Journal of Scheduling
Manuscript Draft

Manuscript Number: JOSH366

Title: Robust Optimization Models for the Discrete Time/Cost Tradeoff Problem

Article Type: Application and Methodologies for P&S

Keywords: Project scheduling; Time/Cost Trade-off; Robust Optimization; Interval Data; Benders Decomposition; Tabu Search

Robust Optimization Models for the Discrete Time/Cost Tradeoff Problem

Öncü Hazır[*,1,2] , Erdal Erel[1], and Yavuz Günalay[3]

[1] Faculty of Business Administration, Bilkent University, 06800, Ankara – Turkey
e-mail: oncuh@bilkent.edu.tr, erel@bilkent.edu.tr

[2] Industrial Engineering Department, Çankaya University, 06530, Ankara – Turkey
e-mail: hazir@cankaya.edu.tr

[3] Faculty of Business, Bahçeşehir University, 34100 Beşiktaş – İstanbul-Turkey
e-mail: yavuz.gunalay@bahcesehir.edu.tr

June 23, 2008

Abstract

The achievement of project goals is affected by various sources of uncertainty. Therefore developing models and algorithms to generate robust project schedules that are less sensitive to disturbances resulting from uncertain factors are essential. This paper addresses robust scheduling in project environments; specifically, we address the discrete time/cost trade-off problem (DTCTP). We formulate the robust DTCTP with three alternative optimization models in which interval uncertainty is assumed for the unknown cost parameters. We develop exact and heuristic algorithms to solve these robust optimization models. Furthermore, we compare the schedules that have been generated with these models on the basis of schedule robustness.

Keywords: Project scheduling, Time/Cost Trade-off, Robust Optimization, Interval Data, Benders Decomposition, Tabu Search

[*] Corresponding author: Öncü Hazır,
Address: Industrial Engineering Department, Çankaya University, 06530, Ankara – Turkey
Tel: (90) 312 2844500-4013, Fax: (90) 312 2848043 e-mail address:hazir@cankaya.edu.tr

1. Introduction

In project management, it is often possible to reduce the duration of some activities and therefore expedite the project with additional costs. In this paper, we consider discrete time/cost relationships and address the discrete time/cost tradeoff problem (DTCTP), which is a well-known multi-mode project scheduling problem with practical relevance. Three versions of the problem have been studied in the literature; namely, the deadline problem (DTCTP-D), the budget problem (DTCTP-B) and the efficiency problem (DTCTP-E). In DTCTP-D, given a set of time/cost pairs (modes) and a project deadline of δ, each activity is assigned to one of the possible modes so that the total cost is minimized. Quite the opposite, the budget problem minimizes the project duration while meeting a given budget, B. Lastly, DTCTP-E is the problem of generating efficient time/cost points over the set of feasible project durations. We address the deadline version in this manuscript.

Formally, the DTCTP can be defined as follows: Given a project with a set of n activities along with a corresponding precedence graph $G = (N, A)$, where N is the set of nodes, which includes n activities as well as two dummy nodes, 0 and $n+1$, that are used to indicate the project start and completion instants. $A \subseteq N \times N$ is the set of arcs, which represents the immediate precedence constraints among activities. Each activity j ($j = 1,...,n$) can be performed at one of the $|M_j|$ modes where each mode $m \in M_j$, is characterized by a processing time p_{jm} and a cost c_{jm}.

A mixed integer-programming model of the DTCTP-D can be stated as follows:

$$\text{Min} \sum_{j=1}^{n} \sum_{m \in M_j} c_{jm}\, x_{jm} \qquad (1.0)$$

Subject to

$$\sum_{m \in M_j} x_{jm} = 1 \qquad j = 1,...,n \qquad (1.1)$$

$$C_j - C_i - \sum_{m \in M_j} p_{jm}\, x_{jm} \ge 0 \qquad \forall (i,j) \in A \tag{1.2}$$

$$C_{n+1} \le \delta \tag{1.3}$$

$$C_j \ge 0 \qquad \forall j \in N \tag{1.4}$$

$$x_{jm} \in \{0,1\} \qquad \forall m \in M_j,\ j = 1,...,n \tag{1.5}$$

C_j is the continuous decision variable that symbolizes the completion time of activity j. The binary decision variable x_{jm} assigns modes to the all activities at (1.5). Total cost should be minimized (1.0), while the following constraints are satisfied: a unique mode should be assigned to each activity (1.1); precedence constraints should not be violated (1.2); and the deadline should be met (1.3).

De et al. (1997) has shown that the DTCTP strongly NP-hard. In their survey paper in 1995, they review the problem characteristics, as well as exact and approximate solution strategies. While Demeulemeester et al. (1996, 1998) propose branch and bound to solve the problem exactly, Skutella (1998), Akkan et al. (2005), and Vanhoucke and Debels (2008) propose approximate algorithms.

All the above mentioned studies assume complete information and a deterministic environment; however projects are often subject to various sources of uncertainty that threaten achievement of project objectives. For that reason, it is vital to develop effective robust scheduling algorithms. To minimize the effect of unexpected events on project performance, five fundamental scheduling approaches have been discussed in the literature: stochastic scheduling, fuzzy scheduling, sensitivity analysis, reactive scheduling, and robust (proactive) scheduling (Herroelen and Leus, 2005). In stochastic project scheduling, the activity durations are modeled as random variables and probability distributions are used. Fuzzy project scheduling uses fuzzy membership functions to model activity durations. The effects of parameter changes are investigated in sensitivity analysis. In reactive scheduling, the schedule is modified when a disruption occurs, whereas in robust scheduling anticipation of variability is incorporated into the schedule and schedules that are insensitive to disruptions are generated.

In this paper, we concentrate on robust scheduling and make use of robust optimization to generate robust project schedules. *Robust optimization* is a modeling approach to generate a plan that performs well even in the worst-case scenarios. Robust optimization has attracted many researchers and has been applied to some well-known combinatorial optimization problems such as the shortest path problem during the last decade (Bertsimas and Sim (2003). However, it has been implemented in only a few project scheduling problems as discussed below.

Valls et al. (1998) examine a special resource constrained project scheduling problem (RCPSP) in which the activities might be interrupted for an uncertain period. Yamashita et al. (2008) address the resource availability cost problem (RACP), which minimizes a non-decreasing discrete cost function of resources under the constraint that the project has to be finished by a given deadline. They propose two alternative models: the first model minimizes the sum of the mean and variance of the costs, whereas the second one minimizes the maximum regret function. Both Valls et al. (1998) and Yamashita et al. (2008) follow a scenario-based approach, where a scenario represents a realization of the duration of the activities. On the other hand, Cohen et al. (2008) use interval uncertainty in their recent robust scheduling study that addresses the effects of uncertainty on the continuous time-cost trade-off problem. They model the robust problem using the ARC methodology of Ben-Tal et al. (2004); some of the variables are determined before the realization of the uncertain parameters (non-adjustable variables), while the other variables could be decided after the realization (adjustable variables).

We propose three robust optimization models in which uncertainty is modeled via intervals for the DTCTP-D. Our research differs from the previous studies in the literature regarding both the problem addressed and uncertainty modeling approach followed. Our models address the uncertainty in activity costs. In practice, fluctuations in the exchange rates, factor prices or resource usages result in cost variability. These fluctuations threaten achievement of project cost objectives and it is essential to develop systematic methods to generate robust project schedules, which are less sensitive to uncertainty. We develop exact and heuristic algorithms to solve these robust models and compare the schedules that have been generated with these models on the basis of schedule robustness. The main contribution is the incorporation of uncertainty into a practically relevant project scheduling problem and developing problem specific solution approaches. To the best of our knowledge, this research is the first implementation of robust optimization to the DTCTP.

In the next section, we formulate the robust DTCTP-D using alternative robust optimization models. We propose exact and heuristic algorithms to solve these robust models in section 3. Finally in section 4, we present some computational results and compare the robustness of the schedules generated with these models using some robustness metrics.

2. Robust Discrete Time/Cost Trade-Off Problem with Interval Data

In many real-life projects a tardiness penalty or an opportunity cost may be incurred for each additional time unit the project is late. The cost may include explicit monetary charges, foregone revenue, lost profits, or goodwill losses. Due to these potential costs and early completion benefits, organizations seek on-time completion and aggressively monitor actual progress of these activities. The model proposed in this section addresses project environments in which timely completion of project activities is crucial. A frequently encountered practice that favors early completions is Build-Operate-Transfer (BOT) projects.

BOT model has been widely accepted in both developed and developing countries for the last few decades as it functions as an alternative financing mechanism in undertaking large investment projects.

In BOT projects, a public service or an infrastructure investment is performed and operated for a specific period by a private enterprise, and then transferred to a public institution. The operating period is usually long, often more than 10 years, so that the investment could be paid off. To give an example of the model, a firm constructs a private toll road, and operates it for some time and then transfers the right to operate to public. If the firm could complete construction earlier, it can extend the operating period and therefore increase the profit.

When deviations from the baseline plan are observed and they are judged to threaten the completion of these activities on time, project managers usually allocate extra resources such as additional workers or extra machinery to these activities. As the activity durations generally depend on the resource allocation, it is usually possible to achieve time plan goals by allocating additional resources. These additional allocations create fluctuations in the amount of resources allocated to each activity and result in cost uncertainty. They also seriously affect the profitability of the projects. From this point of view, protection against deviations in total cost becomes the key concern of project managers. Hence, in this paper, we fix the activity durations and assume that activity costs can take values in the intervals, i.e., $c_{jm} \in \left[\underline{c_{jm}}, \overline{c_{jm}}\right], \forall m \in M_j, j = 1,...,n$.

The traditional minmax (absolute robustness) criterion focuses on the worst-case solution, which corresponds to the scenario where the cost of each activity, c_{jm} is replaced by $\overline{c_{jm}}$, the upper bound of the corresponding interval. Optimization with respect to absolute robustness criterion is equivalent to the classic DTCTP-D with $c_{jm} = \overline{c_{jm}}$. However, this approach of robustness is extremely pessimistic and most likely it is unrealistic. A more realistic approach would be modeling the uncertainty only over a subset of the scenario space. One recent application of this restriction idea is the robust discrete optimization approach proposed by Bertsimas and Sim (2003). They assume that only a subset of the uncertain parameters is allowed to deviate from their estimates; in other words, only Γ of activity cost parameters (out of a total of n) involve random behavior. If $\Gamma = 0$, the influence of the cost deviations is ignored and the deterministic problem with nominal cost values is obtained. In contrast, if $\Gamma = n$, maximal cost deviations are considered and the problem becomes a minmax optimization problem. Therefore, Γ can be regarded as a parameter that reflects the pessimism level of the decision maker. High values of this parameter indicate a risk-averse decision making behavior.

2.1. Model 1

In this section, a MIP model for DTCTP-D using Bertsimas and Sim's approach is presented. We assume that at most $0 \leq \Gamma \leq n$ activities have cost values at their upper bounds and the remaining n-Γ coefficients are forced to be deterministic, i.e., they are set to nominal values: $c_{jm} = \dfrac{\overline{c_{jm}} + \underline{c_{jm}}}{2}$ and $d_{jm} = \overline{c_{jm}} - c_{jm}$, $\forall m \in M_j$, $j = 1,\ldots,n$ .. The restricted uncertainty model, which will be called Model 1 from now on, could be expressed by the following nonlinear formulation:

$$f(\Gamma) = \text{Min}\left\{\sum_{j=1}^{n}\sum_{m\in M_j} c_{jm}x_{jm} + \text{Max}\left\{\sum_{j=1}^{n}\sum_{m\in M_j} d_{jm}x_{jm}u_j : \sum_{j=1}^{n} u_j \leq \Gamma, u \in B^n\right\} : x \in X^D\right\} \tag{2}$$

In this formulation, X^D denotes the set of feasible solutions to the DTCTP-D and the set of coefficients, which are subject to uncertainty, are determined by the binary vector u, i.e. B^n refers to n dimensional binary vector and u chooses Γ of the activities which have the largest cost deviations among n activities. Model 1 assumes the activities with the same cost intervals are equally uncertain and all activities are likely to have cost values at the upper bounds. In real life, criticalities of the activities are crucial as well. Considering this, we propose two additional novel models that integrate the criticality of project networks and then compare the performances of all the three models.

2.2. Criticality-Based Uncertainty Models

In the pervious model, Model 1, we assumed that a subset of the activities has cost values at their upper bounds and the remaining coefficients are set to their nominal values. While determining this subset, only the activity costs are considered but the activity durations are ignored. However, cost and time are interdependent as they both depend on the amount of resource allocation. Activities having large slacks (i.e. non-critical activities) provide flexibility in resource allocation. It is possible to delay their starting times or to elongate the durations via lowering the amount of resource allocations. Due to these flexibilities, these activities involve less risk to achieve the cost targets when compared to the critical activities. On the other hand, in case of disruptions, managers usually allocate more resources to critical activities or in managerial terms crash these activities and this requires extra cost.

The conventional measure of an activity's criticality is the total slack, which is the amount of time by which the completion time of the activity can exceed its earliest completion time without delaying the project completion time. It is a measure of the insensitivity of schedule performance with respect to activity delays. The activities that have no slacks are defined to be critical activities. We propose a new criticality definition that is different from the conventional measure.

In real-life projects, it would make much more sense to evaluate the activity slacks with respect to activity’s duration since the higher the ratio of slack to activity time the higher its capability to compensate for a delay. The reason is that as the activity durations increase the probability of a larger number of disruptions to be observed while the activity is being performed, increases. Thus, we use the slack/duration ratio to assess criticality of activities and define the activities that have slack values less than *100ξ %* of the activity duration as *potentially critical activities*, i.e. $CR = \{ j = 1,...,n \ : TS_j / p_j \leq \xi \}$ where TS_i refers to the total slack of activity *i*. *100ξ %* will be called slack duration threshold (SDT) from now on. In this study, we set the SDT to *25 %,* i.e. $\xi = 0.25$. When compared with the classical definition, this new definition enlarges the criticality set.

2.2.1. Model 2

Model 1 is unrealistically pessimistic as the activity slacks are disregarded and the worst-case costs are allocated to activities with ample slacks make. To eliminate this over pessimism, the activities with cost values at the upper bounds are chosen from the critical ones in the criticality-based robust model. Given the mode assignments, only Γ controls the pessimism level in model 1, however in the new model the critical activity set and Γ control the pessimism level. The following model represents the criticality-based approach.

$$f(\Gamma) = \text{Min}\left\{ \sum_{j=1}^{n} \sum_{m \in M_j} c_{jm} x_{jm} + \text{Max}\left\{ \sum_{j=1}^{n} \sum_{m \in M_j} d_{jm} x_{jm} u_j \ : \ \sum_{j \in CR} u_j \leq \Gamma, u \in B^n \right\} : x \in X^D \right\} \tag{3}$$

Note that the only difference between (1) and (2) is the set of activities which can have cost values at the upper bounds. *CR* refers to the set of potentially critical activities in (3). In this new approach, criticality definition becomes crucial. For our problem, slacks are defined with respect to a specific mode assignment. As the mode assignments change, the slack distribution among the activities also changes. To illustrate the differences between the models, we use the simple network in Figure 1, which is adapted from the example of De et al. (1995). The project has a deadline of $\delta = 6$. Each activity has two mode alternatives characterized by the triplet, (p_{jm}, c_{jm}, $\overline{c_{jm}}$) shown above the nodes.

[Insert Figure 1 here]

Table 1 depicts the objective function values of the deterministic models ($\Gamma = 0$) and two robust models (with $\Gamma = 1, 2$). The rows of the table show the feasible mode combinations for the activities. Optimal solutions are marked with "*" and given a mode combination, the critical activities are underlined. Note that activity slacks depend on mode assignments.

[Insert Table 1 here]

It can be observed from the table that Γ and $\Gamma_{CR,}$ which control the level of pessimism in the restricted uncertainty model and criticality-based model respectively, are effective on the choice of activity modes. When both of the models are compared, Model 2 is less pessimistic than Model 1 as it considers less risk premium in costs. The following proposition proves this fact.

Proposition 1: Model 1 is more pessimistic than Model 2.

Proof: See Appendix A

Corollary 1: The optimal solution of Model 2 provides a lower bound to Model 1.

Proof: See Appendix B

In the following subsection, we propose an alternative criticality-based approach, which lies in between the Model 1 and Model 2 regarding the level of conservatism.

2.2.2. Model 3

This model also accounts for the cost deviations in the non-critical activity set, but unlike the first criticality model, the critical activities have priority over non-critical ones. Given the mode combinations, Γ activities which have cost values at their upper bounds are chosen from the critical activity set firstly. If the cardinality of the critical activity set is less than Γ, the remaining units are chosen from the non-critical activity set. While calculating the cost deviations both for the critical and non-critical activities, activities that have the cost coefficients influencing the objective most are given the priority.

3. Solution Algorithms

3.1. Benders Reformulation of Model 1

In this section, we show how to solve Model 1 using a Benders Decomposition algorithm.

Proposition 2: Model 1 could be formulated as follows:

$$\text{Min} \sum_{j=1}^{n} \sum_{m \in M_j} c_{jm} x_{jm} + z$$

subject to

$$\begin{aligned}
& z - \sum_{j=1}^{n} \sum_{m \in M_j} u_j^k d_{jm} x_{jm} \geq 0 && k = 1, \ldots, K \\
& \sum_{j=1}^{n} \sum_{m \in M_j} w_j^s p_{jm} x_{jm} \leq \delta && s = 1, \ldots, S \\
& \sum_{m \in M_j} x_{jm} = 1 && j = 1, \ldots, n \qquad (4) \\
& x_{jm} \in \{0, 1\} && \forall m \in M_j, \; j = 1, \ldots, n \\
& z \geq 0,
\end{aligned}$$

where, $u^k = (u_1{}^k, \ldots, u_n{}^k)$, for $k = 1, \ldots, K$ are the extreme points of the polytope $U = \{ u \in R^n : \sum_{j \in N} u_j \leq \Gamma,\ 0 \leq u_j \leq 1,\ j = 1, \ldots, n \}$. S refers to the total number of paths between node 0 to $n+1$ in $G\ (N, A)$ and w_j^s, $j = 1, \ldots, n$ is the elements of the node-path incidence vector, w^s of the path s, $s = 1, \ldots, S$; i.e., $w_j^s = \begin{cases} 1 & \text{if node } j \text{ belongs to path } s \\ 0 & \text{otherwise} \end{cases}$

Proof: See Appendix C

Enumerating all the extreme points and paths is burdensome, so we use a relaxation approach and generate the constraints as needed. We propose the following Benders Decomposition algorithm to solve the problem exactly.

Solution Algorithm:

Introduce an additional index t to the notation to denote the values at iteration t.

1. Start with an initial solution,

   $\overline{x^1} \in X^0 = \{ (x_{11},...,x_{1|M_1|},...,x_{n1},...,x_{n|M_n|}) : x_{jm} \in B, \forall m \in M_j, j = 1,...,n; \sum_{m \in M_j} x_{jm} = 1, j = 1,...,n\}$

   Set $z^0 = -\infty$, $t = 1$.

2. Solve $SP_1{}^t(\overline{x^t}) : \eta^t = \underset{s=1,...,S}{\text{Max}}\{\sum_{j=1}^{n}\sum_{m \in M_j} w_j^s p_{jm} \overline{x_{jm}^t}\}$ and

   Solve $SP_2{}^t(\overline{x^t}) : \psi^t = \text{Max}\{\sum_{j=1}^{n}\sum_{m \in M_j} d_{jm} \overline{x_{jm}^t} u_j : \sum_{j=1}^{n} u_j \le \Gamma , 0 \le u_j \le 1, j = 1,...,n \}$

   let u^t be the optimal solution.

3. If $\eta^t > \delta$ then

   Find the longest path and its incidence vector $\overline{w^t}$ and set $\overline{u^t} = u^t$.

   $X^t = X^{t-1} \cap \{ x \in X^0 : \sum_{j=1}^{n}\sum_{m \in M_j} \overline{w_j^t} p_{jm} x_{jm} \le \delta \}$

   If ($\psi^t > z^{t-1}$)

   $X^t = X^t \cap \{ x \in X^0 : z \ge \sum_{j=1}^{n}\sum_{m \in M_j} \overline{u_j^t} d_{jm} x_{jm} \}$

   Else

   If ($\psi^t > z^{t-1}$)

   $X^t = X^{t-1} \cap \{x \in X^0 : z \ge \sum_{j=1}^{n}\sum_{m \in M_j} \overline{u_j^t} d_{jm} x_{jm}\}$

   Else

   Stop and report $\overline{x^t}$ as the optimal solution.

   End if

4. Solve the relaxed master problem, MP^t:

$$\varphi^t = \text{Min } \{\sum_{j=1}^{n} \sum_{m \in M_j} c_{jm} x_{jm} + z : x \in X^t\}.$$

Let x^t be the optimal solution and z^t be the optimal objective function.

5. $t = t+1$, $\overline{x^t} = x^{t-1}$.

6. Return to Step 2.

Note that solving the first subproblem, namely $SP_1(\bar{x})$ is equivalent to determining the length of the critical path (C_{n+1}) with respect to the given mode assignment. If $C_{n+1} \le \delta$, the problem is feasible, otherwise it is infeasible. In addition to the feasibility cuts, optimality cuts are inserted through solving an additional LP, $SP_2(\bar{x})$. A greedy algorithm could provide a solution that is close to optimum, easily. This algorithm orders $\sum_{m \in M_j} d_{jm} \overline{x_{jm}}$ values and this order identifies Γ of the activities that affect the objective function the most. Benders Decomposition is known to exhibit slow convergence. Therefore, we include several features to accelerate its convergence and solve large scale problem instances optimally.

Algorithmic Enhancements:

The computational efficiency of Benders Algorithm depends on three issues: (i) the number of iterations; (ii) the time needed to solve the SP at each iteration; (iii) the time needed to solve the MP at each iteration.

In order to decrease the number of iterations, we add multiple cuts to the model at each iteration. Instead of finding the critical path and adding a single feasibility cut at each iteration, we determine the longest K paths and insert K cuts. To find these paths, we use a modification of the well-known Yen's (1971) K-shortest loopless paths (KSP) algorithm. KSP lists K shortest paths between a given source-destination pair in the directed graph without revisiting the same node (loopless paths). Note that since PERT networks are acyclic, critical path problems may be solved as shortest path problems with cost parameters equal to the negative of the activity durations. K is in fact a parameter, which affects the computational efficiency. It is observed that as K increases, the number of iterations required attaining global convergence decreases, whereas the time and the computer effort demanded for solving each MP at each iteration increases. In order to decide on the level of parameter K, we performed 30 pretests involving problems with different sizes. As a result, we decided on the following strategy: K is fixed to 2 in the first 10 iterations, later K is increased at each iteration until $K = 15$.

The major difficulty in this decomposition lies in the solution of the MP, which may become a very large 0–1 programming problem. To solve the MP efficiently, we propose the following enhancements:

1) We use preprocessing to eliminate some of the modes so that the number of decision variables in the MP is decreased. We apply the preprocessing techniques by Akkan et al. (2005) to eliminate long and short modes.

2) We solve the LP relaxations first and generate cuts from the fractional solutions. Then, integrality constraints are added and the algorithm is restarted (McDaniel and Devine, 1977).

3) Not all of the master problems are solved to optimality, i.e. feasibility cuts are generated from heuristic solutions. However, the last MP should be solved to optimality.

4) In order to solve the MP efficiently, we develop a customized branch-and-cut algorithm that groups the variables as special ordered sets (SOS) and that incorporates an effective neighborhood search strategy.

We solve the MP iterations approximately at a relative optimality tolerance level of $\varepsilon = 2\ \%$ until feasibility is satisfied. In the test runs, we tried $\varepsilon = 1, 2, 3, 4, 5\ \%$. The algorithm run fastest with $\varepsilon = 2\ \%$. This means that the incumbent solution of each MP is guaranteed to be within $2\ \%$ of the optimal value. In the following sections, we report the quality and the computational effort requirements of this solution as the *truncated solution.* From this point on, we solve the MP iterations to optimality so that they provide a lower bound on the cost of the optimal solution.

In this paper, we branch on sets of variables instead of branching on individual variables. A set of variables, in which at most one variable in the set is allowed to be nonzero, forms a "Special Ordered Set of Type 1" (SOS_1). In the DTCTP, the constraint set (1.1) represents an SOS_1. Branching strategies have large impacts on the size of the branch-and-bound tree and the computation time. Branching on SOS_1 instead of a single variable has some advantages such as the tree is more balanced (Linderoth and Savelsbergh (1999)). We assign an order to the variables among SOS_1 by using the activity durations as weights.

In our algorithm, the MP includes knapsack constraints. Previous computational studies show that cover cuts are effective for problems with knapsack constraints (Atamturk and Savelsbergh, 1999). Cutting planes derived from knapsack constraints may be strengthened by using SOS_1. These strengthened constraints are called Generalized Upper Bound Constraints (GUBs).

We develop the branching tree using optimization software CPLEX. Branching nodes are selected using a best-estimate search strategy, a strategy in which one chooses the node estimated to have the best feasible integer solution obtainable. In our branch-and-cut tree, we insert the GUB cuts up to five times the total number of constraints. Adding excessive number of cuts works to improve the global lower bound. In order to improve the global upper bound, we integrate the Relaxation Induced Neighborhoods (RINS) heuristic that works to find good feasible solutions early in the search process. This heuristic explores the neighborhood using information contained in the relaxation of the MIP model. Danna et al. (2005) show RINS to be efficient in finding good feasible solutions early in the process.

Given the complex structure of Models 2 and 3 due to criticality requirement, we use Tabu Search (TS) to solve the model and obtain good quality approximate solutions. Details of the algorithm and the parameter tuning are discussed in the next section.

3.2. Tabu Search and Parameter Settings

TS is a local-search improvement heuristic proven to be effective to solve many difficult combinatorial optimization problems (Hazir et al., 2008a). It has a penalty mechanism to avoid getting trapped at local optima by forbidding or penalizing moves that cause cycling among solution points previously visited. These forbidden moves are called "tabu". The short term memory keeps track of move attributes that have changed during the recent past and these attributes become tabu for a specific number of iterations. Under some conditions called *aspiration criterion*, tabu status of a move can be overridden. There are two commonly used strategies to obtain good solutions: diversification is used to direct the search into less visited regions of the search space, whereas intensification is used to fully explore a certain region.

Local search-based algorithms may not result in high quality solutions for the DTCTP-D, since it is not simple to identify a feasible solution in the neighborhood of the current solution; classical move operators do not guarantee feasibility. To overcome this shortcoming, we apply the features proposed by Kulturel-Konak et al. (2003), which are specially designed to solve constrained optimization problems. Their algorithm uses an adaptive penalty function which encourages the search to proceed through a portion of the infeasible region, namely the "*near feasibility threshold (NFT)*". The generated solutions are penalized according to their distances to the feasibility region. Their method requires few parameters and is shown to be insensitive to parameter changes.

In our TS algorithm, each solution is represented with a mode assignment vector. Infeasible mode assignments are allowed with some penalties. The algorithm starts the exploration in the infeasible region with the least cost solution, the solution in which the activities are assigned to the longest modes. By using a cost-based fitness function that is composed of total project cost and an adaptive penalty cost, it keeps searching for feasible and efficient directions until the stopping criterion is satisfied. In order to fully explore the neighborhood of the generated solutions, we examine the entire neighborhood with single mode decreasing and increasing moves.

To find the best parameters, some test problems of varying problem sizes are solved for a wide range of system parameters. In the test runs, we test a tabu list of size 5, 7, and 10. The best solutions are achieved with a tabu list of size 7. We define the aspiration criterion so that tabu status of a move can be overridden if it leads to a solution better than the incumbent solution. The stopping criterion is set to 10,000 iterations after observing that it is sufficient to obtain convergence for the test instances. In order to direct the search into less-visited regions of the search space and escape from local optima, we use a simple diversification strategy. If the incumbent solution is not updated for 1,000 iterations, the algorithm restarts with a randomly generated neighbor of the initial solution; the tabu list is initialized and the move values are recalculated according to the new solution.

4. Computational Experiments and Results

4.1. Experimentation

We use a subset of the random instances generated by Akkan et al. (2005) to test the proposed measures and algorithms. Their test bed defines the deterministic problem and mainly the network structure parameters, the number of modes per activity and the tightness of the deadline define an instance. However, in order to define the robust problem, two additional parameters, namely robustness level (Γ) and uncertainty factor (ψ), are required.

Mainly two parameters define the network structure: complexity index (CI) and the coefficient of network complexity (CNC). CI is a measure developed by Bein et al. (1986) to assess how far the given network is from being series–parallel. It is defined to be the minimum number of node reductions required to reduce a given two terminal directed acyclic graph into a single-arc graph, when used together with series and parallel reductions. Assessing the distance of a given network from being series–parallel is important for this study because DTCTP with series–parallel graphs could be solved quickly (Demeulemeester et al., 1996). The second complexity measure, CNC is developed by Pascoe (1966) and defined to be the ratio of the number of arcs to the number of nodes.

The number of modes per activity is randomly generated with discrete uniform distribution using interval U[2, 10]. To compute the deadline for each instance, first the minimum possible project duration, *Tmin* (length of the critical path with shortest modes), and the maximum possible project duration, *Tmax* (length of the critical path with longest modes), are calculated. Then, the deadline is set as follows:

$$\delta = Tmin + \theta\,(Tmax - Tmin) \quad \text{with } 0 \leq \theta \leq 1 \qquad (12)$$

Concave (ccv), convex (cvx), and neither concave nor convex functions (hyb) are used to generate the costs. The uncertainty factor represents the rate by which the variables d_{jm} are allowed to change around c_{jm}, i.e. $d_{jm} = \psi c_{jm}$, where ψ is uniformly distributed in [0.1, 1]. Table 2 summarizes the parameters of the test bed.

[Insert Table 2 here]

Three instances are solved for each parameter setting. All the algorithms are implemented in C programming language on a Sun UltraSPARC 12x400 MHz workstation with 3 GB RAM. Optimization software CPLEX 9.1 is used to solve the linear and integer programs.

4.2. Computational Results

Computational experiments demonstrate that the parameter Γ reflects the risk attitude of the decision maker and highly affects the schedule construction. As the decision makers become more risk-averse, larger Γ values should be employed in the model so that schedules with lower worst-case costs can be generated. However, these schedules usually perform worse when nominal costs are attained. In practice the expected activity cost is used as the nominal cost. Figure 2 sketches the impact of Γ on schedule generation and shows the multi-criteria behavior of the robust problem.

[Insert Figure 2 here]

Every point in Figure 2 represents the optimal solution of Model 1 for a specific project network with $n = 85$ under a given Γ. Note that the cost of each activity is represented with two parameters: the nominal cost and the worst-case cost. As it is illustrated in the figure, total nominal cost ($\sum_{j=1}^{n} \sum_{m \in M_j} c_{jm} x_{jm}$) and total worst-case cost ($\sum_{j=1}^{n} \sum_{m \in M_j} \overline{c_{jm}} x_{jm}$) of a schedule typically conflict, i.e., a schedule with a lower nominal cost yields higher cost at the worst-case. However, there may be exceptional instances such as the solution generated with $\Gamma = \lfloor 0.05n \rfloor$ in Figure 2, which is a dominated solution.

Besides the level of pessimism of the decision maker, we investigate the effects of various complexity parameters on the solution efficiency of Model 1 and summarize the results in Table 3. For each Γ setting, 36 different project instances are solved. The results of the exact procedure are presented under the column labeled with “Optimum”; the average number of linear and integer master problems solved and the average CPU time in seconds to solve the instances are reported under the columns “LP Iter.”, “IP Iter.”, “CPU(s)”, respectively.

[Insert Table 3 here]

The results of the approximate method are depicted under the column called "Truncated Solution", within which, the percentage of problem instances that the optimal solution has been found, the average percentage deviation from the optimal solution, the maximum percentage deviation from the optimal solution and the average CPU time reduction with truncation are reported under the columns "Ins Opt (%)", "Avg Dev (%) ", "Max Dev (%)", "Dec CPU (%)", respectively.

3 reveals that the network complexity, which is measured with CNC, and the pessimism level are effective on the computational effort given in CPU seconds. The instances with complex network structure and higher pessimism level require more computational effort. Among these influential factors, CNC has already been shown to be influential in solving the deterministic problems. When the CPU time for solving the deterministic problem and the robust problem are compared, we conclude that considerably higher computational effort is required when the notions of uncertainty and robustness are incorporated into the model. Finally, the truncation-based heuristic may be used as a solution alternative for large scale instances, as it is shown to be generating quick solutions that are very close to the optimal solution.

In the following section we introduce some robustness measures to assess the robustness of the generated schedules and compare the effectiveness of the proposed robust project scheduling models by using these measures.

4.3. Comparison of the Proposed Models

In this section, we first give a brief review of the metrics to evaluate schedule robustness. Afterwards, we perform computational experiments and evaluate the generated schedules by using several robustness measures.

4.3.1. Robustness Measures

Existing robust scheduling studies generally address machine environments and often follow scenario-based approaches, where scenarios for job attributes are required to be defined. They basically employ two types of robustness measures: direct measures, which are derived from realized performances, and heuristic approaches, which utilize simple surrogate measures. Computational burden of optimizing direct measures is generally higher when compared to surrogate measures. We refer the readers to Sabuncuoğlu and Gören (2005), for a detailed examination of the measures. Since achieving project completion time and project cost targets are crucial for project managers, we evaluate the robustness of project schedules both in terms of cost and time. The evaluation is based on the following measures in this paper.

I. Cost-Based Measures

a) Expected Realized Cost

The cost of performing each activity is represented with two parameters: the nominal cost and the worst-case cost. We assume that the nominal cost of a mode is equal to the expectation over all the scenarios corresponding to possible alternatives in practice. Therefore, for a given schedule the summation of the nominal costs over all activities defines the expected realized cost of the project schedule. The schedule which has the minimal expected realized cost is chosen as the most robust schedule.

b) Worst-Case Cost

The upper bounds of the cost intervals define the worst-case costs, the maximal cost among all possible scenarios. Hence the summation of the upper bounds of the cost intervals over all activities characterizes the worst-case realized cost of a schedule. The schedule that has the smallest worst-case cost (among all schedules) is chosen. This is a risk-averse approach and corresponds to the minimax objective in decision analysis.

c) Cost of the Reference Scenario

This measure concentrates on a specific scenario in which critical activities are realized at the worst-case costs with the remaining activities being realized at the nominal costs. The schedule that has the smallest cost with respect to this scenario is selected.

II. Time-Based Measures

a) Average Total Slack

Average of the slacks, or equivalently the sum of the slacks over all the activities could be used to assess schedule robustness. The schedules with larger average total slacks are preferred in terms of robustness.

b) Percentage of Potentially Critical Activities

Delays in critical activities may result in delays in the project completion time. Therefore, schedules with a smaller number of critical activities are preferred in terms of robustness. We employ the ratio of the number of potentially critical activities to the total number of activities in the project as a robustness measure.

c) Project Buffer Size

Buffers are protection mechanisms against uncertainty in the duration of activities. Project buffers are inserted to the end of projects to avoid possible delays in project completion. We use the ratio of project buffer size to project deadline as a robustness measure. The larger the project buffer, the more protected the project against disruptions.

For other measures and testing methodology of the measures, we refer the readers to Hazir et al. (2008b).

4.3.2. Computational Analysis

Computational experiments are carried out to evaluate the performance of the models under different problem settings and the models are compared using the above-mentioned robustness measures. For comparison purposes we set pessimism level to $\Gamma = \lfloor 0.25n \rfloor$. In the computational study, for each setting 3 different instances are solved, hence each model is tested with 36 problems. Table 4 compares the introduced models with 6 robustness measures. While comparing, Model 1 is taken as the reference model and percent differences between robustness measure of the reference model and the criticality-based models are reported. For each robustness measure, the average percentage deviation from the reference model and the percentage of problem instances that the model dominates the reference model are reported in the rows % Dev, % Dom, respectively. We also report the paired t-test confidence intervals (CI) for percent differences between robustness measures of model 1 and criticality-based models. 95 % confidence level is used in these intervals. Furthermore, statistical significance of the differences is reported. The last row expresses whether minimization or maximization of the measure is preferred in terms of robustness.

[Insert Table 4 here]

When worst-case robustness measure is considered, Model 1 dominates the criticality-based models. This finding is consistent with the argument that Model 1 is over-pessimistic. Model 1 is better in most of the problem instances when the expected realized cost of the models is compared. However, in this case differences among model performances are small. When time based measures are considered, Model 1 is dominated by Models 2 and 3. Model 2 minimizes the number of critical activities and it is more protected against disruptions since it has larger project buffers. It could also be observed that the performance of Model 3 lies between the performances of the Model 1 and 2 for majority of the robustness measures.

The parameter Γ reflects the risk attitude of the decision makers. As the decision makers become more risk-averse, larger Γ values should be used so that schedules with lower worst-case costs could be generated. Note that Model 2 is not so sensitive to the changes in parameter Γ when compared to other models. This is basically due to the criticality requirement. The second parameter that affects model characteristics is the slack/duration threshold (SDT). Lower SDT results in different reactions in criticality-based models. Since the risk premiums are incurred for only critical activities in Model 2, as the threshold decreases, the number of critical activities is reduced so that total risk premium decreases and majority of the activities take their nominal costs. However, in Model 3, a reduction in the threshold might increase the total risk premiums, as risks of non-critical activities are also incorporated.

Besides the model characteristics, we investigate the solution efficiency and report the results in Table 5. This table illustrates the average CPU time to solve the problem instances for each model. The problems are classified according to the coefficient of network complexity (CNC), which is the ratio of the number of arcs to the number of nodes. Every iteration of the criticality-based models requires higher computational efforts due to model complexity and we run TS sufficiently long (10,000 iterations) to increase the quality of the best solution.

[Insert Table 5 here]

5. Conclusions

In this paper, we have proposed three models to formulate the robust DTCTP. These models assume interval uncertainty for activity costs and generate schedules that are protected against variability in costs. In order to solve the models, we have developed both exact and approximate algorithms. The first model is solved exactly by using Benders Decomposition; the other two criticality-based models are rather complex and solved approximately by a tabu search algorithm. The main advantage of Model 1 is that it could be solved exactly. However, the limitation is that the activities with the same cost intervals are assumed to be equally uncertain and all activities are likely to have cost values at the upper bounds.

To evaluate the performance of the algorithms under various problem settings, we have conducted computational experiments. We have assessed the robustness of the schedules generated by the algorithms by using several cost and time-based robustness measures. When the worst-case cost of the generated schedules is considered, Model 1 is found to be advantageous; yet it is usually dominated by the criticality-based models regarding time-based robustness measures. Furthermore, Model 1 is found to be more pessimistic when compared to its alternatives.

The models developed in this manuscript address the requirement to generate robust project schedules that are less sensitive to uncertainty. In addition, they provide decision support to managers in project planning under uncertainty. As a future extension of this research, robust optimization models could be formulated for Multi-Mode Resource Constrained Project Scheduling Problem (MRCPSP), which allows the use of both renewable and nonrenewable resources. For this new generalized problem setting, in addition to the uncertainty costs, the uncertainty in activity durations or in resource requirements or in resource availabilities should also be addressed.

References

1. A. Atamturk, M. W. P. Savelsbergh, "Integer-Programming Software Systems," Annals of Operations Research, 140, pp. 67-124, 2005.

2. C. Akkan, A. Drexl, and A. Kimms, "Network decomposition-based benchmark results for the discrete time–cost trade-off problem," European Journal of Operational Research, 165, pp. 339-358, 2005.

3. W. W. Bein, J. Kamburowski, and M. F. M. Stallmann, "Optimal reduction of two-terminal directed acyclic graphs," SIAM Journal on Computing, 21, pp. 1112-1129, 1992.

4. J. F. Benders, "Partitioning procedures for solving mixed variables programming problems," Numerische Mathematic, 4, pp. 238-252, 1962.

5. A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski, "Adjusting robust solutions of uncertain linear programs," Mathematical Programming, 99, pp. 351–376, 2004.

6. D. Bertsimas, M. Sim, “Robust Discrete Optimization and Network Flows,” Mathematical Programming, 98, pp. 49-71, 2003.

7. I. Cohen, B. Golany, A. Shtub, “The Stochastic Time/Cost Trade-off Problem: A Robust Optimization Approach,” Networks, 49, pp. 175-188, 2007.

8. E. Danna, E. Rothberg, C. Le Pape, “Exploring Relaxation Induced Neighborhoods to Improve MIP Solutions,” Mathematical Programming Series. A 102: pp. 71–90, 2005.

9. P. De, E.J. Dunne, J.B. Ghosh, C.E. Wells, “The discrete time/cost trade-off problem revisited,” European Journal of Operational Research, 81, pp. 225-238, 1995.

10. P. De, E.J. Dunne, J.B. Ghosh, C.E. Wells, “Complexity of the discrete time/cost trade-off problem for project networks,” Operations Research, 45, pp. 302-306, 1997.

11. E. Demeulemeester, W. Herroelen, S.E. Elmaghraby, “Optimal procedures for the discrete time/cost trade-off problem in project networks,” European Journal of Operational Research, 88, pp. 50–68, 1996.

12. E. Demeulemeester et al., “New computational results for the discrete time/cost trade-off problem in project networks,” Journal of the Operational Research Society, 49, pp. 1153-1163, 1998.

13. Ö. Hazır, Y. Günalay and E. Erel, “Customer Order Scheduling Problem: A Comparative Metaheuristics Study,” International Journal of Advanced Manufacturing Technology, 37, pp. 589–598, 2008a.

14. Ö. Hazır, M. Haouari, and E. Erel, “Robustness Measures and a Scheduling Algorithm for the Discrete Time/Cost Trade-off Problem,” Working Paper, Faculty of Business Administration, Bilkent University, Turkey, 2008b.

15. W. Herroelen, R. Leus, “On the merits and pitfalls of critical chain scheduling,” Journal of Operations Management, 19, pp. 559-577, 2001.

16. W. Herroelen, R. Leus, “Project scheduling under uncertainty – survey and research potentials,” European Journal of Operational Research, 165, pp. 289–306, 2005.

17. S. Kulturel-Konak and et al., “Exploiting Tabu Search Memory in Constrained Problems,” INFORMS Journal on Computing, 16(3), pp. 241–254, 2004.

18. J. Linderoth, M.W.P. Savelsbergh, “A Computational Study of Search Strategies for Mixed Integer Programming,” INFORMS Journal on Computing, 11, pp. 173–187, 1999.

19. D. McDaniel and M. Devine, "A Modified Benders’ Partitioning Algorithm for Mixed Integer Programming," Management Science, 24, pp. 312-379, 1977.

20. T. L. Pascoe, “Allocation of resources–CPM,” Revue Française de Recherche Opérationelle, 38, pp. 31–38, 1966.

21. I. Sabuncuoğlu, S. Gören, 2005. A review of reactive scheduling research: proactive scheduling and new robustness and stability measures. Technical Report, IE/OR 2005-02, Department of Industrial Engineering, Bilkent University, Ankara.

22. M. Skutella, "Approximation Algorithms for the Discrete Time–Cost Trade-off Problem," Mathematics of Operations Research, 23, pp. 195-203, 1998.

23. M. Vanhoucke, D. Debels, "The Discrete time/cost trade-off problem under various assumptions exact and heuristic procedures," Journal of Scheduling (in press), 2008.

24. V. Valls and et al. "Project Sheduling with Stochastic Activity Interruptions. In J. Werglarz, ed., Project Scheduling: Recent Models, Algorithms and Applications. Boston: Kluwer Academic Publishers, pp. 333-353, 1998.

25. D. S. Yamashita, V.A. Armentano and M. Laguna, "Robust Optimization Models for Project Scheduling with Resource Availability Cost," Journal of Scheduling (in press), 2008.

26. J. Y. Yen, "Finding the K Shortest Loopless Paths in a Network," Management Science, 17, pp. 712–716, 1971.

Appendix A:

Proof of Proposition 1:

Given a schedule $x \in X^D$, define the risk premiums considered in Model 1 and 2 as:

$$g_1(x) = \underset{u \in B^n}{\text{Max}} \left\{ \sum_{j=1}^{n} \sum_{m \in M_j} d_{jm} x_{jm} u_j \; : \; \sum_{j=1}^{n} u_j \leq \Gamma \right\}, \tag{5}$$

$$g_2(x) = \underset{u \in B^n}{\text{Max}} \left\{ \sum_{j=1}^{n} \sum_{m \in M_j} d_{jm} x_{jm} u_j \; : \; \sum_{j \in CR} u_j \leq \Gamma \right\},$$

where $CR = \{j: TS_j/p_j \leq \xi\,,\ \ j = 1,\ldots,n\}$, (6)

The feasibility sets are: $U_1 = \left\{ u \in B^n : \sum_{j=1}^{n} u_j \leq \Gamma \right\}$, and $U_2 = \left\{ u \in B^n : \sum_{j \in CR} u_j \leq \Gamma \right\}$.

From its definition of CR is a subset of $\{1,\ldots,n\}$; hence for any $u \in U_2$, then $u \in U_1$ as well, i.e. $U_2 \subseteq U_1$. Besides, both of the models defined in (5) and (6) have the same objective function, thus Model 1 is a relaxation of Model 2. For any IP, a relaxation constitutes an upper bound for a maximization problem, hence

$$g_2(x) \leq g_1(x) \qquad \forall x \in X^D \tag{7}$$

Q.E.D.

Appendix B:

Proof of Corollary 1:

Let z_1 and z_2 represent optimal objective value of Model 1 and Model 2 respectively. Then

$$z_1 = \text{Min}\ \{h_1(x) : x \in X^D\} \tag{8}$$

$$z_2 = \text{Min}\ \{h_2(x) : x \in X^D\} \tag{9}$$

where $h_i(x) = \sum_{j=1}^{n} \sum_{m \in M_j} c_{jm}\ x_{jm} + g_i(x)$, $x \in X^D$, $i = 1,2$.

Let x^* is the vector that minimizes $h_1(x)$, i.e., $z_1 = h_1(x^*)$. Then by definition of $h_1(x)$,

$$z_1 = h_1(x^*) = \sum_{j=1}^{n} \sum_{m \in M_j} c_{jm}\, x^*_{jm} + g_1(x^*) \tag{10}$$

Using Proposition 1 we get

$$z_1 \geq \sum_{j=1}^{n} \sum_{m \in M_j} c_{jm}\, x^*_{jm} + g_2(x^*) \tag{11}$$

and right hand side of the relationship (11) is the evaluation of $h_2(x)$ at a feasible solution x^*. Thus $z_1 \geq h_2(x^*)$, and z_2 is the minimum value of $h_2(x)$. So

$z_1 \geq z_2$

Q.E.D.

Appendix C:

Proof of Proposition 2:

We could reformulate (1) as:

$$f(\Gamma) = \text{Min}\left[\sum_{j=1}^{n} \sum_{m \in M_j} c_{jm}\, x_{jm} + g(x) : x \in X^D\right], \text{ where} \tag{12}$$

$$g(x) = \text{Max}\left\{\sum_{j=1}^{n} \sum_{m \in M_j} d_{jm} x_{jm} u_j : \sum_{j=1}^{n} u_j \leq \Gamma, u \in B^n\right\} \tag{13}$$

Given a solution vector, $x \in X^D$, $g(x)$ is a knapsack problem of which LP relaxation has binary optimal solutions (see Theorem 1 of Bertsimas and Sim 2003), hence $g(x)$ could be rewritten as:

$$g(x) = \text{Max}\left\{\sum_{j=1}^{n} \sum_{m \in M_j} d_{jm} x_{jm} u_j : \sum_{j=1}^{n} u_j \leq \Gamma,\ 0 \leq u_j \leq 1,\ j = 1, \ldots, n\right\} \tag{14}$$

This knapsack problem has a non-empty feasible set, for any given Γ. Let U be the polytope that defines the feasible set, then U can be defined as a convex combination of K vertex,

$u^k = (u_1^{\,k},...,u_n^{\,k}),$ for $k = 1,...,K$ and one of them is the optimum solution of $g(x)$. Thus we have,

$$g(x) = \underset{1\le k\le K}{\text{Max}} \left\{ \sum_{j=1}^{n} \sum_{m\in M_j} d_{jm} x_{jm} u_j^{\,k} \right\} \tag{15}$$

Note that the polytope does not depend on the solution vector, x. Therefore the same set of extreme points could be used in solution of $g(x)$. If we set $z = g(x)$ and define a constraint for each extreme point of U, (12) could be reformulated as:

$$\underset{x\in X^D, z}{\text{Min}} \left\{ \sum_{j=1}^{n} \sum_{m\in M_j} c_{jm} x_{jm} + z : z \ge \sum_{j=1}^{n} \sum_{m\in M_j} u_j^{\,k} d_{jm} x_{jm} \,,\ k = 1,..,K \right\} \tag{16}$$

Similarly, the deadline constraint could be defined as the longest path in graph G should not be more than the time limit, δ. Combining the path constraints to satisfy deadline feasibility and (16), Model 1 given in equation (2) is reformulated as equation set (4). Note that we add the non-negativity constraint on z for the sake of algorithmic convenience; otherwise z could be unrestricted in sign.

Q.E.D.

Figure 1
Click here to download Figure: Figure 1.doc

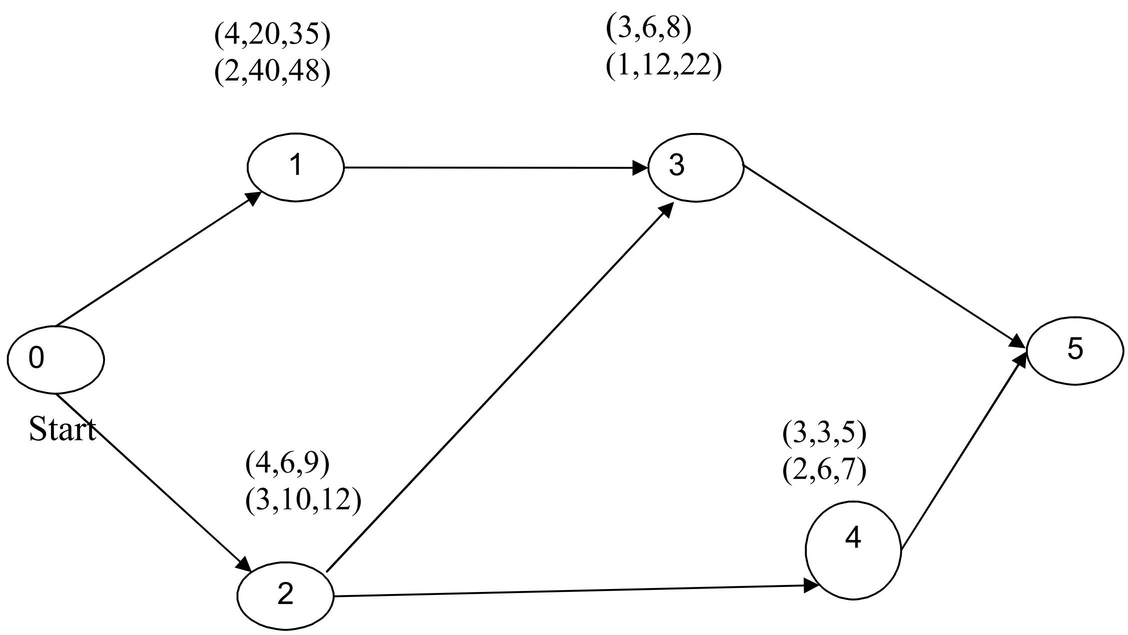

Figure 1. The Example Network (Robust Problem)

Figure 2
Click here to download Figure: Figure 2.doc

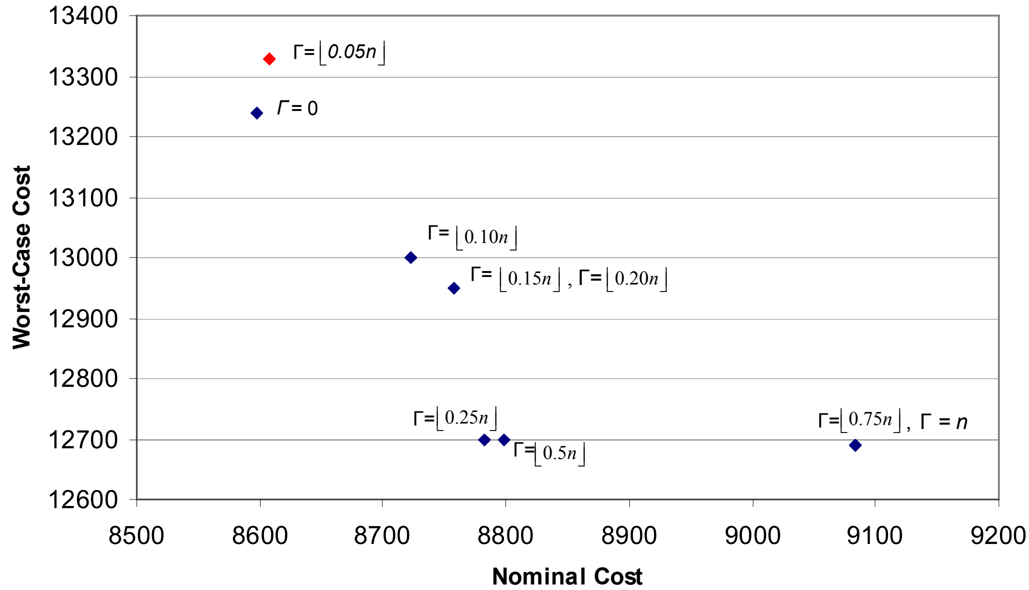

Figure 2. The Impact of Pessimism Level on Schedule Generation

Table 1
Click here to download Table: Table 1.doc

Table 1. Comparison of Robust Models

	Activity(j)				C_5	$f(\Gamma = 0)$	$d_{jm} = \overline{c_{jm}} - c_{jm}$				Robust Objective: $f(\Gamma)$			
	1	2	3	4			$j = 1$	$j = 2$	$j = 3$	$j = 4$	$\Gamma = 1$	$\Gamma_{CR} = 1$	$\Gamma = 2$	$\Gamma_{CR} = 2$
Feasible Mode Combinations	2	2	2	2	5	68	8	2	10	1	78	68	86	68
	2	2	2	1	6	65	8	2	10	2	75	67	83	69
	2	2	1	2	6	62	8	2	2	1	70	64	72	66
	2	2	1	1	6	59	8	2	2	2	67	61	69*	63
	1	2	2	2	5	48	15	2	10	1	63	63	73	63
	1	2	2	1	6	45	15	2	10	2	60	60	70	62*
	1	1	2	2	6	44*	15	3	10	1	59*	59 *	69*	62 *
	2	1	2	2	6	64	8	3	10	1	74	67	82	68

Table 2
Click here to download Table: Table 2.doc

Table 2. Experimental Setting for Solving the Robust Problem

Parameter	Level(s)
CI	13
CNC	5, 6, 7, 8
Number of modes	U[2,10]
Deadline parameter (θ)	0.15
Nominal cost function	ccv, cvx, hyb
Uncertainty factor (ψ)	U[0.1, 1.0]
Pessimism Level (Γ)	0, $\lfloor 0.25n \rfloor$, $\lfloor 0.5n \rfloor$, $\lfloor 0.75n \rfloor$

Table 3
Click here to download Table: Table 3.doc

Table 3. Summary of Computational Results of Model 1

			Optimum			Truncated Solution			
			LP Iter.	IP Iter.	CPU(s)	Ins Opt (%)	Avg Dev (%)	Max Dev (%)	Dec CPU (%)
Robust	CNC	5	16.32	19.23	1400.96	81.82	0.03	0.42	25.03
		6	22.42	27.33	10427.91	100	0.00	0.00	15.04
		7	23.00	31.14	18044.09	90.91	0.01	0.22	16.38
		8	20.28	31.88	19139.61	84.00	0.03	0.37	16.20
	Γ	$\lfloor 0.25n \rfloor$	19.42	26.84	8772.72	87.10	0.02	0.31	21.64
		$\lfloor 0.5n \rfloor$	20.40	27.73	10789.31	87.10	0.02	0.37	18.07
		$\lfloor 0.75n \rfloor$	21.77	27.58	11690.94	93.55	0.01	0.42	14.81
Deterministic	CNC	5	15.00	12.00	410.87	44.44	0.21	0.82	27.77
		6	22.50	15.25	1935.93	100	0.00	0.00	21.42
		7	22.00	17.17	7277.80	71.00	0.02	0.12	28.81
		8	21.14	31.29	10974.72	75.00	0.01	0.04	30.44

Table 4
Click here to download Table: Table 4.doc

Table 4. Model Comparison with Robustness Measures

		Cost Based Measures			Time Based Measures		
		Expected Cost	**Worst Case Cost**	**Reference Scenario**	**Average Slack**	**% Critical**	**Project Buffer**
Model 2	**% Dev**	3.60*	14.56*	-17.33*	10.37*	-72.58*	3.17*
	CI	(2.51, 4.69)	(13.29, 15.84)	(-18.86, -15.81)	(8.45, 12.28)	(-77.98, -67.18)	(2.82, 3.52)
	% Dom	13.89	0	100	100	100	100
Model 3	**% Dev**	0.67	9.21*	-9.24*	2.97*	-17.67*	0.57*
	CI	(-0.27, 1.61)	(7.96, 10.43)	(-11.41, -7.07)	(1.46, 4.48)	(-22.22, -13.12)	(0.39, 0.76)
	% Dom	38.89	2.78	94.44	80.56	88.89	91.67
Optimization Criterion		Min	Min	Min	Max	Min	Max

* indicates that % Dev is significant at 5% level

Table 5

Click here to download Table: Table 5.doc

Table 5. Models and Computational Requirements

		CPU (sec)		
		Model 1	**Model 2**	**Model 3**
CNC	**5**	1400.96	3567.47	3309.628
	6	10427.91	6311.57	6094.823
	7	18044.09	17332.73	16075.57
	8	19139.61	38854.95	37804.99