.00	120	120	120	120	120	120	120	120	120	120	120	120
$n = 30$ β						0.10					
RS	0.2			0.5			0.7			1.0		
θ	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs
1.00	107	101	87	105	87	79	114	85	81	120	120	120
1.25	120	120	120	120	120	120	120	120	120	120	120	120
1.50	120	120	120	120	120	120	120	120	120	120	120	120
1.75	120	120	120	120	120	120	120	120	120	120	120	120
2.00	120	120	120	120	120	120	120	120	120	120	120	120
$n = 60$ β						0.05					
RS	0.2			0.5			0.7			1.0		
θ	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs
1.00	63	98	65	81	88	84	116	71	88	120	120	120
1.25	120	120	120	120	120	120	120	120	120	120	120	120
1.50	120	120	120	120	120	120	120	120	120	120	120	120
1.75	120	120	120	120	120	120	120	120	120	120	120	120
2.00	120	120	120	120	120	120	120	120	120	120	120	120
$n = 60$ β						0.10					
RS	0.2			0.5			0.7			1.0		
θ	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs
1.00	63	92	68	81	87	86	116	74	87	120	120	120
1.25	120	120	120	120	120	120	120	120	120	120	120	120
1.50	120	120	120	120	120	120	120	120	120	120	120	120
1.75	120	120	120	120	120	120	120	120	120	120	120	120
2.00	120	120	120	120	120	120	120	120	120	120	120	120
$n = 90$ β						0.05					
RS	0.2			0.5			0.7			1.0		
θ	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs
1.00	34	104	57	81	89	86	112	76	89	120	120	120
1.25	120	120	120	120	120	120	120	120	120	120	120	120
1.50	120	120	120	120	120	120	120	120	120	120	120	120
1.75	120	120	120	120	120	120	120	120	120	120	120	120
2.00	120	120	120	120	120	120	120	120	120	120	120	120
$n = 90$ β						0.10					
RS	0.2			0.5			0.7			1.0		
θ	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs	BP	LRp	LRs
1.00	34	104	58	81	87	88	112	76	90	120	120	120
1.25	120	120	120	120	120	120	120	120	120	120	120	120
1.50	120	120	120	120	120	120	120	120	120	120	120	120
1.75	120	120	120	120	120	120	120	120	120	120	120	120
2.00	120	120	120	120	120	120	120	120	120	120	120	120

Table 9: Number of Feasibly Solved Instances

$n = 30$	β	0.05							
	RS	0.2		0.5		0.7		1.0	
θ		BP	LR	BP	LR	BP	LR	BP	LR
1.00		0.04	3.54	0.03	0.90	0.03	0.70	0.03	0.51
1.25		0.08	5.63	0.05	1.84	0.04	1.61	0.03	1.18
1.50		0.08	8.44	0.05	3.04	0.05	2.80	0.03	2.18
1.75		0.08	11.91	0.05	4.42	0.05	4.26	0.03	3.36
2.00		0.08	16.09	0.05	5.99	0.05	5.90	0.03	4.70
$n = 30$	β	0.10							
	RS	0.2		0.5		0.7		1.0	
θ		BP	LR	BP	LR	BP	LR	BP	LR
1.00		0.04	3.53	0.03	0.90	0.03	0.70	0.03	0.51
1.25		0.08	5.68	0.05	1.85	0.04	1.62	0.03	1.18
1.50		0.08	8.28	0.05	3.04	0.05	2.82	0.03	2.19
1.75		0.08	11.83	0.05	4.46	0.05	4.28	0.03	3.40
2.00		0.08	16.18	0.05	6.01	0.05	5.93	0.03	4.72
$n = 60$	β	0.05							
	RS	0.2		0.5		0.7		1.0	
θ		BP	LR	BP	LR	BP	LR	BP	LR
1.00		0.09	20.52	0.07	3.73	0.07	2.32	0.08	1.96
1.25		0.19	34.26	0.13	8.22	0.11	5.85	0.08	5.29
1.50		0.19	51.08	0.13	14.57	0.11	10.76	0.08	10.22
1.75		0.19	70.85	0.13	23.17	0.11	17.24	0.08	16.86
2.00		0.19	92.50	0.13	32.90	0.11	24.83	0.08	24.89
$n = 60$	β	0.10							
	RS	0.2		0.5		0.7		1.0	
θ		BP	LR	BP	LR	BP	LR	BP	LR
1.00		0.09	20.55	0.07	3.74	0.07	2.33	0.08	1.97
1.25		0.19	34.63	0.13	8.19	0.11	5.92	0.08	5.40
1.50		0.19	51.69	0.13	14.62	0.11	10.84	0.08	10.29
1.75		0.19	71.67	0.13	23.29	0.11	17.44	0.08	16.93
2.00		0.19	93.31	0.13	33.06	0.11	25.19	0.08	25.08
$n = 90$	β	0.05							
	RS	0.2		0.5		0.7		1.0	
θ		BP	LR	BP	LR	BP	LR	BP	LR
1.00		0.15	60.37	0.13	8.90	0.13	5.67	0.15	4.87
1.25		0.34	92.72	0.24	20.83	0.20	15.35	0.15	14.50
1.50		0.35	131.16	0.24	36.59	0.20	28.80	0.15	29.13
1.75		0.35	175.79	0.24	55.21	0.20	44.97	0.15	47.37
2.00		0.35	225.99	0.24	75.98	0.20	63.27	0.15	68.14
$n = 90$	β	0.10							
	RS	0.2		0.5		0.7		1.0	
θ		BP	LR	BP	LR	BP	LR	BP	LR
1.00		0.16	60.63	0.13	8.95	0.13	5.70	0.15	4.89
1.25		0.34	93.11	0.24	20.78	0.20	15.33	0.15	14.54
1.50		0.35	131.78	0.24	36.60	0.20	28.82	0.15	29.15
1.75		0.35	176.31	0.24	55.08	0.20	45.38	0.15	47.78
2.00		0.35	227.48	0.24	75.80	0.21	63.79	0.15	68.78

Table 10: Average Run–Time Performance

$n=30$ β	0.05															
RS	0.2				0.5				0.7				1.0			
θ	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs
1.00	1.58	0.41	0.41	0.45	0.57	0.13	0.13	0.13	0.48	0.09	0.11	0.09	0.17	0.04	0.00	0.00
1.25	1.74	0.40	0.38	0.39	0.69	0.16	0.14	0.14	0.62	0.13	0.12	0.11	0.27	0.09	0.00	0.00
1.50	1.88	0.49	0.38	0.40	0.79	0.23	0.15	0.15	0.74	0.19	0.13	0.12	0.37	0.14	0.00	0.00
1.75	2.02	0.58	0.38	0.40	0.90	0.30	0.15	0.15	0.86	0.25	0.13	0.12	0.46	0.20	0.00	0.00
2.00	2.15	0.66	0.39	0.41	1.00	0.37	0.16	0.16	0.97	0.31	0.14	0.13	0.54	0.24	0.00	0.00
$n=30$ β	0.10															
RS	0.2				0.5				0.7				1.0			
θ	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs
1.00	3.01	0.78	0.77	0.83	1.10	0.26	0.24	0.25	0.92	0.18	0.21	0.17	0.34	0.07	0.00	0.00
1.25	3.30	0.75	0.71	0.75	1.32	0.30	0.27	0.27	1.18	0.24	0.24	0.22	0.53	0.18	0.00	0.00
1.50	3.54	0.91	0.72	0.76	1.52	0.44	0.29	0.28	1.39	0.36	0.25	0.23	0.70	0.28	0.00	0.00
1.75	3.78	1.06	0.73	0.78	1.71	0.57	0.30	0.29	1.61	0.47	0.25	0.24	0.86	0.37	0.00	0.00
2.00	3.99	1.20	0.74	0.78	1.88	0.69	0.30	0.30	1.80	0.58	0.26	0.24	1.02	0.46	0.00	0.00
$n=60$ β	0.05															
RS	0.2				0.5				0.7				1.0			
θ	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs
1.00	1.91	0.62	0.38	0.39	0.63	0.17	0.13	0.11	0.39	0.08	0.07	0.05	0.25	0.05	0.00	0.00
1.25	2.03	0.54	0.36	0.38	0.72	0.19	0.13	0.11	0.50	0.11	0.08	0.06	0.36	0.12	0.00	0.00
1.50	2.14	0.63	0.37	0.39	0.79	0.25	0.14	0.12	0.60	0.17	0.09	0.07	0.47	0.19	0.00	0.00
1.75	2.25	0.71	0.38	0.40	0.87	0.31	0.15	0.13	0.69	0.22	0.09	0.07	0.57	0.25	0.00	0.00
2.00	2.36	0.79	0.39	0.41	0.94	0.37	0.15	0.13	0.78	0.27	0.09	0.07	0.66	0.31	0.00	0.00
$n=60$ β	0.10															
RS	0.2				0.5				0.7				1.0			
θ	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs
1.00	3.63	1.16	0.71	0.75	1.23	0.32	0.25	0.22	0.76	0.16	0.14	0.09	0.49	0.10	0.00	0.00
1.25	3.86	1.00	0.71	0.72	1.38	0.36	0.26	0.22	0.96	0.21	0.16	0.12	0.69	0.23	0.00	0.00
1.50	4.06	1.16	0.71	0.73	1.52	0.47	0.27	0.24	1.13	0.31	0.17	0.13	0.88	0.35	0.00	0.00
1.75	4.24	1.31	0.73	0.74	1.65	0.58	0.28	0.25	1.30	0.40	0.17	0.14	1.06	0.46	0.00	0.00
2.00	4.41	1.44	0.73	0.77	1.77	0.68	0.29	0.26	1.45	0.49	0.18	0.14	1.23	0.57	0.00	0.00
$n=90$ β	0.05															
RS	0.2				0.5				0.7				1.0			
θ	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs
1.00	2.22	0.58	0.40	0.42	0.67	0.19	0.11	0.09	0.43	0.09	0.05	0.04	0.29	0.07	0.00	0.00
1.25	2.35	0.69	0.38	0.43	0.76	0.23	0.12	0.10	0.51	0.13	0.07	0.06	0.38	0.13	0.00	0.00
1.50	2.46	0.76	0.39	0.43	0.85	0.29	0.13	0.11	0.59	0.19	0.08	0.06	0.46	0.19	0.00	0.00
1.75	2.57	0.84	0.39	0.45	0.93	0.35	0.13	0.12	0.67	0.24	0.08	0.07	0.53	0.25	0.00	0.00
2.00	2.68	0.91	0.40	0.46	1.00	0.41	0.14	0.12	0.74	0.29	0.08	0.07	0.61	0.30	0.00	0.00
$n=90$ β	0.10															
RS	0.2				0.5				0.7				1.0			
θ	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs	LST	BP	LRp	LRs
1.00	4.21	1.09	0.76	0.74	1.30	0.36	0.21	0.19	0.83	0.18	0.10	0.08	0.56	0.14	0.00	0.00
1.25	4.44	1.27	0.71	0.78	1.46	0.43	0.23	0.19	0.99	0.25	0.14	0.11	0.72	0.25	0.00	0.00
1.50	4.64	1.40	0.73	0.81	1.61	0.55	0.24	0.21	1.13	0.35	0.15	0.12	0.86	0.36	0.00	0.00
1.75	4.82	1.54	0.75	0.84	1.75	0.66	0.25	0.22	1.26	0.44	0.15	0.13	1.00	0.46	0.00	0.00
2.00	4.98	1.66	0.74	0.83	1.88	0.76	0.27	0.23	1.39	0.53	0.16	0.14	1.14	0.55	0.00	0.00

Table 11: Average Gap between Lower Bound and Upper Bound

ought to be combined with a parallel construction scheme. The BP results are slightly worse, but given that BP works much faster, the results indicate that BP is state–of–the–art. As expected, the lower bound defined by the makespan oriented heuristic LST is poor. The effect of the parameters is as follows. Increasing RS values reduce the gap between lower and upper bound. It is remarkable to note that for $RS = 1.0$ where resource constraints do not exist, BP is not capable to find the optimum solution while the Lagrangean relaxation based approach does. The number of activities does not affect the Lagragean relaxation based approach decidedly. The performance of BP, however, declines, if the number of activities grows and resource constraints are tight. The impact of the deadline on the performance of the Lagrangean relaxation based procedure is very small. For BP, increasing deadlines reduce the performance. The strongest influence on the performance of the heuristics comes from the discount rate β. If β increases, the gap between lower and upper bound increases, too. Note, for $\beta = 0$ every feasible solution would be an optimum solution. Hence, this result is reasonable. Putting all together, one could say that the Lagrangean relaxation based approach is more robust than BP, because only the discount rate matters.

Finally, we like to investigate, if Lagrangean relaxation pays off for computing upper bounds. As a point of reference, we use the optimum solution of the resource–unconstrained instance, NoR for short. Table 12 shows the outcome of that study by presenting the percentage deviation from the best known lower bound computed as

$$100 \cdot \frac{NPV^{UB} - NPV^{LB^*}}{NPV^{LB^*}}$$

where NPV^{UB} is the upper bound and NPV^{LB^*} is the best known lower bound for instances which were feasibly solved. And indeed, the Lagrangean relaxation of the resource constraints improves the bound derived by ignoring the resource constraints decidedly. These improvements are better the more activities there are, and, of course, the lower the value RS is. The deadline has no effect.

5 Conclusions

In this paper, we have studied the resource–constrained project scheduling problem with net present value objective. After reviewing the literature, we have concluded that sophisticated upper bounds have not been proposed so far. Hence, our aim was to close this gap. To do so, we have discussed the Lagrangean relaxation of the resource constraints which yields a total weighted completion time problem which can be solved in polynomial time. Subgradient optimization has been used to search for good Lagrangean multipliers. Furthermore, we have derived a class of heuristics which use the Lagrangean relaxation results for computing feasible solutions. An in–depth computational study has then been performed to investigate the performance. The test–bed has been constructed on the basis of the well–known ProGen instances giving 14400 instances in total. We put special emphasis on the question of how to choose the discount rates and argued that it is a must to state clearly how periods should be interpreted in the real–world in order to have a senseful meaning. By looking at the literature closely, we found that some of the data used before is far from being realistic. The results of the computational prove that the upper bounds are indeed tight. The Lagrangean relaxation based heuristic finds feasible

$n = 30$ β	0.05							
RS	0.2		0.5		0.7		1.0	
θ	NoR	LR	NoR	LR	NoR	LR	NoR	LR
1.00	0.73	0.37	0.20	0.12	0.10	0.08	0.00	0.00
1.25	0.62	0.27	0.16	0.09	0.08	0.06	0.00	0.00
1.50	0.62	0.28	0.17	0.10	0.09	0.07	0.00	0.00
1.75	0.62	0.29	0.17	0.11	0.09	0.07	0.00	0.00
2.00	0.63	0.30	0.17	0.11	0.09	0.08	0.00	0.00
$n = 30$ β	0.10							
RS	0.2		0.5		0.7		1.0	
θ	NoR	LR	NoR	LR	NoR	LR	NoR	LR
1.00	1.41	0.71	0.38	0.23	0.20	0.15	0.00	0.00
1.25	1.18	0.51	0.31	0.18	0.16	0.12	0.00	0.00
1.50	1.19	0.53	0.32	0.19	0.17	0.13	0.00	0.00
1.75	1.19	0.55	0.33	0.21	0.17	0.14	0.00	0.00
2.00	1.20	0.55	0.33	0.21	0.18	0.15	0.00	0.00
$n = 60$ β	0.05							
RS	0.2		0.5		0.7		1.0	
θ	NoR	LR	NoR	LR	NoR	LR	NoR	LR
1.00	1.15	0.58	0.25	0.14	0.07	0.04	0.00	0.00
1.25	0.87	0.31	0.18	0.09	0.06	0.04	0.00	0.00
1.50	0.88	0.32	0.19	0.10	0.06	0.04	0.00	0.00
1.75	0.88	0.33	0.19	0.10	0.06	0.05	0.00	0.00
2.00	0.89	0.34	0.19	0.11	0.06	0.05	0.00	0.00
$n = 60$ β	0.10							
RS	0.2		0.5		0.7		1.0	
θ	NoR	LR	NoR	LR	NoR	LR	NoR	LR
1.00	2.33	1.21	0.47	0.27	0.13	0.09	0.00	0.00
1.25	1.67	0.59	0.35	0.17	0.11	0.08	0.00	0.00
1.50	1.67	0.61	0.36	0.18	0.11	0.08	0.00	0.00
1.75	1.68	0.63	0.36	0.19	0.11	0.09	0.00	0.00
2.00	1.68	0.64	0.37	0.20	0.12	0.10	0.00	0.00
$n = 90$ β	0.05							
RS	0.2		0.5		0.7		1.0	
θ	NoR	LR	NoR	LR	NoR	LR	NoR	LR
1.00	1.34	0.52	0.26	0.15	0.07	0.05	0.00	0.00
1.25	1.15	0.35	0.19	0.09	0.06	0.04	0.00	0.00
1.50	1.15	0.35	0.19	0.09	0.06	0.05	0.00	0.00
1.75	1.14	0.36	0.19	0.10	0.06	0.05	0.00	0.00
2.00	1.15	0.37	0.19	0.11	0.07	0.05	0.00	0.00
$n = 90$ β	0.10							
RS	0.2		0.5		0.7		1.0	
θ	NoR	LR	NoR	LR	NoR	LR	NoR	LR
1.00	2.59	1.00	0.51	0.30	0.13	0.09	0.00	0.00
1.25	2.19	0.65	0.35	0.16	0.12	0.08	0.00	0.00
1.50	2.19	0.67	0.36	0.18	0.12	0.09	0.00	0.00
1.75	2.19	0.69	0.37	0.19	0.12	0.10	0.00	0.00
2.00	2.18	0.69	0.37	0.20	0.13	0.10	0.00	0.00

Table 12: Average Gap between Upper Bound and Best Known Lower Bound

solutions with a deviation from that bound below 0.8% on average for all parameter level combinations in the study. But, it has also been revealed that a heuristic proposed by Baroum and Patterson is almost equally good in terms of the deviation from the upper bound, that it is extremely better in terms of the run-time, but that it is weak in finding feasible solutions when the deadline is tight.

Acknowledgement

We are indebted to Andreas Drexl for his steady support and to Peter Nippel for his advice.

References

[1] BAROUM, S.M., PATTERSON, J.H., (1996), The Development of Cash Flow Weight Procedures for Maximizing the Net Present Value of a Project, Journal of Operations Management, Vol. 14, pp. 209–227

[2] BAROUM, S.M., PATTERSON, J.H., (1999), An Exact Solution Procedure for Maximizing the Net Present Value of Cash Flows in a Network, in: Weglarz, J., (ed.), Project Scheduling — Recent Models, Algorithms and Applications, Boston, Kluwer, pp. 107–134

[3] BEY, R.B., DOERSCH, R.H., PATTERSON, J.H., (1981), The Net Present Value Criterion: Its Impact on Project Scheduling, Project Management Quarterly, Vol. 12, pp. 35–45

[4] BŁAŻEWICZ, J., LENSTRA, J.K., RINNOOY KAN, A.H.G., (1983), Scheduling Subject to Resource Constraints: Classification and Complexity, Discrete Applied Mathematics, Vol. 5, pp. 11–24

[5] BRUCKER, P., DREXL, A:, MÖHRING, R.H., NEUMANN, K., PESCH, E., (1999), Resource–Constrained Project Scheduling: Notation, Classification, Models, and Methods, European Journal of Operational Research, Vol. 112, pp. 3–41

[6] CHERKASSKY, B.V., GOLDBERG, A.V., (1995), On Implementing Push–Relabel Method for the Maximum Flow Problem, in: Balas, E., Clausen, J., (eds.), Proceedings of the 4th Conference on Integer Programming and Combinatorial Optimization, Lecture Notes in Computer Science, Vol. 920, Berlin, Springer, pp. 157–171

[7] CHRISTOFIDES, N., ALVAREZ–VALDES, R., TAMARIT, J.M., (1987), Project Scheduling with Resource Constraints: A Branch and Bound Approach, European Journal of Operational Research, Vol. 29, pp. 262–273

[8] DAYANAND, N., PADMAN, R., (1997), On Modelling Payments in Projects, Journal of the Operational Research Society, Vol. 48, pp. 906–918

[9] DAYANAND, N., PADMAN, R., (1999), On Payment Schedules in Contractors Client Negotiations in Projects: An Overview of the Problem and Research Issues, in: Weglarz, J., (ed.), Project Scheduling — Recent Models, Algorithms and Applications, Boston, Kluwer, pp. 477–508

[10] DE REYCK, B., HERROELEN, W., (1998), An Optimal Procedure for the Resource–Constrained Project Scheduling Problem with Discounted Cash Flows and Generalized Precedence Relations, Computers & Operations Research, Vol. 25, pp. 1–17

[11] DOERSCH, R.H., PATTERSON, J.H., (1977), Scheduling a Project to Maximize Its Present Value: A Zero–One Programming Approach, Management Science, Vol. 23, pp. 882–889

[12] ELMAGHRABY, S.E., (1990), Project Bidding under Deterministic and Probabilistic Activity Durations, European Journal of Operational Research, Vol. 49, pp. 14–34

[13] ERENGUC, S.S., TUFEKCI, S., ZAPPE, C.J., (1993), Solving Time/Cost Trade–off Problems with Discounted Cash Flows Using Generalized Benders Decomposition, Naval Research Logistics, Vol. 40, pp. 25–50

[14] FISHER, M.L., (1981), The Lagrangean Relaxation Method for Solving Integer Programming Problems, Management Science, Vol. 27, pp. 1–18

[15] GRINOLD, R.C., (1997), The Payment Scheduling Problem, Naval Research Logistics, Vol. 19, pp. 123–136

[16] HELD, M., WOLFE, P., CROWDER, H.P., (1974), Validation of Subgradient Optimization, Mathematical Programming, Vol. 6, pp. 62–88

[17] HERROELEN, W.S., VAN DOMMELEN, P., DEMEULEMEESTER, E.L., (1997), Project Network Models with Discounted Cash Flows a Guided Tour Through Recent Developments, European Journal of Operational Research, Vol. 100, pp. 97–121

[18] ICMELI, O., ERENGUC, S.S., (1994), A Tabu Search Procedure for the Resource Constrained Project Scheduling Problem with Discounted Cash Flows, Computers & Operations Research, Vol. 21, pp. 841–853

[19] ICMELI, O., ERENGUC, S.S., (1996), A Branch and Bound Procedure for the Resource Constrained Project Scheduling Problem with Discounted Cash Flows, Management Science, Vol. 42, pp. 1395–1408

[20] ICMELI, O., ERENGUC, S.S., (1996), The Resource Constrained Time/Cost Trade-off Project Scheduling Problem with Discounted Cash Flows, Journal of Operations Management, Vol. 14, pp. 255–275

[21] ICMELI, O., ERENGUC, S.S., ZAPPE, C.J., (1993), Project Scheduling Problems: A Survey, International Journal of Operations & Production Management, Vol. 13, pp. 80–91

[22] ICMELI TUKEL, O., ROM, W.O., (1998), Analysis of the Characteristics of Projects in Diverse Industries, Journal of Operations Management, Vol. 16, pp. 43–61

[23] KAZAZ, B., SEPIL, C., (1996), Project Scheduling with Discounted Cash Flows and Progress Payments, Journal of the Operational Research Society, Vol. 47, pp. 1262–1272

[24] KOLISCH, R., PADMAN, R., (1997), An Integrated Survey of Project Scheduling, Working Paper 463, University of Kiel

[25] KOLISCH, R., SPRECHER, A., (1997), PSPLIB — A Project Scheduling Problem Library, European Journal of Operational Research, Vol. 96, pp. 205–216

[26] KOLISCH, R., SPRECHER, A., DREXL, A., (1995), Characterization and Generation of a General Class of Resource–Constrained Project Scheduling Problems, Management Science, Vol. 41, pp. 1693–1703

[27] MÖHRING, R.H., SCHULZ, A.S., STORK, F., UETZ, M., (1998), Resource Constrained Project Scheduling: Computing Lower Bounds by Solving Minimum Cut Problems, Working Paper 620, Technical University of Berlin

[28] NEUMANN, K., ZIMMERMANN, J., (1998), Exact and Heuristic Procedures for Net Present Value and Resource Levelling Problems in Project Scheduling, Working Paper WIOR–538, University of Karlsruhe

[29] ÖZDAMAR, L., ULUSOY, G., (1995), A Survey on the Resource–Constrained Project Scheduling Problem, IIE Transactions, Vol. 27, pp. 574–586

[30] PADMAN, R., SMITH–DANIELS, D.E., (1993), Early–Tardy Cost Trade–offs in Resource Constrained Projects with Cash Flows: An Optimization–Guided Heuristic Approach, European Journal of Operational Research, Vol. 64, pp. 295–311

[31] PADMAN, R., SMITH–DANIELS, D.E., SMITH–DANIELS, V.L., (1997), Heuristic Scheduling of Resource–Constrained Projects with Cash Flows, Naval Research Logistics, Vol. 44, pp. 365–381

[32] PINDER, J.P., MARUCHECK, A.S., (1996), Using Discounted Cash Flow Heuristics to Improve Project Net Present Value, Journal of Operations Management, Vol. 14, pp. 229–240

[33] RUSSELL, A.H., (1970), Cash Flows in Networks, Management Science, Vol. 16, pp. 357–373

[34] RUSSELL, R.A., (1986), A Comparison of Heuristics for Scheduling Projects with Cash Flows and Resource Restrictions, Management Science, Vol. 32, pp. 1291–1300

[35] SEPIL, C., ORTAC, N., (1997), Performance of the Heuristic Procedures for Constrained Projects with Progress Payments, Journal of the Operational Research Society, Vol. 48, pp. 1123–1130

[36] SMITH–DANIELS, D.E., AQUILANO, N.J., (1987), Using a Late–Start Resource–Constrained Project Schedule to Improve Project Net Present Value, Decision Sciences, Vol. 18, pp. 617–630

[37] SMITH–DANIELS, D.E., PADMAN, R., SMITH–DANIELS, V.L., (1996), Heuristic Scheduling of Capital Constrained Projects, Journal of Operations Management, Vol. 14, pp. 241–254

[38] SMITH–DANIELS, D.E., SMITH–DANIELS, V.L., (1987), Maximizing the Net Present Value of a Project Subject to Materials and Capital Constraints, Journal of Operations Management, Vol. 7, pp. 33–45

[39] TAVARES, L.V., (1986), Multicriteria Scheduling of a Railway Renewal Program, European Journal of Operational Research, Vol. 25, pp. 395–405

[40] TAVARES, L.V., (1987), Optimal Resource Profiles for Program Scheduling, European Journal of Operational Research, Vol. 29, pp. 83–90

[41] YANG, K.K., TALBOT, F.B., PATTERSON, J.H., (1992), Scheduling a Project to Maximize Its Net Present Value: An Integer Programming Approach, European Journal of Operational Research, Vol. 64, pp. 188-198

[42] YANG, K.K., TAY, L.C., SUM, C.C., (1995), A Comparison of Stochastic Scheduling Rules for Maximizing Project Net Present Value, European Journal of Operational Research, Vol. 85, pp. 327–339