A heuristic procedure for solving discrete time-cost tradeoff problems in repetitive projects

Qian Zhang¹ and Xin Zou²*

¹ State Grid Energy Research Institute, Beijing 102209, China
² Department of Economic Management, North China Electric Power University, Baoding 071003, China
*e-mail: zoux788@126.com

Abstract. This paper studies a discrete time-cost tradeoff problem in repetitive projects, in which each activity has several alternative modes and must be scheduled in sequence from the first unit to the last unit. The objective is to minimize the total project cost consisting of direct costs and idle costs of all activities while meeting a pre-specified deadline. A heuristic procedure based on the linear programming relaxation is presented to solve this problem with the goal of finding high-quality solutions for medium- or large-size problems within a short period of time. Computational experiments are conducted to verify the effectiveness of the proposed procedure.

1. Introduction

The discrete time-cost tradeoff problem (DTCTP) has received widely attention in the construction industry for the purpose of time management and cost control. In the DTCTP, the duration (processing time) of an activity is assumed to be a discrete, non-increasing function of the amount of a single non-renewable resource (money) allocated to it. In the literature, the DTCTP can be divided into two sub-problems, i.e., the deadline problem with objective to minimize the project cost while meeting a pre-specified deadline, and the budget problem with the goal to minimize the project duration without exceeding a given budget. This paper focuses on the deadline problem in repetitive projects.

Repetitive projects, such as highways, high-rise buildings, pipelines, and housing projects, are characterized by a number of units and a set of activities that need to be repeated in each unit for the length of the job. Due to the feature of repetition, an effective way to schedule repetitive projects lies in eliminating waste from resource waiting for preceding resources to finish their work by maintaining work continuity of resources. Traditional methods based on network scheduling algorithms are not suitable for optimizing the schedule of repetitive projects because they do not take resource work continuity into account. As such, many studies have been conducted to implement the time-cost tradeoff analysis in repetitive projects. For examples, Reda [1] adopted the linear programming theory to deal with the deadline problem while preventing all activities from having interruptions. A strictly application of resource work continuity may lead to a longer project duration and therefore an increase in project indirect cost. Therefore, Senouci and Eldin [2] developed a model to determine the time-cost profile of repetitive projects by considering the impacts of activities modes and interruptions simultaneously. With the rapid development of computer-based techniques, many studies have applied meta-heuristic algorithms such as genetic algorithms (GAs) to solve the DTCTP in repetitive projects. Hyari et al. [3] integrated GA and the Pareto ranking method to constitute the Pareto optimal front.
Long and Ohsato [4] proposed a GA-based model to minimize both project duration and total cost corresponding to different levels of relative importance of the two objectives. Ezeldin and Soliman [5] developed a hybrid technique of combining GAs with dynamic programming approach to handle the DTCTP under uncertainty. Hegazy et al. [6] presented a GA-based distributed scheduling model for projects involving multiple distributed sites by integrating the time-cost tradeoff analysis and the concept of soft logic, in which units in an activity could be scheduled in parallel, in sequence, or part in parallel and part in sequence. In addition, Zhang et al. [7] employed GA to optimize the execution mode of each activity in repetitive projects with consideration of time-cost tradeoffs, where the work sequence of units in each activity can be changed arbitrarily.

Mathematical modeling methods based on optimization theories is able to preserve the optimality of solutions. However, they are too time-consuming to deal with large-size problems. Meta-heuristic algorithms have been widely applied in solving project scheduling problems to search for optimal or near-optimal solutions. Their performance depends strongly on the proper implementation and fine tuning of parameters, and therefore may be varied owing to different user experience. In this paper, a heuristic procedure without needing parameter tuning is developed to solve the DTCTP in repetitive projects. Then, computational experiments are conducted to verify the effectiveness of the proposed procedure.

2. Problem description

Suppose that we are given a repetitive project with a set \( A = \{0, 1, \ldots, n + 1\} \) of activities and a set \( U = \{1, \ldots, m\} \) of units. Without loss of generality, we assume that activities 0 and \( n + 1 \) is the single dummy start and end activities, respectively. For each activity \( i \in A \), modes are represented with index \( k \in M_i = \{1, \ldots, \bar{k}_i\} \) and characterized by a tuple \((d_{ik}, c_{ik})\) of time and cost at mode \( k \). We assume \( k < k' \) for \( k, k' \in M_i \), implies \( d_{ik} < d_{ik'} \) and \( c_{ik} > c_{ik'} \). Hence, 1 and \( \bar{k}_i \) are the shortest and longest modes of activity \( i \), respectively. Let \( r_{ik} \) be the daily idle cost of activity \( i \) at mode \( k \). For dummy activities 0 and \( n + 1 \), we set \( k_0 = \bar{k}_{n+1} = 1 \), \( d_{01} = d_{n+1,1} = 0 \), \( c_{01} = c_{n+1,1} = 0 \), and \( r_{01} = r_{n+1,1} = 0 \). The precedence relations between activities are represented by the set \( E \in A \times A \), where each pair of activities \( (p, i) \in E \) indicates that activity \( p \) immediately precedes activity \( i \). Let \( \delta \) be the deadline, that is, an upper bound on the project completion time.

Three sets of decision variables are defined: (1) \( x_{ik} \in \{0, 1\} \), which takes a value of one if all units in activity \( i \) is executed in mode \( k \), and zero otherwise; (2) \( s_{ij} \in R_+ \), which represents the start time of activity \( i \) in unit \( j \); and (3) \( z_{ik} \in R_+ \), which represents the idle time of activity \( i \) at mode \( k \). Formally, the DTCTP in repetitive projects can be described by the following mixed-integer linear programming model (MILP model, for short):

\[
\min \sum_{i \in A} \sum_{k \in M_i} (x_{ik}c_{ik} + z_{ik}r_{ik}) 
\]

\[
\sum_{k \in M_i} x_{ik} = 1, \quad i \in A
\]

\[
s_{pj} + \sum_{k \in M_p} x_{pk}d_{pjk} \leq s_{ij}, \quad (p, i) \in E; j \in U
\]

\[
s_{ij} + \sum_{k \in M_i} x_{ij}d_{ijk} \leq s_{i,j+1}, \quad i \in A; j \in U \setminus \{m\}
\]

\[
s_{n+1,m} \leq \delta
\]

\[
z_{ik} \geq \sum_{j \in U \setminus \{m\}} (s_{i,j+1} - s_{ij} - d_{ijk}) - \delta(1 - x_{ik}), \quad i \in A; k \in M_i
\]

where \( s_{ij}, z_{ik} \geq 0; x_{ik} \in \{0, 1\} \).

The objective function (1) minimize the sum of direct costs and idle costs of all activities. Constraints (2) guarantee that each activity is executed in only one mode. Constraints (3) preserve the
precedence relations between activities at each unit. Constraints (4) ensure that each activity is scheduled in sequence from the first unit to the last unit. Constraint (5) forces the project to be completed before the given deadline $\delta$. Constraints (6) are used to evaluate the value of $z_{ik}$ for each $i \in A$ and $k \in M_1$.

3. Heuristic procedure

The proposed procedure for solving the MILP model consists of two modules, namely initialization and improvement. The initialization module aims to construct a feasible MILP solution according to Property 1, while the improvement module is adopted to improve the initial feasible MILP solution based on Property 2.

**Property 1:** Let $LPR$ be the linear programming relaxation of the MILP model and $d^{LP}_{ij} = \sum_{k \in M_1} x^{LP}_{ijk}$ be the duration of activity $i$ in unit $j$ determined by the optimal LPR solution $x^{LP}_{ij}$. Selecting the mode $k_i$ with $d_{ijk_i} \leq d^{LP}_{ij}$ for each activity $i$ leads to a feasible solution of the MILP model.

**Property 2:** Let $ES_{ij}(k_h; h \in A)$ and $LC_{ij}(k_h; h \in A)$ be the earliest start and latest completion times of activity $i$ in unit $j$ when each activity $h \in A$ is executed in mode $k_h$, respectively. The mode $k_i + 1$ is feasible, on condition that the modes of all other activities are remained unchanged, if and only if $LC_{ij}(k_h, k_i + 1; h \in A, h \neq i) \leq LC_{ij}(k_h; h \in A)$ is satisfied for each $j \in U$.

3.1 Initialization

According to Property 1, a feasible MILP solution is obtained using the following four steps:

1. Determining $d^{LP}_{ij}$ for each activity $i$ in each $j$ by solving the LPR model.
2. For each activity $i \in A$, if there exists a mode $k_i$ such that $d_{ijk_i} \leq d^{LP}_{ij} \leq d_{ijk_i+1}$ is satisfied, then the mode is assigned to activity $i$; otherwise, the longest mode $k_i$ is selected. Then, a feasible MILP solution is obtained and let $UB_1 = \sum_{i \in A} c_{ik_i}$ denotes total direct cost corresponding to this solution.
3. Let the actual duration of activity $i$ in unit $j$ be $d_{ij} = d_{ijk_i}$. Using a linear programming model shown in Eq. (7) to improve the feasible MILP solution by minimizing the total idle cost that is denoted as $UB_2$.
4. Evaluating the total project cost by $UB = UB_1 + UB_2$.

\[
\begin{align*}
\text{min } UB_2 &= \sum_{i \in A} r_i(s_{im} - s_{i1}) \\
sp_{pj} + dp_{pj} &\leq sl_{jp}, \quad (p, i) \in E; j \in U \\
s_{ij} + dp_{ij} &\leq s_{i,j+1}, \quad i \in A; j \in U \backslash \{m\} \\
sp_{n+1,m} &\leq \delta
\end{align*}
\]

3.2 Improvement

This module aims to reduce the total project cost of the initial MILP solution by replacing the mode of each activity $i \in A$ from the current $k_i$ to a cheaper one without leading to an infeasible solution. The details are as follows:

1. Determining $ES_{ij}(k_h; h \in A)$ and $LC_{ij}(k_h; h \in A)$ for each activity $i$ in each unit $j$ according to the current mode assignment.
2. Updating the set $F = \{i \in A | LC_{ij}(k_h, k_i + 1; h \in A, h \neq i) \leq LC_{ij}(k_h; h \in A), j \in U\}$.
3. If $F = \emptyset$, then the procedure stops and a local optimal solution is obtained. Otherwise, for each activity $i \in F$, replacing its current mode $k_i$ with $k_i + 1$ and evaluating the corresponding total project cost $UB^i = UB^i_1 + UB^i_2$, where $UB^i_1 = UB_1 + c_{ikk_i+1} - c_{ikk_i}$, and $UB^i_2$ is determined using the linear programming model shown in Eq. (7).
4. Finding the activity $i^*$ satisfying $UB^{i^*} - UB = \min_{i \in F} \{UB^i - UB\}$. If $UB^{i^*} - UB < 0$ is true,
let $k_{l}^{*} = k_{l} + 1$, $UB_{1}^{*} = UB_{1}^{l}$, $UB_{2}^{*} = UB_{2}^{l}$, and return to (1); otherwise, the procedure terminates and a local optimal solution is generated.

4. Computational experiments

This section conducts computational experiments to analyze the performance of the proposed heuristic procedure for solving the DTCTP in repetitive projects. All test problems are run on a PC with i7-4700MQ CPU, 8GB RAM memory, and Window 7 operating system.

Each test problem is generated using the following five types of parameters: number of activities ($|A|$), number of units ($m$), number of modes ($k_{i}$) for each activity $i$, coefficient of network complexity (CNC), and deadline tightness ($\theta$). Table 1 lists the settings for these parameters. When the values of these parameters are determined, a random instance is generated by the method adopted in Değirmenci and Azizoğlu [8].

### Table 1 Parameters of test problems

| Parameter                          | Values                     |
|-----------------------------------|----------------------------|
| Number of activities ($|A|$)      | 40, or 60                  |
| Number of units ($m$)             | 20, or 40                  |
| Number of modes ($k_{i}$) for each activity $i$ | 1.8, or 2.1 |
| Coefficient of network complexity (CNC) | Discrete uniform distribution over the set of $\{2,3,...,10\}$ (DU[2,10]) or $\{11,12,...,20\}$ (DU[11,20]) |
| Deadline tightness ($\theta$)     | 0.15, 0.3, 0.45, 0.6, 0.75, or 0.9 |

Ten instances are generated randomly for each parameter level combination, which gives a total of $2 \times 2 \times 2 \times 6 \times 6 \times 10 = 960$ random instances. Tables 2-4 show the computational results of the proposed procedure for different parameters $|A|$, $k_{i}$, $m$, CNC, and $\theta$.

### Table 2 Computational results of the proposed procedure for different $|A|$ and $k_{i}$

| $k_{i}$ | $|A|=40$                   | $|A|=60$                   |
|---------|-----------------------------|-----------------------------|
|         | DU[2,10] | DU[11,20] | DU[2,10] | DU[11,20] |
| Number of instances | 240 | 240 | 240 | 240 |
| Avg. CPU (seconds)   | 9.67 | 12.18 | 24.73 | 41.15 |
| Max. CPU (seconds)   | 27.12 | 31.05 | 96.15 | 159.76 |
| Avg. Dev. (%)        | 2.2 (134) * | 2.93 (104) | 2.48 (177) | 3.2 (68) |

* The number in the parenthesis indicates the number of stances solved to optimality.

### Table 3 Computational results of the proposed procedure for different $m$ and CNC

| $m$ | CNC=1.8 | CNC=2.1 |
|-----|---------|---------|
|     | 20 | 40 | 20 | 40 |
| Number of instances | 240 | 240 | 240 | 240 |
| Avg. CPU (seconds)   | 29.69 | 14.27 | 23.15 | 26.31 |
| Max. CPU (seconds)   | 84.19 | 75.33 | 96.15 | 159.76 |
| Avg. Dev. (%)        | 2.61 (131) | 2.69 (148) | 3.08 (131) | 2.36 (73) |

### Table 4 Computational results of the proposed procedure for different $\theta$

| $\theta$ | 0.15 | 0.3 | 0.45 | 0.6 | 0.75 | 0.9 |
|----------|------|-----|------|-----|------|-----|
| Number of instances | 160 | 160 | 160 | 160 | 160 | 160 |
| Avg. CPU (seconds)   | 38.53 | 35.08 | 23.33 | 18.44 | 14.50 | 6.73 |
| Max. CPU (seconds)   | 159.76 | 124.53 | 96.15 | 77.13 | 81.82 | 31.72 |
| Avg. Dev. (%)        | 3.59 (35) | 2.8 (54) | 1.34 (80) | 1.19 (86) | 0.77 | 0.06 |
From these Tables, we find that the proposed procedure can provide high-quality solution for all instances within a short period of time. Specifically, all instances can be optimized within 160 seconds with an average deviation of 2.7%. Of the 960 instances, 50.31% can be solved to optimality. On the other hand, the CPU time and average deviation from optimality of the procedure reduces significantly as the deadline tightness ($\theta$) decreases. For example, when the value of $\theta$ is increased from 0.15 to 0.9, the average deviation reduces from 3.59% to 0.06%, and the number of instances solved to optimality increases from 35 to 127.

5. Conclusions
This paper proposed a heuristic procedure for the discrete time-cost tradeoff problem in repetitive projects. Compared with previous exact models and meta-heuristic algorithms, the proposed procedure owns the capability of finding optimal or high-quality solutions within a short period of time. Another advantage to the procedure is independent of tuning of parameters. The planner is encouraged to adopt the procedure developed in this paper to schedule large-size projects when other methods fail to find exact solutions within a reasonable amount of time.

Acknowledgments
The authors would like to acknowledge the National Natural Science Foundation of China (71701069), Natural Science Foundation of Hebei province of China (G2018502127), and the Research on differentiated management of comprehensive plan under state owned enterprise reform.

References
[1] Reda, R. M. (1990). RPM: Repetitive project modeling. Journal of Construction Engineering and Management, 116(2): 316-330.
[2] Senouci, A. B., Eldin, N. N. (1996). Dynamic programming approach to scheduling of non-serial linear project. Journal of Computing in Civil Engineering, 10(2): 106-114.
[3] Hyari, K. H., El-Rayes, K., El-Mashaleh, M. (2009). Automated trade-off between time and cost in planning repetitive construction projects. Construction Management and Economics, 27(8): 749-761.
[4] Long, L. D., Ohsato, A. (2009). A genetic algorithm-based method for scheduling repetitive construction projects. Automation in Construction, 18(4): 499-511.
[5] Ezeldin, A. S., Soliman, A. (2009). Hybrid time-cost optimization of nonserial repetitive construction projects. Journal of Construction Engineering and Management, 135(1): 42-55.
[6] Hegazy, T., Elhakeem, A., & Elbeltagi, E. (2004). Distributed scheduling model for infrastructure networks. Journal of Construction Engineering and Management, 130(2), 160-167.
[7] Zhang, L., Zou, X., Qi, J. (2015). A trade-off between time and cost in scheduling repetitive construction projects. Journal of Industrial & Management Optimization, 11(4): 1423-1434.
[8] Değirmenci, G., Azizoğlu, M. (2013). Branch and bound based solution algorithms for the budget constrained discrete time/cost trade-off problem. Journal of the Operational Research Society, 64(10): 1474-1484.