Asia-Pacific Journal of Operational Research
Vol. 38, No. 4 (2021) 2050057 (28 pages)

DOI: 10.1142/S0217595920500578

Maximizing the Net Present Value in Project Scheduling Under Periodic Inflation

Mahboobe Peymankar
Department of Industrial Engineering Faculty of Engineering
Ferdowsi University of Mashhad Mashhad, Iran
peymankar@mail.um.ac.ir

Mohammad Ranjbar*
Department of Industrial Engineering Faculty of Engineering
Ferdowsi University of Mashhad Mashhad, Iran
m_ranjbar@um.ac.ir

Received 20 December 2019
Accepted 27 September 2020
Published 25 March 2021

We investigate a project scheduling problem in which cash flows are periodically variable, and we aim to maximize the net present value (NPV). For each activity, a set of cash flows is considered where each one pertains to a particular period. This setting is compatible with inflation rates that may well occur in some countries with unstable economic situations where the occurrence times of inflation are sometimes known in advance. In this case, the project can be scheduled more suitably to abate probable pitfalls. In this paper, we investigate the problem in two settings: deterministic and stochastic cash flows. In the deterministic case, in each period we consider the expected value of cash flows of each activity as a constant and develop an integer linear programming model in conjunction with a branch-and-bound algorithm to solve the problem optimally. Moreover, we develop a multi-stage stochastic programming (MSSP) model to formulate the stochastic version of the problem. Using a set of randomly generated test instances and extensive computational results, we analyze the performance of our developed solution approaches. In addition, we compare the deterministic and stochastic models and analyze the sensitivity of the most important parameters.

Keywords: Project scheduling; net present value; branch-and-bound algorithm; multi-stage stochastic programming.

1. Introduction

One of the significant goals in the field of project management and scheduling is the maximization of project-based organizations' profit, commonly expressed using the net present value (NPV) criterion. In the literature, a project usually refers to a set

*Corresponding author.

of precedence-related activities where each one is associated with a duration and a cash flow. In this project, activities should be scheduled in such a way that the NPV is maximized. This goal would appear to be a major challenge because cash flows of many real and long-lasting construction projects may change over time. Inflation frequently happens in the prices of the project resources and this is one of the integral parts of cost overruns. Mahamid and Dmaidi (2013) and Prajapati *et al.* (2016) indicated that the top three important factors leading to cost overruns especially in big construction projects are political situation, fluctuation of materials price and economic instability.

In some particular cases, the price of project resources increases greatly and sharply which seems to be a serious challenge for project managers. For example, the US government withdrew from the Iran nuclear deal on November 5 of 2018 and announced it six months beforehand. This led to a high inflation rate in Iran such that the home price tripled during six months after sanction imposition. Moreover, several more economic sanctions have been imposed on Iran during recent decades that all had been notified in advance (Wikipedia: Sanctions against Iran).

Moreover, sudden changes in climate, e.g., the temperature or precipitation, will affect natural gas, crude oil, crops and electricity prices. Besides, the central bank's interest rate policies change asset prices, e.g., gold, currency, stock and house prices. Furthermore, a technological revolution such as shale or electric vehicle affects the price of related commodities, e.g., crude oil. Although economic fundamentals are typically hard to predict, there are significantly related cases that have been recognized, e.g., the 2008 recession that was predicted by some people (Breuss, 2018; Nyberg, 2010; Obstfeld *et al.*, 2009). They were alleged to have been making failed predictions, whereas they were right.

Accordingly, we can assume that periods and their corresponding changes in cash flows are known in advance. Represented as new information about project costs, this change may cause rescheduling the project to rise to new challenges. Consequently, we can improve the NPV by planning a better baseline schedule for the project activities. In thispaper, we assume that the execution phase of a project can be divided into a few time periods where cash flows and discount rates might increase from a period to its immediate next one due to economic inflation. We consider two settings to formulate the problem: deterministic and stochastic. In the deterministic case, we ignore the variance of cash flows of each activity in each period, and we consider its expected value as a constant. In contrast, in the stochastic case probable cash flows are factored in under a set of scenarios. We use a reactive scheduling policy that revises an existing schedule in the light of a new inflation rate. To do so, the problem is formulated through a stochastic programming model.

The contributions of this paper are threefold: (1) we formulate the project scheduling problem with periodic inflation under deterministic setting as an integer linear programming model; (2) we develop a branch-and-bound (B&B) algorithm for the deterministic case, including efficient lower and upper bounds; and (3) We

develop a multi-stage stochastic programming (MSSP) model to formulate the problem in the stochastic setting.

The remainder of this paper is organized as follows. Related work is discussed in Sec. 2. Section 3 includes our developments in the deterministic case including problem formulation, an introductory example, and a B&B algorithm. Our stochastic programming model is developed in Sec. 4. In Sec. 5, computational results are discussed to show the competencies of our solution approaches. Finally, a summary and outlook on future research are given in Sec. 6.

2. Related Work

There are numerous papers in the field of project scheduling in the literature (see Demeulemeester and Herroelen, 2006). One of the well-known problems in this field is the resource-unconstrained project scheduling problem for maximizing the net present value (NPV), shown as *max-npv*. In this problem, activities are scheduled subject to temporal precedence relations and a predetermined project deadline. The concept of the NPV in project scheduling, as a project financial index, was firstly introduced by Battersby (1967). The nonlinear problem presented by Russell (1986) was transformed into the equivalent linear program and represented as an efficient procedure by Grinold (1972). Kamburowski (1990) proposed another procedure for maximizing the NPV in the generalized network according to Grinold's approach. We refer the interested readers to Herroelen *et al.* (1999), where the various problems in this field were classified.

Demeulemeester (1996) developed a recursive search algorithm performing on a partial tree structure in which the precedence constraints are satisfied and positive cash flows are scheduled as early as possible whereas negative cash flows are scheduled as late as possible. After that, De Reyck (1996) developed an optimal scheduling procedure for maximizing the NPV by considering unconstrained resources and generalized precedence relations. Schwindt and Zimmermann (2001) designed a steepest ascent approach for maximizing the NPV of projects. Vanhoucke (2006) employed the ideas presented in Demeulemeester (1996), De Reyck (1996) and Schwindt and Zimmermann (2001) to develop a hybrid recursive search algorithm. He aimed to maximize the NPV of a project under generalized precedence relations, shown as *max-npv-gpr*. So far, this algorithm has been the most efficient solution approach for *max-npv* and *max-npv-gpr* in the literature.

Although most of the research studies in the field of project scheduling aim to maximize the NPV have assumed that cash flows of activities are constant, Elmaghraby and Herroelen (1990) alluded the idea of time-dependent cash flows. They assumed that the cash flow of each activity is dependent on its finish time. Moreover, Etgar and Shtub (1997) considered incentive payments (penalties) for early (late) event occurrences and presumed that the costs of resources over time are time-dependent. To solve the problem, they developed a branch-and-bound (B&B) algorithm. Vanhoucke *et al.* (2001a) considered the unconstrained project

scheduling problem with discounted cash flows, where the net cash flows were assumed to be linearly dependent on the finish times of corresponding activities. They also developed a very efficient exact recursive search algorithm. This algorithm starts by constructing an early tree and continues with a current tree which is updated recursively in such a way that activities with negative (positive) cash flows are pushed towards the project deadline (starting time). In addition to this, this recursive search procedure was utilized by Vanhoucke *et al.* (2001b) in a B&B algorithm to find the optimal solution of the resource-constrained version of the problem. Finally, Vanhoucke *et al.* (2003) studied the unconstrained project scheduling problem with discounted cash flows. In that research, they assumed that the cash flow of each activity is dependent on its finish time. They also presumed that cash outflows occur when an activity is finished, whereas cash inflows are acquired as progress payments at the end of some periods.

Another related category of research papers to ours is project scheduling problems with uncertain parameters (Hazir and Ulusoy, 2019). The most frequent assumption is the stochastic duration of activities. Buss and Rosenblatt (1997), Tilson *et al.* (2006) and Sobel *et al.* (2009) considered the exponential distribution function for the duration of activities and adopted continues-time Markov chains to model the problem. Their methods are suitable for small projects because the number of states in Markov chain grows exponentially when the size of the project increases. Wiesemann *et al.* (2010) addressed this problem when the duration of activities and cash flows are described by discrete scenarios and employed the method developed by Benati (2006) to devise a B&B algorithm. Three models based on the chance-constrained model, the expected value model and the chance maximization model was developed by Zhao *et al.* (2016) to apply uncertainty in the problem. Then, an estimation of distribution algorithm was developed to solve the problem. Zheng *et al.* (2018) first used the time buffering method and proactive scheduling model to construct a robust schedule and then proposed two reactive scheduling models to adjust the developed baseline schedule. Moreover, three heuristic approaches based on tabu search algorithm and variable neighborhood search algorithm were designed to solve the problem. Liang *et al.* (2019) proposed the NPV and an expected penalty cost to measure the robustness quality and solution robustness when activity durations are uncertain. In addition to this, they develop a two-stage algorithm which integrates tabu search algorithm and simulated annealing algorithm. Their results indicated that the proactive project schedules with composite robustness to effectively protect the payment plan from disruptions by means of allocating appropriate time buffers and achieve a remarkable performance. There are a few articles in the literature that have studied the project scheduling problem with *max-npv* goal and time-dependent cash flows. In those research works, cash flows were often considered either decreasing or increasing over time based on a linear function or a curve. This function, which most often has been assumed to be linear, seems not to be applicable to our setting.

3. The Deterministic Case

3.1. *Problem description and modeling*

In this section, we describe the deterministic version of the problem as follows. Consider an activity-on-node (AON) network, represented by a graph $G = (N, A)$ with a set of nodes N, which defines the project activities, and a set of arcs A, which shows the temporal precedence relations among the activities. This network includes two dummy activities numbered 0 and $n+1$. The duration of each activity $i \in N$ is denoted by d_i where $d_0 = d_{n+1} = 0$, and the project must be completed by a given deadline (δ). The precedence relations among activities are represented in standard form, i.e., start-to-start precedence relations with a minimal time lag l_{ij} between activities $(i, j) \in A$. Each activity has a cash flow, where negative cash flows illustrate costs and positive ones correspond to benefits. As previously mentioned, economic changes and inflation may lead to noticeable upticks in the predicted cash flows and discount rates. We presume that the project horizon is divided into $r+1$ periods where cash flow of each activities increases from each period e to period $e+1$, but it is supposed to be constant within periods. As a result, we define r as the number of change points in cash flows over project time horizon, namely change point e indicates the exact border between periods $e-1$ and e. It is worthwhile to mention that durations of different periods are not necessarily equal. Furthermore, the objective of the problem is to maximize the NPV of the project. We also assume that activities are non-preemptive and all required resources are unconstrained. In Table 1, the parameters of the problem are described.

Regardless of the finish times of activities, it is assumed that the cash flow of each activity is determined at the start of the activity, but it is paid and incurred whenever the activity is finished. In other words, if activity i finishes at time t

Table 1. The defined parameters of the problem.

Parameter	Definition
$N = \{0, \ldots, n+1\}$	the set of activities with indices i and j
A	the set of precedence relations
Suc_i	the set of direct successors of activity i
Pred_i	the set of direct predecessors of activity i
d_i	duration of activity i
δ	project deadline
r	number of change points
$CP = \{cp_0 = 0, cp_1, \ldots, cp_r cp_{r+1} = \delta\}$	the set of change points
$P = \{1, 2, \ldots, r+1\}$	the set of periods with index e
c_{ie}	the cash flow of activity i when it starts in period e
l_{ij}	the start-to-start time lag between activities i and j
α_e	the discount rate of period e
$T = \{0, \ldots, \delta\}$	the set of times with index t
c_{it}^{npv}	the NPV of activity i if it finishes at time t
cpl	the critical path length of the project

where $t \in (cp_{e-1}, cp_e]$ and $t - d_i \in [cp_{e'-1}, cp_{e'})$, we have $c_{it}^{\text{npv}} = c_{ie'} \exp(-\alpha_e(t - cp_{e-1}) - [\sum_{f=1}^{e-1} \alpha_f (cp_f - cp_{f-1})])$. It should be noted that we consider fixed price contracts in which cost of each activity is determined at the beginning of that. This assumption makes the problem more challenging and signifies the role of scheduling. We also presume costs (benefits) are paid (received) at finish times. In other words, the cost of an activity only depends on its start time period and inflation rates in upcoming periods have no impact on it. However, we have to consider inflation rates in calculation of the NPV.

To formulate this problem, the following decision variable is used.

$$x_{it} = \begin{cases} 1; & \text{If activity } i \text{ finishes at time } t \\ 0; & \text{Otherwise.} \end{cases}$$

In the following, the deterministic model (DM) is presented for the deterministic version of the problem.

$$\text{DM}: \ \text{Max} \sum_{i \in N} \sum_{t=0}^{\delta} c_{it}^{\text{npv}} x_{it} \tag{1}$$

$$\text{s.t.} \ \sum_{t=0}^{\delta} x_{it} = 1 \quad \forall\, i \in N \tag{2}$$

$$\sum_{t=0}^{\delta} t x_{jt} - \sum_{t=0}^{\delta} t x_{it} \geq l_{ij} + d_j - d_i; \quad \forall\, (i,j) \in A \tag{3}$$

$$x_{it} \in \{0,1\}; \quad \forall\, i \in N, \forall\, t \in T \tag{4}$$

The maximization of the net presented value is presented in objective function (1). Constraints (2) enforce a unique finish time for each activity. Constraints (3) present the start-to-start minimal time lag precedence constraints. Finally, constraints (4) define the x_{it} as binary variables.

3.2. *An example of the deterministic case*

Consider the project network of Fig. 1, including five non-dummy activities and three periods. In this instance, each node indicates an activity, the above number shows its duration and three below numbers show the corresponding cash flows of three periods. As can be seen, cash flows embody an inflated circumstance. Moreover, the number associated with each arrow indicates the start-to-start minimal time lag between the beginning and ending activities of the arrow. We consider two change points at $cp_1 = 7$ and $cp_2 = 12$. The project must be finished by $\delta = 22$ and the discount rates in the three periods are assumed to be $\alpha_1 = 0.05$, $\alpha_2 = 0.055$ and $\alpha_3 = 0.06$, respectively.

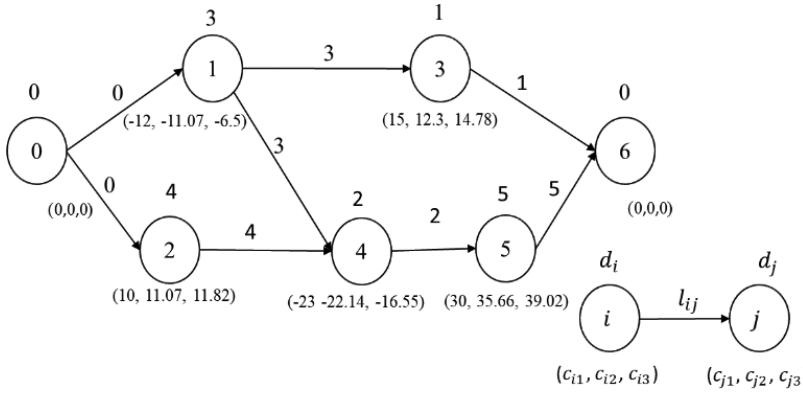

Fig. 1. Precedence network of the example project.

Table 2. Cash flows of different periods.

Activity (i)	c_{i1}	c_{ie}	
		$e = 2$	$e = 3$
1	−12	−11.07	−6.50
2	10	11.07	11.82
3	15	12.30	14.78
4	−23	−22.14	−16.55
5	30	35.66	39.02

Moreover, Table 2 presents new cash flows $c_{ie} = \frac{c_{i(e-1)}(1+\kappa_{ie})}{1+\mu_e}$ for each activity $i \in N$ and period $e \in P$, where κ_{ie} indicates the inflation rate of activity i in period e and $\mu_e = \frac{\sum_{i=1}^{n} \kappa_{ie}}{n}$.

To cope with this situation, one option is to dismiss both change points and inflated cash flows. In this case, the DM results in the schedule of Fig. 2 with an NPV of 10.4 cost units. Another option for dealing with the problem is to consider the first change point and ignore the other one. In this case, the output of the DM is the schedule of Fig. 3 with an NPV of 13.26 cost units.

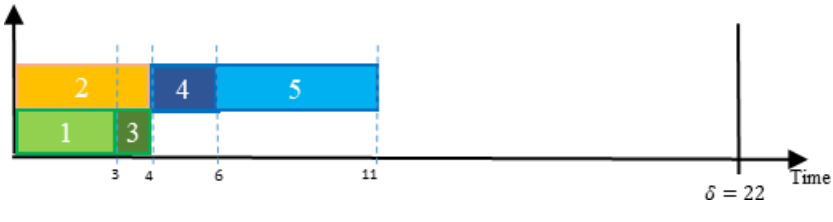

Fig. 2. The optimal schedule obtained regardless of change points.

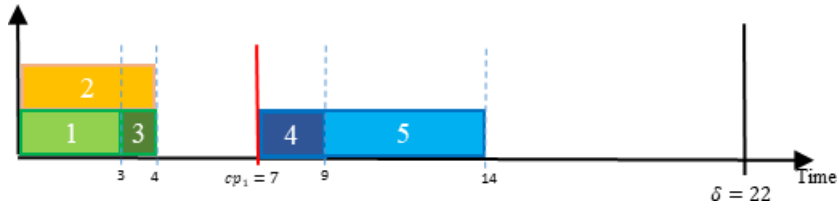

Fig. 3. The optimal schedule with one change point, i.e., $cp_1 = 7$.

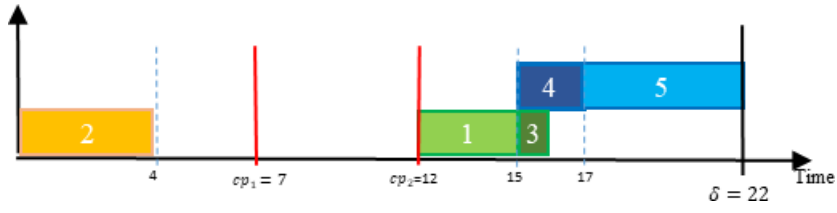

Fig. 4. The optimal schedule with two change points: $cp_1 = 7$ and $cp_2 = 12$.

Alternatively, we can schedule the activities by considering both change points that occur in time moments 7 and 12, respectively. By doing so, the DM gives us the schedule depicted in Fig. 4 and results in an NPV of 16.4 cost units.

3.3. *Sketch of the solution approach*

To solve optimally the deterministic version of the problem, we design a two-phase solution approach. In the first phase, the start period of each activity is determined using a B&B algorithm, and in the second phase, activities are scheduled subject to given periods. In the second phase, for each solution obtained from phase 1, we modify the project network using the network modification procedure, described in Sec. 3.3, and then apply the recursive search algorithm (RSA) developed by Vanhoucke (2006) to find the corresponding best solution.

3.4. *A branch-and-bound algorithm*

The search tree of our B&B algorithm has at most n levels wherein each level the start period of at least one activity is determined. Moreover, we employ a depth-first strategy to extend the search tree. Since the cash flows and durations of two dummy activities 0 and $n + 1$ are zero, the starting periods of these activities are periods 1 and $r + 1$, respectively. Furthermore, the earliest and latest feasible start periods of each real activity $j \in N$, shown as $P_j^{\min}, P_j^{\max}$ respectively, are determined according to the earliest and latest start times of the activity.

Each node in the search tree defines the assignment of (at least) one activity to one of its feasible start periods. In other words, the branches of each node denote

feasible time periods in which a particular activity can start. In some cases, determining the start period of an activity might result in fixing the start periods of some other precedence-related activities. We assume that activities are considered in the search tree based on the non-decreasing order of their number of (direct and indirect) successors (using the largest activity number as a tie-breaker).

We consider various types of precedence relations in the project network, i.e., finish-to-start, start-to-start, start-to-finish and finish-to-finish, with minimal time lags. Therefore, we first need to transform the project network into a standard form, including only start-to-start precedence relations with minimal time lags. After that, the modified label correcting algorithm, developed by Orlin *et al.* (1993) is utilized to calculate the earliest and the latest start time of each activity $i \in N$, shown as es_i and ls_i respectively.

The earliest and latest start times of the activities of the example project in conjunction with their feasible start periods are displayed in Table 3.

In branching phase, the start period of each activity i, shown as P_i, is initialized as $P_i = P_i^{\min}$. If s_i indicates the start time of activity i, it has to be in period $[cp_{P_i-1}, cp_{P_i})$. By fixing P_i, we know that $P_j \leq P_i; \forall\, j \in \text{Pred}_i$. It is worthwhile to note that for each pair of activities i and j, where j is a direct or indirect predecessor of i, if $P_j^{\min} = P_i$, then we must fix P_j as $P_j = P_i$. In the search tree, when the start periods of all activities are determined over a path from the root node to a leaf, we must use the backtracking operator to search all other possible solutions. In other words, the backtracking operator generates multiple nodes at the same level, branched from the same node in the upper level. Whenever the backtracking operator is utilized, a new start period is considered for the activity, and this new branch is created when $P_i + 1 \leq \min\{P_i^{\max}, \min_{j \in \text{Suc}_i}\{P_j\}\}$.

The search tree of the example project is depicted in Figs. 5(a) and 5(b). In each node, the numbers shown at the first and second rows denote the number of nodes and the number of activities whose start periods are fixed at the corresponding level, respectively. This should be noted that the activity index written on the left side indicates the main considered activity and others, given at the right side, show the predecessors of the main one. Moreover, periods of activities determined in the second row are shown in the last row.

Considering the example project, we first choose activity 5 to determine its start period. As can be seen in Fig. 5(a), this activity starts in its first feasible period, i.e.,

Table 3. The earliest and latest start times and feasible start periods.

Activity (i)	es_i	ls_i	$P_i^{\min}$	$P_i^{\max}$
1	0	12	1	3
2	0	11	1	2
3	3	21	1	3
4	4	15	1	3
5	6	17	1	3

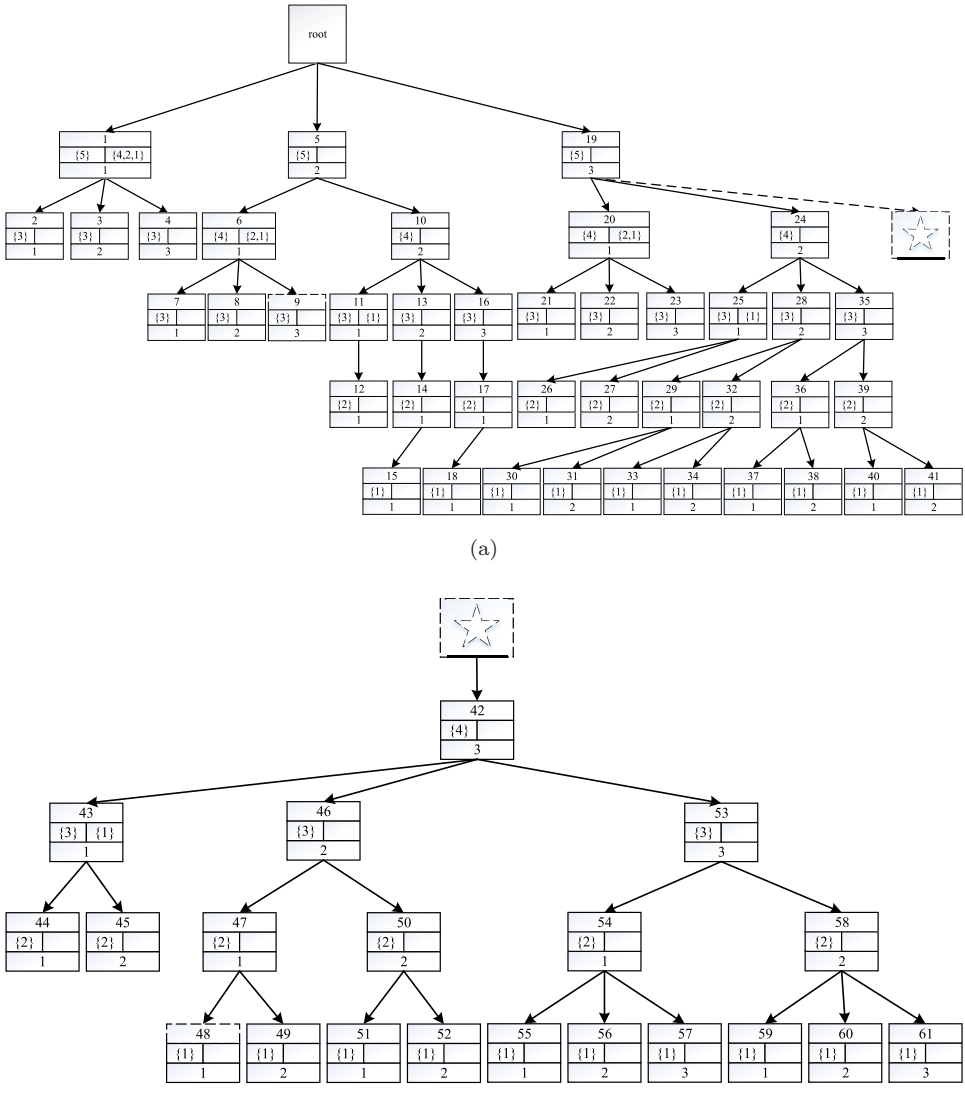

Fig. 5. (a) The search tree of the example project-part 1, (b) The search tree of the example project-part 2.

$P_5 = 1$. Three direct and indirect predecessors of activity 5, i.e., activities 1, 2 and 4, start at the same period because we have $P_4^{\min} = P_2^{\min} = P_1^{\min} = P_5$. Therefore, in the first node, the start period of the four mentioned activities are fixed. Afterward, the start period of activity 3 is calculated as described in the previous node. Having determined the start period of activity 3, we use the backtracking operator. Consequently, node 3 is created to shift the main considered activity of this

level, i.e., activity 3, to the next period ($P_3 = 2$). The branching and backtracking processes continue until all feasible start periods of activities are enumerated. In the example project, there are 61 nodes and 34 different feasible solutions obtained by the first phase. We must apply the RSA to each of which to find the corresponding best schedules. To do so, we modify the project network by inserting a set of new precedence constraints, described in Sec. 3.3. The RSA builds an early tree based on the critical path idea in the project network with generalized precedence relations. Then, it iteratively searches the tree for sets of activities to shift. In order to find solutions with better NPVs, these activities are shifted forwards in time within a calculated displacement period.

3.5. *The network modification procedure*

To find the best schedule corresponding to each solution acquired from the B&B algorithm, we must first modify the project network through adding some new nodes and arrows. To do so, r new nodes with numbers $n+2$ to $n+r+1$ must be added to the network. Moreover, node $n+2$ must be connected to dummy activity 0 so as to start at time cp_1. For this purpose, two arrows are added to the network where the first one connects node 0 to node $n+2$ with a minimal time lag $l_{0,n+2} = cp_1$ and the other connects node $n+2$ to node 0 with a minimal time lag $l_{n+2,0} = -cp_1$.Thereafter, two new arrows are added to link node $n+1$ and $n+r+1$ with minimal time lags of $l_{n+1,n+r+1} = \delta - cp_r$ and $l_{n+r+1,n+1} = cp_r - \delta$. These precedence constraints ensure that there is a period with a length of $\delta - cp_r$ between the last change point and the project deadline. Following, two new arrows with minimal time lags $l_{i,i+1} = cp_{i-n} - cp_{i-n-1}$ and $l_{i+1,i} = cp_{i-n-1} - cp_{i-n}$ are added between nodes i and $i+1$ where $i \in \{n+2, \ldots, n+r$. These precedence constraints guarantee that all periods are sequenced in the correct order and predetermined lengths. As a result, the network is transformed into a generalized form of precedence relations.

In the example project, since $r = 2$, we first add two dummy nodes with indices 7 and 8. Then, two arrows are inserted between nodes 0 and 7 with minimal and maximal time lags of 7 and -7, respectively. Besides, the start time of node 7 is $cp_1 = 7$. Thereafter, two arrows are added between nodes 7 and 8 with $l_{7,8} = cp_2 - cp_1 = 5$ and $l_{8,7} = cp_1 - cp_2 = -5$. Finally, we add two arrows between nodes 6 and 8 with time lags of $l_{6,8} = cp_2 - \delta = -10$ and $l_{8,6} = \delta - cp_2 = 10$. The modified network is illustrated in Fig. 6.

Now, consider a solution obtained from the B&B algorithm in which activity $i; i = 1, \ldots, n$ must start in interval $[cp_e, cp_{e+1}); e = 0, \ldots, r$. To modify the project network, if $e \in \{1, \ldots, r-1\}$, we insert an arrow originating from node $n+e+1$ and ending to node i with a time lag of $l_{n+e+1,i} = 0$. We also add another arrow from node i to node $n+e+2$ with a time lag of $l_{i,n+e+2} = 1$. Moreover, if $e = 0$, we insert a single arrow from node i to node $n+2$ with a time lag of $l_{i,n+2} = 1$.

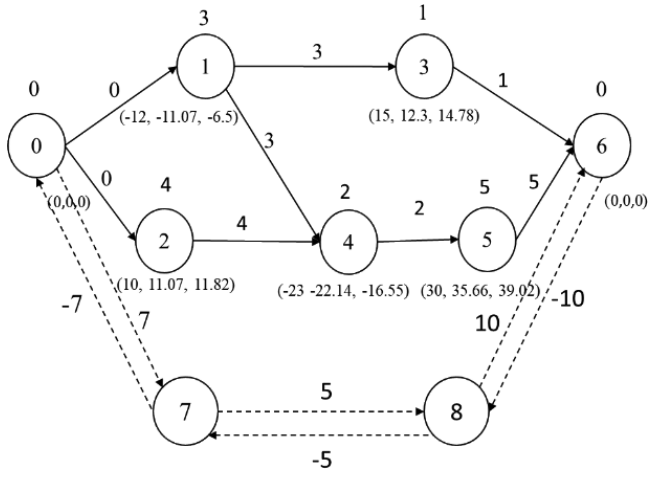

Fig. 6. The modified network of the example project.

Furthermore, if $e = r$, we add only one arrow from node $n + r + 1$ to node i with a time lag $l_{n+r+1,i} = 0$.

Consider the example project and the solution $P_1 = 2, P_2 = 1, P_3 = 3, P_4 = 3$ and $P_5 = 3$, obtained from the B&B algorithm. The modified network with newly added arrows is depicted in Fig. 7. This new network is given as an input to the RSA and results in the following solution with an NPV of 15.57: $s_0 = 0, s_1 = 9, s_2 = 0, s_3 = 12, s_4 = 12, s_5 = 14, s_6 = 22, s_7 = 7, s_8 = 12$.

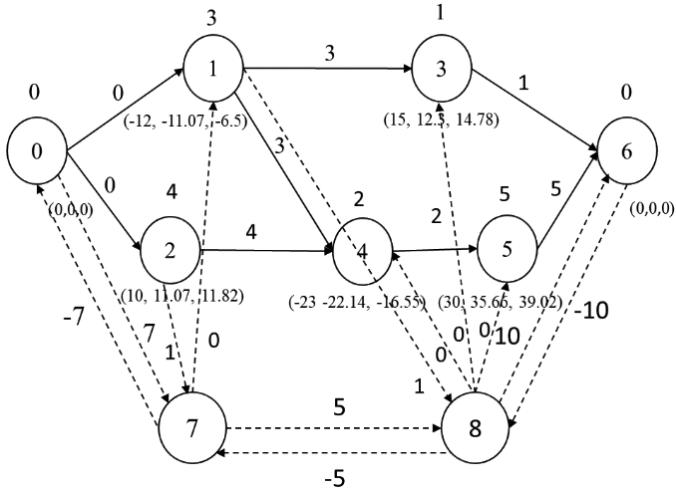

Fig. 7. The modified network of the example project corresponding to the solution $P_1 = 2, P_2 = 1, P_3 = 3, P_4 = 3, P_5 = 3$.

3.6. *The lower bound*

We develop a lower bound (LB) to generate a feasible solution for the problem in the root node of the search tree. Finding a good lower bound leads to improvement in the performance and efficiency of the B&B algorithm because it causes to fathom many redundant nodes of the search tree. Our developed LB is constructed based on a heuristic greedy search algorithm whereby a feasible start period is determined for each activity. Subsequently, the RSA is utilized to obtain the corresponding best schedule. Moreover, whenever a better solution is found, the LB is updated.

To obtain an LB, we develop a heuristic algorithm in both forward and backward versions where the latter is presented as a pseudo-code in Algorithm 1. The main idea of this algorithm is to fix the start period of activities that have the largest variations in their cash flows. To this end, we first construct a matrix $\Omega = [\omega_{ie}]_{n\times(r+1)}$ including n rows, indicating n non-dummy activities, and $r+1$ columns, denoting the number of periods. If an activity can be assigned to no period, we write down "-" in the corresponding cell of Ω, otherwise we calculate the remaining entries as $\omega_{ie} = \frac{|c_{ie}|}{\max_{\forall\, e\in EP_i}|c_{ie}|}$, where EP_i shows eligible periods of the start time of activity i. We also use the symbol "$\times$"to specify the ineligible activities.

In the first step of the backward heuristic algorithm, we define two sets SA and EA as the scheduled and eligible activities, respectively. In addition to this, EP_i is defined for activity $i = 1, \ldots, n$ as $EP_i = \{P_i^{\min}, \ldots, P_i^{\max}\}$. Next, activity $i^* \in EA$ which shows the largest range of variation, i.e., $i^* = \arg\max_{\forall\, i\in EA}\{\max_{\forall\, e}(\omega_{ie}) - \min_{\forall\, e}(\omega_{ie})\}$, is selected. Thereafter, activity i^* is assigned to start period e^* in such a way that for activities with positive (negative) cash flows $e^* = \arg\max_{\forall\, e}\{\omega_{i^*e}\}$ ($e^* = \arg\min_{\forall\, e}\{\omega_{i^*e}\}$). Subsequently, activity i^* is added to set SA, and set EA and matrix Ω are updated subsequently. In order to update set EA, each activity $i \in N\backslash\{SA \cup EA\}$ for which $\mathrm{Suc}_i \subseteq SA$ is included in EA. Additionally, matrix Ω is updated using replacing symbol * with $\omega_{i^*e^*}$ ($EP_{i^*} = \emptyset$). After the assignment of activity i^* to period e^*, we might be able to limit or even fix the start period of some predecessors of activity i^* according to the rules described in Sec. 3.2. In other words, EP_j might be updated for some $j \in \mathrm{Pred}_{i^*}$. This process is repeated until EA is not null.

To improve the quality of our developed LB, we apply the forward heuristic algorithm in which we follow the same steps of the backward one but from the

Algorithm 1. Pseudo-code of the backward heuristic algorithm for the LB.

1. Initialize $SA = \{0, n+1\}, EA = \{i \in N\backslash SA : \mathrm{Suc}_i \subseteq SA\}, EP_i$ for $i = 1, \ldots, n$

While $EA \neq \emptyset$ **do**

2. Let $i^* = \arg\max_{\forall\, i\in EA}\{\max_{\forall\, e\in EP_i}(\omega_{ie}) - \min_{\forall\, e\in EP_i}(\omega_{ie})\}$
3. If i^* has positive cash flows, then let $e^* = \arg\max_{\forall\, e\in EP_{i^*}}\{\omega_{i^*e}\}$.
4. Else let $e^* = \arg\min_{\forall\, e\in EP_{i^*}}\{\omega_{i^*e}\}$.
5. Assign activity i^* to period e^*, $SA \leftarrow i^*$ and update EA and Ω.

End while

beginning of the project. For this purpose, EA is initialized as $EA = \{i \in N \backslash SA : \text{Pred}_i \subseteq SA\}$ and updated by adding each activity $i \in N \backslash \{SA \cup EA\}$ for which $\text{Pred}_i \subseteq SA$. We consider LB as the best-found solution from both the forward and backward versions of the heuristic algorithm.

For the example project, we first apply the backward version of the developed heuristic algorithm in which SA, EA, EP_i and Ω are initialized as follows.

$$SA = \{0, 6\}, \quad EA = \{3, 5\},$$

$$EP_1 = EP_3 = EP_4 = EP_5 = \{1, 2, 3\}, EP_2 = \{1, 2\}$$

$$\Omega = \begin{bmatrix} \times & \times & \times \\ \times & \times & - \\ 1 & 0.81 & 0.98 \\ \times & \times & \times \\ 0.76 & 0.91 & 1 \end{bmatrix}.$$

Regarding matrix Ω, activity 5 is selected and since it has a positive cash flow, this activity is assigned to period 3. Consequently, sets SA, EA are updated as $SA = \{0, 6, 5\}, EA = \{4, 3\}$. Moreover, the eligible start period of activity 4 must be updated as $EP_4 = \{3\}$ and matrix Ω is subsequently updated as follows.

$$\Omega^{(1)} = \begin{bmatrix} \times & \times & \times \\ \times & \times & - \\ 1 & 0.81 & 0.98 \\ 1 & 0.96 & 0.71 \\ - & - & * \end{bmatrix}.$$

Next, activity 4 is selected because it has the largest range of variation in cash flows. The eligible start period of this activity is $EP_4 = \{1, 2, 3\}$, its feasible start period is $e = 3$ and matrix $\Omega^{(1)}$ is updated as $\Omega^{(2)}$. This process will continue until the start periods of all activities are determined. Matrix $\Omega^{(4)}$ indicates the final solution of this example.

$$\Omega^{(2)} = \begin{bmatrix} \times & \times & \times \\ 0.9 & 1 & - \\ 1 & 0.81 & 0.98 \\ - & - & * \\ - & - & * \end{bmatrix}, \quad \Omega^{(3)} = \begin{bmatrix} \times & - & - \\ 0.9 & 1 & - \\ * & - & - \\ - & - & * \\ - & - & * \end{bmatrix}, \quad \Omega^{(4)} = \begin{bmatrix} * & - & - \\ - & * & - \\ * & - & - \\ - & - & * \\ - & - & * \end{bmatrix}$$

When activity 3 is assigned to period 1 in matrix $\Omega^{(3)}$, activity 1 cannot start at period 2 or 3, hence periods 2 and 3 must be removed from the eligible period of activities 1, implying $EP_1 = \{1\}$. Moreover, the eligible start period of activity 2 is $EP_2 = \{1, 2\}$ and its feasible start period is $e = 2$. Now, the start periods

of all activities are feasible and we apply the RSA to this solution resulting in an NPV of 14.07. Regarding the forward version of the heuristic algorithm, matrix Ω is updated two times where the final one is reached as below:

$$\Omega^{(2)} = \begin{bmatrix} - & - & * \\ - & * & - \\ - & - & * \\ - & - & * \\ - & - & * \end{bmatrix}.$$

Using the RSA and $\Omega^{(2)}$, we reach an NPV of 14.47 leading to $LB = \max\{14.07, 14.47\} = 14.47$.

3.7. *The upper bound*

For each node of the search tree, we can predict the maximum value of the NPV before branching that node. If the upper bound (UB) obtained for a particular node is less than the best lower bound found so far, there is no need to branch that node. To obtain a UB for each node, we modify the project network based on the nodes placed on the path from the root to that node of the search tree. Giving this network to the RSA as an input, we find the corresponding upper bound of that node.

Having applied the lower and upper bound procedures to the search tree of the example project, we obtain the search tree displayed in Fig. 8 in which the above number of each node indicates the UB and the nodes for which $UB \leq LB$ are

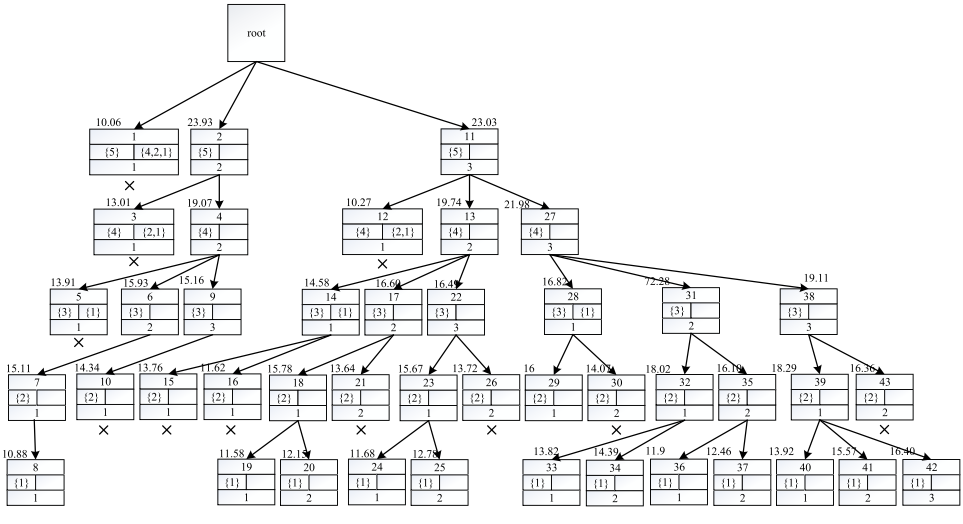

Fig. 8. The search tree of the example project truncated using the LB and UB procedures.

fathomed. As can be seen, the search tree has been truncated and the number of nodes was reduced to 42.

4. The Stochastic Case

4.1. *A multi-stage stochastic problem modeling*

In this section, we formulate the stochastic version of the problem as a MSSP model. This formulation imposes a reactive scheduling policy in each predetermined period. Generally speaking, we divide the time horizon into a number of periods, where each period has its own probabilistic inflation rates and prices except period one, indicating the curret time and all prices are known there. In each period, we have to decide which activities must start based on realized data and probabilistic iflation rates in upcoming periods.

Our stochastic program has $(r+1)$ stages and includes a sequence of random parameters $\Gamma_2, \ldots, \Gamma_{r+1}$ defined on a discrete probability space. In the problem at hand, stage one corresponds to period $e = 1$ in which no stochastic parameter has to be taken into account whereas stage $e; e \geq 2$ corresponds to random parameter Γ_e. In a MSSP, a scenario is a realization of random parameters $\Gamma_2, \ldots, \Gamma_{r+1}$, shown as $(\Gamma_2^\gamma, \ldots, \Gamma_{r+1}^\gamma)$, and can be represented as a scenario tree. The different realizations of $\Gamma_e; e \geq 2$ imply different possibilities of the cash flow corresponding to period e. Dupačová (1995) indicated that the random parameter of the last stage only influences the objective function. To present stochasticity of cash flows in all periods as a discreet scenario tree, we define a set of scenario Γ where the number of scenarios, shown by $|\Gamma|$, is a finite number. This set describes the uncertainty of cash flows where $c_{ie\gamma}$; $\forall i \in N, \forall e \in P\backslash\{1\}, \forall \gamma \in \Gamma$ indicates the cash flow of activity i in period e under scenario γ, which is incurred with probability p_γ where $\sum_{\forall \gamma \in \Gamma} p_\gamma = 1$. In other words, if each real activity $i \in N$ is going to start at time $t < cp_1$, the initial cash flow c_{i1} is factored in. Moreover, if activity i starts at period $e([cp_{e-1}, cp_e); \forall e \in P\backslash\{1\})$ and scenario $\gamma \in \Gamma$ occurs, $c_{ie\gamma}$ has to be taken into account. It is worth mentioning that the stochastic parameter $c_{ie\gamma}$ is supposed to remain constant in period e.

In our MSSP model, called MSSPM, the realization of random parameters $\Gamma_2, \ldots, \Gamma_e$ has been detected at cp_{e-1}, and the remaining uncertainty relates to $\Gamma_{e+1}, \ldots, \Gamma_{r+1}$. Furthermore, the distribution of unrealized parameters is conditional on the realization of random parameters from previous periods. According to Dupačová (1995), a MSSP model can be presented by imposing non-anticipativity constraints. A reactive scheduling policy in our developed MSSPM is non-anticipative, namely decisions in each period do not depend on realization of cash flows in the next periods.

The NPV of each activity can be computed in a preprocessing step based upon the start time of the activity. Given that activity $i \in N$ finishes at time $t \in (cp_{e-1}, cp_e]; \forall e \in P$, if it starts in the first period, the corresponding NPV is shown by c_{it}^{npv}, otherwise, it starts in a period e' where $e' > 1$ and the corresponding

NPV is shown with $c_{it\gamma}^{\text{npv}}$. These two quantities can be calculated using the formulas (5) and (6), respectively.

$$c_{it}^{npv} = c_{i1} \exp\left(-\alpha_e(t - cp_{e-1}) - \left[\sum_{f=1}^{e-1} \alpha_f (cp_f - cp_{f-1})\right]\right), \tag{5}$$

$$c_{it\gamma}^{npv} = c_{ie'\gamma} \exp\left(-\alpha_e(t - cp_{e-1}) - \left[\sum_{f=1}^{e-1} \alpha_f (cp_f - cp_{f-1})\right]\right). \tag{6}$$

We introduce the variable z_{it} that gets one if activity i finishes at time $t < cp_1 + d_i$ and gets zero otherwise. Furthermore, we define variable $y_{iet\gamma}$ that gets one if activity i starts in period e and finishes at time $t \geq cp_{e-1} + d_i$ under scenario γ and gets zero otherwise. Let us introduce $T_{ie} = \{d_i + cp_{e-1}, \ldots, \min\{d_i + cp_e - 1, \delta\}\}$ as the set of times at which activity i can be finished if it is started in period e. In the following, the problem formulation is presented.

$$\text{MSSPM}: \ \text{Max} \sum_{i\in N} \sum_{t\in T_{i1}} c_{it}^{npv} z_{it} + \sum_{i\in N} \sum_{e\in P\setminus\{1\}} \sum_{\gamma\in\Gamma} \sum_{t\in T_{ie}} p_\gamma c_{it\gamma}^{npv} y_{iet\gamma}, \tag{7}$$

$$\text{s.t} \sum_{t\in T_{i1}} z_{it} + \sum_{e\in P\setminus\{1\}} \sum_{t\in T_{ie}} y_{iet\gamma} = 1; \quad \forall\, i \in N, \forall\, \gamma \in \Gamma \tag{8}$$

$$\sum_{t\in T_{j1}} t z_{jt} - \sum_{t\in T_{i1}} t z_{it} + \sum_{e\in P\setminus\{1\}} \sum_{t\in T_{je}} t y_{jet\gamma} - \sum_{e\in P\setminus\{1\}} \sum_{t\in T_{ie}} t y_{iet\gamma} \geq l_{ij} + d_j - d_i; \quad \forall\, (i,j) \in A, \forall\, \gamma \in \Gamma \tag{9}$$

$$\sum_{t\in T_{ie}} t y_{iet\gamma} = \sum_{t\in T_{ie}} t y_{iet\gamma'}; \quad \forall i \in N, \forall \gamma, \gamma' \in \Gamma : \gamma < \gamma', (\Gamma_2^\gamma, \ldots, \Gamma_e^\gamma) = (\Gamma_2^{\gamma'}, \ldots, \Gamma_e^{\gamma'}), \forall\, e \in \{P1, r+1\} \tag{10}$$

$$z_{it} \in \{0,1\}; \quad \forall\, i \in N, \forall\, t \in T_{i1} \tag{11}$$

$$y_{iet\gamma} \in \{0,1\}; \quad \forall\, i \in N, \forall\, \gamma \in \Gamma, \forall\, e \in P\setminus\{1\}, \forall\, t \in T_{ie} \tag{12}$$

Maximization of the expected net presented value is presented in objective function (7). Constraints (8) impose a unique finish time to each activity under each scenario. Constraints (9) present the start-to-start minimal time lag precedence constraints. The non-anticipativity Constraints (10) for all $\gamma, \gamma' \in \Gamma$ such that $(\Gamma_2^\gamma, \ldots, \Gamma_e^\gamma) = (\Gamma_2^{\gamma'}, \ldots, \Gamma_e^{\gamma'})$ ensure if activity $i \in N$ starts in period in $e \in P\setminus\{1, r+1\}$, it has an identical start time under scenarios $\gamma, \gamma' \in \Gamma$. Finally, Constraints (11) and (12) define the binary variables.

4.2. *An example of the stochastic case*

This example indicates how considering change point and the data provided by scenarios may change the schedule in comparison with the deterministic case. Let us consider all data of the example presented in Sec. 3 apart from the cash flows of the second and third periods for which we need to define a set of possible realizations for cash flows. We create set of scenarios Γ through the scenario tree depicted in Fig. 9. In this example, we consider three different realization for Γ_2 and Γ_3 leading to $|\Gamma| = 9$ scenarios.

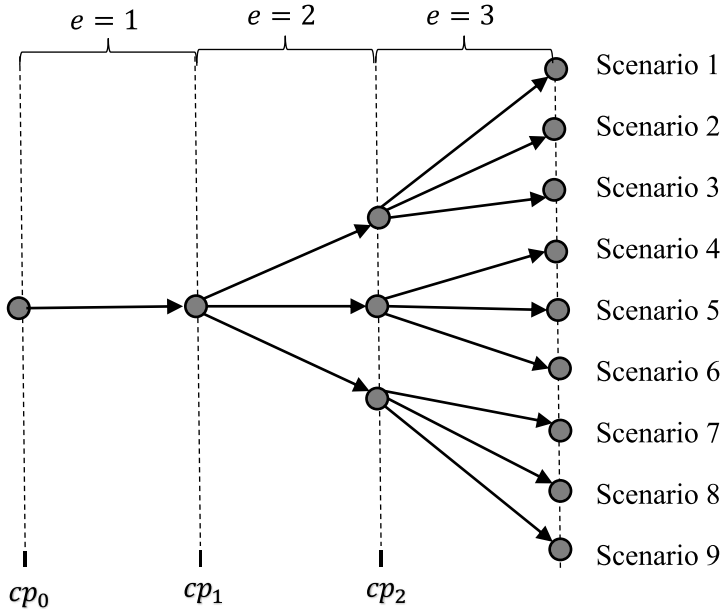

Fig. 9. The scenario tree of the example project.

Table 4. Cash flows in different time periods and scenarios.

Activity (i)	Γ_2			Γ_3			The expected value of cash flow in period e	
	1	2	3	1	2	3	$e = 2$	$e = 3$
1	−11.07	−12.90	−15.71	−18.71	−10.33	−10.41	−14.40	−11.01
2	11.07	9.67	9.43	9.62	12.54	9.61	9.67	9.24
3	12.3	22.57	11.31	13.36	13.28	20.02	14.79	15.34
4	−22.14	−16.12	−25.14	−16.04	−17.71	−23.22	−22.13	−18.36
5	35.66	22.57	27.02	26.73	36.89	24.82	26.55	25.13
Probability of each possibility	0.1	0.3	0.6	0.2	0.3	0.5	1	1

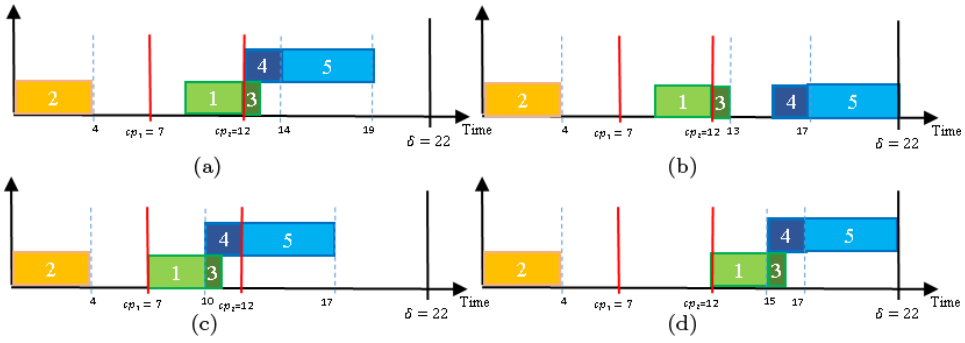

Fig. 10. The optimal schedules of the example obtained from the MSSPM.

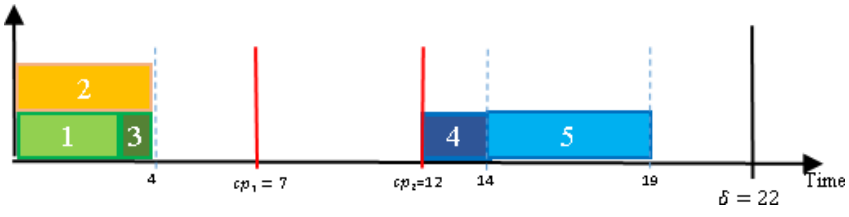

Fig. 11. The optimal schedule of the example obtained from the DM.

The details of the scenarios are presented in Table 4. For instance, the first scenario is constituted through combination of the first elements of Γ_2 and Γ_3 with probability $p_1 = 0.1 \times 0.2 = 0.02$.

The output of the MSSPM is a set of schedules depicted in Fig. 10 with an expected NPV of 12.13 cost units. Figure 10(a) corresponds to scenarios 1 and 2. Moreover, Fig. 10(b) is related to scenario 3, Fig. 10(c) corresponds to scenarios 4, 5 and 6, and Fig. 10(d) matches to scenarios 7, 8 and 9.

As can be seen, the partial schedule before the first change point is the same under all scenarios. Moreover, in scenarios 1, 2 and 3, originated from the first element of Γ_2, an identical partial schedule has been obtained in the second period (Figs. 10(a) and 10(b)), implying non-anticipativity concept. The same feature exists for scenarios created based on the second and third elements of Γ_2.

If we dismiss scenarios data and consider the expected value of cash flows in each period as constant values, we have to solve the example using the DM. In this case, the optimal schedule, depicted in Fig. 11, results in a NPV of 10.26 cost units.

5. Computational Experiments

We performed computational experiments to analyze the performance of our developed solution approaches. Moreover, we evaluated the impact of the project deadline and change points on the optimal solutions and the NPV.

5.1. *Experimental setup*

We coded the B&B algorithm in Visual C++ 2012 and performed all computational experiments on a PC Intel® Core™ i7@ 3.4 GHz processor with 32GB of internal memory. We also used ILOG CPLEX solver 12.8 to solve the DM and MSSPM on the same PC. To validate our developed algorithms, we generated two test sets, i.e., Set1 and Set2. Set1, including 1,080 test instances, was created using the random network generator RanGen (Demeulemeester *et al.*, 2003) to evaluate the B&B algorithm and the DM. To generate the start-to-start minimum time lags between two activities $(i, j) \in A$, we let $l_{ij} = d_i$ where d_i is chosen randomly from $\{1, \ldots, 10\}$. We chose $n = 60, 90, 120, 150$ and $OS = 0.25, 0.5, 0.75$ where OS indicates the order strength, defined as the number of elements of A divided by the theoretically maximum number of precedence-related activities in a project network (Mastor, 1970). We also considered $r = 1, 3, 5$ and $\delta = \beta \times cpl$ where coefficient β takes three values $1.5, 2$ and 2.5. For each combination of n, OS, r and β, we generated 10 random test instances resulting in 1,080 test instances. For these instances, we assumed that change points are integers and uniformly distributed over time horizon $\{1, \ldots, \delta-1\}$. Moreover, the percentage of negative cash flows in the first period is selected randomely based on the discrete uniform distribution$\{25\%, 50\%, 75\%\}$. Moreover, in the first period, positive and negative cash flows are selected casually from discrete uniform distributions $\{10, \ldots, 100\}$ and $\{-100, \ldots, -10\}$, respectively. The cash flow of each activity in the second period is generated based on its initial cash flow multiplied by a random inflation rate. The inflation rate of each activity in subsequent periods is selected randomly from the continuous uniform distribution $U[0\%, 30\%]$. Likewise, the cash flow of each activity in each period $e : e \geq 2$ is generated based on its corresponding value in period $e - 1$. Furthermore, we consider the discount rate of the first period as $\alpha_1 = 0.05$, and for the subsequent periods we calculate the discount rate as $\alpha_e = \epsilon\alpha_{e-1}; e > 2$ where ϵ is randomly chosen from the uniform distribution $U[0\%, 10\%]$.

Set2, including 810 test instances, was created to evaluate the MSSPM and has a great deal in common with Set1. What sets Set2 apart from Set1 is the number of activities and cash flows except for the first period. In Set2, we consider $n = 60, 65$ and 70 because the CPU run times and the required memory strongly depends on the size of the problem and grows very fast. According to the method described in the example of Sec. 4.2, we consider three different realization for cash flows in each period e leading to $|\Gamma| = 3^r$ scenarios. Since we consider $r = 1, 3$, and 5, we have $|\Gamma| = 3, 27$ and 243 scenarios for each test instance. The inflation rate corresponding to each scenario is selected randomly from the uniform distribution $U[0\%, 30\%]$. Therefore, the new cash flows in each time period $e : e \geq 2$ and under each scenario are generated based on their corresponding values in time period $e - 1$. Additionally, the probability of each realization in each period was randomly generated by $U[0, 1]$; all generated probablitiies are normalized so that we have $\sum_{\forall \gamma \in \Gamma} p_\gamma = 1$. Finally, for each combination of n, OS, r, β and r, 10 random test instances were generated resulting in 810 test instances.

Table 5. The average CPU run times (seconds) of the DM.

OS	n							
	60		90		120		150	
	DM	B&B	DM	B&B	DM	B&B	DM	B&B
0.25	9.32	14.56	132.75	139.63	1,943.76	624.16	3,489.54	982.64
0.5	19.74	28.94	547.57	530.72	2,863.65	947.93	—	1,394.31
0.75	49.71	65.41	952.87	894.82	3579.3	1225.74	—	2173.59
Avg.	26.26	36.30	544.40	521.72	2,795.57	932.61	3,489.54	1,516.85

5.2. *Computational results of the deterministic solution approaches*

5.2.1. *Comparative computational performance*

We ran both the DM and the B&B algorithm to solve all generated test instances. As some of the instances are intractable, we imposed a one-hour time limit to our experiments. Table 5 indicates the average CPU run times (seconds) of the DM and the B&B algorithm. As can be seen, which is common to both solution methods is that the more the number of activities, the more required CPU run time. This is due to the fact that when the number of activities increases, the number of required variables and constraints in both the algorithms rise. Despite the project scheduling problems with the makespan minimization goal in which smaller amounts of OS results in more complex and time-consuming instances, for *max-npv* problems, higher amounts of OS make instances harder. This happens because the sign and amount of each activity cash flow may contribute to the optimal schedule of all its predecessors and successors. However, what sets the B&B algorithm apart from the DM is its overall better performance, namely the total average CPU run time for the DM and the B&B algorithm are 1,358.7 and 751.87, respectively. Although the DM outperforms the B&B algorithm for majority of instances with $n = 60$ and they have nearly identical performances for instances with $n = 60$, the superiority of the B&B algorithm is highlighted when instances are more complex, i.e., having larger amounts of n and OS. The last row of Table 5 indicates the average CPU run time (Avg.) based upon different values of n. It is worthwhile to note that none of instances with $n = 60$ and $OS = 0.5, 0.75$ could be solved using the DM.

Due to the imposed time limit, the DM or B&B algorithm might not be able to find optimal solutions for all instances. Table 6 indicates the number of optimal solutions (out of 270 test instances) obtained by the DM model and the B&B

Table 6. Number of found optimal solutions using the DM and the B&B algorithm.

n	60	90	120	150
DM	270	219	164	84
B&B algorithm	270	263	240	205

algorithm. Overall, the B&B algorithm shows better performance, and the DM could find the optimal solution of 737 out of 1,080 while that was 978 for the B&B algorithm.

5.2.2. *Sensitivity analyses*

In this section, we run several experiments to analyze the impact of two significant parameters of the problem, the project deadline and the occurrence times of change points. To this end, we consider the example project illustrated in Sec. 3.2 to understand how optimal NPV may alter when these parameters change.

Figure 12 depicts the impact of the project deadline on the NPV in which we have considered three values for δ as $\delta = 1.5cpl, 2cpl$ and $3cpl$. As can be seen, the higher the amount of project deadline is, the more the NPV will be. This trend indicates that when activities have more slacks, better start times can be chosen for them, leading to a better NPV. Another notable result is that the NPV shows more increase in the first piece of Fig. 12. This seems to indicate that the impact of δ on the NPV gradually decreases when the project deadline rises hugely.

To investigate the impact of occurrence times of change points in the example project, we first consider the second change point as a fixed parameter and alter the first one as $cp_1 = 1, \ldots, 11$. This investigation has been shown in Fig. 13. For $cp_1 = 2, \ldots, 11$, we obtained $s_1 = 12, s_2 = 0, s_3 = 15, s_4 = 15$ and $s_5 = 17$. If we let $cp_1 = 1$, the only change is $s_2 = 1$ because the positive cash flow of this activity increases after the first change point. For $cp_1 = 2, \ldots, 11$, we obtained $s_2 = 0$ due to the time value of money and the larger discount rate of the second period. Consequently, we see a downward trend at the beginning of Fig. 13. Since the optimal schedule do not change for $cp_1 = 2, \ldots, 11$ and apart from activity 3, cash flows of other activities starting in the second period increase, the project will be more profitable if the occurrence time of the first change point rises.

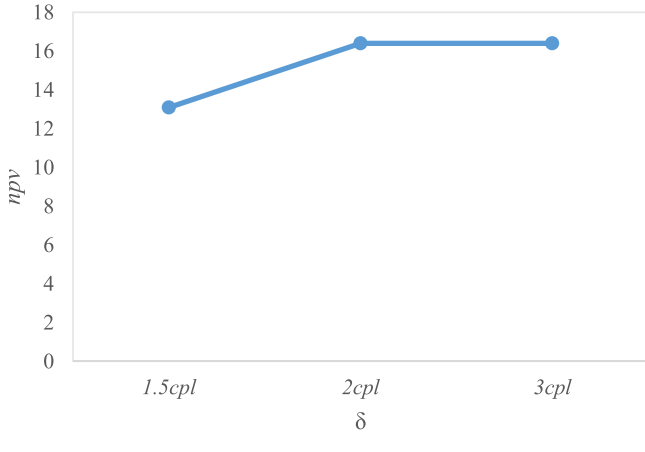

Fig. 12. The impact of δ on the NPV in the example project.

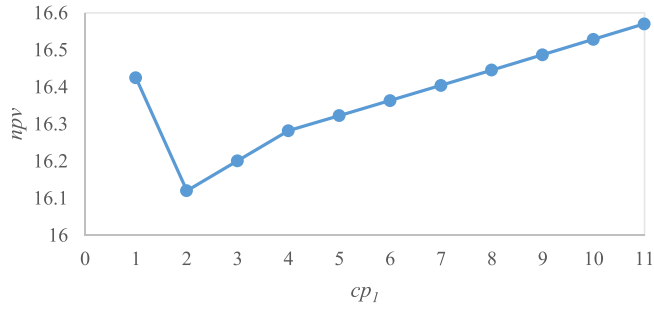

Fig. 13. The impact of cp_1 on the NPV when $cp_2 = 12$.

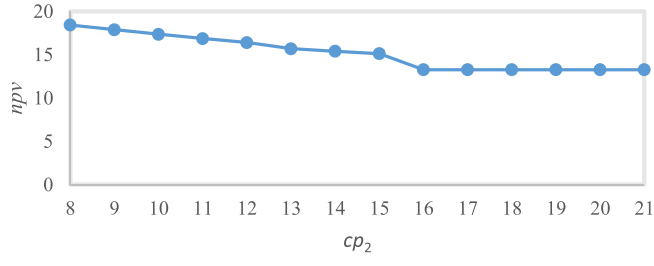

Fig. 14. The impact of cp_2 on the NPV when $cp_1 = 7$.

Similarly, if we consider cp_1 as a constant parameter and alter the second one as $cp_2 = 8, \ldots, 21$, Figure 14 indicates the behavior of the NPV. Despite the overall upward trend in Fig. 13, this figure indicates a downward trend. In other words, the larger the amount of cp_2, the smaller the NPV. Moreover, Fig. 14 implies that the impact of cp_2 is higher than cp_1 because it shows a larger range of variation on the NPV. When $cp_2 = 8$, we obtain $s_2 = 0$ and others start at soon as possible after cp_2. This policy is followed by the optimal schedules until $cp_2 = 12$, but at $cp_2 = 13$ we have $s_1 = 0$ and $s_3 = 3$ because the project must be completed by the given deadline. Consequently, for $cp_2 = 13, 14$ and 15, we have $s_1 = 0$, $s_2 = 0$, $s_3 = 3$ and activities 4 and 5 start as soon as possible after cp_2. While the star times of activities 1, 2 ,3 remain constant in the optimal schedules corresponding to $cp_2 = 13, \ldots, 21$, the start times of activities 4 and 5 change as $s_4 = 7$ and $s_5 = 9$. This is due to the fact that no activity must be completed after the project deadline. The optimal schedule pertaining to $cp_2 = 16$ is also optimal for $cp_2 = 17, \ldots, 21$.

5.3. *Computational results of the stochastic solution approach*

5.3.1. *Computational performance*

All generated test instances were solved by the MSSPM and using the CPLEX. Since finding the optimal solution of some instances was very time and

Table 7. Number of found optimal solutions using the MSSPM.

$\|\Gamma\|$	n		
	60	65	70
3	90	82	79
27	78	67	46
243	54	43	32

memory-consuming, we imposed the one-hour time limit to our experiments. Some instances could not be optimally solved within the given time limit or the memory limitation. The number of optimally solved instances (out of 90 test instances) for the MSSPM is shown in each cell of Table 7. According to Table 7, increasing the number of activities and the number of scenarios cause more complexity of the problem and less found optimal solutions.

Tables 8 indicates the average CPU run times (seconds) of the MSSPM based on different values of n and $|\Gamma|$. As can be seen, the larger n and $|\Gamma|$ and smaller OS is, the more CPU run time is required to solve instances optimally.

5.3.2. *Merit and pitfall of the Stochastic solution approach*

In contrast to the DM that ignores the scenarios data and focus on the expected value of cash flows in each period, the MSSPM considers all scenarios and generates schedules corresponding to each. In order to compare these two policies, we ran all instances of Set2 using the DM. To do so, in each period, we considered the expected value of different realizations of each cash flow as a constant. The corresponding results are presented in Tables 9 and 10. Table 9 indicates that if the stochastic model is utilized, how many percents the objective function of the problem can be improved, and Table 10 summarizes the average CPU run times in conjunction with the number of optimally solved instances (out of 270 instances), shown inside parenthesis. According to Table 9, the stochastic model contributes to more improvement for larger values of OS. In contrast, there is no improvement trend based on the number of activities. However, the MSSPM can create an average improvement of 15% in the expected NPV. On the other hand, by comparing Table 10 to Tables 7 and 8, we conclude that the average CPU run time of the MSSPM is 22 and 26 times

Table 8. The average CPU run times (seconds) of the MSSPM.

$\|\Gamma\|$	n			Avg.
	60	65	70	
3	41.71	69.98	122.80	78.16
27	385.43	892.74	1,094.69	790.95
243	3,199.81	3,457.92	3,521.65	3,393.13
Avg.	1,208.98	1,473.55	1,579.71	1,420.75

Table 9. The average percentage of improvement of objective funtion when we use the MSSPM in comparison to the DM.

OS	*N*		
	60	65	70
0.25	2%	7%	10%
0.5	18%	21%	11%
0.75	28%	24%	15%

Table 10. The average CPU run times (seconds) of the DM and B&B and the number of found optimal solutions.

n	60	65	70
DM	29.21 (270)	42.98 (265)	85.82 (254)
B&B	38.65 (270)	47.21 (270)	79.43 (267)

more than the DM and the B&B algorithm, respectively. Moreover, the MSSPM fails to solve instances having more than 70 activities within one-hour time limit. As expected, the stochastic model can build better solutions but consumes more CPU run times and memory, which grow very fast.

Generally speaking, we conclude that there is a trade-off between the quality of obtained solutions and CPU run times and memory consumption. In an uncertain circumstance, the DM model results in lower quality solution but it can be solved faster whereas the MSSPM achieves better solutions but it needs far more CPU run times and memory.

6. Conclusions and Outlook on Perspective Research

In this paper, we formulated a project scheduling problem under periodic inflation to maximize the NPV. We considered the problem in both deterministic and stochastic settings and developed an integer linear programming model for each case. In addition to this, we developed an efficient branch-and-bound algorithm to solve large instances of the deterministic version of the problem. Using extensive computational experiments, we analyzed and compared the performances of our developed solution approaches. On average, we showed that the stochastic model may lead to 15% improvement in the expected NPV rather than the DM, but it needs much more CPU run time to achieve this superiority. Moreover, we analyzed the impact of the project deadline and the occurrence time of change points on the project NPV.

As a future research opportunity, we recommend developing heuristic or metaheuristic algorithms for the problem. Moreover, developing exact solution approaches like benders decomposition might be an interesting research topic.

References

Battersby, A (1967). *Network Analysis for Planning and Scheduling*. Macmillan.

Benati, S (2006). An optimization model for stochastic project networks with cash flows. *Computational Management Science,* 3(4), 271–284.

Breuss, F (2018). Would DSGE models have predicted the great recession in Austria? *Journal of Business Cycle Research*, 14(1), 105–126.

Buss, AH and MJ Rosenblatt (1997). Activity delay in stochastic project networks. *Operations Research*, 45(1), 126–139.

De Reyck, B (1996). An optimal procedure for the unconstrained *max-npv* project scheduling problem with generalized precedence relations. DTEW Research Report 09642, 1–33.

Demeulemeester, E (1996). An optimal recursive search procedure for the deterministic unconstrained max-npv project scheduling problem. *DTEW Research Report 09603*, 1–15.

Demeulemeester, E and WS Herroelen (2006). *Project Scheduling: A Research Handbook*, Vol. 49. Springer Science & Business Media.

Demeulemeester, E, M Vanhoucke and W Herroelen (2003). RanGen: A random network generator for activity-on-the-node networks. *Journal of Scheduling*, 6(1), 17–38.

Dupačová, J (1995). Multistage stochastic programs: The state-of-the-art and selected bibliography. *Kybernetika-Praha*, 31(2), 151–174.

Elmaghraby, SE and WS Herroelen (1990). The scheduling of activities to maximize the net present value of projects. *European Journal of Operational Research*, 49(1), 35–49.

Etgar, R and A Shtub (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.

Grinold, RC (1972). The payment scheduling problem. *Naval Research Logistics Quarterly*, 19(1), 123–136.

Hazir, O and G Ulusoy (2019). A classification and review of approaches and methods for modeling uncertainty in projects. *International Journal of Production Economics*, 223, 107522.

Herroelen, W, E Demeulemeester and B De Reyck (1999). A classification scheme for project scheduling. In *Project Scheduling*, pp. 1–26. Springer, Boston, MA.

Kamburowski, J (1990). Maximizing the project net present value in activity networks under generalized precedence relations. In *Proc. 21st DSI Annual meeting*, San Diego, pp. 748–750.

Liang, Y, N Cui, T Wang and E Demeulemeester (2019). Robust resource-constrained *max-npv* project scheduling with stochastic activity duration. *OR Spectrum*, 41(1), 219–254.

Mahamid, I and N Dmaidi (2013). Risks leading to cost overrun in building construction from consultants' perspective. *Organization, Technology & Management in Construction: An International Journal*, 5(2), 860–873.

Mastor, AA (1970). An experimental investigation and comparative evaluation of production line balancing techniques. *Management Science*, 16(11), 728–746.

Nyberg, H (2010). Dynamic probit models and financial variables in recession forecasting. *Journal of Forecasting*, 29(1–2), 215–230.

Obstfeld, M, JC Shambaugh and AM Taylor (2009). Financial instability, reserves, and central bank swap lines in the panic of 2008. *American Economic Review*, 99(2), 480–486.

Orlin, J, R Ahuja and T Magnanti (1993). *Network Flows, Theory, Algorithms and Applications*. Vol. 5. New Jersey: Prentice Hall.

Prajapati, S, R Gupta and M Pandey (2016). Causes and effects of cost overrun on construction projects in Madhya Predesh. *International Journal of Engineering Development and Research*, 4(2), 1346–1350.

Russell, RA (1986). A comparison of heuristics for scheduling projects with cash flows and resource restrictions. *Management Science*, 32(10), 1291–1300.

Schwindt, C and J Zimmermann (2001). A steepest ascent approach to maximizing the net present value of projects. *Mathematical Methods of Operations Research*, 53(3), 435–450.

Sobel, MJ, JG Szmerekovsky and V Tilson (2009). Scheduling projects with stochastic activity duration to maximize expected net present value. *European Journal of Operational Research*, 198(3), 697–705.

Tilson, V, MJ Sobel and JG Szmerekovsky (2006). Scheduling projects with stochastic activity duration to maximize EPV. *SSRN eLibrary*.

Vanhoucke, M (2006). An efficient hybrid search algorithm for various optimization problems. *European Conf. Evolutionary Computation in Combinatorial Optimization*.

Vanhoucke, M, E Demeulemeester and W Herroelen (2001a). Scheduling projects with linear time-dependent cash flows to maximize the net present value. *International Journal of Production Research*, 39(14), 3159–3181.

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

Vanhoucke, M, E Demeulemeester and W Herroelen (2003). Progress payments in project scheduling problems. *European Journal of Operational Research*, 148(3), 604–620.

Wiesemann, W, D Kuhn and B Rustem (2010). Maximizing the net present value of a project under uncertainty. *European Journal of Operational Research*, 202(2), 356–367.

Wikipedia, Sanctions against Iran. https://en.wikipedia.org/wiki/Sanctions_against_Iran. Accessed May 18, 2020.

Zhao, C, H Ke and Z Chen (2016). Uncertain resource-constrained project scheduling problem with net present value criterion. *Journal of Uncertainty Analysis and Applications*, 4(1), 12.

Zheng, W, Z He, N Wang and T Jia (2018). Proactive and reactive resource-constrained *max-npv* project scheduling with random activity duration. *Journal of the Operational Research Society*, 69(1), 115–126.

Biography

Mahboobe Peymankar is a PhD student in Industrial Engineering at Ferdowsi University of Mashhad (Iran). She received her BSc and MSc degrees in Industrial Engineering both from Ferdowsi University of Mashhad in 2012 and 2014, respectively. She is interested in the field of Project Scheduling, Machine Scheduling, Supply Chain and Logistics. She has published a research paper in Journal of Production Economics. She also has published several papers in the field of scheduling in Persian journals.

Mohammad Ranjbar is a Professor of Industrial Engineering at Ferdowsi University of Mashhad (Iran). He received his PhD in Industrial Engineering in 2007 from Sharif University of Technology. His research field includes Project Scheduling,

Project Scheduling, Machine Scheduling, Transportation and Logistics and Healthcare Systems. He has published more than 30 papers in high-quality international journals such as European Journal of Operational Research, Computers and Operations Research, Computers and Industrial Engineering, Journal of the Operational Research Society, International Transaction in Operations Research and Journal of Manufacturing Systems.