Robust Scheduling at a Large IT Services Delivery Center

Elvin Çoban  
Özyeğin University  
Istanbul, Turkey  
Email: elvin.coban@ozyegin.edu.tr

Aliza Heching  
IBM Thomas J. Watson Research Center  
Yorktown Heights, USA  
Email: aheching@gmail.com

J. N. Hooker  
and Alan Scheller-Wolf  
Carnegie Mellon University  
Pittsburgh, USA  
Email: jh38,awolf@andrew.cmu.edu

Abstract—We study a project scheduling problem at a large IT services delivery center in which there are unpredictable delays. We apply robust optimization to minimize tardiness while informing the customer of a reasonable worst-case completion time. Due to the impracticality of quantifying joint probability distributions for delay times, we follow the recent practice of using empirically determined uncertainty sets, which to our knowledge have not been applied to service scheduling. To solve instances of a realistic size, we introduce a new solution method based on logic-based Benders decomposition. We show that when the uncertainty set is polyhedral, convexity properties of the problem allow us to simplify the decomposition substantially, leading to a model of tractable size.

I. INTRODUCTION

We analyze a project scheduling problem at a large IT services delivery center in which there are uncertain processing times and unpredictable delays in start times. This study is motivated by a real problem at a global IT services delivery organization. To design a schedule that is not unduly disrupted by contingencies, we formulate a robust optimization problem. Our main objective is to minimize tardiness while informing the customer of a reasonable worst-case completion time in a complex scheduling environment.

Due to the impracticality of quantifying joint probability distributions for delay times, we apply robust optimization with uncertainty sets rather than probabilistic information. An uncertainty set is an empirically determined space of possible outcomes for which one should realistically plan, without encompassing theoretically worst-case scenarios. To our knowledge, uncertainty sets have not previously been applied to service scheduling.

Optimization over uncertainty sets is a challenging two-stage optimization problem that can become intractable in large-scale instances using conventional techniques. We therefore propose a new solution method based on logic-based Benders decomposition. We show that when the uncertainty set is polyhedral, the problem has convexity properties that result in a simplified decomposition that can be much easier to solve. Polyhedral uncertainty sets provide a great deal of flexibility for practical application. The size of the simplified model is proportional to the number of extreme points of the polyhedron. Thus if the polyhedron is a simplex, a natural option for the problem considered here, the model grows only linearly with the number of tasks to be scheduled.

The remainder of the paper is organized as follows. After a review of previous work, we state the deterministic and then the robust scheduling problem. We describe logic-based Benders decomposition and show how it can be applied to robust scheduling. Due to the size of the problem, we apply a second level of Benders decomposition to the Benders master problem. This allows the subproblem to decouple and results in a three-stage model. We then focus on simplicial uncertainty sets, which are those described by a linear inequality and nonnegativity constraints. This allows us to formulate a much simpler two-stage model, based on convexity properties that are proved via a dynamic programming recursion.

II. PREVIOUS WORK

Robust optimization attempts to ensure that an optimal solution remains feasible and near optimal when uncertain parameters change somewhat [1]. Because the joint probability distribution of the parameters is often unavailable in practice, we focus on distribution-free robust optimization in which uncertain parameters belong to an uncertainty set [2].

This approach was first proposed in [3] for robust linear programming in which the columns of the coefficient matrix are uncertain and belong to a convex uncertainty set. Originally, worst-case scenarios were included in the uncertainty set, but this is typically too conservative for practical implementations. To overcome this overconservatism, one can use ellipsoidal uncertainty sets that exclude unlikely outcomes of uncertain parameters, as in [4]–[8]. It can be shown that the robust counterpart of some important generic convex optimization problems under ellipsoidal uncertainty sets are exactly or approximately tractable problems that can be solved by interior point methods. However, the robust counterpart of a linear programming problem becomes a second-order conic problem. Thus, the complexity of the problem increases with the ellipsoidal uncertainty set.

To avoid this increasing complexity, a polyhedral uncertainty set is proposed in [2], [9]. The robust counterpart of a linear programming problem remains a linear programming problem, and the method is more tractable, especially in large-scale settings. The level of conservatism is controlled via restricting the number of parameters that take their worst-case values [9]–[14]. Detailed reviews of robust optimization with uncertainty sets can be found in [1], [15], [16].
### III. THE PROBLEM

The service delivery center employs several hundred agents with different skill sets. The agents process thousands of incoming customer projects each year. A project may consist of several tasks that must be completed in a prescribed order. An agent must have at least the necessary required skills to complete each assigned task.

Based on the data sets we have analyzed, late deliveries result primarily from interruptions that require a task to be set aside temporarily. For example, the agent may require more information from the customer before proceeding. The resulting delay may be several days, during which the agent can work on other tasks, but only if the agent has been assigned other tasks. The schedule must therefore anticipate this possibility when assigning tasks.

Late deliveries on projects may also be caused by variable processing times, which our data indicate can differ by a factor of two or more for tasks of a given type. Although processing times tend to be measured in hours rather than days, long processing times can upset the schedule if several occur in a row.

Interruptions usually occur as processing gets underway, and the agent finds that key information is missing. This, combined with the fact that processing times are comparatively short, suggest that there is no need to adopt a pre-emptive scheduling model. We therefore treat the interrupted task as having a delayed start time. Under this modeling approach, the primary uncertainty is in the release time. We re-compute the schedule on a rolling basis as projects arrive.

### IV. MODELING THE PROBLEM

#### A. Deterministic Model

We first formulate the deterministic version of the problem using notation summarized in Table I. The tardiness cost \( c_j \) may reflect priorities (e.g., important clients). If only the completion time of a project matters, the cost is assigned to the final task of the project, and the other tasks have zero cost. The problem assumes that a precedence graph \((J,E)\) has been specified. This is a directed graph in which \((j,j') \in E\) when task \( j \) must finish before task \( j' \) starts.

Let \( \alpha^+ = \max\{0, \alpha\} \). Assuming no preemption, the problem is

\[
\min \sum_j c_j(s_j + p_j - d_j)^+= \quad (a) \\
S_j^j \subseteq S_j^i, \text{ all } j \quad (b) \\
\text{noOverlap}(s_j, p_j, d_j), \text{ all } j \quad (c) \\
s_j + p_j \leq s_j', \text{ all } (j,j') \in E \quad (d)
\]

The objective function minimizes weighted tardiness of tasks. Constraints (a) ensure that agents have the skills necessary for the tasks they are assigned. Constraints (b) state that tasks are processed after their release times. The noOverlap constraints in (c) have the form noOverlap\((s, p)\), which requires that tasks \(1, \ldots, n\) with processing times \( p = (p_1, \ldots, p_n) \) be given start times \( s = (s_1, \ldots, s_n) \) so that the tasks do not overlap. The constraint therefore requires that the tasks assigned to each agent \( i \) do not overlap. An agent who will be unavailable for a certain period \([t, t']\) should be pre-assigned a dummy task with a fixed start time \( t \) and duration \( t' - t \). Constraints (d) enforce the precedence graph.

We will use the following as a running example. There are 3 tasks and 2 agents, and task 1 must precede task 2. Let \( p = (2, 3, 4), r = (0, 3, 2), \) and \( d = (4, 4, 4) \), with time measured in hours. Both agents have the necessary skills for all 3 tasks, and each \( c_j = 1 \). Then one optimal solution of (1) assigns tasks 1 and 3 to agent 1, and task 2 to agent, so that \( y_1 = (1, 2, 1) \). Each task starts at its release time in this solution, so that \( s = (0, 3, 2) \). Tasks 2 and 3 are 2 hours late, resulting in total tardiness 4.

#### B. Robust Model

The simplest sort of robust scheduling plans for the worst case. Suppose \( \theta = (\theta_1, \ldots, \theta_m) \) is a tuple of uncertain parameters, such as processing times or delays. If we let \( f(x, \theta) \) be the cost of schedule \( x \) for a given \( \theta \), then the worst-case robust scheduling problem is

\[
\min_x \left\{ \max_{\theta \in \Theta} \{f(x, \theta)\} \right\} \quad (2)
\]

Worst-case scheduling is often too conservative, as there is a vanishingly small probability that every parameter will take its worst-case value. A probabilistic approach avoids this difficulty but may be impractical. It is typically impossible to characterize the joint distribution of thousands of parameters.

An uncertainty set \( \Theta \) contains parameter values \( \theta \) that are judged, without the use of probabilities, to be realistic possibilities [9]–[14]. The robust optimization problem becomes

\[
\min_{x \in X} \left\{ \max_{\theta \in \Theta} \{f(x, \theta)\} \right\} \quad (2)
\]
A simple way to specify $\Theta$ is to use $r$-restricted robust scheduling, which supposes that at most $r$ parameters take their worst-case value [14]. However, we consider polyhedral and, in particular, simplicial uncertainty sets, which allow more flexibility in characterizing uncertainty. This complicates solution of the problem, but we show in the following sections that a decomposition approach can yield a practical solution.

For the service center scheduling problem, the uncertain parameters $\theta_i$ in (2) are release time delays and processing times. The release time delays for the $n$ tasks are $\Delta r = (\Delta r_1, \ldots, \Delta r_n)$, and we require that $\Delta r$ belong to uncertainty set $R$. The uncertain processing times are $p + \Delta p$ where $\Delta p = (\Delta p_1, \ldots, \Delta p_n)$ and $\Delta p$ belongs to uncertainty set $P$.

The decision variables $x_i$ are nominally the task assignments $y_j$ and start times $s_j$. However, for a given set of start times $s$, unexpectedly long processing times in the subproblem may cause tasks to overlap. We therefore replace the start times $s_j$ with sequence variables $\sigma_j$ that indicate the position of job $j$ in the sequence. Start times can then be adjusted in the subproblem to avoid overlap. The general robust problem (2) becomes the following

$$\min_{y, \sigma} \left\{ \max_{P \in \mathcal{P}} \{ f(\sigma, y, r + \Delta r, p + \Delta p) \} \mid S_j' \subset S_j \right\}$$  \hspace{1cm} (3)

We calculate $f(\sigma, y, r + \Delta r, p + \Delta p)$ by first constructing a greedy schedule based on the given values of $\sigma, y, r + \Delta r,$ and $p + \Delta p$ and then observing the weighted tardiness of this schedule. This greedy schedule has each agent perform assigned tasks in the order given by $\sigma$. Let $pr(j, \sigma, y)$ be the task that agent $y_j$ performs immediately before task $j$ in sequence $\sigma$. If we assume that $\sigma$ observes the precedence relations in the graph $(J, E)$, we can write

$$f(\sigma, y, r + \Delta r, p + \Delta p) = \sum_j c_j(s_j + p_j + \Delta p_j - d_j)^+$$

where $s_j$ is recursively defined for all $j = 1, \ldots, n$ by

$$s_j = \max \left\{ r_j + \Delta r_j, s_{pr(j, \sigma, y)} + p_{pr(j, \sigma, y)} + \Delta p_{pr(j, \sigma, y)} \right\} \max_{(j', j) \in E} \left\{ s_{j'} + p_{j'} + \Delta p_{j'} \right\}$$  \hspace{1cm} (4)

Each task $j$ starts at the earliest possible time that is defined by the maximum of the three events formulated in (4): (i) the delayed release time of task $j$, (ii) the finish time of agent $y_j$’s previous task, and (iii) the latest finish time of all predecessor tasks. We let $s_{pr(j, \sigma, y)} = -\infty$ if task $j$ is the first task assigned to some agent.

In the example given earlier, suppose $y = (1, 2, 1)$ and $\sigma = (1, 2, 3)$ as in the deterministic solution. Then $(s_1, s_2, s_3)$ in (4) are given by

$$s_1 = \max \{ \Delta r_1, -\infty, -\infty \}$$
$$s_2 = \max \{ 3 + \Delta r_2, s_1 + 2, -\infty \}$$
$$s_3 = \max \{ 2 + \Delta r_3, -\infty, s_1 + 2 \}$$  \hspace{1cm} (5)

If we suppose the uncertainty set $R$ is the simplex described by $|\Delta r| \geq 0 \{ |\Delta r_1 + \Delta r_2 + \Delta r_3| \leq 3 \},$ one of the worst-case outcomes for this solution is $\Delta r = (3, 0, 0)$, resulting in worst-case tardiness 10. We will see that the solution of the robust problem (3) is different, namely $y = (1, 1, 2)$ and $\sigma = (1, 2, 3)$, for which the worst-case tardiness is only 7. A deterministic solution therefore need not be a robust solution.

V. LOGIC-BASED BENDERS DECOMPOSITION

Logic-based Benders decomposition (LBBD) is a generalization of Benders decomposition in which the subproblem can in principle be any combinatorial problem, not necessarily a linear or nonlinear programming problem [17]–[19]. This flexibility has led to the application of LBBD to planning and scheduling problems that naturally decompose into an assignment and a scheduling portion. This approach can reduce solution times by several orders of magnitude relative to conventional methods [20]–[24]. We apply LBBD to the general robust problem (2) as follows.

We first write (2) as

$$\min v \quad v = \max_{\theta \in \Theta} \{ f(x, \theta) \} \quad x \in X$$  \hspace{1cm} (6)

The master problem for (6) is

$$\min z \quad z \geq f(x, \theta^*)$$  \hspace{1cm} (7)

The Benders cuts put lower bounds on $z$ as a function of $x$. Given an optimal solution $\bar{x}$ of (7), the subproblem is

$$\min v \quad v = \max_{\theta \in \Theta} \{ f(\bar{x}, \theta) \}$$  \hspace{1cm} (8)

which is equivalent to $\max_{\theta \in \Theta} \{ f(\bar{x}, \theta) \}$.

In general, Benders cuts are obtained by solving an inference of the subproblem [19]. A very simple cut can be obtained for this model as follows. If $(\theta^*, v^*)$ solves the subproblem, then for any $x$, the restricted worst-case cost is at least $f(x, \theta^*)$. So we have a Benders cut for (7)

$$z \geq f(x, \theta^*)$$  \hspace{1cm} (9)

This is essentially the “regret cut” of [25]. Stronger cuts can be obtained by analyzing the proof of optimality in the subproblem. For example, it may be determined that only certain variable settings $\bar{x}_j$ in the solution of the master play a role in the optimality proof. A cut can be written on this basis, as discussed in Section VII-C.

VI. DECOMPOSITION OF THE SCHEDULING PROBLEM

A. Initial Benders Formulation

We now provide a Benders formulation for the robust scheduling problem (3). The master problem determines agent assignments $y$ and the task sequence $\sigma$:

$$\min z \quad z \geq f(x, \theta^*)$$  \hspace{1cm} (a)
$$\sigma_j < \sigma_{j'}, \quad \text{all } (j, j') \in E \quad (b)$$

Benders cuts

$$y_j \in \{1, \ldots, m\}, \quad \text{all } j \quad (d)$$
The subproblem is formulated as follows:

\[
\max_{s, \Delta r, \Delta p} \sum_j (s_j + p_j + \Delta p_j - d_j)^+
\]

\[
s_j = \max \{ r_j + \Delta r_j, \quad s_{pr(j,\bar{\sigma},\bar{y})} + p_{pr(j,\bar{\sigma},\bar{y})} + \Delta p_{pr(j,\bar{\sigma},\bar{y})} \}, \quad \text{all } j
\]

\[
\Delta p \in P, \quad \Delta r \in R
\]

where \((\bar{\sigma}, \bar{y})\) is the solution of the master problem. To simplify notation, we assume that tardiness costs \(c_j = 1\). The constraints define the starting time of task \(j\) as the maximum of the delayed release time of task \(j\) and the finish time of the task performed by agent \(y_j\) immediately before \(j\). It is no longer necessary for the maximum to take into account the latest finish time of task \(j\)'s predecessors, because the precedence relations in \((J, E)\) are enforced by the master problem.

If \((s^k, \Delta x^k, \Delta p^k)\) solves the subproblem in iteration \(k\) of the Benders algorithm, the simplest Benders cut is a strengthened nogood cut as described above. One can also use the regret cut (9), which is now

\[
z \geq f(\sigma, y, r + \Delta x^k, p + \Delta p^k)
\]

where \(f\) represents total tardiness when \(\sigma, y, \Delta x^k, \text{ and } \Delta p^k\) are fixed. To model this cut as an MILP in the master problem, we introduce a tuple \(s^k\) of start time variables for each Benders cut \(k\). The master problem becomes

\[
\begin{align*}
\min & \quad z \\
\text{s.t.} & \quad S^k_j \subseteq S_{y_j}, \quad \text{all } j \\
& \quad \sigma_j < \sigma_{j'}, \quad \text{all } (j, j') \in E \\
& \quad z \geq \sum_j s^k_j \\
& \quad s^k_j \geq r_j + \Delta x^k_j - d_j, \quad \text{all } j, k \\
& \quad s^k_j \geq r_j + \Delta p^k_j, \quad \text{all } j, k \\
& \quad s^k_j \geq s_{pr(j,\sigma,y)} + p_{pr(j,\sigma,y)} + \Delta p_{pr(j,\sigma,y)}, \quad \text{all } j, k \\
& \quad s^k_j + p_j + \Delta p^k_j \leq s^k_{j'}, \quad \text{all } (j, j') \in E, \quad \text{all } k \\
& \quad y_j \in \{1, \ldots, m\}
\end{align*}
\]

where the decision variables are \(z, s^k, y, \sigma, \text{ and } s^k\). It is straightforward to formulate this as an MILP, but the problem becomes very large as Benders cuts accumulate due to the addition of new variables \(s^k\) with each cut.

\section{B. Three-Stage Decomposition}

To overcome the problem of increasing problem size, one can solve (13) by a second Benders decomposition scheme in which the subproblem decouples by agent classes. We will refer to this as the inner Benders problem. The master problem in the inner scheme contains 0–1 assignment variables \(x_{ij}\), where \(x_{ij} = 1\) when \(y_j = i\). The subproblem contains sequencing variables \(\sigma_j\). This results in the three-stage decomposition of Figure 1. The task assignment and sequence obtained from the inner Benders algorithm is passed to the uncertainty subproblem, which computes the corresponding worst case solution, using MILP if the uncertainty sets are polyhedral. Benders cuts from the uncertainty subproblem are inserted into the master problem, and the process repeats until the optimal value of (13) converges to the best value of the uncertainty subproblem found so far.

The sequencing subproblem sequences tasks to minimize tardiness, subject to agent assignments from the master problem. A project’s tasks are often carried out by a single agent who has all the necessary skills. When this is the case, we can decouple the sequencing subproblem into a separate sequencing problem for each agent, because precedence relations concern only tasks that are part of the same project.

Even if tasks in a project are assigned to two more agents, we can decouple the problem with respect to agent classes. These are the smallest sets of agents whose assigned tasks are not coupled by precedence relations with tasks assigned to other sets of agents. To form agent classes for a given assignment \(\bar{x}\), we construct a graph \(G\) in which vertices correspond to agents. Two vertices \(i, i'\) are connected by an arc when there is a precedence constraint between some job assigned to \(i\) and some job assigned to \(i'\). Then the agent classes correspond to the vertex sets of connected components of \(G\).

When the inner Benders algorithm converges to an optimal solution \((\bar{x}, \bar{\sigma})\), we solve the uncertainty subproblem (11) with \(\bar{x}\) replacing \(\bar{y}\).

\section{VII. Polyhedral Uncertainty Sets}

When the uncertainty sets \(P, R\) are polyhedral, we can simplify the three-stage decomposition of Figure 1 to a more tractable two-stage decomposition. We accomplish this first by showing that the uncertainty subproblem has an extreme point solution, which in turn allows us to combine the sequencing and uncertainty subproblems into a single MILP problem that minimizes tardiness. Because regret cuts are suitable only for an uncertainty subproblem, we will use strengthened nogood cuts as Benders cuts.

\subsection{A. Extreme Point Solution}

To simplify exposition, we suppose from here out that only the start time delays are uncertain, and the processing times are known. The analysis is readily extended to accommodate uncertain processing times. We therefore suppose that the
uncertainty set $R$ is polyhedral. The basic theorem is as follows.

**Theorem 1.** If the uncertainty set $R$ is a bounded polyhedron, then at least one of its extreme points is an optimal solution of the uncertainty subproblem (11).

We omit this and subsequent proofs due to space limitations. The proof of Theorem 1 is based on the fact that (11) can be viewed as maximizing a convex function over the polyhedron $R$. In the example, the extreme points of $R$ are $\Delta r = (0,0,0), (3,0,0), (0,3,0)$, and $(0,0,3)$.

### B. Two-Stage Decomposition

Theorem 1 allows us to simplify the three-stage decomposition of Figure 1 to the two-stage decomposition of Figure 2. We know that tardiness for a given assignment and task sequence is maximized at some extreme point of $R$. We can therefore minimize worst-case tardiness for a given assignment by finding a sequence that minimizes the maximum tardiness over all extreme points. The two subproblems of Figure 1 can therefore be combined into a single subproblem that minimizes tardiness subject to a lower bound on tardiness for each extreme point.

![Figure 2: Two-stage decomposition.](image)

The master problem computes an optimal assignment of tasks to agents:

$$
\min z \\
S_{ij} \subset S_i, \text{ all } i, j \text{ with } x_{ij} = 1 \\
\sum_i x_{ij} = 1, \text{ all } j \\
\text{relaxation and Benders cuts} \\
x_{ij} \in \{0,1\}, \text{ all } i, j
$$

(14)

The relaxation and Benders cuts are described below.

The subproblem minimizes worst-case tardiness $T$. Variable $s_{ij}^\ell$ is the start time of task $j$ when its delay $\Delta r_{ij}^\ell$ is defined by the extreme point $\ell$ of $R$.

$$
\min T \\
T \geq \sum_j (s_{ij}^\ell + p_j - d_j)^+, \text{ all } \ell \tag{a} \\
\text{noOverlap} \left( \sigma(\bar{x}, i), s^\ell(\bar{x}, i), p(\bar{x}, i) \right), \text{ all } i, \ell \tag{b} \\
s_{ij}^\ell \geq r_j + \Delta r_{ij}^\ell, \text{ all } j, \ell \tag{c} \\
s_{ij}^\ell \geq s_{ij'}^\ell + p_{j'}, \text{ all } (j', j) \in E, \text{ all } \ell \tag{d}
$$

The minimum is taken over variables $s_{ij}^\ell$ and $T$. The sequencing variables $\sigma_j$ appear as arguments of the noOverlap constraint to ensure that, for each $i$, the same sequence is used for all extreme points $\ell$.

**Theorem 2.** If the uncertainty set $R$ is a bounded polyhedron, the three-stage and two-stage decompositions are equivalent optimization problems.

The number of constraints in the subproblem (15) grows linearly with the number of extreme points, and therefore only linearly with the number of tasks when the polyhedron is a simplex. In addition, (15) decouples by agent class $I_\alpha$. For each class $I_\alpha$, we have the problem

$$
\min T_\alpha \\
T_\alpha \geq \sum_j (s_{ij}^\ell + p_j - d_j)^+, \text{ all } \ell \\
\text{noOverlap} \left( \sigma(\bar{x}, i), s^\ell(\bar{x}, i), p(\bar{x}, i) \right), \text{ all } i \in I_\alpha, \text{ all } \ell \\
s_{ij}^\ell \geq r_j + \Delta r_{ij}^\ell, \text{ all } j \in I_\alpha, \text{ all } \ell \\
s_{ij}^\ell \geq s_{ij'}^\ell + p_{j'}, \text{ all } (j', j) \in E, \text{ all } \ell
$$

If $T_\alpha^*$ is the minimum tardiness for agent class $\alpha$, the overall minimum tardiness is $\sum_\alpha T_\alpha^*$.

In the example, the initial master problem (14) contains only the assignment constraints. One solution of (14) is the assignment $y = (1,2,1)$, for which $x_{11} = x_{22} = x_{31} = 1$. This implies only one agent class. Because $R$ has 4 extreme points, the subproblem (15) is

$$
\min T \\
T \geq \left( s_1^\ell - 2 \right)^+ + (s_2^1 - 1)^+ + s_3^\ell, \text{ } \ell = 1,2,3,4 \\
\text{noOverlap} \left( \sigma_1(\bar{x}, 3), (s_1^\ell, s_2^\ell), (2,4) \right), \text{ } \ell = 1,2,3,4 \\
s_1^\ell \geq r_3 + 3, s_2^1 \geq r_2 + 3, s_3^3 \geq r_3 + 3; s_1^\ell \geq r_3, \text{ all } j, \ell \\
s_2^1 \geq s_1^\ell + 2, \text{ } \ell = 1,2,3,4
$$

An optimal solution is $\sigma = (1,2,3)$ with $s_1^1 = (3,5,5), s_2^2 = (0,6,2), s_3^3 = (0,3,5)$, with tardiness $T = 10$ as noted earlier.

We can strengthen the master problem (14) with relaxations of the subproblem, which allow the master problem to select more reasonable assignments before many Benders cuts have been accumulated. We use the two relaxations described in [22].

### C. Benders Cuts

As Benders cuts, we use strengthened nogood cuts in the master problem (14). If $T^*$ is the optimal value of the
subproblem when $x = \bar{x}$, the simplest nogood cut is
\[
z \geq \begin{cases} 
T^* & \text{if } x = \bar{x} \\
-\infty & \text{otherwise}
\end{cases} 
\tag{16}
\]

The cut can be strengthened by heuristically removing task assignments and re-solving the subproblem until the minimum tardiness is less than $z^*$.

We use the following heuristic. Let $T^*_\alpha$ be the minimum tardiness in the solution of the subproblem corresponding to agent class $I_\alpha$. Remove a task $j$ such that: (a) task $j$ is completed before its deadline $d_j$ in the subproblem solution; (b) task $j$’s completion time is strictly earlier than the start time of the next task performed by the same agent, or task $j$ is the last task of its project performed by the same agent; (c) task $j$ is performed by an agent who performs no late tasks; and (d) no attempt has already been made to remove task $j$. Re-solve the subproblem (VII-B) for the agent class $I_\alpha$, for which $j \in I_\alpha$. If the minimum value is $T^*_\alpha$, repeat. If the minimum value is less than $T^*_\alpha$, replace task $j$ and repeat. Continue as desired. If $J$ is the set of tasks removed, impose the strengthened nogood cut
\[
z \geq \begin{cases} 
z^* & \text{if } x_j = \bar{x}_j \text{ for all } j \not\in J \\
-\infty & \text{otherwise}
\end{cases} 
\tag{17}
\]

In the example, the nogood cut (16) can be written
\[
T \geq 10(x_{11} + x_{22} + x_{13} - 2)
\]
The heuristic does not remove any of the tasks. However, since there is only one agent class, we can observe that the tardiness remains at least 10 as long as $x_{11} = x_{13} = 1$. The cut can therefore be strengthened to $T \geq 10(x_{11} + x_{13} - 1)$. Symmetry allows one to write the analogous cut $T \geq 10(x_{21} + x_{23} - 1)$. The Benders algorithm converges to the robust solution mentioned earlier.

VIII. Conclusion

We introduced a novel robust scheduling method to solve a project scheduling problem in a large IT service delivery organization. We applied robust optimization with uncertainty sets to model unpredictable delays and provide the customer with a realistic worst-case completion date. We showed how to reduce the problem to tractable size by assuming a simplicial uncertainty set and applying logic-based Benders decomposition. Possible future research includes development of additional Benders cuts and generalization of the convexity results.

REFERENCES