Multi-Dimensional Optimization Model for Schedule Fast-Tracking without Over-stressing Construction Workers

<table>
<thead>
<tr>
<th>Journal:</th>
<th>Canadian Journal of Civil Engineering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manuscript ID</td>
<td>cjce-2018-0544.R1</td>
</tr>
<tr>
<td>Manuscript Type:</td>
<td>Article</td>
</tr>
<tr>
<td>Date Submitted by the Author:</td>
<td>15-Dec-2018</td>
</tr>
</tbody>
</table>
| Complete List of Authors: | Abuwarda, Zinab; University of Waterloo, Civil and Environmental Engineering  
Hegazy, Tarek; University of Waterloo, Civil & Env. Engineering; University of Waterloo, Civil & Env. Engineering |
| Keyword:          | constr. management < Construction, construction (CA) < Computer Applications, Optimization, Fast-Tracking, Scheduling |
| Is the invited manuscript for consideration in a Special Issue?: | Not applicable (regular submission) |
Multi-Dimensional Optimization Model for Schedule
Fast-Tracking without Over-Stressing Construction Workers

by
Zinab Abuwarda¹, and Tarek Hegazy²

1. Assistant Professor
   Department of Civil Engineering
   Port-Said University
   Port Fouad 42526, Egypt
   E-mail: zabuward@uwaterloo.ca

2. Professor, (Corresponding Author)
   Civil and Environmental Engineering Department
   University of Waterloo
   Waterloo, Ontario, N2L 3G1
   Tel: (519) 888-4567 ext: 32174
   E.Mail tarek@uwaterloo.ca
Multi-Dimensional Optimization Model for Schedule
Fast-Tracking without Over-Stressing Construction Workers

Zinab Abuwarda and Tarek Hegazy

ABSTRACT:

Fast-Tracking is an important process to speed the delivery of construction projects. To support optimum fast-tracking decisions, this paper introduces a generic schedule optimization framework that integrates four schedule acceleration dimensions: linear activity crashing; discrete activity modes of execution; alternative network paths; and flexible activity overlapping. Because excessive schedule compression can lead to space congestion and overstressed workers, the optimization formulation uses specific variables and constraints to prevent simultaneous use of overlapping and crashing at the same activity segment. To handle complex projects with a variety of milestones, resource limits, and constraints, the framework has been implemented using the Constraint-Programming (CP) technique. Comparison with a literature case study and further experimentation demonstrated the flexibility and superior performance of the proposed model. The novelty of the model stems from its integrated multidimensional formulation, its CP engine, and its ability to provide alternative fast-track schedules to strictly constrained projects without over stressing the construction workers.

Keywords: Construction, Schedule Compression, Acceleration, Substitution, Crashing, Overlapping, Fast Tracking, Multi-modes, Constraint Programming, Optimization.
INTRODUCTION

The strict deadlines and continuous changes in construction projects mandate efficient techniques to expedite project delivery. In the literature, project expedition has often been referred to as schedule acceleration, crashing, compression, or fast-tracking. In practice, when project delivery needs to be expedited, practitioners and researchers use one (or more) of four main strategies to fast-track the project:

*Linear Activity Crashing*: this decision involves reducing the duration of some activities (mainly critical ones) at the expense of extra direct cost. Typically, crashing is a form of manipulating the activity resources through working overtime hours, multiple shifts, weekends, and/or by adding more workers (overmanning). To apply this strategy, detailed analysis of the linear time-cost function of each activity is needed;

*Discrete Activity-Mode Substitution*: this decision involves selecting among alternative execution options. One example is to outsource the work instead of performing it with in-house resources. For each activity, this decision weighs between discrete options that range from a slow (cheap) mode to a fast (expensive) mode;

*Path-Substitution (Alternative Paths)*: this decision involves the replacement of a set of activities in series (i.e., part of the project network path) by an alternative one. An example is replacing the time consuming cast-in-situ activities (formwork, reinforcement, and concrete works) with faster prefabricated elements (receiving, installation, and alignment). The extra costs of switching to a faster path need to be considered; and
Activity Overlapping: this decision exploits the soft precedence relations between eligible activities and changes their work sequence from being in-series to being partially parallel, to save time. An example is to start running the electrical conduits on a slab when the rebar works are halfway done. Overlapping decisions, however, might involve risks that lead to possible rework and resource over-allocation when both activities use the same type of resource. Therefore, although overlapping doesn’t result in an increase in the direct cost, the time and cost of rework need to be considered.

In practice, expert project managers make combinations of these decisions to address aggressive fast-tracking needs. However, it is not possible in large complex projects with aggressive deadlines to simultaneously consider all the activities’ acceleration options, while satisfying the many constraints related to short-term and long-term milestones, the deadline, and resource limits. Yet, the possible congestion and workers stress associated with excessive fast-tracking can lead to less practical project plans and exacerbate the complexity of the scheduling and optimization tasks.

LITERATURE ON SCHEDULE COMPRESSION

The literature has extensive research on developing optimization models to handle one or more of the four fast-tracking strategies described earlier. This section thus discusses the literature efforts related to each strategy, highlighting the type of optimization technique used (heuristic procedures; traditional mathematical optimization such as Branch-and-Bound; metaheuristic evolutionary techniques; and advanced mathematical tools such as Constraint Programming, CP). Afterwards, this section ends with a
discussion of the integrated efforts that combine multiple strategies, followed by the efforts that handle large-scale problems.

Early studies dealt with linear activity crashing (e.g., Kelly 1961; Meyer and Shaffer 1963; Hendrickson and Au 1989; Pagnoni 1990; Fondahl 1961; Fulkerson 1964; Elmeghraby and Salem 1981; Hajdu 1996). These studies used traditional mathematical tools to find exact optimum solutions for small-size networks, but could not address large-size problems. Accordingly, heuristic methods (e.g., Siemens 1971; Hajdu 1996) were used and were suitable for large-size problems although these methods do not guarantee optimum solutions. With the surge of computational power and AI-based metaheuristic tools such as genetic algorithms, various schedule optimization models were developed. Although their solution quality is higher than heuristic methods, they generally require exponential processing time for medium-size problems and do not guarantee solution optimality (e.g., Feng et al. 1997; Li and Love 1997; Hegazy 1999).

In recent years, schedule compression using multimode discrete time-cost trade-off (TCT) analysis has been studied extensively. Good literature surveys of these efforts can be found in Demeulemeester and Herroelen (2002), Peteghem and Vanhoucke (2010), and Menesi et al. (2013). Recent contributions were made by Chakrabortty et al. (2016) using integer linear programming, Ashuri and Tavakolan (2013) using Shuffled Frog-Leaping metaheuristic optimization, Shahriari (2016) using Genetic Algorithms, Aminbakhsh and Sonmez (2016) using Pareto Front Particle Swarm, and Menesi et al. (2013), Menesi and Hegazy (2014), and Abuwarda and Hegazy (2016b) using Constraint Programming (CP).
The available path-substitution literature is very limited. Gasparini and Qassim (2003) used a superstructure project consisting of several optional interdependent activities and introduced a mixed-integer linear programming model to obtain the optimum set-up. A recent effort by Abuwarda and Hegazy (2016b), that is enhanced in this study also introduced a CP model to select among alternative network paths to determine the most suitable work packages for the project at the early planning stage.

Research on activity overlapping is mostly found in manufacturing. Sample efforts such as Krishnan et al. (1997), Eppinger (1997), Roemer and Ahmadi (2004), and Koyuncu and Erol (2015), examined the sensitivity of various tasks to overlapping and rework evolution. A good comparison among overlapping efforts can be found in Grèze et al. (2014) and Dehghan et al. (2015). In construction, studying overlapping of design-stage activities is more common (e.g., Bogus et al. (2006) and Srour et al (2013)). Khoueiry et al. (2013) optimized overlapping of design-to-construction activities, while Pena-Mora and Li (2001) developed a generic framework for overlapping of design-to-design, design-to-construction, and construction-to-construction activities. In all efforts, overlapping is assumed to expedite the project at the expense of potential rework. Many studies in the literature assign rework as an extension to downstream activities’ duration (e.g., Roemer and Ahmadi (2004), Berthaut et al. (2011), Grèze et al. (2014), Hazini et al. (2014), Khoueiry et al. (2013), Chua and Hossain (2010), Hossain and Chua (2014), and Dehghan et al. (2015)). They estimate rework time and cost as a linear function of the overlapping time, which is the approach used in this paper. Other interesting efforts by Berthaut et al. (2011) and Grèze et al. (2014) also assign the rework to the downstream activity and define overlapping options as discrete modes with associated
rework time and cost, then determine the best overlapping modes using linear programming and heuristic optimization, respectively. Rather than assigning rework only to the downstream activity, Gerk and Qassim (2008) extend the overlap period of both downstream and upstream activities by a factor that is linearly proportional to the degree of overlapping. Recently, some researches (e.g., Wak et al. (2016), Lim et al. (2014), Dehghan and Ruwnapura (2014), and Dehghan et al. (2015)) consider stochastic analysis and rework probability to perform efficient concurrent construction scheduling.

While the above research efforts dealt with individual dimensions in schedule compression and fast-tracking strategies, few other efforts addressed their integration. Roemer and Ahmadi (2004) presented a mathematical model that combined activity crashing and overlapping. Their study in manufacturing, however, considered a set of sequential activities. In other research, Gerk and Qassim (2008) formulated crashing, overlapping, substitution, and resource constraints into a mixed-integer linear model that uses the activity-on-arrow representation with Finish-To-Start relations only. In a later effort, Hazini et al. (2013) developed a heuristic acceleration method that integrates activity crashing, overlapping, and mode-substitution. The model suits small projects with linear time-cost activity relations, and was later enhanced in Hazini et al. (2014) using multi-objective evolutionary optimization. Both models, however, do not consider resource limits.

To produce fast quality solutions for large-scale optimization problems, an advanced mathematical optimization technique, Constraint Programming (CP) has been used increasingly in recent years. CP derives its speed and power from being a logic programming technique (Chan and Hu, 2002). CP has been successfully used to solve
complex combinatorial problems in a wide variety of domains, with particular advantages in scheduling problems (Hentenryck, 2002, and Heipcke, 1999)). To facilitate the use of CP, IBM developed a powerful implementation tool, IBM ILOG CPLEX Optimization Studio (Beck et al. 2011), which is utilized in this research. Interested readers are referred to Laborie and Rogerie (2008) and Laborie et al. (2009) for background on CP and its working mechanism. In the literature, various researchers reported the advantages of CP in solving a variety of resource-constrained scheduling problems (e.g., Liess and Michelon (2008), Liu and Wang (2008), Liu and Shih (2009), and Liu and Wang (2011)). To the authors’ knowledge, however, no CP effort exists that integrates the many facets of schedule compression covered in the proposed model of this paper.

RESEARCH METHODOLOGY

This paper proposes a comprehensive multi-dimensional schedule optimization framework that uses all schedule compression strategies combined to respond to variety of constraints related to deadline, milestones, and multiple resource limits, simultaneously. The optimization framework involves developing a mathematical representation of the scheduling options at five decision levels (Figure 1): (1) at the network level, defining project networks with multiple groups of alternative sub-paths (branches); (2) at the activity level, providing a new representation of the full spectrum of alternative resource assignments that integrates the activity optional multi-modes and possible linear-crashing options; (3) at the relationship level, defining flexible (soft) activity relationships that allow overlapping between activities; (4) at the project level, allowing various practical constraints; and (5) at the project level also, allowing single
and multiple optimization objectives. To facilitate better fast-tracking, the present model allows multiple acceleration strategies to be applied to some activities. However, to specifically consider the need to avoid overstressing resources during excessive fast-tracking, the model incorporates a representation of the activity segment in which a certain acceleration method is used, with constraints on allowing simultaneous acceleration methods to be used on the same activity segment. Based on a mathematical representation of the five levels in Figure 1, a constraint programming (CP) model is developed on the IBM ILOG CPLEX platform. Accordingly, the model output is the optimum combination of activity options and the associated schedule and resource details.

The following section provides a description of the new representations at the different levels, followed by the detailed mathematical model and multiple experimentation on a literature case study to validate the model and present its full capabilities and future extensions.

**NEW REPRESENTATION OF SCHEDULING DECISIONS**

Three new representations are proposed in this paper to define the wide range of scheduling options that the schedule optimization uses to determine an optimum schedule. These are as follows:

**Network-Level: Alternative Paths**

Multipath network is important to support path substitution decisions which involve the replacement of the early selected path by alternative ones as a corrective action to expedite late project delivery. The proposed network representation handles having multiple groups of multiple alternative paths (branches) and can select between them to
simulate the real complex network of construction projects with alternative structures. Precedence relationships that lead to alternative path in the network are depicted by broken arrows in Figure 1. These relationships are only imposed if their path is selected.

**Activity-Level: Detailed Time-Cost-Resource (TCR) Spectrum**

To identify all possible resource assignments for an activity, the proposed model proposes a new representation of alternative resource assignments that integrate the possible modes of execution with possible linear crashing strategies, as shown schematically in Figure 1. The activity discrete execution modes represent alternative construction technologies, ranging from slow-and-cheap to fast-and-expensive, with their duration plotted as points at the bottom of the spectrum (Figure 2). Within each mode, linear crashing options that use either overtime, overmanning, or multiple shifts can be applied to any part (segment) of an activity, as represented by the radial lines that start at each distinctive mode. For illustration, consider an activity that requires 800 workhours and its normal construction mode involves 10 workers over a duration of 10 days (10 workers x 8 hrs/day x 10 days = 800 hrs). To crash this activity to 9 days, the 10 workers will work (10 x 8) x 9 = 720 hrs, thus an extra 80 hrs of work are required as either overtime by the existing workers, or adding extra workers. Thus, various crashing options can be as follows:

1- Crashing by overtime: With 4 overtime hours daily by all the 10 worker, the 80 extra hours will require 2 days. Thus the activity will employ 10 workers, 12 hrs/day for two days, then 8 hrs/day for the remaining 7 days i.e., (10 x 12 x 2+10 x 8 x 7 = 800 hrs).
The overtime segment in this case is 2 days, and goes linearly per crashing day. Thus, to crash the activity to 8 days, the overtime segment will be \((2 \text{ crashing days} \times 2) = 4\) overtime days. Similarly, to crash the activity more to 7 days, the overtime segment will be \((2 \times 3\) days) = 6 days. Afterwards, to crash the activity to 6 days, the overtime strategy alone is not sufficient but it can be combined with an overmanning strategy; or

2- Crashing by overmanning: Using 3 extra workers, the 80 extra hours will require them to work 8 hrs/day for 3.33 days \((80/3/8)\) or roughly 4 days. Thus, the activity will employ 13 workers for 4 days, and 10 workers for the remaining 5 days, i.e., \((13 \times 8 \times 4 + 10 \times 8 \times 5) > 800\) hours. The overmanning segment in this case is 3.33 days (linearly per crashing day). To crash the activity more to 8 days, the overmanning segment will be \((2 \times 3.33\) days) \(\approx 7\) days. To crash the activity more than 8 days, the overmanning strategy alone is not sufficient but can be combined with an overtime strategy.

The calculation above uses a piecewise linear arrangement of the total hours over the days and the workers, with the crashing segments automatically calculated for each activity mode, thus forming a spectrum of the possible Time-Cost-Resource (TCR) arrangements of each activity, as shown in Figure 2. The TCR spectrum is a visual representation that shows all the crashing strategies for all the execution modes. Using the TCR spectrum, crashing is associated with a specific resource utilization plan, not only time and cost. The spectrum is thus used in the schedule optimization to select the activity mode, and within the selected mode, selects the optimum crew formation and the
applicable time segment. This is particularly important when the resources are limited and thus an overtime/overmanning strategies become important for fast-tracking. The spectrum is also important to facilitate relaxing some of the activities that become non-critical during the project fast-tracking. The example TCR spectrum in Figure 2 shows a piecewise-linear spectrum that visually represents multiple options, among which the schedule optimization can select from to crash activity 5 in the case study discussed later. This activity has three modes of 22, 16, and 12-day durations that are represented by the three circles. To simplify the figure, the crashing options within the 22-day mode only are shown. Many options are possible to crash the 22-day normal duration, day-by-day up to 15 days, including overtime, overmanning, or a combination of both, which can also be extended to multiple shifts and working weekends. As an input to this analysis, practical information about the range of options is first specified, including the maximum overtime hours (e.g., 4 hours/day); maximum additional workers (e.g., 2 workers/day), linear productivity-loss function (as proposed by Hazini et al. 2013); and the hourly rates for overtime and overmanning. As such, given any desired crashing level (e.g., 21, 20, to 15 days), a detailed calculation is done to draw all the points and branches of the spectrum (Figure 2) and determine the appropriate resource utilization strategy and its applicable activity segment(s). Branches (a) and (b) in the figure relate to two different crashing strategies. Branch (a) exhibits linear crashing using overtime from 22 days to 17 days, with the cost slope and the overtime segment are defined. Continuing the crashing from 17 days to 15 days, two additional workers were used (representing additional overmanning). Branch (b) shows another strategy starting with overmanning. Because only two extra workers are available, overmanning was sufficient to crash the
activity to 20 days, afterwards, additional overtime hours are needed to do further crashing.

**Relation Level: Overlapping using Flexible Relations**

The third component of the framework inputs (Figure 1) relates to activity overlapping, which is allowed by specifying the activity relationships that have flexibility ranges attached to them. As such, the concepts of flexible relations proposed in Abuwarda and Hegazy (2016a) has been adopted in this paper. In this representation, instead of using the common “FS 3” relationship that indicates a hard lag time of 3 days, a flexible relation is represented in the form of “FS Lag, ML”, where ML is the Minimum Lag. For example, “FS 3, 1” indicating that the initial lag of 3 days can be reduced to 2, or to a minimum of 1 day (i.e., ML = 1). Fig. 3 shows one hard relation (“FS 3”) and two soft relations (“FS 3, 1” and “FS 3, -1”). In the latter case, the lag can be either 3, 2, 1, 0, or -1 days. When the lag is decided to be -1 (as will be determined by optimization), overlapping occurs and entails some rework in the downstream activity, caused by possible coordination challenges.

As reported in most literature studies on overlapping, the calculation of rework time and cost related to overlapping can be challenging. Researchers simplify this calculation by adopting a simple assumption that the penalty cost of overlapping a pair of activities is directly proportional to overlapping time (Cho and Epinnger (2005); Gerk and Qassim (2008); Hazini et al. (2013, 2014); Dehghan and Ruwnapura (2014)). Similarly, the relation between overlap amount and rework time is linear and continuous. For simplicity, the present model adopts this assumption, although the formulation accommodates any
 functional relation between overlapping time and rework time and cost.

MODEL FORMULATION AND IMPLEMENTATION

With all the multi-dimensional options at the network, activity and relationship levels (Figure 1) identified, the model’s mathematical formulation has been coded using Constraint Programing language of the IBM ILOG CPLEX Optimization Studio 12.7. To facilitate inputs and outputs, the model is linked to Microsoft Excel and Microsoft Project. Details of the mathematical equations related to the decision variables, constraints and objective function are as follows:

Decision Variables: the decision variables for each activity $i$ are represented in Equations 1 to 7, as follows: The $Z_{ge}$ binary decision variable is used to include or exclude each alternative path $e$ (1 to $E$) for each group of alternative paths $g$ (1 to $G$) in the project and accordingly, the $X_i$ binary decision variable is used to include or exclude each activity $i$ (1 to $N$), depending on the selection of the path:

Path inclusion: $Z_{ge} \in \{0, 1\}$ $\forall g = 1, \ldots, G; e = 1, \ldots, E$

(1)

Activity inclusion: $X_i \in \{0, 1\}$ $\forall i = 1, \ldots, N$

(2)

As such, all activities in the network will have their $X_i = 1$, except for the activities on an excluded path. This is in addition to three variables (as shown in Figure 4) for activity mode and crashing decisions, activity scheduled start, and activity duration, as follows:
\[ Y_{ikq} \in \{0, 1\} \quad \forall \ i = 1, \ldots, N; \ k = 1, \ldots, K_i; \ q = 1, \ldots, Q_{ik} \quad (3) \]

Where, \( Y_{ikq} \) is a binary decision variable of the activity mode and crashing option, \( N \) is the number of activities, \( K_i \) is the number of modes for activity \( i \), \( Q_{ik} \) is the number of crashing options (e.g., overtime, and overmanning for mode \( k \) of activity \( i \)).

The example in Figure 4 shows sample activity data and a decision \( Y_{ikq} = 1 \) to indicate using mode 1, with crashing strategy 2 (overtime). Accordingly, the activity duration and cost are determined and then modified according to the overlapping strategy. Thus, activity start and accelerated duration become:

Activity Scheduled Start: \( SS_i \) positive integer value \( \forall \ i = 1, \ldots, N \quad (4) \)

Activity accelerated duration: \( d_i \) positive integer value \( \forall \ i = 1, \ldots, N \quad (5) \)

To have the flexibility to use different strategies individually or combined (as needed by the user), two more variables are defined as follows:

Binary variable of (not allowing/allowing) activity crashing: \( U_i \in \{0, 1\} \quad \forall \ i = 1, \ldots, N \quad (6) \)

Binary variable of (not allowing/allowing) activity overlapping: \( W_i \in \{0, 1\} \quad \forall \ i = 1, \ldots, N \quad (7) \)

**Constraint on alternative paths:** To make sure that only one alternative path is included, a simple constraint becomes necessary:
\[
\sum_{g=1}^{G} \sum_{e=1}^{E} Z_{ge} = 1
\]

(8)

**Constraint on excluded activities:** to ensure that all the activities on an excluded path \(e\) (its \(Z_{ge} = 0\)) are excluded, the value of the decision variable \(X_i\) of the excluded activities on this path must be set to zero through the following constraint:

\[
Z_{ge} - X_i = 0 \quad \forall i \in \text{Activities of path } e; \quad g = 1, \ldots, G; \quad e = 1, \ldots, E
\]

(9)

Meeting this constraint ensures that included activities have their \(X_i = 1\), while excluded activities have their \(X_i = 0\) and thus their durations and costs are zeroes.

**Activity mode selection and crashing constraints:** To consider activity mode selection, the model defines multiple modes \(k\) (from 1 to \(K_i\) modes) for each activity. Each activity mode has its linear time-cost spectrum that defines variety of crashing choices \(q\) (from 1 to \(Q_{ik}\)), as shown in the example of Figure 4. Each portion of the spectrum has its cost slope \((CS_{ikq})\) that facilitates cost calculations. As such, activity data is defined using a tuple with parameters \((NT_{ikq}, NC_{ikq}, CT_{ikq}, CS_{ikq}, S_{ikq}, (r_{ik1}, r_{ik2}, \ldots, r_{ikL}))\) that represent the normal time, normal cost, crash time, cost slope, overtime segment (per crashing day), and the amount of \(l\) resources (1 to \(L\)), respectively. This representation is powerful enough to enable the activity to have multiple modes and a piecewise-linear time-cost spectrum of each mode. To select a mode and a crashing strategy, the binary two-dimensional decision variable \(Y_{ikq}\) \((0, 1)\) indicates which activity mode is selected and which crashing strategy is selected (e.g., example in Fig. 4). To
make sure that only one mode of construction is used for each activity, a constraint is added as follows:

\[
\sum_{k=1}^{K_i} \sum_{q=1}^{Q_{ik}} Y_{ikq} = 1
\]

(10)

Based on the value of the \(X_i\) and \(Y_{ikq}\) decision variables, the normal duration of activity \(i\) is calculated as follows:

\[
NT_i = X_i \cdot \sum_{k=1}^{K_i} \sum_{q=1}^{Q_{ik}} NT_{ikq} \cdot Y_{ikq} \quad \forall \quad i = 1,...,N; \quad (11)
\]

Similar expressions to equation (11) are also used to determine the normal cost \(NC_i\), crash duration \(CT_i\), and cost slope \(CS_i\).

**Duration constraints:** In the model, each activity \(i\) can be crashed so that its duration \(d_i\) can take any integer value, from its normal duration (long and cheap) to its crash duration (short and expensive). Each duration value relates to a certain crashing strategy of either overtime, over-manning, or combination of both (as shown in Fig. 4). Thus, constraints are needed to provide upper and lower bounds for activity duration, as follows:

\[
NT_i \geq d_i \geq CT_i \quad \forall \quad i = 1,...,N; \quad (12)
\]

Where, \(NT_i\) and \(CT_i\) are the normal and crash durations of the activity, respectively. Thus, when the activity exhibits no crashing (i.e., \(U_i = 0\)), the activity duration become \((d_i = NT_i)\).
**Overlapping constraints:** To accommodate overlapping, each flexible relationship between an activity $i$ and its predecessor $P_i$ must be defined using three inputs: Relationship type (e.g., FS); Lag time ($Lag_{ip}$); and the Minimum Lag ($ML_{ip}$). For the activities with hard lag relations or in case of overlapping is not permitted for this relation (i.e., $W_i = 0$), then $ML_{ip}$ is equal to the $Lag_{ip}$. For the flexible relations, the model incorporates modified CPM equations to calculate the activities’ Scheduled Start times ($SS_i$) and Scheduled Finish times ($SF_i$). Having any type of relationship between activity $i$ and its predecessor $p_i$. Then, the following constraints become necessary:

Finish-to-Start: $SS_i \geq X_i(SF_p + ML_{ip} \cdot W_i + lag_{ip}(1-W_i)) \quad \forall p \in P_i \quad i = 1,\ldots,N; \quad (13)$

Start-to-Start: $SS_i \geq X_i(SS_p + ML_{ip} \cdot W + lag_{ip}(1-W_i)) \quad \forall p \in P_i \quad i = 1,\ldots,N; \quad (14)$

Finish-to-Finish: $SF_i \geq X_i(SF_p + ML_{ip} \cdot W_i + lag_{ip}(1-W_i)) \quad \forall p \in P_i \quad i = 1,\ldots,N; \quad (15)$

Start-to-Finish: $SF_i \geq X_i(SS_p + ML_{ip} \cdot W_i + lag_{ip}(1-W_i)) \quad \forall p \in P_i \quad i = 1,\ldots,N; \quad (16)$

Where the Scheduled Finish Time ($SF_i$) of an activity $i$ is the sum of the scheduled start time ($SS_i$) plus modified activity duration $D_i$ (defined later in Eq. 21). These equations are generic and can schedule the activities with multiple predecessors of various types.

To determine the overlapping implications (rework time and cost) on the schedule, a detailed evaluation of the “additional” overlapping that a schedule exhibits is required, as compared to the initial schedule. Thus, the scheduling process first evaluates an initial schedule using the same equations above but with all values of $U_i$ and $W_i$ being zero.
Accordingly, the Initial Start time ($IS_i$) and Initial Finish time ($IF_i$) for each activity are determined. These initial start and finish times are used to calculate the initial overlapping ($IO_{ip}$) between each activity $i$ and its immediate predecessor $p$, as follows:

$$IO_{ip} = \max(0, IF_p - IS_i)$$

(17)

Then, during the schedule optimization process, the scheduled overlap is determined as the difference between the Scheduled Finish of the predecessor $SF_p$ and the Scheduled Start $SS_i$ (calculated using Eqs. 13-16) of the Successor $i$, as follows:

$$SO_{ip} = \max(0, SF_p - SS_i)$$

(18)

Accordingly, the additional overlap ($O_{ip}$) due to schedule acceleration is computed, as follows (illustrated in Figure 5):

$$O_{ip} = SO_{ip} - IO_{ip}$$

(19)

An example of the calculations in Eq. 17-19 is illustrated in Figure 5, where an activity $i$ before acceleration is scheduled to overlap one day (initial overlap, Eq. 17) with its predecessor. After acceleration, the schedule is compressed by increasing activity $i$ overlap with its predecessor to 3 days. Thus, the cost of acceleration considers the addition overlap time of 2 days (Eq. 19).

**Rework constraint:** As discussed earlier in the literature, overlapping causes the duration of the successor activity to be extended by the expected rework time ($rw_{ip}$), which is proportional to the additional overlap ($O_{ip}$) time between activity $i$ and its predecessor $p$. This direct relation multiplies the additional overlap time by a constant rate $R_{ip}$. Any other function, however, can be easily incorporated into the present model. In the case of a successor activity that is overlapped with multiple predecessors, the
model calculates the amount of rework to the successor activity as the sum of individual rework in the predecessor activities (as in Cho and Eppinger 2005; Berthaut et al. 2011, Hazini et al. 2013, 2014), as follows:

\[
rw_i = \sum_{p \in p_i} rw_{ip} = \sum_{p \in p_i} R_{ip} \cdot O_{ip} \quad \forall \ p \in \overline{P_i} \quad \forall \ i = 1,...,N; \quad (20)
\]

To consider how the rework of an activity effects the start/finish of its successor(s), the final activity duration \( (D_i) \) becomes as follows:

\[
D_i = d_i + r w_i \quad ; \quad SS_i = SS_i + D_i \quad \forall \ i = 1,...,N;
\]

\[
SF_i = SF_i + D_i \quad \forall \ i = 1,...,N; \quad (21)
\]

**Resource availability constraints:** As typically done, the model assumes that activity resource demand is constant all over the activity duration and the rework extension. Accordingly, resource constraints (one per resource) are expressed in the model such that the sum of the demands of a resource \( (l) \) by all eligible activities in each day \( (t, \) from day 1 to the end of all tasks) must be within the resource availability limit, as follows:

\[
\sum_{i}^{K_i} \sum_{k=1}^{Q_{ik}} (r_{ikl} \cdot Y_{ikq}) \leq R_{lt} \quad \forall i \in \text{eligible activities in day } t; \ t = 1, ..., \max SF_i; l = 1, ..., L \quad (22)
\]

Where, \( r_{ikl} \) is the amount of resource \( (l) \) required by the combined decision of mode \( k \) and crashing strategy \( q \) of activity \( i \). \( R_{lt} \) is the amount of resource \( (l) \) available in day \( t \), and \( L \) is the number of critical resources.

**Integration constraints:** Several constraints may be needed for practical considerations. For example, constraints have been added to represent the desire to
avoid crashing and/or overlapping when a decision is made to change from the normal activity mode (mode 1) to any other mode (from 2 to $K_i$), as follows:

$$U_i + \sum_{k=2}^{K_i} Y_{ik1} \leq 1 \quad \text{and} \quad W_i + \sum_{k=2}^{K_i} Y_{ik1} \leq 1 \quad \forall \ i = 1,\ldots,N; \quad (23)$$

From another perspective, as overlapping and crashing are not mutually exclusive, they can be used simultaneously, but this intensive compression strategy may cause some difficulties during construction. In some situations, therefore, it may be necessary to avoid having both crashing and overlapping occurring simultaneously in one activity. Thus, a constraint may be used, as follows:

$$U_i + W_i \leq 1 \quad \forall \ i = 1,\ldots,N; \quad (24)$$

It is also possible to add a third optional constraint that if both crashing and overlapping occur simultaneously in one activity, they occur in different activity segments, as shown in Fig. 6 and suggested in the literature by Hazini (2013). In the figure, the second segment of the activity is crashed while the first and third segments overlap with other activities.

To consider for this case, a constraint is added to ensure that the activity duration is greater than or equal to the maximum overlap period between the activity and its predecessors (segment 1), plus the maximum overlap period between the activity and its successors (segment 3), plus the required time period to apply crashing (segment 2).

$$D_i \geq \max_{p \in p_i} O_{ip} + \sum_{k=1}^{Q_{ik}} \min_{q=1}^{Q_{ik}} \left( D_{ik} S_{ikq} (NT - d_i) + \max_{s \in S_i} O_{is} \right) \quad ; \quad \forall \ i = 1,\ldots,N; \quad (25)$$
Where $S_{i1q}$ is the overtime segment (per crashing day) required to apply crashing strategy $q$. For example, activity $i$ (e.g., activity 5 in the case study discussed later) uses 3 overtime days to crash the activity for 1 day (i.e., $S_{i1} = 3$ days). The use of activity segments in Figure 5 can possibly reduce rework and the stress on the work team by avoiding the extra congestion due to overlapping and overmanning crashing at the same activity segment. As shown in Figure 6, the rework time due to the overlapping between activity $i$ and its predecessor will be applied at the end of segment 2 where the activity crashing should be applied. This is based on the valid assumption that the selected crashing technique (overtime, overmanning etc.) can be applied also during the rework time to save the project duration.

It is important to mention that the constraint in Eq. 23 is always used in the model to ensure the exclusive use of substitution technique at the activity level, while either the optional constraint in Eq. 24 or the constraint in Eq. 25 is used in the model, based on project manager preference.

**Objective functions:** Based on the definition of the model's decision variables and the above basic constraints, three expressions have been used in the model as alternative objective functions in the optimization ($f_1 = \text{Project Duration}; f_2 = \text{Total Cost};$ and $f_3 = \text{Acceleration Cost}$). The function $f_1$, is represented as the maximum scheduled Finish Time $SF_i$ among all activities as expressed as follows:

$$\text{Project Duration (f1)} = \max SF_i \quad (26)$$

To calculate the total project cost during the optimization process, the extra cost due to activity crashing and mode substitution is considered in the calculation of direct cost whereas the extra costs of overlapping is treated as part of the penalties, as follows:
Total Cost \( (f_2) = Direct Cost + Indirect Cost + Penalties - Incentives \) \( (27) \)

Project direct costs is represented in Eq. 28 as the sum of all normal costs \((NC_i)s\) plus all crashing costs, which is the crash time \((NT_i - d_i)\) multiplied by the cost slope \(CS_i\).

\[
Activities' \ direct \ costs = \sum_{i=1}^{N} NC_i + CS_i.(NT_i - d_i) \quad (28)
\]

The total project indirect cost, on the other hand, is a multiplication of project-duration \((f_i, Eq. 26)\) by the indirect cost per day \((IC)\), which is an input to the model. Thus, \(Indirect \ Cost = f_i \times IC\). The third component of total cost is the penalties (Eq. 29) associated with exceeding the project deadline plus the cost of activity overlapping. For the cost of overlapping, the model assumes it has a direct relationship with the rework time \(R_{ip}\), overlapping cost \(C_{ip}\), and additional overlapping time \(O_{ip}\) between activity \(i\) and its immediate predecessor \(p\). The values of \(C_{ip}\), and \(R_{ip}\) are inputs of the model that can be estimated based on the project manager’s experience:

\[
Penalties = Penalty \ for \ exceeding \ deadline + Cost \ of \ overlapping \nonumber
\]

\[
= max (f_1 - Original \ Project \ Duration, 0) . P + \sum_{i=1}^{N} \sum_{p \in P_i} C_{ip}.R_{ip} \cdot O_{ip} \quad (29)
\]

Where, \((P)\) is the daily penalty (or liquidated damage) for every day beyond the deadline. Finally, the last component of the total cost in Eq. 27 relates to the incentive for early project completion, represented in the model as follows:

\[
Incentives = max (Original \ Project \ Duration - f_1 , 0) \cdot I \quad (30)
\]

Where, \((I)\) is the daily incentive (bonus $/day) for completing the project before the deadline. Based on the above equations, the total cost of schedule acceleration \(f_3\) is expressed, in terms of crashing, overlapping, and mode substitution as follows:

\[
Acceleration \ Cost \ (f_3) = Crashing \ Costs + Overlapping \ Penalties + Substitution \ Costs
\]
\[ N \sum_{i=1}^{N} CS_i \cdot (NT_i - d_i) + \sum_{i=1}^{N} \sum_{p \in p_i} C_{ip}R_{ip} \cdot O_{ip} + \sum_{i=1}^{N} (NC_i - NC_{i1}) \]  \hspace{1cm} (3)

Where, \( NC_i \) is the normal cost of the selected mode of activity \( i \) after optimization and \( NC_{i1} \) is the normal cost of the original mode of the activity (mode 1 with no applied crashing technique).

**COMPARISON WITH LITERATURE**

For validation and experimentation purposes, the developed CP model was applied to a case study that involves both crashing and overlapping to test the efficiency of the model solution and its ability to achieve better solutions. The case study was reported in Hazini et al. (2014) where the optimization was implemented using genetic algorithms. Their model, however, uses the relationships' overlap days as decision variables, and as such, the model is only suitable for schedule compression but cannot handle resource constraints, which require some activities to be delayed to avoid exceeding resource limits. The case study has an initial project duration of 110 days and involves both crashing and overlapping. Fig. 7b,c compares the CP results of the proposed model (Fig. 7c) with the genetic algorithm approach of Hazini et al. (Fig. 7b). The figure shows that CP provides a better solution, both in terms of a shorter schedule (86 days as opposed to 90, with less total cost), in a few seconds of processing time. The figure highlights the overlapped relations and the crashed activities as well.

**MODIFIED CASE STUDY**

To demonstrate the new capabilities of the proposed model, the case study was modified slightly. Figure 8 shows the project network with both soft and hard relations. The data of the activities' modes as well as the possible crashing information are summarized in...
Table 1 (data of activity 5 was used to draw the spectrum of Figure 2). Table 2 provides the rework time and cost associated with each additional overlapping day. For example, if the schedule introduces a 10-day overlap between activity 1 and activity 2, then the associated rework time and cost are 0.2 x 10 = 2 days and 0.2 x 10 x $900 = $1800, respectively.

The initial project duration was prepared with all activities using their initial normal (slow and cheap) modes, and a resource limit of 32 resources per day, resulting in a project duration of 110 days and a total cost of 515,496 (top of Figure 9). After preparing the case study as inputs to the optimization model, various optimization experiments were carried out. The objective of the first two experiments was to minimize total project cost, considering different levels of resource constraints. The optimization results are shown in Figure 9 where the model was able to reach a minimum total cost of $491,750 in 92 days of project duration, using the 32/day resource limit. The middle part of Figure 9 shows the resulting schedule with the selected strategy for each task. The resultant schedule has an optimum mix of crashing (activities 3 and 7), mode substitution (activity 4), and overlapping (activities 1, 2, and activities 1, 3). The second experiment was then carried out with a stricter resource limit of only 22 resources per day. The resulting schedule is shown at the bottom of Figure 9. As expected, the optimum schedule exhibits less use of overlapping to avoid resource over-allocation. This, however, comes at the expense of project duration which reached 110 days (same as initial schedule but using 30% less resources), while total cost ($511,940) is still less than the cost of the initial schedule. It is important to mention that the use of overlapping to compress the schedule is highly dependent on the resource limit. As opposed to most literature studies on overlapping,
the present model considers resource constraints in schedule compression.

Next optimization experiments targeted to minimize project duration (using a resource limit of 32 per day) and testing the advanced features of the model. Two additional experiments were performed, as follows:

(a) Allowing simultaneous crashing and overlapping on same activity segment; and

(b) Preventing simultaneous overlapping and crashing on the same activity segment to avoid workers’ overstress (i.e., use the optional constraint of Eq. 25).

In case (a), the model reached a minimum duration of 75 days with a total cost of $521,002 (Figure 10a). To reach this aggressive duration, some activities use both crashing and overlapping at the same activity segment. Activity 1, for example, uses overtime, yet is overlapped with activity 3 and activity 4.

The result of case (b) in Figure 10b, on the other hand, shows a project duration of 82 days with a total cost of $518,842, which is a little longer than case (a) but is cheaper and exhibits no simultaneous use of the crashing and overlapping strategies on the same activity segment, thus is better able to avoid over stressing the workforce. Based on the model results, formulating the crashing segment in the model facilitated combined crashing, overlapping, and substitution in any given tasks more effectively without over stressing the workers or causing excessive site congestion by allocating the overtime and overmanning strategies to the days in which the activity has no overlapping. Also, based on the results shown in Figures 9 and 10, the optimization
framework performed consistently with the set objectives and constraints and is able to use optimum combinations of a wide range of schedule compression strategies.

COMMENTS AND FUTURE ENHANCEMENTS

Based on the model formulation and results, some comments on the developments and future enhancements are as follows:

- The CP tool proved to be efficient and produces optimal solutions within seconds of processing time;

- The proposed model’s multi-dimensional formulation is another reason behind its powerful performance. The model has generic settings that can be set as “ON” or “OFF”. Accordingly, the model can work as any individual (or combination of) the dimensions discussed in the literature, including: project network with alternative structures; alternative activity modes; activity linear crashing options; alternative overlapping possibilities; resource allocation; time-cost trade-off, etc., in addition to variety of practical constraints;

- The model is able to consider all types of hard and soft (flexible) relationships (FS, SS, SF, and FF) and multiple-predecessors/successors in the overlapping calculations;

- The model uses the full spectrum of activity time-cost-resource options, to clearly define the activity mode and the specific resource implementation for any crashing option along with the applicable portion of the activity duration. This allows the model
to apply activity crashing and overlapping in different activity segments, thus avoids overstressing the workers and reduces the chances of rework;

- To facilitate the application of the model on new cases, and to simplify the entry of the large amount of needed data, particularly for large project, the implemented model has connections among Microsoft Project, Excel, and the CP optimizer to automatically pass the project data and the optimization results. The CP optimization code is also generic and does not need to be changed from one project to another. This automation effort will be extended in the future to have the model work as an add-on to standard project management software that is widely used by practitioners;

- It is noted that all schedule compression studies, including this one, consider that all activities (both short-term and long-term) as having similar priority for compression. In future work, therefore, it is worthwhile to study the value of giving higher priority to accelerating short-term activities to retain the schedule’s residual flexibility to accelerate the project more in the future;

- While the small examples in the paper were suitable to demonstrate and clearly follow the model results, future work will include detailed experiments on large-scale projects to test processing time and solution quality. Initial experiments with the model on a case study of over 1000 activities showed consistent superior performance, which confirms the ability of CP to handle real-life large-scale complex projects, as confirmed by other research by the authors (Menesi et al. 2013), and;

- Future improvements to the model include the use of dependent construction methods for some tasks, in addition to revising the model to allow activity splitting,
which improves the resource allocation and achieves better-leveled resources profiles;

- Another extension of this work is to perform a detailed rework analysis in cases of overlapping design-design, design-construction, and construction-construction activities. This analysis will consider the characteristics of the activities to accurately determine the probability of rework and its impact in terms of time and cost. The analysis also will study adding multiple intermediate reworks to the successor activity depending on the frequent exchange of the evolving information during the overlapping, which can also produce a rework to the predecessor activity.

- One additional ongoing extension to the model is to revise the representation of flexible relations by defining not only a minimum lag value, but also a maximum lag as well. Instead of using two relationships to represent relaxation and overlapping limits, as discussed by Kreter et al. (2016) and Neumann et al. (2003), a single flexible relationship can provide a range of options during optimization;

- While the proposed model focuses on reducing workers “overstress” that results from compressing an activity segment with combined overtime, overmanning, and overlapping, yet, the model does not represent the actual stress level of workers. Thus, one area of future enhancement is to represent the detailed workers’ stress levels as a decision variable. This can also be combined with the use of wearable devices to automate workers’ stress measurement under different work conditions; and also using agent-based simulation to model workers’ behaviors during fast-tracking, and accordingly improve the schedule optimization; and
Since the model was initially designed for project planning before construction, it is currently being extended to optimize schedules during construction, considering the various progress events of all parties that take place on a daily basis.

CONCLUSION

This paper proposed a comprehensive multi-dimensional schedule optimization model that has a combination of network-path substitution, multiple activity modes, piecewise-linear crashing options, and soft activity relations that allow overlapping. The proposed model uses constraint-programming (CP) to determine the optimum combination of schedule compression strategies. Several experiments on a case study project demonstrated the consistent performance of the optimization under different objective functions. The model’s results revealed that optimizing the schedules using a combination of crew formations, activity modes, overlapping, and network paths can meet strict constraints of deadline and limited resources with a considerable cost saving. The results also demonstrated that the optimization is capable of arriving at solutions that reduce workers stress by setting proper constraints to avoid simultaneous crashing and overlapping in same activity segment.

Future work will examine the performance of the model on large-scale construction projects and will apply schedule optimization during the execution phase of projects. This work represents a serious effort to introduce a comprehensive schedule optimization model that is flexible, easy to implement, and can provide the necessary decision support capabilities that are currently missing in existing project management software tools. Such effort can be extended to an integrated decision support system to support
the decision making in the construction industry, and a cost-benefit analysis tool to study the trade-off between the total project duration and cost.

REFERENCES


Dehghan, R., Hazini, K., Ruwanpura, J., 2015 “Optimization of overlapping activities in the design phase of construction projects”, Automation in Construction 59, 81–95, DOI: 10.1016/j.autcon.2015.08.004


Eppinger, S. D., 1997 “A planning method for integration of large-scale engineering systems”, Int. Conf. on Engineering Design, Tampere, Finland, 199–204,


Meyer, W. L., & Shaffer, L. R. 1963 "Extensions Of the critical path method through the application of integer programming", Civil Engineering and Construction Research Series No.2, University Of Illinois, Urbana-Champaign, ASIN: B00D1BCL8O.


List of Tables:

Table 1: Modes and crashing options

Table 2: Rework data
Table 1: Modes and crashing options

<table>
<thead>
<tr>
<th>Activity</th>
<th>Mode</th>
<th>Crashing option</th>
<th>Cost ($1,000)</th>
<th>Duration (days)</th>
<th>Resource need (r)</th>
<th>Productivity Loss %</th>
<th>Cost Slope (CS) /day</th>
<th>OT Segment (S) /crashing-day</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1. Normal</td>
<td>$ 76.4</td>
<td>36</td>
<td>10</td>
<td>10</td>
<td>$1,400</td>
<td>3.8 d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Overtime</td>
<td>$ 76.4</td>
<td>36 to 30</td>
<td>10</td>
<td>15%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1. Normal</td>
<td>$ 72.0</td>
<td>30</td>
<td>10</td>
<td>10%</td>
<td>$1,040</td>
<td>3.2 d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Overtime</td>
<td>$ 72.0</td>
<td>30 to 25</td>
<td>10</td>
<td>10%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1. Normal</td>
<td>$ 70.2</td>
<td>28</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Overtime</td>
<td>$ 72.0</td>
<td>28 to 21</td>
<td>10</td>
<td>10%</td>
<td>$ 656</td>
<td>2.6 d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. OTM*</td>
<td>$ 75.4</td>
<td>20 to 18</td>
<td>12</td>
<td>10%</td>
<td>$ 276</td>
<td>18.0 d</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1. Normal</td>
<td>$ 81.6</td>
<td>48</td>
<td>10</td>
<td>20%</td>
<td>$ 1,760</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Overtime</td>
<td>$ 81.6</td>
<td>48 to 40</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1. Normal</td>
<td>$ 65.4</td>
<td>22</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Overtime</td>
<td>$ 65.4</td>
<td>22 to 17</td>
<td>10</td>
<td>17%</td>
<td>$ 961</td>
<td>3.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. OTM</td>
<td>$ 70.9</td>
<td>16 to 15</td>
<td>12</td>
<td>17%</td>
<td>$ 527</td>
<td>15.0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1. Normal</td>
<td>$ 66.8</td>
<td>24</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Overtime</td>
<td>$ 66.8</td>
<td>24 to 19</td>
<td>10</td>
<td>15%</td>
<td>$ 1,184</td>
<td>3.5</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1. Normal</td>
<td>$ 68.0</td>
<td>20</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Overtime</td>
<td>$ 68.0</td>
<td>20 to 16</td>
<td>10</td>
<td>20%</td>
<td>$ 1,120</td>
<td>2.6 d</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3. OTM</td>
<td>$ 73.4</td>
<td>15 to 12</td>
<td>12</td>
<td>20%</td>
<td>$ 658</td>
<td>14.0 d</td>
</tr>
</tbody>
</table>

*OTM = Overtime + Overmanning
### Table 2: Rework data

<table>
<thead>
<tr>
<th>Activity ID</th>
<th>Predecessor ID</th>
<th>Rework Time</th>
<th>Rework Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0.2</td>
<td>$900</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0.1</td>
<td>$850</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0.15</td>
<td>$1050</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>0.30</td>
<td>$800</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>0.25</td>
<td>$1000</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.2</td>
<td>$1000</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>0.15</td>
<td>$950</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>0.05</td>
<td>$950</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.25</td>
<td>$950</td>
</tr>
</tbody>
</table>
List of Figures:

Figure 1: Components of the proposed schedule optimization framework

Figure 2: Sample activity piecewise-linear crashing spectrum

Figure 3: Representation of a flexible relationship that allow overlapping

Figure 4: Key Decision variables

Figure 5: Calculation of the additional overlaps in Eqs. 17-19

Figure 6: Avoiding multiple acceleration strategies in one activity segment

Figure 7: Comparison of results with a literature case study

Figure 8: Network of the modified case study

Figure 9: Results of the cost minimization experiments

Figure 10: Results of duration minimization experiments
Figure 1: Components of the proposed schedule optimization framework
Figure 2: Sample activity piecewise-linear crashing spectrum

215x279mm (300 x 300 DPI)
Figure 3: Representation of a flexible relationship that allow overlapping

215x279mm (300 x 300 DPI)
Figure 4: Key Decision variables

215x279mm (300 x 300 DPI)
Figure 5: Calculation of the additional overlaps in Eqs. 17-19

215x279mm (300 x 300 DPI)
Figure 6: Avoiding multiple acceleration strategies in one activity segment

215x279mm (300 x 300 DPI)
Figure 7: Comparison of results with a literature case study
Figure 8: Network of the modified case study

215x279mm (300 x 300 DPI)
Figure 9: Results of the cost minimization experiments
Figure 10: Results of duration minimization experiments