# Capital– and resource–constrained project scheduling with net present value optimization

Pieter Leyman [1] and Mario Vanhoucke[1,2,3]

[1]Faculty of Economics and Business Administration, Ghent University, Tweekerkenstraat 2, 9000 Ghent (Belgium), pieter.leyman@ugent.be, mario.vanhoucke@ugent.be
[2]Technology and Operations Management Area, Vlerick Business School, Reep 1, 9000 Ghent (Belgium)
[3]UCL School of Management, University College London, Gower Street, London, WC1E 6BT (United Kingdom)

**Abstract**

In this paper, we study the capital–constrained project scheduling problem with discounted cash flows (CCPSPDC) and the capital– and resource–constrained project scheduling problem with discounted cash flows (CRCPSPDC). The objective of both problems is to maximize the project net present value (NPV), based on three cash flow models. Both problems include capital constraints, which force the project to always have a positive cash balance. Hence, it is crucial to schedule activities in such an order that sufficient capital is available.

The contribution of this paper is threefold. First, we propose three distinct cash flow models, which affect the capital availability during the project. Second, we introduce two new schedulers to improve capital feasibility, one for the CCPSPDC and one for the CRCPSPDC. The schedulers focus on delaying sets of activities, which cause cash outflows to be received at later time instances, in order to reduce capital shortages. Both schedulers are implemented as part of three metaheuristics from literature, in order to compare the metaheuristics' performance. Two penalty functions have been included, one to improve capital feasibility and another to improve deadline feasibility. Third, the proposed procedure has been tested on a large dataset and the added value of the schedulers has been validated. Managerial insights are provided with respect to the impact of key parameters.

**Keywords:** Project scheduling; Net present value; Capital; Metaheuristics

## 1 Introduction

Project scheduling, and in particular the resource–constrained project scheduling problem with duration minimization (RCPSP), has been the topic of a large amount of research over the past few decades. A great deal of focus has been on regular measures of performance, which are non decreasing functions of the activity finish times, as is the case for the RCPSP. Non–regular objectives, i.e. an objective for which the definition given for regular objectives does not hold, have become increasingly popular and varied. We focus on the maximization of the project net present value (NPV), which implies that a negative or positive cash flow is assigned to each activity and the project objective is to schedule all activities in such a way that the NPV is maximized. The distinction can furthermore be made between the max–NPV problem (no renewable resources are included) and the RCPSP with discounted cash flows (RCPSPDC, renewable resources are included).

In this paper, we extend the resource–unconstrained max–NPV problem with capital constraints and three cash flow models, and refer to this problem as the capital–constrained project scheduling problem with discounted cash flows (CCPSPDC). Furthermore, the problem is also extended with the presence of renewable resource constraints, and we refer to this problem as the capital– and resource–constrained project scheduling problem with discounted cash flows (CRCPSPDC). In the max–NPV problem and in the RCPSPDC no limit is set on the cash balance (the sum of the cash inflows received and the cash outflows paid) at any particular time, which implies that the cash balance may very well be negative at certain times during the project duration. The CCPSPDC and CRCPSPDC impose the additional constraints that at no point in time the cash balance, or available capital, can be negative. This way, cash outflows can only be paid if sufficient capital is available, whereas cash inflows add to this capital. We present metaheuristic solution procedures with new scheduling techniques tailored to the needs of the problems, in particular to the capital constraints. The solution procedures in this paper focus on solving capital shortages by delaying negative cash flows in order to move these cash flows later than the capital shortage.

The remainder of this paper is organized as follows. A literature overview is given in section 2. The mathematical problem definitions of the CCPSPDC and the CRCPSPDC are discussed in section 3, whereas two schedulers are the focus of section 4. In section 5 an overview of the metaheuristics employed is given, whereas the computational experiments and their results are analyzed in section 6. We finish with a conclusion and recommendations for future research in section 7.

## 2 Literature overview

Figure 1 provides an overview of the history of the max–NPV problem, since NPV optimization in project scheduling was first introduced by Russell (1970). The years in the figure indicate when the first research on the problem (extension) was conducted. The distinction is made between three large areas of research with respect to the cash inflows of activities, which depend on the negotiations between the contractor and the client:

- **Timing:** the size of the cash inflows can be determined in advance, but the occurrence of payments depends on the actual project schedule. This is typically the case if cash inflows occur at each activity's completion time. As a result, the contractor can only control the timing of cash inflows, but the size is determined by the client.

- **Size:** the payment times are selected in advance (e.g. progress payments every 10 time units), but the size of the payments can only be determined based on the schedule. The contractor can influence the size of the cash inflows, but not their occurrence times.

- **Both:** the timing and size both depend on the project schedule. In this case, the contractor can determine both, but can only employ a limited number of payments (e.g. Dayanand and Padman (1997)), or the payment times depend on the progress of the project (e.g. based on total costs incurred by the contractor (He et al., 2009a)).

Additionally, two of these research classes, namely the research on timing of cash inflows and on the combination of timing and size, have been extended to include capital and multiple activity modes. In the latter case, each activity can be executed with different time–resource combinations.

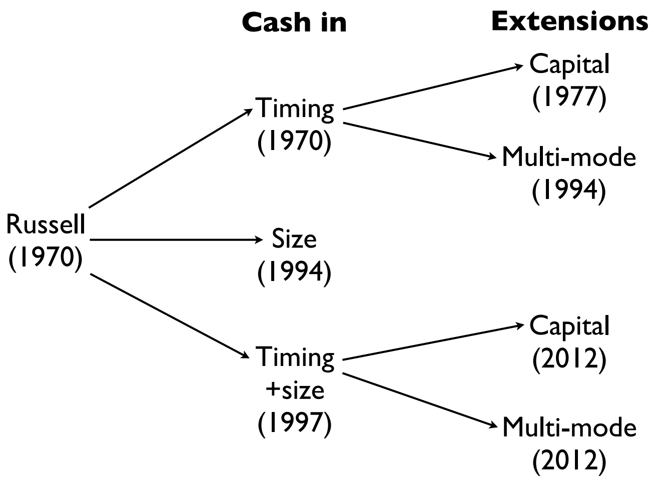

**Figure 1:** History of the max–NPV problem and its extensions.

For an overview of the project scheduling literature with NPV optimization up to 1997, we refer to Herroelen et al. (1997). A summary of more recent work from 1997 on the max–NPV problem and its extensions is displayed in table 1. In the *"Cash in"* columns of the table, we highlight whether the objective includes the determination of the timing and/or size of the cash inflows. The *"Extensions"* columns distinguish the papers in the table based on the inclusion of more specific problem characteristics. A trade–off between multiple modes of an activity (MM) is specified as well. At the bottom of the table, all papers that include capital constraints (C), even those published before 1997, are included since these papers solve a problem closely related to the one of this manuscript.

From the table and figure 1 we conclude that the majority of the research on the max–NPV problem has been on the timing of cash inflows and several extensions. Recent papers on the single– and multi–mode RCPSPDC (Leyman and Vanhoucke, 2015, 2016), however, show that no research has been done on capital restrictions in combination with renewable resource limitations.

In this paper, we contribute to the literature on NPV maximization in project scheduling in three ways. First, we propose a new scheduling technique as part of a metaheuristic approach for the max–NPV problem with capital constraints, and compare with existing work. Second, we focus on the problem with renewable resource and capital constraints, which has not yet been discussed in literature. We introduce a scheduler which handles both restrictions, while optimizing the project NPV. Third, whereas the focus in literature has been on the timing and/or size of cash *inflows*, we model the timing and size of cash *outflows* as part of a general model. These cash outflows are particularly relevant with capital constraints, since their timing and size have a profound impact on the capital level available during the project (section 3).

## 3 Problem definition

In this section, we discuss the problem definitions of the capital–constrained project scheduling problem with discounted cash flows (CCPSPDC), and of its renewable resource–constrained extension the CRCPSPDC.

| | **Cash in** | | **Extensions** | |
|---|---|---|---|---|
| **Authors** | **Timing** | **Size** | **MM** | **C** |
| Dayanand and Padman (1997) | X | X | | |
| Shtub and Etgar (1997) | X | | | |
| Etgar and Shtub (1999) | X | | | |
| Dayanand and Padman (2001a) | X | X | | |
| Dayanand and Padman (2001b) | X | X | | |
| Schwindt and Zimmermann (2001) | X | | | |
| Vanhoucke et al. (2001a) | X | | | |
| Vanhoucke et al. (2001b) | X | | | |
| Vanhoucke et al. (2003) | | X | | |
| Vanhoucke and Debels (2007) | X | | X | |
| He and Xu (2008) | X | X | X | |
| He et al. (2009a) | X | X | X | |
| He et al. (2009b) | X | X | X | |
| He et al. (2014) | X | X | X | |
| Doersch and Patterson (1977) | X | | | X |
| Smith-Daniels and Smith-Daniels (1987) | X | | | X |
| Sung and Lim (1994) | X | | X | X |
| Smith-Daniels et al. (1996) | X | | | X |
| Özdamar and Dündar (1997) | X | | X | X |
| Özdamar (1998) | X | | X | X |
| He et al. (2012) | X | X | X | X |
| **This work** | **X** | | | **X** |

**Table 1:** Literature overview max–NPV problem.

### 3.1 The capital–constrained project scheduling problem with discounted cash flows

A project can typically be represented by a directed graph or network $G(N, A)$ with $N$ used for the project activities or nodes and $A$ the precedence relations or arcs between the nodes $N$. We employ the activity–on–the–node (AoN) representation and assume a time–lag of zero for the precedence relations. Each activity $i$ ($i \in N = \{1, \dots n\}$) has a duration $d_i$, a cash inflow $c_{i,in}$ ($> 0$) and a cash outflow $c_{i,out}$ ($< 0$). Additionally, a start dummy 0 and end dummy $n + 1$ are included. The project has a deadline of $\delta_{n+1}$. The finish times $f_i$ of the activities are the decision variables.

Mathematically, the max–NPV problem can be conceptually formulated as follows:

$$\text{Maximize } \sum_{i=1}^{n} (c_{i,in} + c_{i,out}) \cdot e^{-\alpha f_i} \tag{1}$$

Subject to:

$$f_i \leq f_j - d_j, \quad \forall (i, j) \in A, \tag{2}$$

$$f_{n+1} \leq \delta_{n+1}, \tag{3}$$

$$f_i \in int^+, \quad \forall i \in N \tag{4}$$

The objective function (1) optimizes the project NPV based on a discount rate $\alpha$, and discounts both the cash in– and outflow to the activity finish time. Constraints (2) enforce precedence feasibility. Constraint (3) makes sure the project deadline is met. If no deadline were imposed, cash outflows could be delayed indefinitely. Finally, constraints (4) state that the decision variables should be integer values.

In objective function (1), it is assumed that cash in– and outflows both occur at activity completion time. However, for the extension with capital constraints (CCPSPDC) we consider alternative occurring times for the cash outflows, since the cash outflows have an impact on the capital balance. We propose the following general model for the CCPSPDC, as an extension to the max–NPV model:

$$\text{Maximize } \sum_{i=1}^{n} c_{i,in} \cdot e^{-\alpha f_i} + \sum_{i=1}^{n} \sum_{t=0}^{d_i} c_{i,out} \cdot v_{it} \cdot e^{-\alpha(f_i - d_i + t)} \tag{5}$$

Subject to:

$$C_0 + \sum_{i \in Q_f(t)} c_{i,in} + \sum_{i \in Q_s(t)} \sum_{w=0}^{min(t-(f_i-d_i),d_i)} c_{i,out} \cdot v_{iw} \geq 0, \quad t = 0, \ldots, \delta_{n+1}, \tag{6}$$

$$f_i \leq f_j - d_j, \quad \forall (i,j) \in A, \tag{7}$$

$$f_{n+1} \leq \delta_{n+1}, \tag{8}$$

$$f_i \in int^+, \quad \forall i \in N \tag{9}$$

The objective function (5) optimizes the project NPV, with the first term containing the NPV of the cash inflows. It is assumed that cash inflows always occur upon activity completion. The second term models the cash outflows based on a parameter $v_{it}$ ($v_{it} \in [0;1]$), which holds the fraction of an activity $i$'s cash outflow to be paid at time $f_i - d_i + t$. In order to ensure that the entire cash outflow $c_{i,out}$ of each activity $i$ is assigned to a time period during the activity's duration, we set $\sum_{t=0}^{d_i} v_{it} = 1$. This way, $c_{i,out}$ is distributed over the activity duration in a predefined manner. Constraints (6) model the capital feasibility and ensure that the capital does not become negative at any time unit $t$. $Q_s(t)$ is the set of activities which have been started on and before time $t$, and $Q_f(t)$ is the set of activities finished on and before time $t$. $min(t - (f_i - d_i), d_i)$ in the third sum ensures that for the capital evaluation at time $t$, only cash outflows of activity $i$ up until time $t$ are considered. Constraints (7)–(9) are the same as constraints (2)–(4) in the max–NPV model.

Figure 2 provides an overview of five different applications of the general CCPSPDC model. Each separate graph shows the capital profile of a single activity $i$, with on the horizontal axis the time $t$ and on the vertical axis the capital $C_t$ at time $t$. The values for $C_t$ are calculated based on constraints (6). The bold line in each of the graphs is the capital profile of activity $i$ during its required duration. Each column in the figure displays a different application based on different values for the $v_{it}$ parameters to model the way the cash outflows are spread over the duration, whereas the rows distinguish between the three possibilities regarding the comparative values of $c_{i,in}$ and $c_{i,out}$. In the remainder of this paper, we focus on the three highlighted applications. Models 1 and 3 constitute two extreme cases, since it is assumed that the cash outflows occur at the start and end of an activity respectively. Without loss of generality, it can be stated that model 2 serves as a representation of all model variations in between both extremes, in which the cash outflows are distributed over the time instances between the activity start and finish times.

- Model 1: cash outflows are paid at activity start times ($v_{i,0} = 1$), and hence reductions in available capital occur at the start of an activity. The resulting objective and capital constraints can be simplified and correspond with functions (10) and (11) respectively.

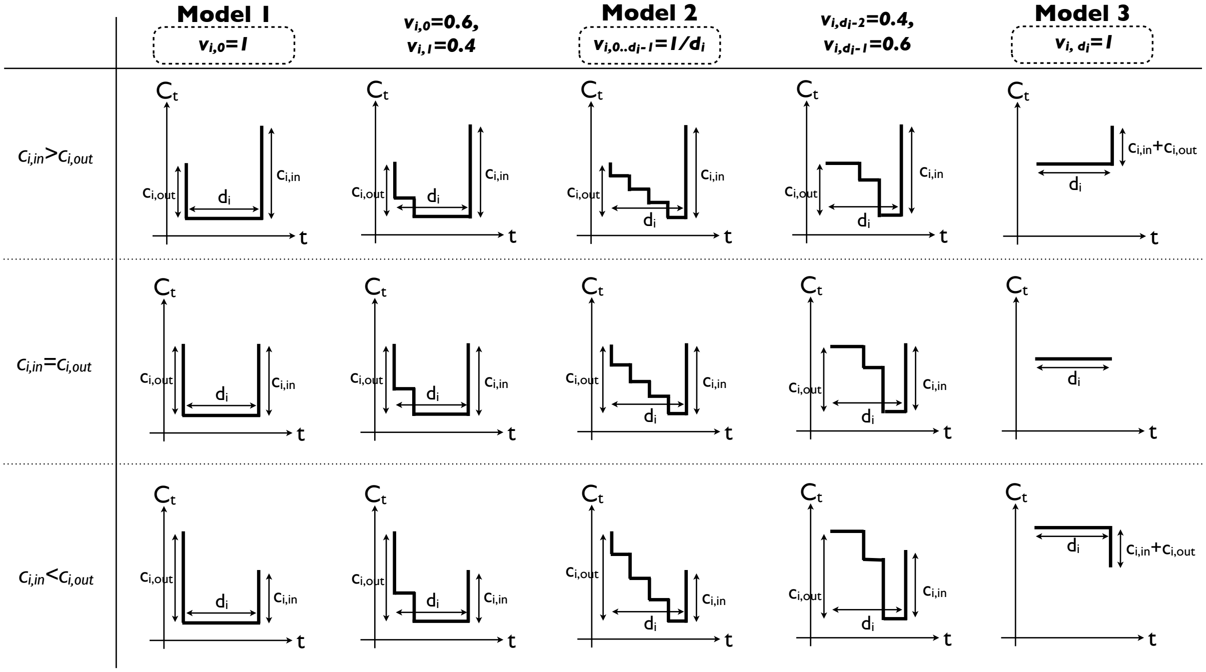

**Figure 2:** Example applications of model CCPSPDC.

$$\text{Maximize} \sum_{i=1}^{n} c_{i,in} \cdot e^{-\alpha f_i} + \sum_{i=1}^{n} c_{i,out} \cdot e^{-\alpha(f_i - d_i)} \tag{10}$$

$$C_0 + \sum_{i \in Q_f(t)} c_{i,in} + \sum_{i \in Q_s(t)} c_{i,out} \geq 0, \quad t = 0, \ldots, \delta_{n+1} \tag{11}$$

- Model 2: cash outflows are paid on a per time unit basis during the activity duration ($v_{i,0\ldots d_i-1} = 1/d_i$). The objective function and capital limitations are adjusted to functions (12) and (13) respectively.

$$\text{Maximize} \sum_{i=1}^{n} c_{i,in} \cdot e^{-\alpha f_i} + \sum_{i=1}^{n} \sum_{t=0}^{d_i-1} c_{i,out}/d_i \cdot e^{-\alpha(f_i - d_i + t)} \tag{12}$$

$$C_0 + \sum_{i \in Q_f(t)} c_{i,in} + \sum_{i \in Q_s(t)} min(t - (f_i - d_i) + 1, d_i) \cdot c_{i,out}/d_i \geq 0, \quad t = 0, \ldots, \delta_{n+1} \tag{13}$$

- Model 3: cash outflows are paid at activity finish times ($v_{i,d_i} = 1$), along with the receipt of the cash inflows. The simplified objective function (14) and capital constraints (15) are shown below.

$$\text{Maximize} \sum_{i=1}^{n} c_{i,net} \cdot e^{-\alpha f_i} \tag{14}$$

$$C_0 + \sum_{i \in Q_f(t)} c_{i,net} \geq 0, \quad t = 0, \ldots, \delta_{n+1} \tag{15}$$

### 3.2 The capital– and resource–constrained project scheduling problem with discounted cash flows

As an extension to the problem discussed in section 3.1, we propose to also include renewable resource (RR) constraints. The mathematical models of section 3.1 have to be extended with renewable resource constraints. Each activity $i$ has a resource demand $r_{ig}$ of type $g$, whereas each RR type $g$ ($g \in R = \{1, \ldots, |R|\}$) has a limited availability of $a_g$. Functions (16) have to be added as additional constraints to the models of section 3.1. $S(t)$ are the set of activities in progress at time $t$.

$$\sum_{i \in S(t)} r_{ig} \leq a_g, \quad \forall g \in R, \ t = 0, \ldots, \delta_{n+1} \tag{16}$$

The CRCPSPDC contains both renewable resource and capital constraints, both of which are fundamentally different as discussed along the following lines:

- Renewable resources have a fixed availability, are decreased at the start time of an activity, and are renewed at activity finish time. Examples include machines and workers, which are owned by the company executing the project and do not need to be purchased (Pritsker et al., 1969; Blazewicz et al., 1983).
- Cumulative resources (CR) have a variable availability, which is not necessarily the same after the completion of an activity (Neumann and Schwindt, 2002). Examples of a cumulative resource are inventory and capital.

As an example, consider the network in figure 3 with a single RR ($a_1 = 3$), an initial capital $C_0$ of 30 and a deadline of 10. The discount rate is assumed to be 1%. The graph displays the example's network structure with a duration, RR demand, and cash out– and inflow for each activity. We assume model 1 from section 3.1 is applied, with cash outflows at the activity start times and inflows at the activity finish times. We consider the optimal solution to the example and its corresponding schedule in three distinct cases.

1. CCPSPDC: the optimal schedule is displayed in the top left of figure 4. The width of each of the blocks corresponds with the activities' durations. The bold line displays the capital level available throughout the project (e.g. $C_3 = 30 - 10 - 20 + 20 = 20$). In the schedule, activity 3 is scheduled at its earliest finish time because of its positive NPV, whereas activity 2 is scheduled at its latest finish time due to its negative NPV. The cumulative NPV of activities 4 and 5 is positive so both are scheduled at their earliest finish time, even though the NPV of activity 4 is negative. No capital shortages occur. The project NPV equals 3.26 ($= -18.65 + 9.05 - 10 + 19.41 - 20 + 19.22 - 24.02 + 28.25$). The only trade–off to be made in this case is between the capital availability and the project NPV.

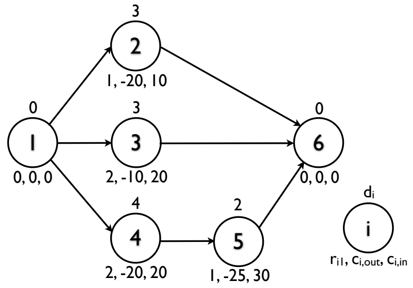

**Figure 3:** Data example.

2. RCPSPDC: the optimal schedule is shown in the top right schedule of figure 4, with the height of each of the blocks corresponding with its RR demand. The additionally introduced right vertical axis displays the RR availability $a_1$. Compared to the top right schedule, activities 4 and 5 have been delayed because activities 3 and 4 can no longer be scheduled in parallel. Since the cumulative NPV of activities 4 and 5 is smaller than the NPV of activity 3, it is best to delay the former activities. It can, however, be observed that the resulting schedule is capital infeasible since the capital drops to -5 between times 7 and 9. The only trade–off in this case is between the RR availability and the project NPV.

3. CRCPSPDC: the optimal schedule can be found at the bottom of figure 4. Due to the limited capital availability, activity 2 has to be scheduled at time 7 since otherwise insufficient capital is available to start activity 5. Alternatively, delaying activity 4 and 5 does not ensure a non–negative capital during the project duration. Hence, this case involves trade–offs between capital availability and project NPV, between RR availability and NPV, but also between capital and RR availability.

Based on the three different example schedules, it can be concluded that more trade–offs are included in the CRCPSPDC, when compared to both the CCPSPDC and the RCPSPDC.

# 4 Scheduling techniques with capital constraints

In this section, we first discuss our proposed scheduler for the CCPSPDC. Second, we propose a more complex version of the scheduler, which also takes the renewable resource constraints into account. Both schedulers consists of three main parts: an initial schedule, a capital feasibility evaluation or improvement step, and a NPV improvement part.

## 4.1 A scheduler for the CCPSPDC

We start with the construction of an initial schedule and evaluate the capital feasibility. If the schedule is capital feasible (*C–Feas*), a NPV improvement is applied, otherwise a penalty function is employed, used to denote that the resulting schedule is capital infeasible (*C–Infeas*).

The proposed scheduler is applied on a single priority list (PL) generated by a metaheuristic (section 5), and is used for each list generated by the metaheuristic. An overview of the overall flow of the scheduler is shown in figure 5.

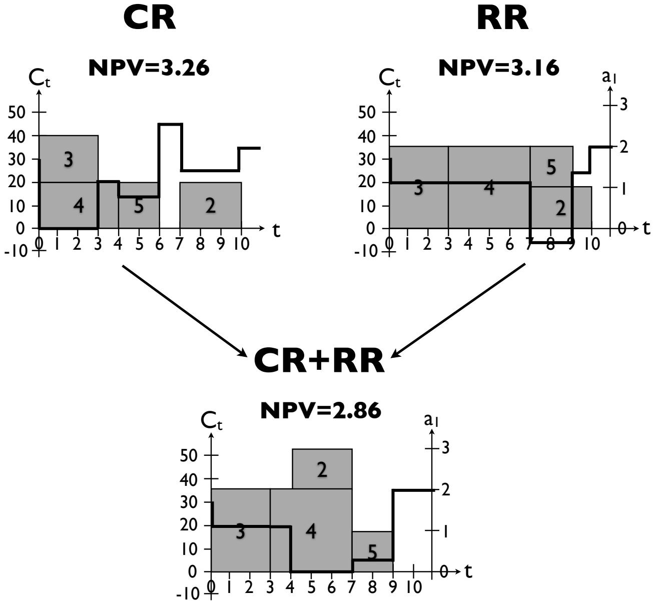

**Figure 4:** Schedules example.

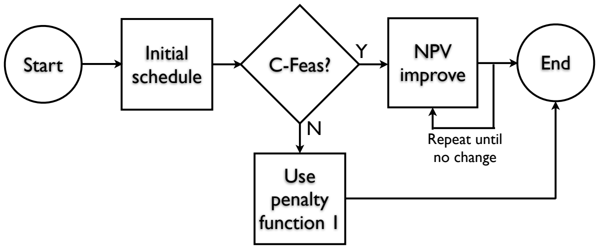

**Figure 5:** Schedule generation flow CCPSPDC.

### 4.1.1 Initial schedule

The initial schedule aims to schedule all activities subject to the precedence relations and to the capital constraints. The activities are scheduled in the order in which they appear in the PL, starting with the first activity in the list. The method aims to find a *C–Feas* finish time and starts from the activity's earliest finish time.

- The partial schedule is *C–Feas*: the finish time of activity $i$ is retained, the capital availability is updated for the entire project duration, and the scheduler continues with the next activity in the PL.

- The partial schedule is *C–Infeas*: the proposed finish time is incremented and the *C–Feas* evaluation is repeated. If the finish time, however, cannot be incremented, i.e. it would become larger than the latest finish time, no *C–Feas* finish time could be found. Activity $i$ is then scheduled at its earliest finish time and the scheduler continues with the next activity in the PL.

### 4.1.2 Capital feasibility evaluation

Once the initial schedule of section 4.1.1 has been completed, we evaluate the capital feasibility of the schedule by calculating the Excess of Capital Request ($ECR$), which is defined as: $ECR = \sum_{t=0}^{\delta_{n+1}} max(0; -C_t)$, with $C_t$ the capital level at time $t$. The $ECR$ adds the shortages of capital of all time units $t$ during the planned project duration for which the capital $C_t$ is negative. The resulting $ECR$ value gives an indication of the capital feasibility of the schedule. A large $ECR$ value on the one hand means that many shortages in capital exist or that the existing shortages are large, or both. A small $ECR$ value on the other hand implies that the shortages are limited both in number of periods and in size. Finally, an $ECR$ value of zero means no capital shortages exist.

If the initial schedule of section 4.1.1 has an $ECR$ value equal to 0, the scheduler continues with the NPV improvement of section 4.1.3. Otherwise, penalty function (17) is applied, after which the schedule is returned to the metaheuristic (section 5).

$$\begin{cases} NPV = NPV_{C-Infeas} \cdot Y_1^{ECR} & \text{if } NPV_{C-Infeas} \geq 0 \\ NPV = \frac{NPV_{C-Infeas}}{Y_1^{ECR}} & \text{otherwise} \end{cases} \tag{17}$$

$NPV_{C-Infeas}$ is the NPV of the *C–Infeas* schedule, whereas $Y_1$ ($Y_1$ ∈[0;1]) is a parameter which is tested in section 6.2. Function (17) ensures that the NPV of a *C–Infeas* schedule is considerably worse than that of any *C–Feas* schedule by reducing $NPV_{C-Infeas}$, based on the size of the $ECR$. A larger $ECR$ value leads to a larger reduction in project NPV.

### 4.1.3 NPV improvement

The NPV improvement is an adjustment of the network–based moves of Leyman and Vanhoucke (2015) to improve the project NPV for the RCPSPDC. For a *C–Feas* schedule, the goal is to improve the project NPV while maintaining capital feasibility, by delaying sets of activities with a negative cumulative NPV. These sets are constructed by taking the precedence relations between activities into account. The method of Leyman and Vanhoucke (2015) is adjusted by omitting the renewable resources and by including a capital feasibility check of each proposed delay. Once the NPV improvement method is completed, the schedule is returned to the metaheuristic.

## 4.2 A scheduler for the CRCPSPDC

The scheduler for the CCPSPDC is extended to cope with the effects of both the capital and renewable resource limitations. Due to the addition of renewable resource constraints, a more complex scheduler is required. We start with the construction of an initial schedule and continue with the capital feasibility improvement method if the initial schedule is *C–Infeas*. Finally, we go into detail about the changes in activity finish times to improve the project NPV.

The proposed scheduler is applied on a single PL generated by a metaheuristic (section 5), and is used for each solution generated by the metaheuristic. An overview of the overall flow of the proposed scheduler is shown in figure 6. In general, the scheduler consists of three parts, namely an initial schedule (section 4.2.1) and two types of delays:

- The first type of delays (*Capital feasibility improvement*) are discussed in section 4.2.2, which aim to improve capital feasibility and delay sets of activities.

- The second type of delays (*NPV improvement*) improve the project NPV by delaying sets of activities with a negative cumulative NPV and are discussed in section 4.2.3.

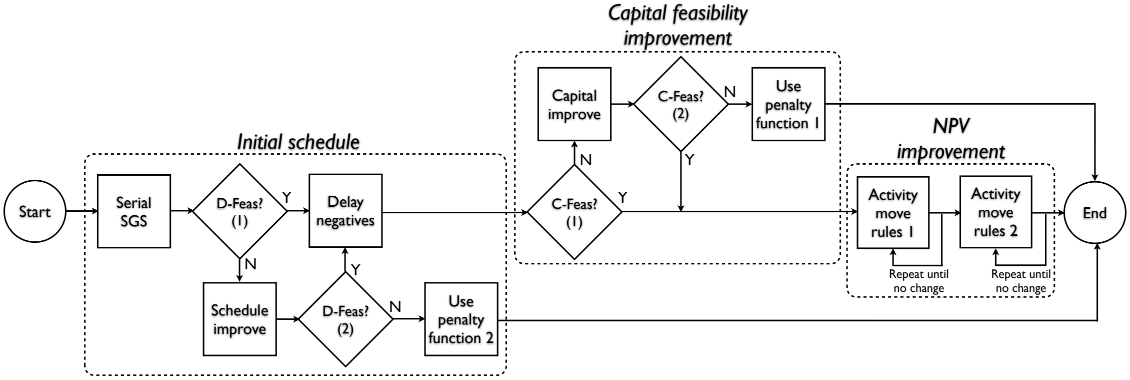

**Figure 6:** Schedule generation flow CRCPSPDC.

### 4.2.1 Initial schedule

For the initial schedule we aim to obtain a deadline feasible (*D–Feas*) earliest start schedule and use the first three steps outlined below. The fourth step improves the earliest start schedule for use in the *Capital feasibility improvement* (section 4.2.2) and *NPV improvement* (section 4.2.3) methods.

1. The serial schedule generation scheme (SSGS) of Kelley (1963) (*Serial SGS* in figure 6) is applied.

2. If the resulting schedule of the first step is deadline–infeasible (*D–Infeas*), the forward–backward improvement method of Li and Willis (1992) is used to reduce the project duration (*D–Feas?(1)* and *Schedule improve* in figure 6). This method terminates if no further reductions in project duration can be made. If the schedule of the first step is *D–Feas* the procedure continues with step 4.

3. If the schedule is still infeasible after step 2 the penalty function of Leyman and Vanhoucke (2015) is applied (*D–Feas?(2)* check and *Apply penalty function 2* in figure 6):

$$\begin{cases} NPV = -Y_2 + NPV_{D-Infeas} \cdot Y_3^{f_{n+1}-\delta_{n+1}} & \text{if } NPV_{D-Infeas} \geq 0 \\ NPV = -Y_2 + \frac{NPV_{D-Infeas}}{Y_3^{f_{n+1}-\delta_{n+1}}} & \text{otherwise} \end{cases} \qquad (18)$$

   $NPV_{D-Infeas}$ is the NPV of the *D–Infeas* schedule, whereas $Y_2$ and $Y_3$ are parameters ($Y_2 > 0$, $Y_3 \in [0; 1]$) which are tested in section 6.2. Function (18) ensures that the NPV of a *D–Infeas* schedule is considerably worse than that of any *D–Feas* schedule by reducing $NPV_{D-Infeas}$ in two ways. First, the parameter $Y_3$ reduces the project NPV based on the difference between the project duration $f_{n+1}$ and the project deadline $\delta_{n+1}$. Second, $Y_2$ subtracts a large positive value from $NPV_{D-Infeas}$.

4. All activities $i$ with a negative NPV are delayed as late as possible (*Delay negatives* in figure 6). These delays are done in the order in which the activities occur in the solution's PL (section 5), starting from the back of the list. If any delays have occurred, the fourth step is repeated until no more changes are applied. The repetition is used to ensure that all activities with a negative NPV are scheduled as late as possible, given the schedule after the previous steps and any delays of other activities.

Once the initial schedule has been generated and the method terminates with step 4, the capital feasibility improvement method of section 4.2.2 is applied. If however no *D–Feas* schedule could be found, the method terminates in step 3 and returns the schedule to the metaheuristic, since a *D–Infeas* schedule is not considered for further improvement.

### 4.2.2 Capital feasibility improvement

The goal of the improvement method is to evaluate the capital feasibility of a *D–Feas* schedule generated by the initial scheduler (section 4.2.1), and to improve the capital feasibility if the schedule is *C–Infeas.* If the schedule under consideration has an *ECR* equal to zero the algorithm continues with the *NPV improvement* of section 4.2.3. If the *ECR* is positive, however, a capital feasibility improvement method is applied to reduce the *ECR* value. Figure 7 gives an overview of the capital feasibility improvement method discussed in this section, which corresponds with *Capital improve*, *C–Feas?(2)* check, and *Use penalty function 1* in figure 6. *C–Feas?(1)* states the condition under which the capital feasibility improvement method is applied.

The goal of the method is to improve a solution's capital feasibility by delaying cash outflows. Delays for activities may require sets of succeeding activities to be delayed as well. These sets can be constructed in two ways, namely based on the network successors and based on the schedule neighbors. In the former case only precedence related activities are included in the set, whereas in the latter case neighboring activities in the project schedule, which may or may not be precedence related, are used. This way, both alternatives consider different sets of activities, which can lead to different changes in the *ECR.* We propose to first apply the schedule–based capital feasibility improvement and second the network–based variant. These delays not only delay sets of activities with a negative cumulative NPV but may also delay sets with a positive cumulative NPV, since the goal is to delay cash outflows to obtain a *C–Feas* solution. We discuss each of the method's large parts (*Check time t*, *$Set_i$ calculations*, *Delay until next positive*, *Delay until next successor* and *Schedule $set_i$*) and the links between the parts in more detail. Table 2 provides an overview of the notations used in the remainder of this section.

**Check time *t*:** the first large part of the capital feasibility improvement method checks the capital feasibility of all time instances $t$, starting from time 0 to the project deadline $\delta_{n+1}$. If the capital $C_t$ at time $t$ is negative then the set *earlySet* is constructed by including all activities $i$ with a start time smaller than or equal to $t$ in *earlySet.* The capital feasibility improvement method aims to improve the capital feasibility at time $t$, by delaying the cash outflows corresponding with some or all of these activities $i$ in *earlySet* to time units later than time $t$. The activities added to *earlySet* are sorted in decreasing order of their appearance in the PL, which implies that the later the position in the PL of an activity $i$ in *earlySet*, the earlier that activity $i$ is considered by the part *$Set_i$ calculations.*

The two possible end nodes in figure 7 for the capital feasibility improvement method are linked with the *Check time t* part in the following way. If the project deadline can be reached without any

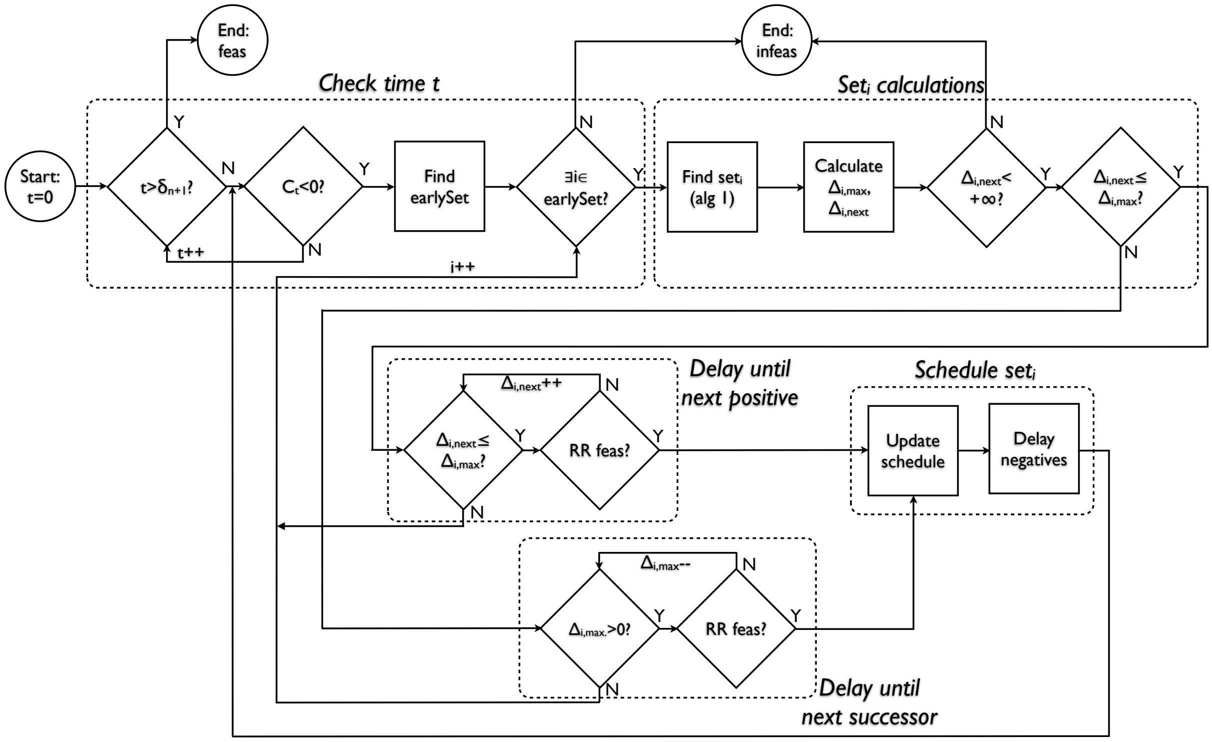

**Figure 7:** Capital feasibility improvement flow.

remaining capital shortages, the procedure terminates in the node *End: feas*, and a *C–Feas* schedule has been found. The scheduler continues with the *NPV improvement* part of the algorithm (section 4.2.3). If, however, a capital shortage still exists at time $t$ but no more unconsidered activities $i$ remain in *earlySet*, the method finishes in the node *End: infeas*, and no *C–Feas* schedule could be found based on the corresponding PL. In this case penalty function (17) is applied, after which the solution is returned to the metaheuristic (section 5).

***Set$_i$* calculations:** the second part starts with an activity $i$ from *earlySet* provided by *Check time t* and aims to find the set of activities $set_i$ which has to be delayed together with $i$, to ensure precedence feasibility. To find $set_i$, algorithm 1 is applied for the network– or schedule–based delays:

- Network–based moves ($sched = false$): the algorithm finds the set of all immediate successors of activity $i$, for which the start time equals the finish time of activity $i$. The algorithm then recursively determines whether any successors of these successors should also be added.
- Schedule–based moves ($sched = true$): the algorithm finds the set of all neighbors of activity $i$, for which the start time equals the finish time of activity $i$. These neighbors are those activities which are scheduled immediately after activity $i$ and which may or may not be precedence related to activity $i$. The algorithm then recursively determines whether any later scheduled neighbors of these neighbors should also be added.

The capital feasibility improvement method aims to delay $set_i$ to reduce the capital shortage

| Notation | Definition |
|---|---|
| $t$ | Time at which capital feasibility is evaluated |
| $C_t$ | Capital available at time $t$ |
| *earlySet* | Set of activities which may be delayed to improve $C_t$ |
| $i$ | Index of activity from *earlySet* currently under consideration |
| $set_i$ | Set of activities to be delayed together with $i$ based on algorithm 1 |
| $k$ | Index for activities in $set_i$ |
| $S_k$ | Set of immediate successors of activity $k$ based on the set of arcs $A$ in $G(N, A)$ |
| $j$ | Index of successor of activity $k$ |
| $B_k$ | Set of neighbors of activity $k$ scheduled after $k$ |
| $h$ | Index of neighbor of activity $k$ |
| $\Delta_{i,max}$ | Maximum allowable delay for $set_i$ based on any immediate successors not in $set_i$ |
| $l$ | Index of earliest activity after time $t$ which can reduce the project *ECR* |
| $\Delta_{i,next}$ | Minimum required delay for $set_i$ based on activity $l$ |
| *Sched* | Boolean used by algorithm 1 to distinguish between network– and schedule–based variant |

**Table 2:** Overview of notations capital feasibility improvement.

at time $t$, and calculates two types of delays. The maximum allowable delay for $set_i$, $\Delta_{i,max}$, based on immediate successors not in $set_i$ is calculated as follows: $\Delta_{i,max} = min(f_j - d_j - f_k | k \in set_i, j \in S_k, j \notin set_i)$. $\Delta_{i,max}$ holds the maximum delay possible for all activities $k$ from $set_i$, based on their successors $j$ not in $set_i$.

**Algorithm 1** Get all successors

```
GetAllSuc (current activity $k$,$set_i$[],boolean $sched$)
  If $sched = false$ (network–based)
      For $j = 1$ to $|S_k|$
          If $f_j - d_j = f_k \wedge j \notin set$
              $set = set \cup \{j\}$
              GetAllSuc ($j$,$set_i$,$sched$)
          End if
      End for
  End if
  If $sched = true$ (schedule–based)
      For $h = 1$ to $|B_k|$
          If $h \notin set$
              $set = set \cup \{h\}$
              GetAllSuc ($h$,$set_i$,$sched$)
          End if
      End for
  End if
  Return $set$
```

In order to reduce the capital shortage at time $t$, a second minimum delay has to be calculated. Activity $i$ from *earlySet* has to be delayed at least until the earliest next activity $l$ ($l \notin set_i$), to reduce the impact of activity $i$ on the capital shortage. Such an activity $l$ furthermore has to be scheduled after time $t$. The goal is to determine a minimum delay for activity $i$ such that its cash outflows are (partially) compensated by the cash inflow of activity $l$. Additionally, we do not wish to delay activity $i$ any further than required to reduce the capital shortage at time $t$, to avoid an unnecessary reduction of the project NPV. The minimum required delay based on such an activity

$l$ is called $\Delta_{i,next}$, and depends on the cash flow model used:

- Model 1 (cash outflows occur at activity start time): activity $i$ should be scheduled later than activity $l$, i.e. $f_i - d_i \geq f_l$, to allow for a proper compensation of $c_{i,out}$.
- Model 3 (cash outflows occur at activity finish time): activity $i$ should have a finish time equal to at least the finish time of activity $l$, i.e. $f_i \geq f_l$.
- Model 2 (cash outflows occur in a stepwise manner during the duration of activity $i$): the required delay should be in–between the delays for model 1 and 3: $f_i - X \geq f_l$, with $X \in \{1; d_i\}$. The value for $X$ determines the part of activity $i$ that has to be delayed after time $t$, in order to reduce the capital shortage at time $t$ by the largest amount possible.

If no such activity $l$ exists, the capital conflict at time $t$ cannot be solved. In that case, $\Delta_{i,next} = +\infty$, the procedure terminates, penalty function (17) is applied and the solution is returned to the metaheuristic. If such an activity $l$ does exist, the procedure compares $\Delta_{i,next}$ and $\Delta_{i,max}$ and either continues with *Delay until next positive* ($\Delta_{i,next} \leq \Delta_{i,max}$) or with *Delay until next successor* ($\Delta_{i,next} > \Delta_{i,max}$).

**Delay until next positive:** this part is applied if $\Delta_{i,next} \leq \Delta_{i,max}$, which implies that a sufficient delay to reach activity $l$ found by the *Set$_i$ calculations* part exists. The goal of *Delay until next positive* is to ensure that a delay of at least $\Delta_{i,next}$ is achieved. This way, the capital feasibility at time $t$ can be improved since the negative cash flow of activity $i$ is (partially) compensated by the positive cash flow of activity $l$. The set $set_i$ is delayed by at least $\Delta_{i,next}$ time units, and subsequently scheduled at the earliest possible time for which the capital increases.

The method starts with a delay of $\Delta_{i,next}$ and checks the RR availability for all activities in $set_i$. In case a feasible delay is found, the procedure continues with the *Schedule set$_i$* part. Otherwise, $\Delta_{i,next}$ is incremented. The RR availability check continues until a feasible delay is found (continue with *Schedule set$_i$*) or until $\Delta_{i,next}$ becomes larger than $\Delta_{i,max}$. The latter case implies that no RR feasible delay of at least $\Delta_{i,next}$ could be found, in which case the procedure returns to the set *earlySet* of the *Check time t* part. The algorithm then continues with the next activity $i$ in *earlySet* and repeats the *Set$_i$ calculations* until a feasible delay is found or until *earlySet* is empty. If no next activity $i$ in *earlySet* exists, the capital feasibility improvement procedure terminates, penalty function (17) is applied and the solution is returned to the metaheuristic.

**Delay until next successor:** this part is applied if $\Delta_{i,next} > \Delta_{i,max}$, which means that activity $i$ cannot reach activity $l$ obtained by the *Set$_i$ calculations* part. The goal of *Delay until next successor* is to delay $set_i$ as late as possible to reduce the capital shortage at time $t$. The set $set_i$ is delayed by at most $\Delta_{i,max}$ time units, and scheduled at its latest possible time given this maximum delay.

The method starts with a delay of $\Delta_{i,max}$ and evaluates the RR availability for all activities in $set_i$. If a feasible delay is found, *Schedule set$_i$* is applied. If the delay is infeasible, $\Delta_{i,max}$ is decreased by 1 and the RR check is repeated. *Delay until next successor* terminates if a feasible delay is found (continue with *Schedule set$_i$*), or if $\Delta_{i,max}$ equals zero (return to the set *earlySet* of the *Check time t* part as done for the *Delay until next positive* part).

**Schedule *set$_i$*:** the fifth part is used if either *Delay until next positive* or *Delay until next successor* has found a feasible delay for $set_i$. The schedule is updated based on the delay, and single activities with a negative NPV are scheduled as late as possible. Once the *Schedule set$_i$* part has been completed, the procedure returns to the *Check time t* part, and searches again for periods $t$ with a negative capital in order to construct a new *set$_i$* to shift.

### 4.2.3 NPV improvement

In this section, we briefly discuss the NPV improvement method. The activity move rules of this section correspond with the steps *Activity move rules 1* (network–based moves) and *Activity move rules 2* (schedule–based moves) in figure 7, and are adapted versions of the activity move rules of Leyman and Vanhoucke (2015) for the RCPSPDC.

Once a schedule is both *D–Feas* and *C–Feas* our algorithm aims to improve the project NPV while maintaining both deadline and capital feasibility. The goal of the NPV improvement move rules is to delay sets of activities with a negative cumulative NPV. Similar to the rules for capital feasibility improvement (section 4.2.2) we distinguish between a network– and schedule–based variant. The network–based delays consider an activity's predecessors and successors based on the project network, whereas the schedule–based delays consider the neighboring activities in the project schedule. We propose to first employ the network–based delays and then the schedule–based delays, in line with the results reported by Leyman and Vanhoucke (2015) for the RCPSPDC. The only adaptation required for the CRCPSPDC, is a capital feasibility evaluation of every RR feasible delay. When the NPV improvement method is completed, the resulting schedule is returned to the metaheuristic.

## 5 Metaheuristics

In this section, we focus on the details of the metaheuristics used. We choose to compare the performance of three metaheuristics, namely a tabu search, a genetic algorithm and a scatter search. These three metaheuristics are selected since they allow for a comparison between single solution, single population and multi population approaches respectively. The specific implementation of each of the metaheuristics is based on three applications from literature, which report the best result to date for the problems studied in the respective papers. All three algorithms can be applied to both the CCPSPDC and the CRCPSPDC. The algorithm parameters are tested in section 6.2. We provide an overview of the application of each metaheuristic in the following paragraphs, but first discuss the solution representation used by all three algorithms.

**Solution representation:** since delays in sections 4.1 and 4.2 are done according to the order in which activities appear in the PL, we employ the topological order (TO) representation first proposed by Valls et al. (2003, 2004), and used for the RCPSP by Debels et al. (2006). The TO representation ensures a precedence feasible ordering of activities and is updated based on actual finish times after the improvement methods discussed in sections 4.2.2 and 4.2.3 have been applied. This way, a population element's initial PL is transformed into a finish time ordered activity list. Ties are broken randomly. For an example of the PL and TO representations, we refer to A.1.

**Tabu search (TS):** we employ the TS of He et al. (2012) for the multi–mode capital–constrained project payment scheduling problem. The authors showed that the proposed TS performs best out of several alternatives. The initial solution's PL is generated randomly and is set as both the best and current solution. The appropriate scheduler of either section 4.1 or 4.2 is applied to evaluate the initial solution. Afterwards, a neighbor is generated from the current solution by applying a two–activity swap. If the neighbor's NPV (based on the scheduler used) is better than the best solution found, the latter is updated along with the current solution, and the reverse of the swap is added to the tabu list with length $L$. For the tabu list, the first–in–first–out principle is used, which means that once the list is full, the swap which has been in the list the longest is removed. If the neighbor's NPV is worse than the best solution found, the algorithm checks whether the swap employed is in the tabu list. If this is not the case, the current solution is updated and the reverse of the swap is added to the tabu list. Otherwise, the current solution is retained. Once the neighbor has been evaluated, and the current solution may have been updated, the TS again generates another neighbor of the current solution. If no change in the current solution has occurred after $C$ neighbors have been generated, a new random PL is generated. New neighboring solutions are generated until a stopping criterion is met.

**Genetic algorithm (GA):** we apply the GA proposed by Leyman and Vanhoucke (2015) for the RCPSPDC. The GA obtained the best results to date and has subsequently been employed for the multi–mode RCPSPDC with different payment models (Leyman and Vanhoucke, 2016), based on extensive computational experiments. For the initial population $|P|$ PLs are randomly generated, with $|P|$ the GA's population size. The scheduler of either section 4.1 or 4.2 is employed for all elements. These $|P|$ elements are then all introduced in the GA's population $P$ on which the selection, crossover and mutation operators are applied. A selection operator is used to select parents used for crossover. The elite selection operator is implemented, which randomly selects one parent from the subset $R$ of the best solutions in the population, and uses four–tournament selection for the second parent. A one–point crossover is subsequently applied on the two selected parents to create two children. Afterwards, a two–activity swap mutation is imposed on each child with a probability of $M$, and the appropriate scheduler is employed. These steps for generating and updating children are repeated until $|P|$ children have been created. Once $|P|$ children have been generated, the best $|R|$ parents are retained and the rest of the parents are replaced by the best $|P|-|R|$ children. The set $R$ is subsequently updated to contain the best elements in the population. Consider that this subset $R$ is also the set from which the first parent is always selected in the elite selection operator. The selection, crossover and mutation operators are then again applied, and this is repeated until a stopping criterion has been met.

**Scatter search (SS):** the SS procedure used in this paper is the algorithm of Van Peteghem and Vanhoucke (2011), which has the best results to date for the multi–mode RCPSP. First, an initial population with size $B \cdot (|B_1| + |B_2|)$ is randomly generated (*Diversification generation*). $B_1$ is the subset of best elements, whereas $B_2$ holds the most diverse elements found. $B$ is a multiplication factor applied for the initial population. The scheduler of either section 4.1 or 4.2 is used to determine the NPV of each element from the initial population. Once the initial population has been generated, the subsets are constructed (*Subset generation*). Solutions are added to $B_1$ if their NPV is better than the best solution currently in $B_1$. Alternatively, if the solution NPV is better than the worst schedule in $B_1$, and if the minimum distance of the new solution to any solution in $B_1$ is greater than $v_1 \cdot n$ ($v_1 \in [0;1]$), the new solution is also added to $B_1$. The minimum distance condition is imposed to ensure some degree of diversity in the set $B_1$. Solutions are added to $B_2$ if they are more diverse from any solution in $B_1$ than any other solution in $B_2$, or if the minimum distance of the new solution to any solution in $B_1$ is greater than $v_2 \cdot n$ ($v_2 \in [0;1]$, $v_2 > v_1$). The distance between two solutions is calculated based on the following formula, similar to Van Peteghem and Vanhoucke (2011):

$$d^f_{p_1,p_2} = \sum_{i=1}^{n} |f_i^{p_2} - f_i^{p_1}| \qquad (19)$$

$p_1$ and $p_2$ are the two solutions which are compared, and $f_i^{p_1}$ and $f_i^{p_2}$ their corresponding finish times for activity $i$. Additionally, if there are less elements in $B_2$ than $|B_2|$, the subset is seeded with random solutions. Solutions are subsequently combined by grouping all pairs from $B_1$ that contain at least one new solution, and all pairs with one solution from $B_1$ and one solution from $B_2$ (*Solution combination*). Two children are generated for every pair based on a one–point crossover. Each of the newly generated elements are scheduled (*Improvement method*), and afterwards both reference sets $B_1$ and $B_2$ are updated based on the same criteria discussed earlier (*Reference set update*). The generation of new elements and subsequent steps are repeated until a stopping criterion has been met.

# 6 Computational results

In this section, we discuss our computational results for both the CCPSPDC and the CRCPSPDC. We first give details of the proposed test data and subsequently configure the proposed schedulers and the three metaheuristics. We clearly show the added value of the steps of the capital feasibility improvement method. We analyze the influence of the data parameters and compare the results for the CCPSPDC with literature. Finally, we provide some managerial insights based on the trade–offs discussed in section 3.2. The stopping criterion based on 5,000 generated schedules as defined by Lova et al. (2009) is employed for all tests, and we assume a discount rate of 1%. All tests were run in Xcode version 5.0.2 on a computer with a 2.5 GHz Dual Core processor.

## 6.1 Test data

**Project network:** we use the same dataset as employed for the RCPSPDC by Vanhoucke (2010), which consists of 2,880 instances, but omit the renewable resources for the CCPSPDC. The parameter settings of the data are shown in table 3 and constitute the first five parameters, with deadline increase the percentage increase of the minimum project duration for the RCPSP. The project deadline is set by increasing the minimum project duration for the RCPSP with this percentage *D–Incr*.

| Parameter | Values | Source |
|---|---|---|
| Number of activities (*Act*) | 25, 50, 75 or 100 | Vanhoucke (2010) |
| Order strength (*OS*) | 0.25, 0.50 or 0.75 | Vanhoucke (2010) |
| Resourse usage (*RU*) | 2 or 4 | Vanhoucke (2010) |
| Resource constrainedness (*RC*) | 0.25, 0.50 or 0.75 | Vanhoucke (2010) |
| Deadline increase (*D–Incr*) | 5, 10, 15 or 20 | Vanhoucke (2010) |
| Profit margin percentage (*PMP*) | 0.33, 0.50 or 0.67 | This paper |
| Cash flow distribution (*CFD*) | 0.33, 0.50 or 0.67 | This paper |
| Capital constrainedness (*CC*) | 0.25, 0.50 or 0.75 | This paper |

**Table 3:** Parameter settings of test instances.

**Cash flows:** we generate cash flow data for each of the 2,880 instances based on two parameters. Smith-Daniels and Smith-Daniels (1987) and Smith-Daniels et al. (1996) state that a project which does not generate capital will not be executed from a capital management point of view, which implies that the total cash inflows of a project have to exceed its total cash outflows. A profit margin percentage (*PMP*) parameter is used to determine the difference between the total cash in– and outflows. The values for the *PMP* parameter are 0.33, 0.50 and 0.67, which correspond with a low, medium and high profit margin respectively. We start from the cash flow data of Vanhoucke (2010) for the RCPSPDC with 100% negative cash flows and apply the *PMP* on this data to generate a total cash inflow.

The distribution of the cash inflows over the activities in the project network also has an impact on the capital feasibility. As such, we introduce a cash flow distribution (*CFD*) parameter, which determines the distribution of the cash inflows over the activities in the project. The value of the *CFD* is set to 0.33, 0.50 or 0.67 which implies that respectively 33% (67%), 50% (50%) or 67% (33%) of the total cash inflow is evenly distributed over the first (second) $n/2$ activities. The division of activities in the first or second half of the project is based on the activity numbers. A low value for the *CFD* means that the cash inflows received by the first (second) $n/2$ activities are relatively small (large), whereas those received by the second $n/2$ activities are relatively large (small). The cash inflow for each of the first $n/2$ activities is equal to $-CFD \cdot \sum_{i=1}^{n} c_{i,out} \cdot \frac{1}{n/2}$, whereas the second $n/2$ activities' cash inflow equals $-(1 - CFD) \cdot \sum_{i=1}^{n} c_{i,out} \cdot \frac{1}{n/2}$. Based on the settings for both the *PMP* and *CFD* parameters, 9 combinations can be generated for each of the 2,880 instances.

**Initial capital:** we define a new parameter capital constrainedness (*CC*), inspired by the definition of resource constrainedness (RC) (Patterson, 1976), as follows:

$$CC = -\frac{\sum_{i=1}^{n} c_{i,out}/n}{C_0} \tag{20}$$

The enumerator of the right hand side holds the average of the cash outflows of all the activities in the project, whereas the denominator is the initial capital $C_0$. Based on a value set for the *CC* on the left hand side, the value for $C_0$ can be calculated. As an example, assume a project with 10 activities and a total cash outflow of -1,000. We wish to set the *CC* to 0.50. In this case, the initial capital $C_0$ is set to -(-1,000/10)/0.50 = 200. Similar to the values chosen for the *RC* in the data of Vanhoucke (2010), we choose to employ values of 0.25, 0.50 or 0.75 for the *CC*. Finally, consider that the *CC* is independent of both the *PMP* and the *CFD*, since different values of both cash flow parameters change the cash inflows of the activities but not the cash outflows.

An overview of the values of the three additional data parameters *PMP*, *CFD* and *CC* can be found at the bottom of table 3. As a result, the total number of test instances is 2,880 x 9 cash flow variants x 3 *CC* levels = 77,760 instances per cash flow model. The 9 cash flow files for each project instance of Vanhoucke (2010) can be found online at *www.projectmanagement.ugent.be* (Research → Project scheduling → Net present value), along with the network data and the best known solutions for the CCPSPDC and the CRCPSPDC.

## 6.2 Algorithm configuration

In this section, we discuss the results of the configuration of our algorithm. We first briefly discuss the settings of the algorithm parameters, and subsequently go into detail about the added value of the proposed scheduler for both the CCPSPDC and the CRCPSPDC. The performance of the three metaheuristics is also analyzed for both problems. All tests are run on 20% of the data presented in section 6.1, by employing each first out of five instances.

|  |  |  | *PMP* |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|
|  |  |  | 0.33 |  | 0.50 |  | 0.67 |  |
|  |  |  | $Y_1$ | $Y_3$ | $Y_1$ | $Y_3$ | $Y_1$ | $Y_3$ |
| **Model 1** | *CFD* | 0.33 | 0.99 | 0.85 | 0.95 | 0.90 | 0.70 | 0.90 |
|  |  | 0.50 | 0.95 | 0.90 | 0.75 | 0.95 | 0.70 | 0.95 |
|  |  | 0.67 | 0.85 | 0.95 | 0.70 | 0.95 | 0.70 | 0.95 |
| **Model 2** | *CFD* | 0.33 | 0.99 | 0.85 | 0.95 | 0.80 | 0.70 | 0.80 |
|  |  | 0.50 | 0.85 | 0.90 | 0.75 | 0.85 | 0.65 | 0.85 |
|  |  | 0.67 | 0.80 | 0.95 | 0.65 | 0.90 | 0.60 | 0.90 |
| **Model 3** | *CFD* | 0.33 | 0.99 | 0.85 | 0.90 | 0.90 | 0.80 | 0.90 |
|  |  | 0.50 | 0.90 | 0.90 | 0.80 | 0.95 | 0.75 | 0.95 |
|  |  | 0.67 | 0.80 | 0.95 | 0.75 | 0.95 | 0.70 | 0.95 |

**Table 4:** Penalty function parameters.

### 6.2.1 Algorithm parameters

$Y_2$ of penalty function (18) is set to 20,000, a value much larger than the NPV of any feasible solution, whereas the values for $Y_3$ can be found in table 4, along with the best found values for the $Y_1$ parameter of penalty function (17) for the capital feasibility. Based on our tests, we found that both values for $Y_1$ and $Y_3$ depend on *PMP* and on *CFD*, although the impact is larger for $Y_1$. Furthermore, minor differences can be observed in the values of both $Y_1$ and $Y_3$ between the three cash flow models. The values for the penalty function parameters are applicable to all three metaheuristics.

The best found values for the parameters of each of the metaheuristics are displayed in table 5. These values are, in general, in line with the values reported by the three applications of the algorithms in literature (He et al. (2012), Leyman and Vanhoucke (2015), and Van Peteghem and Vanhoucke (2011) respectively).

|  | **Parameter** | **Value** |
|---|---|---|
| **TS** | $L$ | 20 |
|  | $C$ | 10 |
| **GA** | $\lvert P\rvert$ | 50 |
|  | $\lvert R\rvert$ | 5 |
|  | $M$ | 0.95 |
| **SS** | $B$ | 9 |
|  | $\lvert B_1\rvert$ | 10 |
|  | $\lvert B_2\rvert$ | 8 |
|  | $v_1$ | 0.30 |
|  | $v_2$ | 0.50 |

**Table 5:** Metaheuristic parameters.

### 6.2.2 CCPSPDC

**NPV improvement:** we evaluate the added value of the NPV improvement, which is applied if the initial schedule is *C–Feas*. We compare the scheduler without the NPV improvement ($NPV_{-}$) and the scheduler with the NPV improvement of Leyman and Vanhoucke (2015) ($NPV_L$) in table 6. In both cases the GA is used as metaheuristic. The comparison between both alternatives is done based on the percentage average deviation from an upper bound (*%AvDev*). The lower the percentage deviation, the better the performance of the algorithm. The upper bound is calculated by the optimal procedure of Vanhoucke et al. (2001b) for the max–NPV problem. Only the instances for which both alternatives found a feasible solution are included. The percentage of *C–Feas* solutions found is displayed in the *%C* columns. We also compare the results based on the percentage average difference between the NPV in both cases (*%AvDiff*), and calculate the p–values ($p$) of the difference between the two options. Based on the results in the table it can be stated that the NPV improvement method has a clear added value for all three models, with the highest added value achieved for model 1.

| | $NPV_-$ | | $NPV_L$ | | $\Delta NPV$ | |
|---|---|---|---|---|---|---|
| | *%AvDev* | *%C* | *%AvDev* | *%C* | *%AvDiff* | p |
| **Model 1** | 11.37 | 94.10 | **5.35** | 94.13 | 7.38 | <0.001 |
| **Model 2** | 7.65 | 96.14 | **2.25** | 96.12 | 6.35 | <0.001 |
| **Model 3** | 4.37 | 98.57 | **0.07** | 98.59 | 4.85 | <0.001 |

**Table 6:** Added value NPV improvement CCPSPDC.

**Metaheuristics:** in table 7 we compare the three metaheuristics. Only instances for which all alternatives found a feasible solution are taken into account. The GA is shown to perform best ($p < 0.001$) for all three models, although it can be observed that the difference between the three alternatives is very small for model 3.

| | **TS** | | **GA** | | **SS** | |
|---|---|---|---|---|---|---|
| | *%AvDev* | *%C* | *%AvDev* | *%C* | *%AvDev* | *%C* |
| **Model 1** | 5.31 | 92.91 | **5.00** | **94.13** | 5.30 | 93.08 |
| **Model 2** | 2.19 | 94.73 | **1.98** | **96.12** | 2.20 | 95.27 |
| **Model 3** | 0.07 | 98.09 | **0.06** | **98.75** | 0.07 | 98.25 |

**Table 7:** Comparison of metaheuristics CCPSPDC.

**Computation times:** table 8 provides an overview of the average computation times in seconds (s) for the three models, given the 5,000 schedules stopping criterion. We distinguish between the average computation time used by the scheduler ($AvCT_S$), the time used by the metaheuristic ($AvCT_M$), and the total computation time ($AvCT_T$). The numbers between brackets display the percentage of the total computation used for the scheduler, metaheuristic and the total procedure respectively. The results in the table indicate that the majority of the computation time is spent applying the scheduler to solutions generated by the GA. The average computation times are the highest for model 2 due to the more complex cash flow profiles (section 3.1).

### 6.2.3 CRCPSPDC

**Capital feasibility improvement:** we test the added value of each part of the capital feasibility improvement, by comparing several options based on the percentage of *C–Feas* solutions found (*%C–Feas*) in table 9, with the best option marked in bold. The alternatives which we consider are the following: $C_{--}$: no capital feasibility improvement; $C_{SD}$: only the schedule–based moves (S)

|  | **CCPSPDC** |  |  | **CRCPSPDC** |  |  |
|---|---|---|---|---|---|---|
|  | $AvCT_S$ | $AvCT_M$ | $AvCT_T$ | $AvCT_S$ | $AvCT_M$ | $AvCT_T$ |
| **Model 1** | 1.04 | 0.16 | 1.21 | 4.78 | 0.16 | 4.94 |
|  | (86.37) | (13.63) | (100.00) | (96.82) | (3.18) | (100.00) |
| **Model 2** | 1.41 | 0.16 | 1.58 | 7.87 | 0.17 | 8.04 |
|  | (89.57) | (10.43) | (100.00) | (97.87) | (2.13) | (100.00) |
| **Model 3** | 0.66 | 0.16 | 0.82 | 4.45 | 0.19 | 4.64 |
|  | (80.37) | (19.63) | (100.00) | (95.83) | (4.17) | (100.00) |

**Table 8:** Computation times (s).

and the delay of activities with a negative NPV (D); $C_{ND}$: only the network–based moves (N) on top of D; $C_{FD}$: the full proposed method ($F$) with first the schedule– and then the network–based moves, including the delay of activities with a negative NPV; $C_{RD}$: first the network– and then the schedule–based moves (reverse: $R$); $C_{F-}$: the proposed method without the delay of activities with a negative NPV.

|  | $C_{--}$ | $C_{SD}$ | $C_{ND}$ | $C_{FD}$ | $C_{RD}$ | $C_{F-}$ |
|---|---|---|---|---|---|---|
| **Model 1** | 43.94 | 78.86 | 77.00 | **80.19** | 79.31 | 76.36 |
| **Model 2** | 57.58 | 86.94 | 85.80 | **87.77** | 87.10 | 86.16 |
| **Model 3** | 95.22 | 97.39 | 97.36 | **97.42** | 97.36 | 97.30 |

**Table 9:** Added value capital feasibility improvement CRCPSPDC (*%C–Feas*).

Based on the results in table 9, we conclude that the proposed capital feasibility improvement method, with first the schedule– and then the network–based moves, performs best for all three models. As illustrated in A.1, the schedule–based variant of the improvement method offers a more efficient way to reduce capital shortages, which leads to a higher capital feasibility. Hence, the schedule–based delays should be applied first for all three models. The added value of delaying single activities with a negative NPV is made clear as well by comparing the *%C–Feas* values for the $C_{FD}$ and $C_{F-}$ options. Based on the higher feasibility for $C_{FD}$, we conclude that the single activity delays have an added value by allowing more capital shortages to be solved (A.2). It can, however, be observed that for model 3 only small differences can be found between the different options for the capital feasibility improvement method, but that the method allows for an improvement nonetheless when compared to $C_{--}$.

**NPV improvement:** the proposed scheduler without any NPV improvement ($NPV_{-}$) is compared with the scheduler with the NPV improvement of Leyman and Vanhoucke (2015) ($NPV_L$) in table 10, based on the GA. Only instances for which a feasible solution could be found by both $NPV_{-}$ and $NPV_L$, are included in the *%AvDev* and *%AvDiff* calculations in the table. *%D* is the percentage of deadline feasible solutions found. Based on the results in the table, we conclude that the NPV improvement method has a significant, albeit small, added value for all three models. The added value is relatively small due to the rules for capital feasibility improvement delaying sets of activities, independent of the set's cumulative NPV. As such, sets of activities with a negative cumulative NPV may already have been delayed, in particular by the *Delay until next successor* part of the capital feasibility improvement method of section 4.2.2 and by the *Delay negatives* step.

**Metaheuristics:** in table 11 we compare the performance of the three metaheuristics. We also

|  | $NPV_-$ |  |  | $NPV_L$ |  |  | $\Delta NPV$ |  |
|---|---|---|---|---|---|---|---|---|
|  | *%AvDev* | *%C* | *%D* | *%AvDev* | *%C* | *%D* | *%AvDiff* | *p* |
| **Model 1** | 33.59 | 80.19 | 98.86 | **33.56** | 80.39 | 99.01 | 0.08 | <0.001 |
| **Model 2** | 31.88 | 87.77 | 99.02 | **31.81** | 87.13 | 99.11 | 0.18 | <0.001 |
| **Model 3** | 30.34 | 97.45 | 99.02 | **30.30** | 97.58 | 99.09 | 0.05 | <0.001 |

**Table 10:** Added value NPV improvement CRCPSPDC.

compare with 10,000 random schedules. Only instances for which all alternatives found a feasible solution are taken into account. We conclude that the GA performs best for all three models (lowest *%AvDev* and highest *%C*), with the SS second best. Additionally, all p–values, based on a pairwise comparison between each of the four alternatives, are smaller than 0.001, indicating a significant difference. Only the difference between TS and 10,000 randoms had a slightly bigger p–value of 0.004 for model 1.

|  | **TS** |  | **GA** |  | **SS** |  | **10,000 randoms** |  |
|---|---|---|---|---|---|---|---|---|
|  | *%AvDev* | *%C* | *%AvDev* | *%C* | *%AvDev* | *%C* | *%AvDev* | *%C* |
| **Model 1** | 38.02 | 72.99 | **33.62** | **80.39** | 35.98 | 74.49 | 38.02 | 72.49 |
| **Model 2** | 35.36 | 79.69 | **31.49** | **87.13** | 33.60 | 83.11 | 35.39 | 80.74 |
| **Model 3** | 33.20 | 92.26 | **29.91** | **97.58** | 31.69 | 95.36 | 33.29 | 91.79 |

**Table 11:** Comparison of metaheuristics CRCPSPDC.

The differences in performance between the three metaheuristics (tables 7 & 11) can be explained along the following lines:

- The TS is a single solution–based metaheuristic, whereas the GA and SS are population–based metaheuristics. As a result, there is no interaction between different solutions in the TS, whereas the GA and SS on the contrary allow for interaction by making use of a crossover operator, resulting in a better performance for the latter.

- A TS algorithm may be too local and can have issues with respect to diversity as a result (Gendreau and Potvin, 2010).

- Based on section 5, it can be stated that the GA employs an elite set and as a result requires a higher mutation rate (Reeves, 2010). This way, the GA implicitly distinguishes between a high quality set and between a diverse set, similarly to the SS algorithm. The main difference with the SS is the lack of distance functions to permit entrance into both sets.

- The use of the elite set in the GA bears resemblances to evolutionary path relinking, namely the combination of high quality solutions (Resende et al., 2010).

- Based on these final two observations, we can state that the GA contains elements of both a SS and of a path relinking algorithm, but that the inclusion of these elements in the GA framework leads to better results than in the SS framework.

To mitigate these issues, and to show the validity of our explanation, we have adjusted the three metaheuristics in the following way:

- In the TS we implement some form of interaction by applying a one–point crossover. The generated child is used as new starting solution, whereas the father is the best results obtained after the previous run of 500 generations, and the mother is a randomly generated element. Furthermore, for generating a neighboring solution, the TS now applies a two–activity swap operator repeatedly, in order to increase diversity. Our tests indicate that a factor 4 is appropriate. The resulting algorithm is referred to as TS' in the remainder of this section.

- The GA' algorithm is more similar to the "standard" GA framework (Reeves, 2010), and uses a four–tournament selection for both parents instead of the elite selection. A lower mutation rate of 5% is also used, along with a population size of 90 and only the best element is retained after each generation. These parameter settings have been tested and displayed the best results.

- The SS' algorithm omits the distance function on the diverse set $B_2$ and applies a two–activity swap with a mutation rate of 10% on all new elements generated with one parent from the high quality set $B_1$ and one parent from $B_2$. This way, the diversity is amplified in the SS' algorithm in comparison to the SS algorithm. In doing so, we introduce several elements more typical for a GA in the SS framework.

All additional tests with respect to the TS', GA' and SS' algorithms have been run for the CRCPSPDC with model 2, since this model can be seen as a middle ground in the general model of section 3. A summary of the results is displayed in table 12, which shows the differences between the six metaheuristics based on *%AvDev*, the average project NPV of the feasible instances found by all alternatives (*AvNPV*), and *%C*. We can conclude that:

- The TS' and SS' have a better performance than the TS and SS respectively, which validates our claims with respect to the shortcomings in the TS and SS frameworks for the problem under consideration.

- The results of the GA' are worse than those of the GA, which highlights the added value of including some SS and path relinking concepts in the GA.

- GA' still obtains better results than the improved SS', which shows that the GA framework allows for a higher performance than the SS framework on the CRCPSPDC. Leyman and Vanhoucke (2015) obtained similar results when they compared the GA and a more "standard" GA implementation with the SS of Vanhoucke (2010) for the RCPSPDC. It can furthermore be stated that a GA typically employs randomization, whereas a SS uses memory–based strategies (Martí et al., 2006), which shows that randomization is more important given the NPV objective and the proposed schedulers.

| | **TS** | **TS'** | **GA** | **GA'** | **SS** | **SS'** |
|---|---|---|---|---|---|---|
| *%AvDev* | 35.40 | 35.24 | 31.53 | 32.00 | 33.64 | 32.99 |
| *AvNPV* | 3,831.74 | 3,853.80 | 4,109.67 | 4,074.26 | 3,954.48 | 3,996.73 |
| *%C* | 79.69 | 80.59 | 87.13 | 87.31 | 83.11 | 84.37 |

**Table 12:** Additional comparison of metaheuristics CRCPSPDC (model 2).

**Computation times:** an overview of the average computation times for the CRPCPSDC is included in the right part of table 8. The conclusions are similar as for the CCPSPDC (left part of the table), and show that the majority of the computation time is used for the scheduler, and that the procedure is the slowest for model 2. The computation times for the scheduler are considerably higher for the CRCPSPDC compared to those for the CCPSPDC, which was to be expected given the more complex scheduler due to the additional resource limitations.

**Convergence:** we analyze the convergence of the GA based on the number of generations until the 5,000 schedules stopping criterion is reached. Figure 8 displays the average project NPV for the three models as the number of generations of the GA increases for the CRCPSPDC. The average number of generations is 54, and an average of 1.85 schedules are used for a single application of the scheduler.

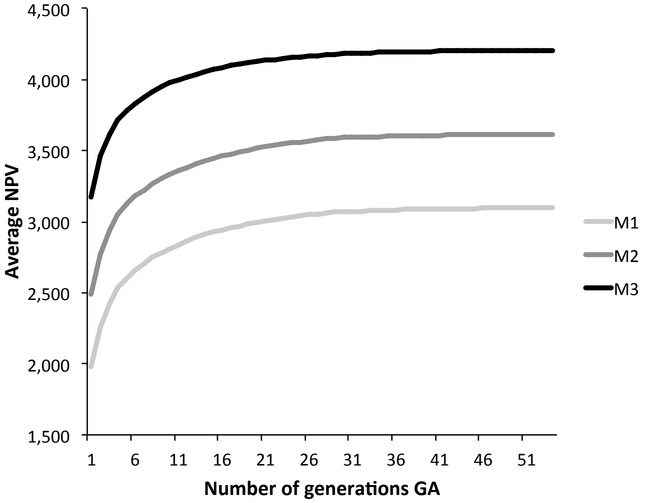

**Figure 8:** Convergence GA.

## 6.3 Discussion & comparison

In this section, we discuss the best results for the CCPSPDC (table 13) and for the CRCPSPDC (table 14). We compare the results for the CCPSPDC with a composite heuristic, which is based on two algorithms from literature. The composite algorithm splits the problem into two parts, namely optimize the project NPV and reduce capital shortages, similar to Smith-Daniels et al. (1996). The proposed heuristic employs the best method of Smith-Daniels et al. (1996) to make decisions with respect to capital, but uses the exact recursive method of Vanhoucke et al. (2001b) for the max–NPV problem. In the remainder of this section, the composite procedure is abbreviated as SD–V, or Smith–Daniels–Vanhoucke.

From the results in tables 13 and 14 the following general conclusions can be drawn:

- The GA outperforms the SD–V algorithm for all three models (table 13). The p–values of the comparison between both algorithms are smaller than 0.001 for the three models.
- Capital feasibility is the hardest to obtain with a small or a large number of activities. For a low number of activities, i.e. 25, only a small number of possible sequences exist in which the

activities can be scheduled. This implies that capital shortages are harder to solve, because fewer activities can be moved to reduce shortages. For a high number of activities, i.e. 100, a great many ways exist in which activities can be delayed. Hence, it is more difficult to find the best delays to solve capital shortages.

- The impact of the distribution of the cash inflows over activities (*CFD*) has a larger impact on the project NPV and on capital feasibility, compared to the profit of the project (*PMP*).
- The inclusion of renewable resources makes the problem harder to solve (higher *%AvDev* and lower *%C* when comparing tables 13 and 14). In the CRCPSPDC (table 14), imposing a stronger RR restriction (a higher *RU* and *RC* value) makes it easier to construct a *C–Feas* schedule, but at the cost of a lower project NPV.
- The computation times in table 14 give an indication of the drivers of algorithm complexity in terms of the data parameters. The number of activities and the resource usage have the largest impact.

The detailed results from both tables can also be found online at *www.projectmanagement.ugent.be*.

| | | **Model 1** | | | | **Model 2** | | | | **Model 3** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | **GA** | | **SD–V** | | **GA** | | **SD–V** | | **GA** | | **SD–V** | |
| | | *%AvDev* | *%C* | *%AvDev* | *%C* | *%AvDev* | *%C* | *%AvDev* | *%C* | *%AvDev* | *%C* | *%AvDev* | *%C* |
| *Act* | 25 | 3.50 | 87.84 | 5.51 | 79.62 | 1.41 | 92.25 | 2.39 | 88.19 | 0.06 | 98.33 | 0.10 | 97.67 |
| | 50 | 3.84 | 97.13 | 6.33 | 93.13 | 1.43 | 98.80 | 2.49 | 96.98 | 0.03 | 99.99 | 0.01 | 99.94 |
| | 75 | 4.86 | 97.85 | 7.86 | 94.28 | 1.85 | 99.09 | 3.15 | 96.93 | 0.05 | 99.96 | 0.06 | 99.67 |
| | 100 | 6.01 | 95.07 | 9.65 | 89.76 | 2.58 | 96.08 | 4.33 | 92.16 | 0.08 | 96.27 | 0.14 | 95.45 |
| *OS* | 0.25 | 4.49 | 98.61 | 6.75 | 95.94 | 1.91 | 99.78 | 2.88 | 98.96 | 0.04 | 100.00 | 0.01 | 99.98 |
| | 0.50 | 4.95 | 95.70 | 8.13 | 88.92 | 1.92 | 98.06 | 3.38 | 93.74 | 0.06 | 99.54 | 0.09 | 99.26 |
| | 0.75 | 4.29 | 89.10 | 7.32 | 82.75 | 1.60 | 91.82 | 3.01 | 87.99 | 0.07 | 96.37 | 0.15 | 95.30 |
| *D–Incr* | 5 | 4.54 | 92.87 | 7.33 | 86.41 | 1.79 | 95.95 | 3.04 | 92.15 | 0.05 | 98.55 | 0.08 | 98.14 |
| | 10 | 4.56 | 94.24 | 7.35 | 88.80 | 1.81 | 96.51 | 3.08 | 93.46 | 0.05 | 98.65 | 0.08 | 98.19 |
| | 15 | 4.59 | 95.12 | 7.41 | 90.25 | 1.83 | 96.79 | 3.11 | 94.19 | 0.06 | 98.68 | 0.08 | 98.19 |
| | 20 | 4.62 | 95.66 | 7.46 | 91.35 | 1.84 | 96.97 | 3.13 | 94.47 | 0.06 | 98.66 | 0.08 | 98.21 |
| *PMP* | 0.33 | 4.98 | 90.22 | 8.42 | 82.48 | 2.20 | 93.12 | 3.91 | 87.58 | 0.12 | 96.12 | 0.18 | 95.14 |
| | 0.50 | 4.80 | 95.57 | 7.64 | 90.61 | 1.88 | 97.52 | 3.15 | 95.05 | 0.04 | 99.79 | 0.06 | 99.43 |
| | 0.67 | 4.02 | 97.64 | 6.24 | 94.51 | 1.42 | 99.02 | 2.30 | 98.06 | 0.01 | 100.00 | 0.00 | 99.98 |
| *CFD* | 0.33 | 8.89 | 85.44 | 14.98 | 74.42 | 3.73 | 90.05 | 6.74 | 82.33 | 0.10 | 95.91 | 0.24 | 94.55 |
| | 0.50 | 3.32 | 98.48 | 5.24 | 95.75 | 1.14 | 99.72 | 1.90 | 98.95 | 0.02 | 100.00 | 0.00 | 100.00 |
| | 0.67 | 2.53 | 99.50 | 3.71 | 97.43 | 0.91 | 99.90 | 1.25 | 99.41 | 0.05 | 100.00 | 0.00 | 100.00 |
| *CC* | 0.25 | 2.51 | 98.78 | 4.60 | 96.34 | 0.64 | 99.41 | 1.33 | 98.47 | 0.04 | 99.49 | 0.02 | 99.21 |
| | 0.50 | 5.06 | 94.18 | 8.04 | 88.14 | 1.96 | 96.50 | 3.41 | 93.01 | 0.05 | 98.58 | 0.09 | 98.27 |
| | 0.75 | 6.47 | 90.46 | 9.92 | 83.11 | 2.97 | 93.75 | 4.70 | 89.22 | 0.07 | 97.84 | 0.13 | 97.06 |
| *Overall* | | 4.58 | 94.47 | 7.39 | 89.20 | 1.82 | 96.55 | 3.09 | 93.56 | 0.06 | 98.64 | 0.08 | 98.18 |

**Table 13:** Comparison with literature CCPSPDC.

| | | **Model 1** | | | **Model 2** | | | **Model 3** | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | *%AvDev* | *%C* | *AvCT (s)* | *%AvDev* | *%C* | *AvCT (s)* | *%AvDev* | *%C* | *AvCT (s)* |
| *Act* | 25 | 16.89 | 73.21 | 0.55 | 15.37 | 83.07 | 0.71 | 14.45 | 97.67 | 0.49 |
| | 50 | 28.85 | 83.80 | 2.67 | 27.13 | 91.52 | 4.00 | 25.93 | 99.52 | 2.20 |
| | 75 | 38.58 | 83.20 | 6.04 | 36.59 | 91.18 | 9.11 | 35.71 | 98.72 | 5.02 |
| | 100 | 47.45 | 78.81 | 12.34 | 45.59 | 85.96 | 18.92 | 44.61 | 94.35 | 10.15 |
| *OS* | 0.25 | 32.71 | 85.64 | 6.78 | 31.17 | 93.60 | 10.88 | 30.53 | 98.84 | 5.52 |
| | 0.50 | 35.06 | 79.00 | 5.25 | 32.89 | 87.84 | 7.87 | 31.32 | 98.16 | 4.34 |
| | 0.75 | 31.92 | 74.62 | 4.17 | 29.80 | 82.36 | 5.80 | 28.23 | 95.70 | 3.53 |
| *RU* | 2 | 26.79 | 72.90 | 2.02 | 24.47 | 84.80 | 2.97 | 23.19 | 97.53 | 1.87 |
| | 4 | 38.66 | 86.61 | 8.78 | 37.69 | 91.06 | 13.40 | 36.90 | 97.60 | 7.05 |
| *RC* | 0.25 | 32.16 | 76.37 | 4.79 | 29.84 | 85.95 | 7.17 | 28.22 | 97.24 | 4.02 |
| | 0.50 | 33.61 | 80.64 | 5.52 | 31.84 | 88.47 | 8.37 | 30.71 | 97.67 | 4.50 |
| | 0.75 | 33.87 | 82.26 | 5.90 | 32.22 | 89.37 | 9.01 | 31.19 | 97.79 | 4.88 |
| *D–Incr* | 5 | 36.25 | 65.41 | 3.97 | 33.03 | 77.54 | 5.98 | 30.79 | 94.24 | 3.44 |
| | 10 | 33.51 | 80.24 | 5.06 | 31.44 | 89.38 | 7.59 | 30.09 | 98.55 | 4.30 |
| | 15 | 32.32 | 85.31 | 5.93 | 30.71 | 91.78 | 8.79 | 29.75 | 98.71 | 4.72 |
| | 20 | 31.64 | 88.05 | 6.65 | 30.37 | 93.02 | 10.37 | 29.60 | 98.76 | 5.39 |
| *PMP* | 0.33 | 32.47 | 74.59 | 6.63 | 30.57 | 82.50 | 10.54 | 29.64 | 94.86 | 5.58 |
| | 0.50 | 33.59 | 80.95 | 5.41 | 31.65 | 89.15 | 8.30 | 30.19 | 98.72 | 4.35 |
| | 0.67 | 33.58 | 83.73 | 4.16 | 31.67 | 92.15 | 5.71 | 30.29 | 99.12 | 3.46 |
| *CFD* | 0.33 | 49.21 | 64.10 | 4.82 | 45.59 | 73.26 | 7.63 | 40.33 | 94.42 | 4.01 |
| | 0.50 | 31.19 | 85.86 | 4.71 | 29.34 | 94.34 | 6.44 | 28.47 | 99.12 | 3.84 |
| | 0.67 | 23.74 | 89.29 | 6.67 | 22.39 | 96.19 | 10.48 | 21.83 | 99.16 | 5.54 |
| *CC* | 0.25 | 30.88 | 93.54 | 5.54 | 30.15 | 96.87 | 8.26 | 30.13 | 98.54 | 4.58 |
| | 0.50 | 33.82 | 77.82 | 5.50 | 31.35 | 87.51 | 8.46 | 29.98 | 97.54 | 4.54 |
| | 0.75 | 35.82 | 67.90 | 5.16 | 32.70 | 79.41 | 7.83 | 30.03 | 96.62 | 4.27 |
| *Overall* | | 33.24 | 79.75 | 5.40 | 31.32 | 87.93 | 8.18 | 30.05 | 97.57 | 4.46 |

**Table 14:** Best results CRCPSPDC.

## 6.4 Managerial insights

In order to evaluate the different trade–offs for the CRCPSPDC (section 3.2), we generate additional project data with the network generator RanGen (Demeulemeester et al., 2003). The focus lies on testing the impact of the *OS*, *RC* and *CC* values on the project NPV and capital feasibility in more detail. An overview of the data parameters is given in table 15, with a total of 32,805 instances (= 5 instances x 9 *OS* levels x 9 *RC* levels x 9 *CC* levels x 3 *PMP* levels x 3 *CFD* levels), applicable to each of the three models. The cash flow data, with three levels for the *PMP* and *CFD* factors, is generated similarly as done in section 6.2. The additional instances generated are available online at *www.projectmanagement.ugent.be*, along with the cash flow data and reported results for each instance.

We have applied an ANOVA to determine the effect of *OS*, *RC* and *CC* on the average project NPV ($AvNPV$) for the three models. All single–factor and two–factor cross effects have a p–value lower than 0.001, except for the combination of *RC* and *CC* for all models (p–values of nearly 1) and the cross effect of *OS* and *CC* for model 3 (p–value of 0.264). The three–factor term ($OS$*$RC$*$CC$) obtained a p–value of 1 for the three models and is not considered. Based on significant p–values for the single– and most of the two–factor effects, it is worthwhile to investigate the impact of the factors *OS*, *RC* and *CC* further. We continue the remainder of the analysis with model 2,

| **Parameter** | **Values** |
|---|---|
| Number of activities ($Act$) | 30 |
| Resourse usage ($RU$) | 4 |
| Deadline increase ($D\text{–}Incr$) | 10 |
| Profit margin percentage ($PMP$) | 0.33, 0.50 or 0.67 |
| Cash flow distribution ($CFD$) | 0.33, 0.50 or 0.67 |
| Order strength ($OS$) | 0.10 to 0.90 in steps of 0.10 |
| Resource constrainedness ($RC$) | 0.10 to 0.90 in steps of 0.10 |
| Capital constrainedness ($CC$) | 0.10 to 0.90 in steps of 0.10 |

**Table 15:** Parameter settings insights.

since this model is a middle ground in the general model of section 3. Furthermore, aside from the aforementioned difference for model 3, the same conclusions apply to all three models, and by extension to the general model. We analyze the impact of the network structure ($OS$), the resource availability ($RC$), and the capital availability ($CC$) on the project NPV ($AvNPV$) and on the capital feasibility ($\%C$). The latter can be seen as a measure for the required capital management by the contractor, since a higher (lower) feasibility implies that it is easier (harder) to schedule the project given the available capital. Typically, a higher NPV and higher capital feasibility are preferable for the contractor, because the project then has a greater contribution to the company profits and requires a smaller focus on the optimization of the available capital.

Figure 9 displays the single–factor effects of $OS$, $RC$ and $CC$ on $\%C$ and $AvNPV$, whereas figure 10 shows the two–factor cross effects. The following guidelines can be stated for the contractor:

- **Network**
  - The contractor can on average increase the project NPV and reduce the need for capital management by making a project more parallel (lower $OS$ value). This way, cash inflows can be received earlier, resulting in a higher NPV and a lower initial capital requirement (figure 9).
  - In more parallel projects the effect of the renewable resource availability plays a greater role. More capital management is on average required in case the project has a high resource availability (lower $RC$ value), whereas the contractor should pay more attention to the project NPV if fewer resource units can be used (top two graphs of figure 10).
  - In more serial projects with a relatively high initial capital (low $CC$ value) the contractor should focus on properly optimizing the project NPV, whereas decreases in capital availability warrant on average a greater focus on capital management (middle two graphs of figure 10).
- **Resource availability**
  - A high resource availability reflects favorably on the project NPV but requires more capital management, whereas the opposite holds true for a low resource availability (figure 9).
  - Capital management is particularly crucial in case of a low initial capital and a large availability of renewable resources (bottom two graphs of figure 10).
- **Capital availability**

- Lowering the initial capital mainly results in a higher degree of capital management, but has on average little effect on the project NPV (figure 9).
- Increasing the initial capital decreases the project NPV on average, since less profitable projects can then be executed. The major question for the contractor then becomes whether the resulting project NPV is worth the required capital investment (figure 9).

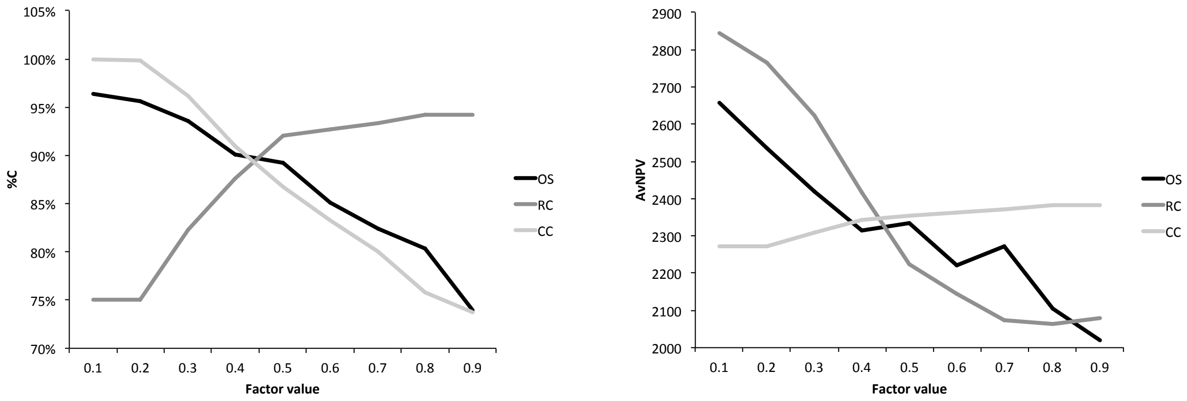

**Figure 9:** Impact of single–factor effects $OS$, $RC$ and $CC$ on $\%C$ and $AvNPV$ (model 2).

# 7 Conclusions & future research

In this paper, we have presented new scheduling techniques for the capital–constrained project scheduling problem with discounted cash flows (CCPSPDC) and for the capital– and resource–constrained project scheduling problem with discounted cash flows (CRCPSPDC). The objective is to schedule the project activities to maximize the project net present value (NPV). Capital constraints impose that the sum of the cash inflows received and cash outflows paid, given an initial capital, cannot be negative at any point in time during the project duration. We apply three cash flow models, as part of a general model for the distribution of cash outflows, to both problems.

In order to solve capital shortages, activities can be delayed together with the set of succeeding activities. The distinction is made between two types of delays to reduce the capital shortage. The first type delays sets of activities such that cash outflows leading to a capital shortage are compensated by cash inflows of other activities. The second type delays activities until a succeeding activity is reached, in which case later iterations of the procedure allow for additional delays. Once the capital shortages have been solved, activity move rules are applied, which improve the project NPV by delaying sets of activities with a negative cumulative NPV. The proposed scheduling techniques have been implemented as part of three metaheuristics from literature, along with two penalty functions, one to improve deadline feasibility and another to improve capital feasibility.

Extensive computational experiments have shown the added value of the delay rules for both the capital feasibility improvement and for the NPV improvement. The best results for the CCPSPDC have been favorably compared with a composite method based on literature.

As a first future research avenue, the proposed models and scheduling techniques can be applied in a multi–mode context. In doing so, the contractor can decide between different modes for each

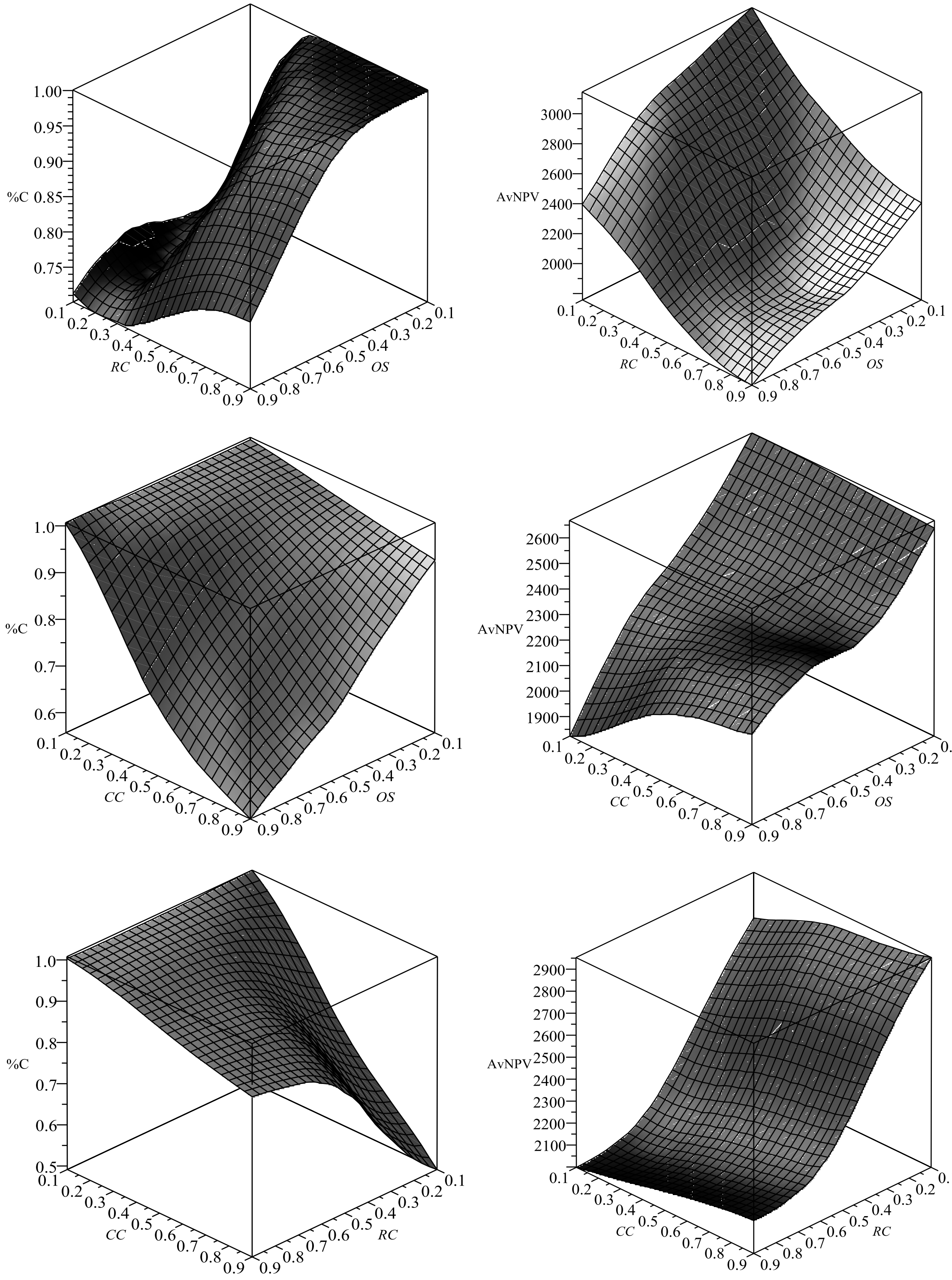

**Figure 10:** Impact of two–factor cross effects $OS$, $RC$ and $CC$ on $\%C$ and $AvNPV$ (model 2).

activity, and this allows for the inclusion of different applications per activity of the general model for cash outflows.

Second, whereas the focus of this manuscript has been on the cash outflows and capital restrictions, it may be interesting to extend these models to include different scenarios for the cash inflows as well (table 1).

Finally, including a client–contractor negotiation process would allow for the determination of the timing and size of the cash inflows based on interactions between both parties. Furthermore, the timing and size of cash outflows can be linked to the negotiation with subcontractors. This way, the entire interaction process between the three parties (i.e. client, contractor and subcontractor) can be integrated and optimized.

# A Examples of capital feasibility improvement with renewable resources

In this appendix, we illustrate the capital feasibility improvement method of section 4.2.2 with two examples. A.1 shows the network– and schedule–based variants of the improvement method, whereas A.2 details the added value of delaying activities with a negative NPV for the capital feasibility.

## A.1

**Example data:** the example of figure 1 is used to illustrate the concepts of the capital feasibility improvement method for model 3. We make use of a single RR with availability of 5, assume a deadline of 19, and an initial capital $C_0$ of 20. The discount rate is set to 1%. The initial schedule of the example is constructed based on the PL (1, 3, 5, 2, 4, 6, 7, 10, 9, 8, 11, 12, 13). Activity 4 is delayed by 1 time unit due to its negative net cash flow. The resulting initial schedule can be found on the right in figure 1, with the time on the horizontal axis, the RR level $a_1$ on the left vertical axis and the capital $C_t$ at time $t$ on the right vertical axis. The bold line indicates the available capital during the project duration, with decreases and increases corresponding with cash out– and inflows respectively. The capital at time 0 equals the initial capital of 20, whereas the capital at each time $t$ is equal to the sum of $C_0$ and the net cash flows of all activities completed no later than time $t$. E.g. $C_5$ = 20 ($C_0$) - 15 ($c_{2,net}$) - 25 ($c_{4,net}$) + 5 ($c_{5,net}$) = -15. The capital at time 19 is the sum of $C_0$ and the cash flows of all activities and equals 60. Based on this schedule it is clear that a capital shortage exists of 15 at times 5 and 6, which leads to an $ECR$ value equal to 30.

**Network–based moves:** the network–based capital feasibility improvement is first applied. The *Check time t* part leads to an *earlySet* consisting of activities 4 and 2 at time 5. Activity 4 occurs last in the PL, so this activity is used first for the *Set$_i$ calculations* part. The corresponding set $set_4$ consists of activities 4 and 7. The earliest cash inflow to occur after $f_4$ = 5 is that of activity 6 at time 8. As such, $\Delta_{4,next}$ equals 3 (= 8 - 5). $\Delta_{4,max}$ is determined by the successors not in $set_4$ of both activities 4 and 7. Only activity 10 meets these criteria and with its finish time of 11 this leads to $\Delta_{4,max}$ being equal to 2. Since $\Delta_{4,max}$ is smaller than $\Delta_{4,next}$, the *Delay until next successor* part of the capital feasibility improvement method is used next. The RR availability is evaluated for a delay of $\Delta_{4,max}$ by $set_4$, which is feasible. Hence, both activities 4 and 7 are delayed by the *Schedule set$_i$* part with $\Delta_{4,max}$ time units, and the capital and RR levels are updated. Finally, the *Schedule* $set_i$ part checks whether any activities with a negative NPV can be delayed, but this is not the case.

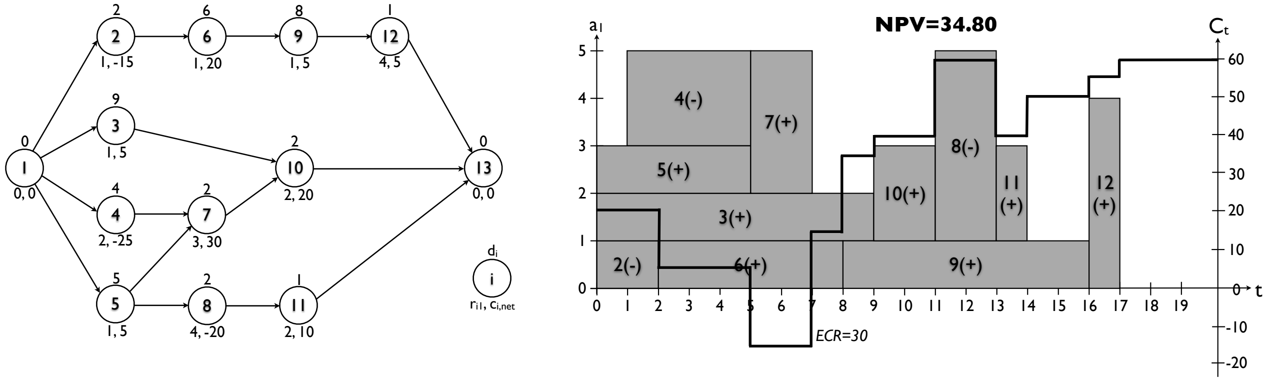

**Figure 1:** Data and initial schedule example 1.

The procedure returns to the *Check time t* part and once again evaluates the capital feasibility at time 5, based on the left schedule of figure 2. Since the capital level at time 5 is positive $t$ is incremented to 6, at which the available capital is also non–negative. The procedure continues to time 7, which has a negative capital so *earlySet* is constructed and consists of activities 4 and 2. $Set_i$ *calculations* is applied for activity 4 and $set_4$ consists of activities 4, 7 and 10. $\Delta_{4,next}$ is calculated based on activity 6 and equals 1. $\Delta_{4,max}$ equals 8 since the only successor not in $set_4$ of any of the three activities is the end dummy. *Delay until next positive* is applied but the method cannot find any RR feasible delay for $set_4$. As such the capital feasibility improvement method backtracks to *earlySet* in the *Check time t* part. $Set_i$ *calculations* is done for activity 2 which results in $set_2$ consisting of activities 2, 6, 9 and 12. $\Delta_{2,next}$ is determined based on activity 6 and equals 6, whereas $\Delta_{2,max}$ is 2 due to the project deadline. The *Delay until next successor* part is applied, but no feasible delays can be found. The procedure returns to *earlySet*, but all activities in the set have been considered, so the capital feasibility improvement method terminates since the capital shortage at time 7 could not be solved. The resulting schedule after the network–based moves of the capital feasibility improvement method have been applied is shown in the left schedule of figure 2. The capital shortage has both been reduced (*ECR* equal to 15 instead of 30) and moved later in time, but the project NPV has decreased due to the delay of a set of activities with a positive cumulative NPV (activities 4 and 7).

**Schedule–based moves:** the schedule–based variant of the capital feasibility improvement method is started based on the left schedule of figure 2. For the capital shortage at time 7, *earlySet* consists of activities 4 and 2. $Set_i$ *calculations* is first applied for activity 4 since this activity appears after activity 2 in the PL. The set $set_4$ holds activities 4, 7, 10, 8 and 11, the last two of which would never have been included in the network–based version of the improvement method. $\Delta_{4,next}$ equals 1 based on the positive cash flow of activity 6 and $\Delta_{4,max}$ is 5 because of the end dummy activity 13 as successor of activity 11. Since $\Delta_{4,next}$ is smaller than $\Delta_{4,max}$, the procedure continues with the *Delay until next positive* part. A delay of 1 time unit is RR feasible for $set_4$, so the *Schedule* $set_i$ part is used next. The activity finish times for all activities in $set_4$ are updated, as are the capital and RR levels. No activities with a negative NPV can be delayed as part of *Schedule* $set_i$. The procedure returns to part *Check time t*, but no capital shortages remain so a

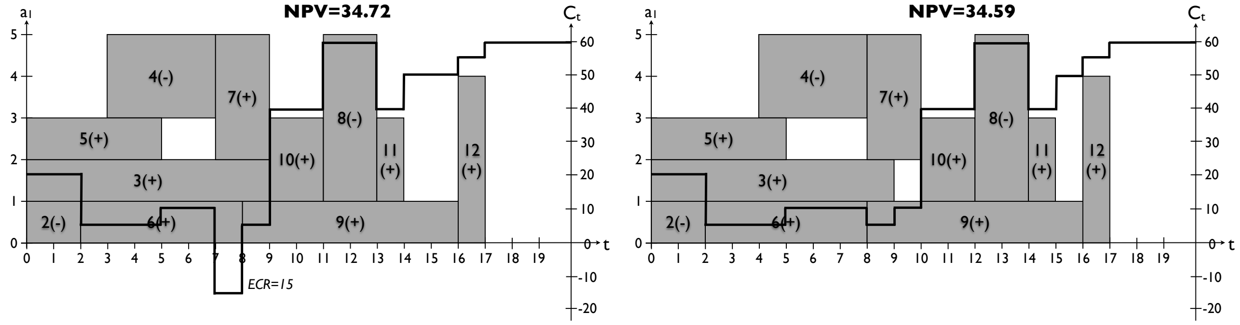

**Figure 2:** Schedules example 1 after network– & schedule–based capital feasibility improvement.

*C–Feas* schedule has been found and the method terminates. The schedule after both variants of the capital feasibility improvement method have been applied can be found in the right schedule of figure 2.

**Solution representation:** recall that the PL of the schedule in figure 1 was (1, 3, 5, 2, 4, 6, 7, 10, 9, 8, 11, 12, 13). Based on the right schedule of figure 2, the updated list, or TO, becomes (1, 2, 5, 6, 4, 3, 7, 10, 8, 11, 9, 12, 13), with the tie between activities 4 and 6 broken randomly in favor of activity 6.

**Alternative:** assume that the schedule–based variant of the capital feasibility improvement is first applied for the same example. We again start from the schedule in figure 1. Activities 4 and 7 are first delayed by 2 time units in the same manner as done before. The available capital at time 7 is, however, negative and *earlySet* consists of activities 4 and 2. Unlike for the network–based improvement, a delay of activity 4, along with activities 7, 8, 10 and 11 is now possible. This delay of 1 time unit would not have been considered in the network–based step. The resulting schedule is *C–Feas* and is displayed in figure 3. It is important to consider that the schedule–based moves are more efficient for the example, since fewer moves are required to obtain a *C–Feas* schedule. Our results of section 6.2 confirm that it is indeed best to first apply the schedule–based variant of the capital feasibility improvement method.

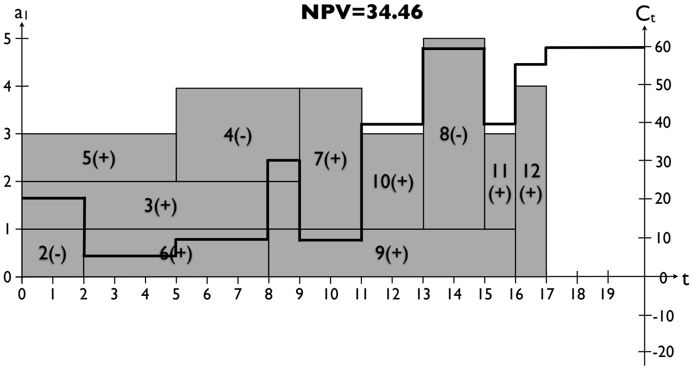

**Figure 3:** Alternative schedule example 1.

## A.2

In step 4 of section 4.2.1 and the *Schedule* $set_i$ part of section 4.2.2, we stressed the need for delaying single activities with a negative NPV as much as possible given both the precedence relations and the RR constraints. Let us illustrate the reasoning behind these delays with the simple example for model 3 shown in figure 4. The project has a RR availability of 4, a deadline of 14 and an initial capital $C_0$ of 10. The PL used is (1, 3, 2, 4, 5, 6), and the discount rate is 1%. The top right schedule in figure 4 shows the initial schedule of the example with the omission of step 4 (i.e. the delaying of activities with a negative NPV). We observe a capital shortage of 5 at times 5 and 6.

We apply the capital feasibility improvement method and start with the schedule–based variant. We aim to solve the capital conflict at time 5 and can use both activities 2 and 3. $set_2$ only contains activity 2 but cannot be scheduled later than time 5, so a delay of $set_2$ cannot solve the capital shortage. $set_3$ on the contrary contains activities 3, 4, 5 and 6 and can be delayed to time 6 beyond the conflict at time 5. Additional delays until time 8 are, however, required for the set, due to capital shortages. Delaying activity 2 cannot solve any of these conflicts, so no change with respect to activity 2 occurs. No further delays are possible because of the project deadline of 14. Applying the network–based variant of the capital feasibility improvement method can also not solve the capital conflicts. The resulting final infeasible schedule is shown on the bottom left of figure 4.

If activity 2 is on the contrary delayed to time 5, $set_2$ contains both activities 2 and 4 at time 5. Both activities are delayed by 2 time units ($\Delta_{2,next} = 4$, $\Delta_{2,max} = 2$) and the capital shortage at time 5 is solved. Now a shortage at time 7 occurs but this can be solved by delaying $set_2$ ($= \{2, 4, 6\}$) by 2 time units. The resulting and optimal schedule is *C–Feas* and can be found on the bottom right of figure 4.

Based on this example, we can conclude that the delays of single activities with a negative NPV are indeed necessary since otherwise we may not be able to solve some capital shortages, as is the case in the bottom left schedule of figure 4. These conclusions are furthermore verified in our computational experiments in section 6.2.

# Acknowledgements

We acknowledge the support provided by the ”Bijzonder Onderzoeksfonds” (BOF) and by the National Bank of Belgium (NBB) for the project with contract number BOF12GOA021.

# References

Blazewicz, J., Lenstra, J., and Rinnooy Kan, A. (1983). Scheduling subject to resource constraints: notation, classification and complexity. *Discrete Applied Mathematics*, 5:11–24.

Dayanand, N. and Padman, R. (1997). On modelling payments in project scheduling. *Management Science*, 48(9):906–918.

Dayanand, N. and Padman, R. (2001a). Project contracts and payment schedules: the client's problem. *Management Science*, 47(12):1654–1667.

Dayanand, N. and Padman, R. (2001b). A two stage heuristic for scheduling payments in projects. *Annals of Operations Research*, 102:197–220.

Debels, D., De Reyck, B., Leus, R., and Vanhoucke, M. (2006). A hybrid scatter search/electromagnetism meta–heuristic for project scheduling. *European Journal of Operational Research*, 169:638–653.

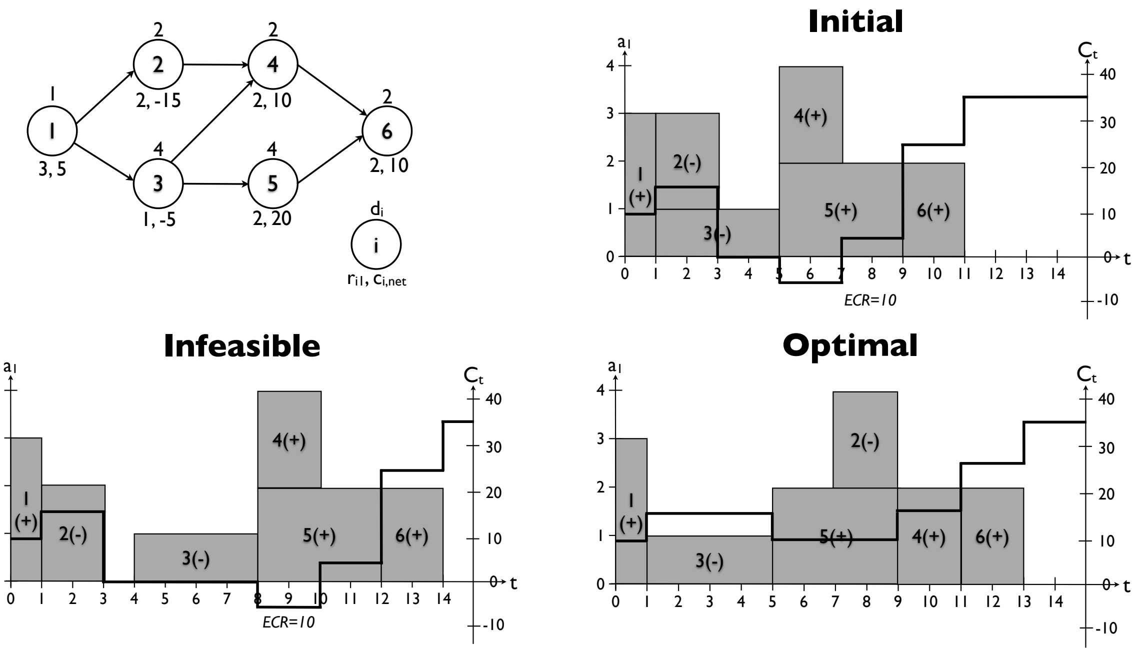

**Figure 4:** Data and schedules example 2.

Demeulemeester, E., Vanhoucke, M., and Herroelen, W. (2003). Rangen: A random network generator for activity-on-the-node networks. *Journal of Scheduling*, 6:17–38.

Doersch, R. and Patterson, J. (1977). Scheduling a project to maximize its net present value: A zero-one programming approach. *Management Science*, 23(8):882–889.

Etgar, R. and Shtub, A. (1999). Scheduling project activities to maximize the net present value - the case of linear time-dependant cash flows. *International Journal of Production Research*, 37(2):329–339.

Gendreau, M. and Potvin, J.-Y. (2010). Tabu search. In Gendreau, M. and Potvin, J.-Y., editors, *Handbook of metaheuristics*, pages 41–59. Springer.

He, Z., Liu, R., and Jia, T. (2012). Metaheuristics for multi-mode capital-constrained project payment scheduling. *European Journal of Operational Research*, 223:605–613.

He, Z., Liu, R., and Xu, Y. (2009a). Client perspective based multimode project payment scheduling problem and its heuristic algorithm. *Systems Engineering – Theory & Practice*, 29(2):70–77.

He, Z., Wang, N., Jia, T., and Xu, Y. (2009b). Simulated annealing and tabu search for multi–mode project payment scheduling. *European Journal of Operational Research*, 198:688–696.

He, Z., Wang, N., and Li, P. (2014). Simulated annealing for financing cost distribution based project payment scheduling from a joint perspective. *Annals of Operations Research*, 213:203–220.

He, Z. and Xu, Y. (2008). Multi–mode project payment scheduling problem with bonus–penalty structure. *European Journal of Operational Research*, 189:1191–1207.

Herroelen, W., Van Dommelen, P., and Demeulemeester, E. (1997). Project networks with discounted cash flows: A guided tour through recent developments. *European Journal of Operational Research*, 100:97–121.

Kelley, J. J. (1963). The critical-path method: Resources planning and scheduling. In Muth, J. and Thompson, J., editors, *Industrial Scheduling*, pages 347–365. Prentice-Hall, New Jersey.

Leyman, P. and Vanhoucke, M. (2015). A new scheduling technique for the resource–constrained project scheduling problem with discounted cash flows. *International Journal of Production Research*, 53(9):2771–2786.

Leyman, P. and Vanhoucke, M. (2016). Payment models and net present value optimization for resource-constrained project scheduling. *Computers & Industrial Engineering*, 91:139–153.

Li, K. and Willis, R. (1992). An interative scheduling technique for resource-constrained project scheduling. *European Journal of Operational Research*, 56:370–379.

Lova, A., Tormos, P., Cervantes, M., and Barber, F. (2009). An efficient hybrid genetic algorithm for scheduling projects with resource constraints and multiple execution modes. *International Journal of Production Economics*, 117:302–316.

Martí, R., Laguna, M., and Glover, F. (2006). Principles of scatter search. *European Journal of Operational Research*, 169:359–372.

Neumann, K. and Schwindt, C. (2002). Project scheduling with inventory constraints. *Mathematical Methods of Operations Research*, 56:513–533.

Özdamar, L. (1998). On scheduling project activities with variable expenditure rates. *IIE Transactions*, 30:695–704.

Özdamar, L. and Dündar, H. (1997). A flexible heuristic for a multi–mode capital constrained project scheduling problem with probabilistic cash inflows. *Computers & Operations Research*, 24(12):1187–1200.

Patterson, J. (1976). Project scheduling: The effects of problem structure on heuristic scheduling. *Naval Research Logistics*, 23:95–123.

Pritsker, A., Watters, L., and Wolfe, P. (1969). Multiproject scheduling with limited resources: A zero-one programming approach. *Management Science*, 16(1):93–108.

Reeves, C. (2010). Genetic algorithms. In Gendreau, M. and Potvin, J.-Y., editors, *Handbook of metaheuristics*, pages 109–139. Springer.

Resende, M., Ribeiro, C., Glover, F., and Martí, R. (2010). Scatter search and path–relinking: Fundamentels, advances, and applications. In Gendreau, M. and Potvin, J.-Y., editors, *Handbook of metaheuristics*, pages 87–107. Springer.

Russell, A. (1970). Cash flows in networks. *Management Science*, 16:357–373.

Schwindt, C. and Zimmermann, J. (2001). A steepest ascent approach to maximizing the net present value of projects. *Mathematical Methods of Operations Research*, 53:435–450.

Shtub, A. and Etgar, R. (1997). A branch and bound algorithm for scheduling projects to maximize net present value: the case of time dependent, contingent cash flows. *International Journal of Production Research*, 35(12):3367–3378.

Smith-Daniels, D., Padman, R., and Smith-Daniels, V. (1996). Heuristic scheduling of capital constrained projects. *Journal of Operations Management*, 14:241–254.

Smith-Daniels, D. and Smith-Daniels, V. (1987). Maximizing the net present value of a project subject to materials and capital constraints. *Journal of Operations Management*, 7(1-2):33–45.

Sung, C. and Lim, S. (1994). A project activity scheduling problem with net present value measure. *International Journal of Production Economics*, 37:177–187.

Valls, V., Ballestín, F., and Quintanilla, S. (2004). A population–based approach to the resource–constrained project scheduling problem. *Annals of Operations Research*, 131:305–324.

Valls, V., Quintanilla, S., and Ballestín, F. (2003). Resource–constrained project scheduling: A critical activity reordering heuristic. *European Journal of Operational Research*, 149:282–301.

Van Peteghem, V. and Vanhoucke, M. (2011). Using resource scarceness characteristics to solve the multi-mode resource-constrained project scheduling problem. *Journal of Heuristics*, 17:705–728.

Vanhoucke, M. (2010). A scatter search procedure for maximizing the net present value of a resource-constrained project with fixed activity cash flow. *International Journal of Production Research*, 48(7):1983–2001.

Vanhoucke, M. and Debels, D. (2007). The discrete time/cost trade–off problem: extensions and heuristic procedures. *Journal of Scheduling*, 10:311–326.

Vanhoucke, M., Demeulemeester, E., and Herroelen, W. (2001a). Maximizing the net present value of a project with linear time-dependant cash flows. *International Journal of Production Research*, 39:3159–3181.

Vanhoucke, M., Demeulemeester, E., and Herroelen, W. (2001b). On maximizing the net present value of a project under renewable resource constraints. *Management Science*, 47(8):1113–1121.

Vanhoucke, M., Demeulemeester, E., and Herroelen, W. (2003). Progress payments in project scheduling problems. *European Journal of Operational Research*, 148:604–620.