Decomposition Algorithm for Optimizing Multi-server Appointment Scheduling with Chance Constraints

Yan Deng* Siqian Shen†

October 1, 2014

Abstract

We schedule appointments with random service durations on multiple servers with operating time limits. We minimize the costs of operating servers and serving appointments, subject to a joint chance constraint limiting the risk of server overtime. Using finite samples of the uncertainty, we formulate the problem as a mixed-integer linear program, and propose a two-stage programming framework where we keep cross-server decisions and constraints at the first stage, so that the remaining problem features a server-wise decomposable structure. The first-stage problem is a variant of the chance-constrained binary packing problem discussed in Song et al. (2014), in which appointments are packed to servers with a probabilistic overtime-free guarantee. The second-stage problem seeks feasible appointment schedules given appointment-to-server assignments. We solve the problems at both stages by methods of coefficient strengthening, bounding, and branch-and-cut with diverse forms of cutting planes. We also investigate variants that consider bounded appointment waiting time, expected overtime/waiting penalty, and use individual chance constraints to limit server overtime. We test instances with diverse problem and sample sizes to demonstrate the computational efficacy of different approaches.

Key words: stochastic service operations; chance constraints; mixed-integer linear programming; decomposition; cutting-plane algorithms

1 Introduction

Problems of scheduling appointments on multiple servers arise in many service applications. They involve complex optimization problems with discrete decision variables and large-scale uncertain data (e.g., random service durations, resource availability, and operational cost). Due to the problem complexity, the existing literature often separately optimizes the problems of allocating appointments to servers and scheduling the appointments on individual

*Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor; email: yandeng@umich.edu

†Corresponding author; Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor; email: siqian@umich.edu

servers. However, according to the results of related studies in the context of operating room allocation and surgery scheduling (see, e.g., Deng et al. 2014), better performance can be achieved by integrating the two decision processes, which helps avoiding myopic solutions and results in economies of scale. Meanwhile, risk measures that ensure low chance of the occurrence of undesirable outcomes, have been less used in the studies of stochastic service operations, as compared to minimizing the expected penalty of incurring undesirable outcomes. In particular, one can employ a risk measure to formulate a *chance-constrained program* (CCP) for directly monitoring the Quality of Service (QoS) in related systems.

In this paper, we investigate a generic modeling framework and solution algorithms for problems of chance-constrained multi-server appointment scheduling (CC-MAS). We optimize the use of multiple servers with operational time limits to serve appointments with random service durations. We formulate a chance-constrained program that minimizes the total costs of operating servers and serving appointments, while bounding the risk of server overtime. Each appointment is appointed a start time that needs to be in a requested visiting time window, and customers for an appointment will arrive punctually at the appointed start time. We also consider several CC-MAS variants, which respectively (i) employ individual chance constraints to bound the overtime risk, (ii) minimize the expected penalty in the length of overtime, and (iii) limit the waiting time of each appointment given the tardiness of previous appointments. We develop various approaches based on coefficient strengthening, bounding, and cutting-plane generation for solving the generic CC-MAS model and also generalize them to each of the variants. We test randomly generated instances and compare the computational efficacy of each approach.

1.1 Literature Review of Allocation and Scheduling

The CC-MAS problem in this paper is closely related to the well-studied *parallel machine scheduling* problems in the literature, although the latter often consider separate problems of server allocation and job scheduling. In the allocation literature, Blake and Donald (2002), Ozkarahan (2000), Jebali et al. (2006) assume deterministic service time which may not reflect the true nature of the related operations. Denton et al. (2010) propose a two-stage stochastic mixed-integer linear programming model for allocating surgeries to operating rooms (ORs) and minimizing the cost of operating ORs under uncertain surgery durations. Shylo et al. (2012) extend this work and introduce a chance-constrained model to bound the risk of running ORs overtime. They solve the model as a deterministic convex program based on an assumption that surgery durations follow independent Normal distributions.

On the other hand, the scheduling literature mainly focuses on planning start time of appointments on a single stochastic server given a fixed sequence of appointment arrivals. Typical methodologies include simulation, queueing theories (e.g., Brahimi and Worthington 1991, Hassin and Mendel 2008) and stochastic optimization (e.g., Denton and Gupta 2003, Erdogan and Denton 2013). A common goal is to balance conflicting metrics such as appointment waiting time, server idle time and overtime. Different from the traditional objective of minimizing the expectation of those outcomes, Sarin et al. (2014) use the conditional-value-at-risk (CVaR) measure to seek risk-averse solutions for single/parallel server scheduling. As compared to the value-at-risk (VaR) measure used in this paper, the CVaR model provides a good approximation of the corresponding VaR model and a feasible solution that can be

obtained by solving a linear program. In all of these studies, service durations are assumed to follow some known distribution, such as Uniform, Gamma, Normal or Log-normal due to a better fit to the data or to the context of the scheduling system considered, or Exponential distribution due to the Markovian assumption in queuing models. In this paper, we also assume full distributional information for the uncertain service durations of a given set of appointments.

1.2 Review of Relevant Methodologies

CCPs are used for guaranteeing the QoS and bounding the risk of having undesirable outcomes within small controllable values. They are generally intractable because the feasible regions are nonconvex and computing probabilities requires a multi-dimensional integral. Some existing studies solve CCPs using convex approximations (e.g., Nemirovski and Shapiro 2007) and scenario approximations (e.g., Nemirovski and Shapiro 2006). In this paper, we generate finite samples of the random service durations and derive a sampling-based formulation to approximate the true CCP. Theoretical results of applying the Sample Average Approximation (SAA) method to evaluate solution bounds and feasibility for general CCPs have been investigated in the literature (e.g., Pagnoncelli et al. 2009, Luedtke and Ahmed 2008). It is worth mentioning that the SAA approach neither requires specific distribution forms nor independence of service durations of different appointments.

The main challenge of this paper arises in solving the CC-MAS variant with a joint chance constraint enforcing overtime-free with a high probability, which we replace using binary logic variables and linear constraints involving undesirable big-M coefficients. Luedtke et al. (2010), Küçükyavuz (2012) propose strong valid inequalities for solving such mixed-integer linear programming (MILP) reformulations of CCPs with only the right-hand side vector being random. Alternatively, Sen (1992), Dentcheva et al. (2000), Ruszczyński (2002) reformulate CCPs as integer programs using the concept of efficient points of the random parameter. The generic CC-MAS model in this paper, however, involves a joint chance constraint with recourse variables, random technology matrix, and discrete variables of server operations. Luedtke (2013) proposes pioneer work on solving two-stage CCPs with recourse decisions, and develops valid inequalities through lifting and uses them in a branch-and-cut framework. Other investigations of theories and computations of joint CCPs include Qiu et al. (2014), Song et al. (2014), Tanner and Ntaimo (2010), which we will review in details and generalize for solving the CC-MAS variants in later sections. Moreover, one of our approaches (i.e., the dual decomposition method in Section 3.3.3) is related to the approaches in Watson et al. (2010), Ahmed et al. (2014), Ahmed (2013) proposed for solving various stochastic/chance-constrained programs with recourse.

The main contribution of this paper is the deployment of multiple decomposition and cutting-plane methods for solving variants of the CC-MAS problem. The decomposition procedures generally create separable subproblems, each of which is associated with a server or a realized scenario, and contains disjoint sets of decisions and constraints. Constraints that need to jointly hold across servers or scenarios are not formulated explicitly, but enforced iteratively through cutting planes. The work can be generalized a broader class of problems with decomposable structures, e.g., network problems with multiple subgraphs and correlated network-flow decisions. Also, the decomposition framework are not restricted to problems

with joint chance constraints, and can be extended to solve general stochastic programs with individual chance constraints and with recourse penalty taken into account in the objective.

The rest of the paper is organized as follows. We provide a generic modeling framework in Section 2 and classify two sets of linear constraints with or without the uncertain parameters and recourse variables. Section 3 first proposes a two-stage programming framework where we keep cross-server decisions and constraints at the first stage, so that the remaining problem features a server-wise decomposable structure. We then discuss various methods of coefficient strengthening, cutting-plane generation, and bounding to solve the problems at both stages. Section 4 investigates three CC-MAS variants, and generalizes the previous decomposition approaches for each variant. Section 5 computes instances with diverse problem and sample sizes to demonstrate the efficacy of each proposed approach. We conclude the paper and state future research directions in Section 6.

2 Problem Statement and Formulation

Consider scheduling a set I of appointments with random durations $\xi_i,\ \forall i \in I$, to a set J of servers over a finite time horizon H. We assume a finite support for ξ as $\{\xi^w\}_{w\in\Omega}$ and without loss of generality each realization $\xi^w = [\xi_i^w,\ i \in I]^\mathsf{T}$ is realized with equal probability $1/|\Omega|$. Each server j operates from time 0 to time T_j. We assign each appointment $i \in I$ a server and a start time within a requested time window $[\underline{a}_i, \overline{a}_i]$. It is worth noting that this time-window constraint differentiates our model from a stochastic bin packing problem where appointments can be packed "seamlessly". We assume that customers for an appointment always arrive at the appointed start time and are served according to a first-in-first-out (FIFO) rule. Let $c_j^1 > 0$ be the cost of operating server j, and $c_{ij}^2 \geq 0$ be the cost of serving appointment i on server j, $\forall i \in I,\ j \in J$. We aim to minimize the cost of operating servers and serving all the appointments, subject to low risk of having servers running overtime.

We define variables $x_j \in \{0,1\}$ for all $j \in J$ such that $x_j = 1$ if we operate server j, and $x_j = 0$ otherwise. Define variables $y_{ij} \in \{0,1\}$ for all $i \in I$ and $j \in J$ such that $y_{ij} = 1$ if appointment i is assigned to server j, and $y_{ij} = 0$ otherwise. Let a continuous variable $s_i \geq 0$ represent the appointed start time of every appointment i. Define binary variables $z_{i'i}$ for each pair of appointments $(i',\ i)$ such that $z_{i'i} = 1$ if appointment i' is ahead of i, and $z_{i'i} = 0$ otherwise. We formulate the CC-MAS problem as:

$$\texttt{CC-MAS}: \quad \min \ \sum\nolimits_{j\in J} c_j^1 x_j + \sum\nolimits_{i\in I}\sum\nolimits_{j\in J} c_{ij}^2 y_{ij} \ ^{[1]} \tag{1}$$

$$\text{s.t. } (x,y,z,s) \in Q$$

$$\mathbb{P}\Big\{(x,y,z,s) \in \mathcal{Q}(\xi)\Big\} \geq 1-\epsilon. \tag{2}$$

where Q is a fixed region and $\mathcal{Q}(\xi)$ is a region parameterized by the random vector ξ. (2) is a joint chance constraint that limits the risk of $(x,y,z,s) \notin \mathcal{Q}(\xi)$ by $\epsilon \in (0,1)$. We detail the two sets of constraints as follows.

$$Q = \Big\{(x,y,z,s) \in \{0,1\}^{|J|} \times \{0,1\}^{|I|\times|J|} \times \{0,1\}^{|I|\times(|I|-1)} \times \mathbb{R}_+^{|I|} :$$

[1] In the rest of this paper, we present the objective function in a vector form as $c^1x + c^2y$.

$$\sum\nolimits_{j\in J} y_{ij} = 1,\ y_{ij} \le x_j \quad \forall i \in I,\ j \in J \tag{3a}$$

$$y_{ij} + y_{i'j} - 1 \le z_{ii'} + z_{i'i} \le 1,$$

$$1 - z_{ii'} \ge y_{ij} - y_{i'j},\ 1 - z_{ii'} \ge y_{i'j} - y_{ij}, \quad \forall i, i' \in I,\ i \ne i',\ j \in J \tag{3b}$$

$$\underline{a}_i \le s_i \le \overline{a}_i \quad \forall i \in I \tag{3c}$$

$$s_i \ge -\mathcal{M}^1_{i'i}(1 - z_{i'i}) + s_{i'} \quad \forall i, i' \in I,\ i \ne i' \Big\}. \tag{3d}$$

(3a) ensure every appointment being assigned to an operating server. For any two appointments i and i', (3b) ensure that i is appointed to start either before ($z_{ii'} = 1$) or after i' ($z_{ii'} = 0$) if they are allocated to the same server, and enforce $z_{ii'} = 0$ otherwise. (3c) ensure that the appointed start time of each appointment i is within the requested time window. (3d) link the appointment start time to their relative orders. Here $\mathcal{M}^1_{i'i}$ is a large constant, such that constraints (3d) are relaxed when $z_{i'i} = 0$. We set $\mathcal{M}^1_{i'i} = \overline{a}_{i'} - \underline{a}_i$.

In each scenario $w \in \Omega$, the realization $\mathcal{Q}(\xi^w)$ of the random region $\mathcal{Q}(\xi)$ is:

$$\mathcal{Q}(\xi^w) = \Big\{(x, y, z, s) : \exists t^w \in \mathbb{R}_+^{|I|} \text{ such that}$$

$$t^w_i \ge s_i, \quad \forall i \in I. \tag{4a}$$

$$t^w_i \ge -\mathcal{M}^2_{i'iw}(1 - z_{i'i}) + t^w_{i'} + \xi^w_{i'} \quad \forall i, i' \in I,\ i \ne i'. \tag{4b}$$

$$t^w_i + \xi^w_i \le \mathrm{T}_j + \mathcal{M}^3_{ijw}(1 - y_{ij}) \quad \forall i \in I,\ j \in J \Big\}, \tag{4c}$$

which involves recourse variable t^w_i as the actual start time of every appointment i. Here $\mathcal{M}^2_{i'iw}$ and $\mathcal{M}^3_{ijw}$ are large coefficients that will relax the corresponding constraints when $z_{i'i} = 0$ and $y_{ij} = 0$, respectively. We set $\mathcal{M}^2_{i'iw} = H - \underline{a}_i$ and $\mathcal{M}^3_{ijw} = H - \mathrm{T}_j$. First, (4a) ensure that the actual start time of each appointment is no earlier than the appointed start time. They also indicate that a server will become idle if it finishes an appointment before the appointed start time of the next one. (4b) enforce the FIFO rule, i.e., appointment i can start only if all the appointments that arrive earlier have been served. Finally, we enforce no overtime in all servers by (4c): The left-hand side represents the actual finishing time of appointment i in scenario w, which is required to be no later than T_j if appointment i is allocated to server j.

In the rest of the paper, we use the joint chance constraint (2) and the following equivalent reformulation interchangeably:

$$\sum\nolimits_{w\in\Omega} \mathbb{I}\left\{(x, y, z, s) \in \mathcal{Q}(\xi^w)\right\} \ge |\Omega| - \theta$$

where $\mathbb{I}\{\cdot\}$ is an indicator function that returns 1 if $\cdot$ is true and 0 otherwise, and $\theta = \lfloor \epsilon|\Omega| \rfloor$.

3 Decomposition and Coefficient Strengthening

3.1 A Two-stage Programming Framework

We propose a two-stage programming framework which is different from a traditional decomposition method of separating decisions that are made before realizing the uncertainty

and those after. Instead, we keep cross-server decision variables and constraints in a relaxed master problem at the first stage, so that the remaining problem can be decomposed by server.

Consider the master problem formulated in the space of (x, y):

$$\texttt{MP}: \quad \min\Big\{c^1x + c^2y : (3a),\ (x,y) \in \mathcal{A} \cap \{0,1\}^{|J|} \times \{0,1\}^{|I|\times|J|}\Big\}$$

where $\mathcal{A} = \big\{(x,y) : \exists s, z \text{ satisfy } (2),\ (3b)-(3d)\big\}$. We enforce $(x,y) \in \mathcal{A}$ by cutting planes. Given a solution $(\hat{x}, \hat{y})$, we explicitly know the set $J(\hat{x}) = \{j \in J : \hat{x}_j = 1\}$ of operating servers, and sets $I_j(\hat{y}) = \{i \in I : \hat{y}_{ij} = 1\}$ of appointments allocated on each server $j \in J(\hat{x})$. We reformulate some of the remaining constraints in CC-MAS as follows.

Consider any server $j \in J(\hat{x})$. There are $|I_j(\hat{y})|$ appointments. We define binary variables $u_{ik},\ \forall i \in I_j(\hat{y})$ and $k = 1, \ldots, |I_j(\hat{y})|$, such that $u_{ik} = 1$ if appointment i is scheduled as the k^{th} one, and $u_{ik} = 0$ otherwise. We replace (3b) with

$$\sum\nolimits_{k=1}^{|I_j(\hat{y})|} u_{ik} = 1 \quad \forall i \in I_j(\hat{y}) \tag{5a}$$

for ensuring a valid order of appointments. For $k = 1, \ldots, |I_j(\hat{y})|$, define continuous variables r_k and γ_k representing the appointed start time and the actual start time of the k^{th} appointment, respectively. We replace (3c) and (3d) respectively with

$$\sum\nolimits_{i\in I_j(\hat{y})} \underline{a}_i u_{ik} \le r_k \le \sum\nolimits_{i\in I_j(\hat{y})} \overline{a}_i u_{ik} \quad \forall k = 1, \ldots, |I_j(\hat{y})| \tag{5b}$$

$$\text{and} \qquad r_k - r_{k-1} \ge 0 \quad \forall k = 2, \ldots, |I_j(\hat{y})|. \tag{5c}$$

Also, we replace constraints (4a)–(4c) with

$$\gamma^w_k \ge r_k \quad \forall k = 1, \ldots, |I_j(\hat{y})| \tag{6a}$$

$$\gamma^w_k \ge \gamma^w_{k-1} + \sum\nolimits_{i\in I_j(\hat{y})} \xi^w_i u_{ik-1} \quad \forall k = 2, \ldots, |I_j(\hat{y})| \tag{6b}$$

$$\gamma^w_{|I_j(\hat{y})|} + \sum\nolimits_{i\in I_j(\hat{y})} \xi^w_i u_{i|I_j(\hat{y})|} \le \mathrm{T}_j \tag{6c}$$

for all $w \in \Omega$. We then present the following separation problem to check whether solution $(\hat{x}, \hat{y})$ satisfies all the constraints in CC-MAS that are relaxed in MP.

$$\begin{aligned} \texttt{SP}(\hat{x},\hat{y}):\ \max \quad & 0 \\ \text{s.t.} \quad & \sum\nolimits_{w\in\Omega} \mathbb{I}\big\{(6a)-(6c),\ j \in J\big\} \ge \lceil (1-\epsilon)|\Omega| \rceil \\ & (5a)-(5c) \quad \forall j \in J(\hat{x}) \\ & u_{ik} \in \{0,1\},\ \forall i = I_j(\hat{y}),\ k = 1, \ldots, |I_j(\hat{y})|,\ j \in J(\hat{x}) \\ & r_k,\ \gamma^w_k \ge 0,\ \forall w \in \Omega,\ k = 1, \ldots, |I_j(\hat{y})|,\ j \in J(\hat{x}) \end{aligned} \tag{7}$$

The above reformulation does not contain the big-$\mathcal{M}^1$, -$\mathcal{M}^2$ and -$\mathcal{M}^3$ coefficients. If $\texttt{SP}(\hat{x},\hat{y})$ is infeasible, we cut off solution $(\hat{x}, \hat{y})$ by generating a cut of the form:

$$\sum\nolimits_{j\in J(\hat{x})} \sum\nolimits_{i\in I_j(\hat{y})} y_{ij} \le |I| - 1 \tag{8}$$

which is essentially a strengthened no-good cut used in literature to cut off an undesirable binary solution. (Appendix A presents its derivation.) Intuitively, the inequality requires a future appointment-to-server assignment decision to be different from the current one.

Moreover, the current form of MP does not take into account any scheduling machinery and overtime risk. Thus it is likely to create solutions that can trivially make the separation problem infeasible. In this paper, we consider a strengthened MP by adding a proxy of the joint chance constraint:

$$\sum\nolimits_{w\in\Omega} \mathbb{I}\left\{\sum\nolimits_{i\in I} \xi_i^w y_{ij} \le \mathrm{T}_j x_j \ \forall j \in J\right\} \ge |\Omega| - \theta \tag{9}$$

which enforce the total duration of assigned appointments not to exceed T_j for every server j with the required overtime-free reliability. The finishing time of an operating server equals to the total service durations plus possible idle time between adjacent appointments. Therefore, (9) is a relaxation of (2) that is built with master-problem variables x and y. We show in later computation that this strengthening step reduces computational time significantly.

3.2 Extended Formulation and Strengthened Big-M Coefficients for the Strengthened MP

We reformulate constraint (9) as integer linear inequalities and present an extended formulation of the strengthened MP as

$$\begin{aligned}
\texttt{MP}^*: \quad & \min\ c^1 x + c^2 y \\
& \text{s.t.} \ \sum\nolimits_{i\in I} \xi_i^w y_{ij} \le \mathrm{T}_j x_j + \mathrm{M}_j^w \kappa_w \ \forall j \in J,\ w \in \Omega && \text{(10a)} \\
& \sum\nolimits_{w\in\Omega} \kappa_w \le \theta = \lfloor \epsilon |\Omega| \rfloor && \text{(10b)} \\
& \text{(3a)},\ (x,y) \in \mathcal{A},\ x \in \{0,1\}^{|J|},\ y \in \{0,1\}^{|I|\times|J|},\ \kappa \in \{0,1\}^{|\Omega|}.
\end{aligned}$$

For each scenario w, we define variable $\kappa_w \in \{0,1\}$ such that $\kappa_w = 0$ implies that the constraints $\sum_{i\in I} \xi_i^w y_{ij} \le \mathrm{T}_j x_j$ should be satisfied. Coefficients $\mathrm{M}_j^w,\ \forall j \in J$ are sufficiently large such that the corresponding constraints are relaxed when $\kappa_w = 1$. We set $\mathrm{M}_j^w = H - \mathrm{T}_j$ unless stated otherwise. This is referred to as *the default choice* later. The extended formulation with naively-chosen big-M coefficients tends to have a weak linear programming (LP) relaxation. We apply two methods in the literature for strengthening big-M coefficients.

Qiu et al. (2014) propose an iterative method that repeats plugging the latest-attained coefficients into an LP model to compute improved values. In each iteration n, we compute:

$$\begin{aligned}
\mathrm{M}_j^w(n) = \max\ & \sum\nolimits_{i\in I} \xi_i^w y_{ij} - \mathrm{T}_j x_j && \text{(11a)} \\
\text{s.t.}\ & \sum\nolimits_{i\in I} \xi_i^w y_{ij} \le \mathrm{T}_j x_j + \mathrm{M}_j^w (n-1) \kappa_w \ \forall j \in J,\ w \in \Omega && \text{(11b)} \\
& \text{(3a)},\ \text{(10b)},\ (x,y) \in \mathcal{A} \\
& x \in [0,1]^{|J|}, y \in [0,1]^{|I|\times|J|}, \kappa \in [0,1]^{|\Omega|}.
\end{aligned}$$

Initially, we set $M_j^w(0)$ as the default choice.

Song et al. (2014) consider another method that avoids accounting for constraints of multiple scenarios at a time. To strengthen M_j^w, we first compute

$$\mathrm{m}_j^w(w') = \max_{(x,y)\in\mathcal{D}_{w'}} \left\{ \sum\nolimits_{i\in I} \xi_i^w y_{ij} - \mathrm{T}_j x_j \right\} \quad \forall w' \in \Omega \tag{12}$$

where $\mathcal{D}_{w'} = \left\{ (x,y) : (3a), \sum_{i\in I} \xi_i^{w'} y_{ij'} \le \mathrm{T}_{j'} x_{j'}, \forall j' \in J,\ (x,y) \in \mathcal{A} \cap \{0,1\}^{|J|} \times \{0,1\}^{|I|\times|J|} \right\}$. Sort $\{\mathrm{m}_j^w(w')\}_{w'\in\Omega}$ in a nondecreasing order such that $\mathrm{m}_j^w(\sigma_1) \le \cdots \le \mathrm{m}_j^w(\sigma_{|\Omega|})$ with $\sigma_1, \ldots, \sigma_{|\Omega|}$ being a permutation of scenarios. Let $\mathcal{P}$ be the feasible region of MP* in the space of (x,y). It is a conjunction of at least $(|\Omega| - \theta)$ subregions among $\{\mathcal{P} \cap \mathcal{D}_w\}_{w\in\Omega}$. So any feasible solution must fall in the conjunction $\cup_{n=1}^{\theta+1} \mathcal{D}_{\sigma_n}$, which leads to:

$$\sum\nolimits_{i\in I} \xi_i^w y_{ij} - \mathrm{T}_j x_j \le \mathrm{m}_j^w(\sigma_{\theta+1}). \tag{13}$$

We thus set $\mathrm{M}_j^w = \mathrm{m}_j^w(\sigma_{\theta+1})$.

Remark 1. Alternatively, one can solve 0-1 single knapsack problems:

$$\overline{\mathrm{m}}_j^w(w') = \max_y \left\{ \sum\nolimits_{i\in I} \xi_i^w y_{ij} - \mathrm{T}_j : \sum\nolimits_{i\in I} \xi_i^{w'} y_{ij} \le \mathrm{T}_j,\ y_{ij} \in \{0,1\},\ \forall i \in I \right\} \quad \forall w' \in \Omega.$$

which are obtained from (12) by fixing $x_j = 1$ and relaxing $\sum_{i\in I} \xi_i^{w'} y_{ij'} \le \mathrm{T}_{j'} x_{j'},\ \forall j' \in J \setminus \{j\}$. Note that $(\overline{\mathrm{m}}_j^w(w'))^+$ [2]provides an upper bound for $\mathrm{m}_j^w(w')$. One can sort $\left\{(\overline{\mathrm{m}}_j^w(w'))^+\right\}_{w'\in\Omega}$ to have $(\overline{\mathrm{m}}_j^w(\sigma_1))^+ \le \cdots \le (\overline{\mathrm{m}}_j^w(\sigma_{|\Omega|}))^+$, and set M_j^w to $(\overline{\mathrm{m}}_j^w(\sigma_{\theta+1}))^+$.

3.3 Decomposition Methods for Solving MP*

We develop three methods for solving MP* as alternatives to solving the problem directly with strengthened big-M coefficients. We first describe a branch-and-cut algorithm using lifted valid inequalities (Section 3.3.1), then discuss two bounding methods: One develops bounds by optimizing over individual scenarios (Section 3.3.2), and the other employs a dual decomposition algorithm to develop bounds based on a Lagrangian relaxation (Section 3.3.3).

3.3.1 Branch-and-cut algorithm

Luedtke (2013) proposes a class of valid inequalities used in a branch-and-cut algorithm for solving two-stage CCPs. We apply the method and obtain the inequalities from lifting

$$\sum\nolimits_{i\in I} \xi_i^w y_{ij} - \mathrm{T}_j x_j \le \mathrm{m}_j^w(w) \quad \forall j \in J$$

which are valid inequalities for $\mathcal{P} \cap \mathcal{D}_w$ but not in general for the feasible region $\mathcal{P}$, to a form of (10a) with a strengthened coefficient for κ_w. Note that (13) holds for any feasible (x,y), we thus use a lifting coefficient of $\mathrm{m}_j^w(\sigma_{\theta+1}) - \mathrm{m}_j^w(w)$ and attain lifted valid inequalities as:

$$\sum\nolimits_{i\in I} \xi_i^w y_{ij} - \mathrm{T}_j x_j \le \mathrm{m}_j^w(w) + \big(\mathrm{m}_j^w(\sigma_{\theta+1}) - \mathrm{m}_j^w(w)\big)\kappa_w,\ \forall j \in J. \tag{14}$$

The branch-and-cut procedures focuses on solving MP* without constraints (10a). For any tentative solution $(\hat{x}, \hat{y}, \hat{\kappa})$, if $(\hat{x}, \hat{y})$ fails to satisfy the constraints of some scenario w with $\hat{\kappa}_w = 0$, a cut (14) is generated.

[2] $(\cdot)^+ = \max\{0, \cdot\}$.

3.3.2 Decomposition-based bounding

We consider scenario-based subproblems:

$$v(w, \mathcal{S}) = \min \left\{ c^1 x + c^2 y : (x, y) \in \mathcal{D}_w \setminus \mathcal{S} \right\} \quad \forall w \in \Omega \tag{15}$$

where $\mathcal{S}$ is a set of (x, y) vertices violating the joint chance constraint (9). In this algorithm, we use $\mathcal{D}_w \setminus \mathcal{S}$ to approximate $\mathcal{P} \cap \mathcal{D}_w$. Exclusion of vertices is enforced by appending no-good cuts (8).

Proposition 1. Let $\mathcal{P}^*$ be the complete set of optimal solutions of $\texttt{MP}^*$ in the space of (x, y). For any set $\mathcal{S} \subseteq \{0,1\}^{|J|} \times \{0,1\}^{|I|\times|J|}$, if $\mathcal{P}^* \nsubseteq \mathcal{S}$ then $v(\sigma_{\theta+1}, \mathcal{S})$ is a lower bound for the optimal objective value, where $\{\sigma_n\}_{n=1}^{|\Omega|}$ is a permutation of scenarios such that

$$v(\sigma_1, \mathcal{S}) \geq \cdots \geq v(\sigma_{|\Omega|}, \mathcal{S}).$$

Proof. Consider some $(x^*, y^*) \in \mathcal{P}^* \setminus \mathcal{S}$. The chance constraint (9) implies that there must exist some $n \in \{1, \ldots, \theta + 1\}$ such that $(x^*, y^*) \in \mathcal{D}_{\sigma_n}$. Therefore,

$$c^1 x^* + c^2 y^* \geq \min\{c^1 x + c^2 y : (x, y) \in \mathcal{D}_{\sigma_n} \setminus \mathcal{S}\} = v(\sigma_n, \mathcal{S}) \geq v(\sigma_{\theta+1}, \mathcal{S}).$$

This idea is similar to the previous scenario-based method for strengthening big-M coefficients by Song et al. (2014). □

The algorithm is an iterative process of improving the best-found lower bound $\underline{B}$ and upper bound $\overline{B}$. In each iteration, for a fixed set $\mathcal{S}$, we compute $v(w, \mathcal{S})$, $\forall w \in \Omega$, and then update $\underline{B}$ with the $(\theta+1)^{\text{th}}$ largest value, i.e., $\underline{B} = v(\sigma_{\theta+1}, \mathcal{S})$. Any solution $(\hat{x}, \hat{y})$ to (15) is feasible to $\texttt{MP}^*$ if it also satisfies (9), and that can be checked by simple arithmetic since (9) does not involve recourse decision. If $(\hat{x}, \hat{y})$ is feasible, we can update $\overline{B}$, while *all* attained subproblem solutions are added to the set $\mathcal{S}$ to be excluded from future consideration.

Let $(\overline{x}, \overline{y})$ be a solution at which we attain $\overline{B}$. We claim that $(\overline{x}, \overline{y})$ is optimal when $\underline{B} \geq \overline{B}$. This result can be easily shown by contradiction, for which we assume an optimal solution (x^*, y^*) and $c^1 x^* + c^2 y^* < c^1 \overline{x} + c^2 \overline{y}$. Apparently, $(x^*, y^*) \notin \mathcal{S}$, otherwise it should have been used to update $\overline{B}$. By Proposition 1, $\underline{B} \leq c^1 x^* + c^2 y^* < c^1 \overline{x} + c^2 \overline{y} = \overline{B}$, leading to a contradiction.

We outline the bounding method in Algorithm 1. The algorithm is finitely convergent as in each iteration we remove some solution(s) from the feasible region which contains a finite number of binary solutions (x, y). There are some implementation details that improve computational efficiency. One can respectively initialize $\underline{B}$ and $\overline{B}$ as 0 and the optimal objective value of a robust counterpart of $\texttt{MP}^*$ (i.e., by requiring $\epsilon = 0$ in the chance constraint (9)); in each iteration, recompute a subproblem only if its incumbent solution violates the inequalities generated in last iteration; check the feasibility for a subproblem solution only if its objective values falls in $[\underline{B}, \overline{B})$.

3.3.3 Dual decomposition

In this section, we consider a dual decomposition method for optimizing $\texttt{MP}^*$. We make copies of decision variables x and y, denoted by x^w and y^w, for each scenario $w \in \Omega$, and

Algorithm 1 A decomposition-based bounding method

Initialize $\overline{B}$ and $\underline{B}$.
$\mathcal{S} \leftarrow \emptyset$.
while $\underline{B} < \overline{B}$ **do**
  for $w \in \Omega$ **do**
    compute $v(w, \mathcal{S})$ and collect the corresponding optimal solution $(\hat{x}(w), \hat{y}(w))$.
    if $(\hat{x}(w), \hat{y}(w))$ is feasible **then**
      $\overline{B} \leftarrow v(w, \mathcal{S})$.
    end if
  end for
  sort $v(w, \mathcal{S}), \forall w \in \Omega$ such that $v(\sigma_1, \mathcal{S}) \geq \cdots \geq v(\sigma_{|\Omega|}, \mathcal{S})$.
  $\underline{B} \leftarrow v(\sigma_{\theta+1}, \mathcal{S})$.
  $\mathcal{S} \leftarrow \mathcal{S} \cup_{w \in \Omega} \{(\hat{x}(w), \hat{y}(w))\}$
end while

reformulate $\texttt{MP}^*$ as:

$$\min\ (1/|\Omega|) \sum\nolimits_{w \in \Omega} \left(c^1 x^w + c^2 y^w\right) \tag{16a}$$

$$\text{s.t.}\ \sum\nolimits_{i \in I} \xi_i^w y_{ij}^w \leq \mathrm{T}_j x_j^w + \mathrm{M}_j^w \kappa_w\ \forall j \in J,\ w \in \Omega \tag{16b}$$

$$\sum\nolimits_{w \in \Omega} A_w y^w = b \tag{16c}$$

$$\text{(10b)},\ \kappa \in \{0,1\}^{|\Omega|}$$

$$(x^w, y^w) \in \mathcal{F} = \left\{(x, y) \in \mathcal{A} \cap \{0,1\}^{|J|} \times \{0,1\}^{|I| \times |J|} : \text{(3a)}\right\} \quad \forall w \in \Omega$$

where (16c) are nonanticipativity constraints (NAC) to enforce identical y^w, $\forall w \in \Omega$. (Note that for all $w \in \Omega$, given any fixed y^w, the best solution for x^w subject to $(x^w, y^w) \in \mathcal{F}$ is uniquely given by $x_j^w = \max_{i \in I} y_{ij}^w,\ \forall j \in J$ under the assumption that $c^1 > 0$. Therefore, identical values of y^w across all scenarios suffice for the values of x^w being identical.) The goal is to create a structure that is decomposable by scenario after relaxing cross-scenario constraints, e.g., NAC and the knapsack constraint (10b), and compute bounds for $\texttt{MP}^*$ using a Lagrangian relaxation.

To the best of our knowledge, there are two notable work on that has applied Lagrangian relaxation approach for solving general CCPs. Watson et al. (2010) relax both NAC and the knapsack constraint, and solve the problem by progressive hedging. Ahmed et al. (2014) relax NAC only, but show that the corresponding Lagrangian relaxation is computable through solving a set of scenario subproblems and then solving a single knapsack problem. In our attempt, we relax both sets of constraints - associating them with dual variables to form a relaxation function. We demonstrate that a maximization of the relaxation function over some dual variable in fact has closed-form solution, leading to a similar result with Ahmed et al. (2014). After that, we present another bounding method, which is a simple extension of the method proposed by Ahmed (2013). The closed-form result allows us to simplify one major computational step.

We introduce a variable $\rho \in \mathbb{R}_+$ and a vector λ associated with (10b) and (16c), respec-

tively, and formulate the Lagrangian relaxation as:

$$\begin{aligned}\ell(\rho,\lambda,\mathcal{S}) = \min\ \ &(16a) + \rho\left(\sum\nolimits_{w\in\Omega}\kappa_w - \theta\right) + \lambda\left(\sum\nolimits_{w\in\Omega} A_w y^w - b\right)\\ \text{s.t.}\ \ &(16b),\ (x^w,y^w)\in\mathcal{F}\setminus\mathcal{S},\ \kappa_w\in\{0,1\}\quad \forall w\in\Omega\end{aligned}$$

where $\mathcal{S}$ is a set of (x,y)-vertices excluded from consideration. We give an analogy of Proposition 1 as follows.

Proposition 2. Let $\mathcal{P}^*$ be the complete set of optimal solutions of $\mathtt{MP}^*$ in the space of (x,y). For any set $\mathcal{S}\subseteq\{0,1\}^{|J|}\times\{0,1\}^{|I|\times|J|}$, if $\mathcal{P}^*\nsubseteq\mathcal{S}$ then $\ell(\rho,\lambda,\mathcal{S})$ is a lower bound for the optimal objective value, for any $\rho\geq 0$ and λ.

The proof is presented in Appendix B. We rearrange $\ell(\rho,\lambda,\mathcal{S})$ into:

$$\begin{aligned}\ell(\rho,\lambda,\mathcal{S}) &= -\theta\rho - b\lambda + \sum\nolimits_{w\in\Omega}\begin{pmatrix}\min_{x,y,\kappa} & \rho\kappa + \lambda A_w + (cx+dy)/|\Omega| \\ \text{s.t.} & \sum_{i\in I}\xi_i^w y_{ij}\leq \mathrm{T}_j x_j + \mathrm{M}_j^w\kappa, \forall j\in J \\ & (x,y)\in\mathcal{F}\setminus\mathcal{S},\ \kappa\in\{0,1\}\end{pmatrix}\\ &= -\theta\rho - b\lambda + \sum\nolimits_{w\in\Omega}\min\left\{\rho + h_1^w(\lambda,\mathcal{S}), h_0^w(\lambda,\mathcal{S})\right\}. \qquad (17)\end{aligned}$$

The second equality follows from letting $\kappa=1$ and $\kappa=0$. For each scenario $w\in\Omega$, we detail the subproblems $h_1^w(\lambda,\mathcal{S})$ and $h_0^w(\lambda,\mathcal{S})$ as

$$\begin{aligned}h_1^w(\lambda,\mathcal{S}) &= \min_{x,y}\left\{\lambda A_w + (cx+dy)/|\Omega| : (x,y)\in\mathcal{F}\setminus\mathcal{S}\right\}\\ h_0^w(\lambda,\mathcal{S}) &= \min_{x,y}\left\{\lambda A_w + (cx+dy)/|\Omega| : \sum\nolimits_{i\in I}\xi_i^w y_{ij}\leq \mathrm{T}_j x_j,\ (x,y)\in\mathcal{F}\setminus\mathcal{S}\right\}.\end{aligned}$$

Proposition 3. For fixed λ and $\mathcal{S}$, maximizing $\ell(\rho,\lambda,\mathcal{S})$ over $\rho\geq 0$ has a close-form solution:

$$\max_{\rho\geq 0}\{\ell(\rho,\lambda,\mathcal{S})\} = \overline{\ell}(\lambda,\mathcal{S}) = -b\lambda + \sum\nolimits_{n=1}^{|\Omega|-\theta} h_0^{\sigma_n}(\lambda,\mathcal{S}) + \sum\nolimits_{n=|\Omega|-\theta+1}^{|\Omega|} h_1^{\sigma_n}(\lambda,\mathcal{S}).$$

where $(\sigma_n)_{n=1}^{|\Omega|}$ is a permutation of scenarios such that

$$h_0^{\sigma_1}(\lambda,\mathcal{S}) - h_1^{\sigma_1}(\lambda,\mathcal{S}) \leq \cdots \leq h_0^{\sigma_{|\Omega|}}(\lambda,\mathcal{S}) - h_1^{\sigma_{|\Omega|}}(\lambda,\mathcal{S}).$$

Proof. Consider the monotonicity of $\ell(\rho,\lambda,\mathcal{S})$ over ρ:

- For $\rho\in\left[0, h_0^{\sigma_1}(\lambda,\mathcal{S}) - h_1^{\sigma_1}(\lambda,\mathcal{S})\right)$, $\ell(\rho,\lambda,\mathcal{S}) = -b\lambda + (|\Omega|-\theta)\rho + \sum_{w\in\Omega} h_1^w(\lambda,\mathcal{S})$, which increases in ρ;
- For $\rho\in\left[h_0^{\sigma_{|\Omega|}}(\lambda,\mathcal{S}) - h_1^{\sigma_{|\Omega|}}(\lambda,\mathcal{S}), +\infty\right)$, $\ell(\rho,\lambda,\mathcal{S}) = -b\lambda - \theta\rho + \sum_{w\in\Omega} h_0^w(\lambda,\mathcal{S})$, which decreases in ρ;
- For $\rho\in\left[h_0^{\sigma_n}(\lambda,\mathcal{S}) - h_1^{\sigma_n}(\lambda,\mathcal{S}), h_0^{\sigma_{n+1}}(\lambda,\mathcal{S}) - h_1^{\sigma_{n+1}}(\lambda,\mathcal{S})\right)$ with some n,

  $$\ell(\rho,\lambda,\mathcal{S}) = -b\lambda + (|\Omega| - n - \theta)\rho + \sum\nolimits_{k=1}^{n} h_0^{\sigma_k}(\lambda,\mathcal{S}) + \sum\nolimits_{k=n+1}^{|\Omega|} h_1^{\sigma_k}(\lambda,\mathcal{S}).$$

  Therefore, for $n\in\{1,\ldots,|\Omega|-\theta-1\}$, it increases in ρ; for $n = |\Omega|-\theta$, it is constant; for $n\in\{|\Omega|-\theta+1,\ldots,|\Omega|-1\}$, it decreases in ρ.

Thus, a maximizer is $\rho^* = h_0^{\sigma_{|\Omega|-\theta}}(\lambda, \mathcal{S}) - h_1^{\sigma_{|\Omega|-\theta}}(\lambda, \mathcal{S})$, which yields the proposed maximum objective value. □

The proposition is based on the assumption that all the scenarios are equal likely. Consider a more general case where each scenario realizes with probability p_w. A similar derivation will lead to the same conclusion as what Ahmed et al. (2014) illustrated in Section 2.1 of their paper.

Based on the decomposition of Lagrangian relaxation, Algorithm 2 outlined below has a bounding process similar with Algorithm 1, and its finite convergence can also be justified similarly. The difference is that, in Algorithm 2, we update lower bound using $\overline{\ell}(\lambda, \mathcal{S})$, and perform an inner loop of subgradient method for updating λ. We will discuss more implementation details in later computation.

Algorithm 2 A dual-decomposition-based bounding method

Initialize $\overline{B}$ and $\underline{B}$.
$\mathcal{S} \leftarrow \emptyset$.
while $\overline{B} - \underline{B} > \epsilon$ **do**
  initialize λ.
  while termination condition of the subgradient method is not satisfied **do**
    update λ.
    for $w \in \Omega$ **do**
      compute $h_1^w(\lambda, \mathcal{S})$.
      compute $h_0^w(\lambda, \mathcal{S})$ and collect the corresponding optimal solution $(\hat{x}(w), \hat{y}(w))$.
      if $(\hat{x}(w), \hat{y}(w))$ is feasible **then**
        $\overline{B} \leftarrow c^1\hat{x}(w) + c^2\hat{y}(w)$.
      end if
    end for
  end while
  $\underline{B} \leftarrow \overline{\ell}(\lambda, \mathcal{S})$.
  $\mathcal{S} \leftarrow \mathcal{S} \cup_{w\in\Omega} \{(\hat{x}(w), \hat{y}(w))\}$
end while

3.3.4 Projection cuts

In Algorithms 1 and 2, there may be a large number of no-good cuts generated. Here we consider a set of auxiliary inequalities proposed by Song et al. (2014). For a general CCP, these inequalities define the projection of the LP feasible region of an extended formulation onto the space of original variables (i.e., variables that are in the original CCP). With these inequalities, one can avoid using artificial variables which are as many as the number of scenarios. Corresponding to an extended formulation, there are exponentially many projection inequalities, but we focus on the one that our current solution violates the most.

Given a solution $(\hat{x}, \hat{y}) \in [0,1]^{|J|} \times [0,1]^{|I|\times|J|}$, let

$$\Omega^* = \left\{ w \in \Omega : \exists j \in J \text{ s.t. } \sum\nolimits_{i\in I} \xi_i^w \hat{y}_{ij} - \mathrm{T}_j \hat{x}_j > 0 \text{ and } \mathrm{M}_j^w > 0 \right\},$$

and $j(w) : \Omega \to J$ be a function in scenario w such that

$$j(w) \in \text{argmax}_{j \in J} \left\{ \left(\sum\nolimits_{i \in I} \xi_i^w \hat{y}_{ij} - \mathrm{T}_j \hat{x}_j \right) / \mathrm{M}_j^w : \mathrm{M}_j^w > 0 \right\}.$$

We check whether $(\hat{x}, \hat{y})$ satisfies the following projection inequality – if not, add the inequality to the formulation.

$$\sum\nolimits_{w \in \Omega^*} \frac{1}{|\Omega| \cdot \mathrm{M}_{j(w)}^w} \left(\sum\nolimits_{i \in I} \xi_i^w y_{ij^*(w)} - \mathrm{T}_{j(w)} x_{j(w)} \right) \leq \epsilon. \tag{18}$$

One can use big-M coefficients that are attained by some coefficient strengthening method. We discuss the implementation details in later computation.

3.4 Server-wise Decomposition for $\mathtt{SP}(\hat{x}, \hat{y})$

Given a master-problem solution $(\hat{x}, \hat{y})$, we focus on solving $\mathtt{SP}(\hat{x}, \hat{y})$ in this section. As formulated in Section 3.1, $\mathtt{SP}(\hat{x}, \hat{y})$ is a mixed-integer feasibility problem with a joint chance constraint. We can still resort to an extended formulation and apply strengthening methods to improve big-M coefficients. Alternatively, the branch-and-cut algorithm described in Section 3.3.1 is also applicable. In this section, we exploit a special structure of appointment scheduling on multiple servers and develop a server-wise decomposition method.

3.4.1 Server-wise decomposition and branch-and-cut

In the formulation of $\mathtt{SP}(\hat{x}, \hat{y})$, the chance constraint (7) is a cross-server constraint, while the rest are server-based constraints. We use a branch-and-cut method which is a simple extension of the integer L-shaped method studied in Laporte and Louveaux (1993). We keep (7) in a master problem and enforce the remaining constraints by cuts generated from server-based subproblems.

We define binary variables $q_w,\ \forall w \in \Omega$ to activate ($q_w = 0$) and deactivate ($q_w \neq 0$) constraints (6a)–(6c) in individual scenarios. At a branch-and-bound node β, we consider an LP formulation as:

$$\begin{aligned}
\mathcal{N}(\beta, \Theta): \quad \max\ & \sum\nolimits_{w \in \Omega} q_w \\
\text{s.t.}\ & \sum\nolimits_{w \in \Omega} q_w \leq \theta && (19) \\
& \sum\nolimits_{w \in C} q_w \geq 1 \quad \forall C \in \Theta && (20) \\
& q_w = 0,\ \forall w \in \Omega_0(\beta),\ q_w = 1,\ \forall w \in \Omega_1(\beta) \\
& 0 \leq q_w \leq 1,\ \forall w \in \Omega \setminus \big(\Omega_0(\beta) \cup \Omega_1(\beta)\big).
\end{aligned}$$

Here $\Omega_0(\beta)$ and $\Omega_1(\beta)$ are disjoint subsets of Ω where q_w is fixed to 0 and 1, respectively. (19) is from the probabilistic guarantee and (20) are cover inequalities for enforcing server-based constraints which are not explicitly formulated here. In particular, $\Theta \subseteq 2^\Omega$ is a set of *scenario covers*. Each cover C is a subset of scenarios that server-based constraints cannot be jointly satisfied. Note that $\mathtt{SP}(\hat{x}, \hat{y})$ is a feasibility problem, but at each node β we create

a non-zero objective for $\mathcal{N}(\beta, \Theta)$. The purpose is not to trigger any bound in the branch-and-bound tree, but drive to pick a solution q with a larger support. We will explain the desirability of a large support for q immediately.

If $\mathcal{N}(\beta, \Theta)$ is infeasible, we fathom the node β. If it produces a non-integral solution, we branch on a fractional variable and append two new nodes. If it produces an integral solution $\hat{q}$, we consider the following subproblems for every $j \in J(\hat{x})$:

$$
\begin{aligned}
\mathcal{G}_j(\hat{q}): \quad & \max\ 0 \\
& \text{s.t. } (5a) - (5c) \\
& \quad u_{ik} \in \{0,1\},\ r_k \geq 0,\ \forall i \in I_j(\hat{y}),\ k = 1, \ldots, |I_j(\hat{y})| \\
& \quad \forall w \in \Omega \text{ with } \hat{q}_w = 0: \qquad (6a) - (6c)
\end{aligned}
$$

If $\mathcal{G}_j(\hat{q}), \forall j \in J(\hat{x})$ are feasible, we can terminate the procedure since a feasible solution to $\texttt{SP}(\hat{x}, \hat{y})$ is found. Otherwise, we add the current combination of "active" scenarios as a cover, i.e., $\Theta = \Theta \cup \{\{w \in \Omega : \hat{q}_w = 0\}\}$, and resolve $\mathcal{N}(\beta, \Theta)$. Note that a solution $\hat{q}$ with a larger support results in less restricted subproblems $\mathcal{G}_j(\hat{q}),\ \forall j \in J$ which are more likely to be feasible – leading to termination of the procedure; even if some subproblem is infeasible, such a $\hat{q}$ leads to a stronger cover inequality. We outline the steps in Algorithm 3.

3.4.2 Irreducibly infeasible subsystem for identifying scenario covers

Here we explore an initialization for set Θ better than simply setting it to an empty set. We identify scenario covers based on irreducibly infeasible subsystem (IIS) of an LP model where we relax integrality restriction and require that the scenario-based constraints are satisfied in all scenarios. The method is considered by Tanner and Ntaimo (2010) for solving CCPs arising in vaccine planning. A similar idea of applying IIS to develop cutting planes is seen in Codato and Fischetti (2006), where they develop combinatorial Benders cuts for solving general MILP problems.

For notation simplicity we first rewrite the scenario-based constraints (6a)–(6c) in a vector form as: $\Lambda^1 r + \Lambda^2_w u + \Lambda^3 \gamma^w \leq \zeta$, and consolidate all the constraints of $\texttt{SP}(\hat{x}, \hat{y})$ except for (7) into a vector form as $F^1 r + F^2 u \leq f$. Here ζ and f are appropriately sized vectors, while $\Lambda^1,\ \Lambda^3,\ \Lambda^2_w,\ \forall w \in \Omega$ and $F^1,\ F^2$ are appropriately sized matrices. We consider a variant of $\texttt{SP}(\hat{x}, \hat{y})$ where the scenario-based constraints are enforced in a fixed set of scenarios $\Omega' \subseteq \Omega$. The LP relaxation of this problem is given as:

$$
\begin{aligned}
& \max\ 0 \\
& \text{s.t. } \Lambda^1 r + \Lambda^2_w u + \Lambda^3 \gamma^w \leq \zeta \quad \forall w \in \Omega' \\
& \quad F^1 r + F^2 u \leq f \\
& \quad 0 \leq u \leq \mathbf{1},\ r \geq 0,\ \gamma^w \geq 0,\ \forall w \in \Omega'
\end{aligned}
$$

Based on Gleeson and Ryan (1990), IIS of the above LP formulation are in one-to-one

Algorithm 3 A server-decomposition-based branch-and-cut algorithm

Initialize Θ (e.g., $\emptyset$).
NodeList $\leftarrow \{$a node with $\Omega_0 = \Omega_1 = \emptyset\}$.
choose some node $\beta \in$ NodeList.
solve $\mathcal{N}(\beta, \Theta)$.
if attain an integral solution $\hat{q}$ **then**
  solve $\mathcal{G}_j(\hat{q}),\ \forall j \in J(\hat{x})$.
  if any $\mathcal{G}_j(\hat{q})$ infeasible **then**
    $\Theta \leftarrow \Theta \cup \{\{w \in \Omega : \hat{q}_w = 0\}\}$.
    go to Step 4.
  else
    claim $\mathtt{SP}(\hat{x}, \hat{y})$ feasible and exit.
  end if
else
  if attain a non-integral solution $\hat{q}$ **then**
    choose $w \in \Omega$ such that $\hat{q}_w \in (0, 1)$.
    create new nodes β_1 and β_2, such that:
      $\Omega_0(\beta_1) \leftarrow \Omega_0(\beta) \cup \{w\}$, $\Omega_1(\beta_1) \leftarrow \Omega_1(\beta)$.
      $\Omega_0(\beta_2) \leftarrow \Omega_0(\beta)$, $\Omega_1(\beta_2) \leftarrow \Omega_1(\beta) \cup \{w\}$.
    NodeList $\leftarrow$ NodeList $\cup\{\beta_1, \beta_2\}$
  end if
end if
NodeList $\leftarrow$ NodeList$\backslash\{\beta\}$.
if NodeList is empty **then**
  claim $\mathtt{SP}(\hat{x}, \hat{y})$ infeasible and exit.
else
  go to Step 3.
end if

correspondence with the supports of vertices of the following polyhedron:

$$\Pi(\Omega') = \Bigg\{(\phi, \pi): \quad \begin{array}{l} \pi^\mathsf{T} f + \sum_{w\in\Omega'} \phi_w^\mathsf{T} \zeta \le -1 \\ \pi^\mathsf{T} F^1 + \sum_{w\in\Omega'} \phi_w^\mathsf{T} \Lambda^1 \ge 0 \\ \pi^\mathsf{T} F^2 + \sum_{w\in\Omega'} \phi_w^\mathsf{T} \Lambda_w^2 \ge 0 \\ \phi_w^\mathsf{T} \Lambda^3 \ge 0,\ \forall w \in \Omega' \\ \pi \ge 0,\ \phi_w \ge 0,\ \forall w \in \Omega' \end{array} \Bigg\}.$$

Given a vertex $(\hat{\phi}, \hat{\pi})$ of $\Pi(\Omega')$, we recover a scenario cover according to:

$$C(\hat{\phi}, \hat{\pi}) = \left\{w \in \Omega' : \hat{\phi}_w \neq 0\right\} = \left\{w \in \Omega' : \max_{d=1,\ldots,\dim(\phi_w)} \left(\hat{\phi}_w\right)_d > 0\right\}.$$ [3]

Note that a cover inequality tends to be stronger when $|C|$ is small. Therefore, to find a

[3] $\dim(\cdot)$ returns the dimension of $\cdot$.

cover we propose solving the following LP problem:

$$\min \left\{ \sum\nolimits_{w\in\Omega'} \psi_w : \psi_w \geq e^{\mathsf{T}}\phi_w,\ \forall e \in \mathcal{E},\ (\phi,\pi) \in \Pi(\Omega') \right\}. \tag{21}$$

where $\mathcal{E}$ is a complete set of appropriately sized unit vectors. Once some cover C is identified, we can prevent it from occurring again in our future search by removing some scenario $w \in C$ from the set of scenarios that we intend to satisfy, i.e., Ω'. We then continue searching until (21) is infeasible which suggests that the incumbent Ω' contains scenarios that can be jointly satisfied. We outline the steps in Algorithm 4. Although it does not suffice for identifying

Algorithm 4 An IIS method for identifying scenario covers

1: $\Omega' \leftarrow \Omega$.
2: **repeat**
3: solve (21).
4: **if** attain solution $(\hat{\pi}, \hat{\phi})$ **then**
5: recover $C(\hat{\pi}, \hat{\phi})$.
6: $\Theta \leftarrow \Theta \cup \{\{w \in \Omega' : \hat{\phi}_w \neq 0\}\}$.
7: pick a scenario $w \in C(\hat{\pi}, \hat{\phi})$ and $\Omega' \leftarrow \Omega' \setminus \{w\}$.
8: **end if**
9: **until** (21) is infeasible.

all the scenario covers for $\texttt{SP}(\hat{x}, \hat{y})$, it provides a good initialization for Θ in Algorithm 2.

4 Variant Models of CC-MAS

4.1 Multiple Chance Constraints

Consider a variant of CC-MAS where we enforce low overtime risk at servers individually. In this case, we have one chance constraint for each server, replacing (2) with:

$$\sum\nolimits_{w\in\Omega} \mathbb{I}\left\{(x, y, z, s) \in \mathcal{Q}_j(\xi^w)\right\} \geq |\Omega| - \lfloor \epsilon_j |\Omega| \rfloor.$$

Polyhedron $\mathcal{Q}_j(\xi^w)$ is defined by constraints (4a), (4b) and the j^{th} set of constraints in (4c). Scalar ϵ_j is the risk tolerance for every server j. In this case, $\texttt{MP}^*$ adapts the following $|J|$ individual chance constraints to replace the current joint chance constraint (9):

$$\sum\nolimits_{w\in\Omega} \mathbb{I}\left\{ \sum\nolimits_{i\in I} \xi_i^w y_{ij} \leq \mathrm{T}_j x_j \right\} \geq |\Omega| - \lfloor \epsilon_j |\Omega| \rfloor.$$

In the corresponding extended formulation, we then have one set of artificial variables $\{\kappa_w^j \in \{0,1\}\}_{w\in\Omega}$ for every server j.

The separation problem $\texttt{SP}(\hat{x}, \hat{y})$, without a joint chance constraint that links decisions across servers, becomes completely server-wise decomposable. We then replace $\texttt{SP}(\hat{x}, \hat{y})$ with $|J(\hat{x})|$ server-based subproblems as:

$$\max\ \left\{ 0 : \sum\nolimits_{w\in\Omega} \mathbb{I}\big\{(6a) - (6c)\big\} \geq |\Omega| - \lfloor \epsilon_j |\Omega| \rfloor,\ (5a) - (5c) \right\}$$

for all $j \in J(\hat{x})$. Each of them is a two-stage CCP with continuous recourse decisions. We can still resort to an extended formulation and apply strengthening methods to improve big-M coefficients. Alternatively, one can apply the branch-and-cut algorithm described in Section 3.3.1 or the traditional integer L-shape method by Laporte and Louveaux (1993).

4.2 Constraints on Appointment Waiting

The amount of appointment waiting time is another important metric to evaluate service quality in some application. Essentially, low waiting is in conflict with low overtime, because a schedule with short intervals between arrivals tends to have short overtime but high waiting, and vice versa. In this CC-MAS variant, we seek a good tradeoff by enforcing an upper limit W_i on waiting for every appointment as:

$$t_i^w - s_i \leq \mathrm{W}_i \tag{4d}$$

for all $w \in \Omega$. In general, the time-window restriction differentiates our problem from a stochastic bin packing problem where solving only MP* is sufficient. It is worth noting that a waiting-time restriction of the form (4d) has a similar effect: it makes the sequencing decisions non-trivial, which can be seen through an example presented in Appendix C. Note that constraints (4d) are added to the group of inequalities defining $\mathcal{Q}(\xi^w)$ for all $w \in \Omega$, and will not affect the structure of the problem. All the solution methods can still be applied.

4.3 Recourse Cost in the Objective

Consider a class of CC-MAS variants where we take into account the costs of recourse decisions. Liu et al. (2014) extend the traditional CCP by considering recourse cost incurred in satisfied scenarios and unsatisfied scenarios with respect to the chance constraint. The authors propose a class of valid "optimality cut" derived from the valid "feasibility cut" by Luedtke (2013) that was discussed in Section 3.3.1. The approach is applicable to our problem, whereas we focus on how to adapt the previous decomposition algorithms for CC-MAS to solve this variant.

We first focus on a model that penalizes overtime, so that we control the overtime magnitude in the violating scenarios. Let c_j^3 be the unit penalty cost for overtime in server j. Define a variables $o_j^w \in \mathbb{R}_+$ as the overtime of every server j in each scenario w. We formulate the new objective function as:

$$c^1 x + c^2 y + (1/|\Omega|) \sum\nolimits_{w \in \Omega} \sum\nolimits_{j \in J} c_j^3 o_j^w$$

and add constraints $o_j^w \geq t_i^w + \xi_i^w - \mathrm{T}_j - \mathcal{M}_{ijw}^3(1 - y_{ij}),\ \forall i \in I$ to calculate the length of overtime on every server j in each scenario w.

The two-stage programming framework. The two-stage framework is still applicable, with adaptive changes described in the following. In the master problem, we introduce a continuous variable δ to represent the second-stage overtime cost and use optimality cuts to

enforce its optimal value. The objective functions of MP and MP* both change to $c^1x+c^2y+\delta$. The separation problem SP$(\hat{x},\hat{y})$ adapts a non-zero objective and we rewrite it as:

$$\begin{aligned}\Delta(\hat{x},\hat{y}) = \min\ & \frac{1}{|\Omega|}\sum\nolimits_{w\in\Omega}\sum\nolimits_{j\in J(\hat{x})} c_j^3\left(\gamma^w_{|I_j(\hat{y})|}+\sum\nolimits_{i\in I_j(\hat{y})}\xi_i^w u_{i|I_j(\hat{y})|}-\mathrm{T}_j\right)^+ \quad [4]\\ \text{s.t. }& (5a)-(5c),\ \forall j\in J(\hat{x}), \quad (7)\end{aligned}$$

Given any master-problem solution $(\hat{x},\hat{y},\hat{\delta})$, we generate a feasibility cut (8) if $\Delta(\hat{x},\hat{y}) = +\infty$, and generate the following optimality cut if $\Delta(\hat{x},\hat{y}) > \hat{\delta}$:

$$\delta \geq (\Delta(\hat{x},\hat{y})-L)\left(\sum\nolimits_{j\in J(\hat{x})}\sum\nolimits_{i\in I_j(\hat{y})} y_{ij} - |I| + 1\right) + L$$

where L is a lower bound for $\Delta(x,y)$ given any master-problem solution (x,y), e.g., $L=0$. This cut is a simple composition of the general-form optimality cut proposed by Laporte and Louveaux (1993), and the strengthened no-good cut (8).

The strengthened master problem and decomposition. The inclusion of optimality cuts does not affect the feasible region in the (x,y) space, so all the methods in Section 3.2 and Section 3.3 are still applicable. By replacing c^1x+c^2y with $c^1x+c^2y+\delta$, and labeling optimality cuts generated in earlier stages also as (3a), we can reuse all the notation and algorithm steps.

The separation problem and its decomposition. Now that the separation problem of the variant has a non-zero objective, we introduce a continuous variable η and add optimality cuts to force it being an upper bound for the total overtime cost. The problem solved at each node β becomes:

$$\begin{aligned}&\min\ \ \eta\\ &\text{s.t. } \eta \geq \left(\sum\nolimits_{j\in J(\hat{x})} g_j(\hat{q}(n)) - L\right)\left(1-\sum\nolimits_{w:(\hat{q}(n))_w=0} q_w\right)+L\\ &\qquad\qquad\qquad\qquad\qquad\qquad \forall n=1,\dots,N \qquad (22)\\ &\quad (19),\ (20),\ q_w=0,\ \forall w\in\Omega_0(\beta),\ q_w=1,\ \forall w\in\Omega_1(\beta)\\ &\quad q_w\in[0,1],\ \forall w\in\Omega\setminus\big(\Omega_0(\beta)\cup\Omega_1(\beta)\big)\end{aligned}$$

where $g_j(q)$ is the optimal objective value of a new server-based subproblem formulated as:

$$\begin{aligned}g_j(\hat{q}) = \min\ & \frac{c_j^3}{|\Omega|}\sum\nolimits_{w:\hat{q}_w\neq 0}\left(\gamma^w_{|I_j(\hat{y})|}+\sum\nolimits_{i\in I_j(\hat{y})}\xi_i^w u_{i|I_j(\hat{y})|}-\mathrm{T}_j\right)^+\\ \text{s.t. }& (6a),\ (6b),\ \forall w\in\Omega,\ (6c),\ \forall w\in\Omega:\hat{q}_w=0\\ & (5a)-(5c),\ u_{ik}\in\{0,1\},\ \forall i\in I_j(\hat{y}),\ k=1,\dots,|I_j(\hat{y})|\\ & r_k\geq 0,\ \gamma_k^w\geq 0,\ \forall k=1,\dots,|I_j(\hat{y})|,\ w\in\Omega.\end{aligned}$$

The server-based subproblem adapts a non-zero objective that computes the expected overtime cost. (22) present optimality cuts, in which $\hat{q}(n),\ \forall n=1,\dots,N$ are some solutions

[4] function $(\cdot)^+$ returns the maximum of $\cdot$ and 0.

attained earlier. Intuitively, if we activate all the scenarios that were active under $\hat{q}(n)$ (i.e., $\sum_{w:(\hat{q}(n))_w=0} q_w = 0$), the cost must not be lower than the overtime cost attained given $\hat{q}(n)$. Algorithm 3 adapts additional bounding and objective-computing steps: First, we fathom a node β right after solving $\mathcal{N}(\beta, \Theta)$ if the attained optimal objective $\geq \overline{B}$. Second, for some node solution $(\hat{q}, \hat{\eta})$, compute $G = \sum_{j \in J(\hat{x})} g_j(\hat{q})$. Update $\overline{B}$ with G if $G < \overline{B}$; If $G \leq \hat{\eta}$, fathom the node, otherwise we add an optimality cut (22) and resolve $\mathcal{N}(\beta, \Theta)$. As there is no change occurring to the feasible region of the separation problem, the IIS method in Section 3.4.2 is still applicable.

Consider another model that penalizes customer waiting. Let $c_i^4 \in \mathbb{R}$ be a penalty cost for appointment i waiting and the new objective function is:

$$c^1 x + c^2 y + (1/|\Omega|) \sum_{w \in \Omega} \sum_{i \in I} c_i^4 (t_i^w - s_i)^+$$

The two-stage programming framework is still applicable, so are the methods proposed for solving the master problem. Given a solution $(\hat{x}, \hat{y})$, we introduce a variable d_{jk}^w and force it being an upper bound for the waiting cost incurred by the k^{th} appointment on server j in scenario w by the following constraints:

$$d_{jk}^w \geq 0,\ d_{jk}^w \geq c_i^4(\gamma_k^w - r_k) - \mathcal{M}_i^w(1 - u_{ik}),\ \forall i \in I_j(\hat{y}).$$

The separation problem then has an objective function $\sum_{j \in J(\hat{x})} \sum_{k=1}^{|I_j(\hat{y})|} \sum_{w \in \Omega} d_{jk}^w$. which is decomposable by server. So the server-wise decomposition method can still be applied.

5 Computational Results

We test our approaches for solving CC-MAS problem instances of scheduling surgeries with uncertain durations to operating rooms on a daily basis ($H = 24$ hrs). We use 2012 surgery data kept by a hospital in Michigan. We fit the mean and variance of surgery durations to the real data and generate random samples based on a multivariate Lognormal distribution, which has been considered well characterizing random surgery durations in the literature (see, e.g., Gul et al. 2011). Rooms have heterogeneous operating time limits ($\mathrm{T}_j = 4 \sim 15$, $j \in J$) and operating cost ($c_j^1 = 8 \sim 18,\ j \in J$). We use a homogenous service cost $c_{ij}^2 = 1,\ \forall i \in I,\ j \in J$. Each surgery has a requested time window $[\underline{a}_i, \overline{a}_i]$ with equal probability chosen from the following three time intervals: [0, 6], [6, 12], and [0, 12]. We set $\epsilon = 0.1$.

All computations are performed on a Linux workstation with four 3.4 GHz processors and 16 GB memory. All involved optimization models are solved by CPLEX 12.6 via ILOG Concert Technology. We test two types of instances respectively involving $|I| = 10$ surgeries, $|J| = 5$ operating rooms, and $|I| = 20$ surgeries and $|J| = 10$ operating rooms. For each instance we test three sample sizes $|\Omega|$ =20, 200, and 2000. We report the result averages by running 10 trails of each combination of problem instance and sample size. The CPU time limit is set to 7200 seconds. An entry "-" in the following result tables indicates that not all ten instances are solved within time limit, and thus no result average is presented.

Two-stage programming framework: We first test how the two-stage programming framework performs by comparing it with directly solving the extended formulation of CC-MAS under the default choice of big-M coefficients (Direct). We implement the framework in a branch-and-cut procedure: At each node we solve the master problem on some restricted region. If an integral solution $(\hat{x}, \hat{y})$ is attained, we solve $\texttt{SP}(\hat{x}, \hat{y})$, of which the solution infeasibility will trigger to generate a no-good cut (8) to the master problem. With respect to the master-problem model, we test two options MP and MP*. All the CCPs involved in the two-stage framework are reformulated under the default big-M choices and solved directly without using any proposed solution methods.

Table 1: Total solution time and number of branched nodes

Instance	$\|\Omega\|$	Direct		MP+SP			MP*+SP		
		total	#node	total	#node	#cut	total	#node	#cut
$\|J\| = 5$	20	269.8	169356	366.2	23	8	1.3	421	3
$\|I\| = 10$	200	-	5333*	4854.3	14320	64	54.6	6503	3
	2000	-	29*	-	879*	87*	-	13199*	5*

*: not all tested cases are solved within time limit. We report the average number of nodes and cuts counted up to when terminating each instance regardless of their solution status.

Table 1 presents the total solution time and the number of branch-and-bound nodes (#node) processed. It also reports the number of no-good cuts (#cut) generated when using the two-stage decomposition framework. Because all three computational schemes fail to solve most of the instances with $|J| = 10, |I| = 20$ within the time limit, we present the results for instances of $|J| = 5, |I| = 10$ only. The table indicates that MP*+SP performs much better in terms of both CPU time and the number of nodes branched. In particular, the original formulation with big-$\mathcal{M}^1$, -$\mathcal{M}^2$, and -$\mathcal{M}^3$ coefficients is replaced by MP* (or MP) and SP which are much easier to compute. Strengthening the master problem with the joint chance constraint (9) reduces the number of iterations and cuts, leading to further reduction of the solution time.

Big-M coefficient strengthening: We implement the following two schemes to compare methods of strengthening the big-M coefficients discussed in Section 3.2.

- $\texttt{MP}^*_{\texttt{iter}}$+SP: Strengthen M_j^w by iteratively solving the LP problem (11) until $\|\mathrm{M}(n) - \mathrm{M}(n-1)\|_2$ is adequately small (where $\mathrm{M}(n) = [\mathrm{M}_j^w(n),\ j \in J,\ w \in \Omega]^\mathsf{T}$).
- $\texttt{MP}^*_{\texttt{scen}}$+SP: Set M_j^w to $\mathrm{m}_j^w(\sigma_{\theta+1}),\ \forall j \in J,\ w \in \Omega$, obtained by solving and sorting optimal results of problem (12).

Table 2 presents the time of solving MP* (reported in columns "mp (sec)"), excluding the time of solving $\texttt{SP}(\hat{x}, \hat{y})$. We report the results of MP*+SP as benchmark, which use the default big-M coefficients. For the above two schemes, in addition we report the time spent on computing strengthened coefficients (reported in "str (sec)"), and the new coefficients as percentages of the default choice (str%).

Table 2: Solution time on solving $\texttt{MP}^*$ and some coefficient strengthening details

Instance	$\lvert\Omega\rvert$	$\texttt{MP}^*$+SP	$\texttt{MP}^*_{\texttt{iter}}$+SP			$\texttt{MP}^*_{\texttt{scen}}$+SP		
		mp (sec)	mp (sec)	str (sec)	str%	mp (sec)	str (sec)	str%
$\lvert J\rvert = 5^\dagger$	20	0.4	0.3	1.1	15.4%	0.1	7.9	1.0%
$\lvert I\rvert = 10$	200	48.7	11.4	20.4	16.0%	1.5	725.4	1.1%
	2000	-	15.8	1917.7	16.0%	5.3	5325.0	1.0%

†: most of the tested instances with $|J| = 10, |I| = 20$ are not solved within time limit, thus their results are not presented.

According to the table, the approach used by $\texttt{MP}^*_{\text{scen}}$+SP excels in the effectiveness of strengthening – yielding smaller coefficients and the strengthened $\texttt{MP}^*$ takes shorter time to solve. However, the strengthening procedure itself takes much longer time. Compared to that, the iterative approach by $\texttt{MP}^*_{\text{iter}}$+SP for updating the big-M coefficients attains a better tradeoff.

Scenario-wise decomposition: We compare the three scenario decomposition methods proposed in Section 3.3 for solving $\texttt{MP}^*$. All of them avoid using big-M coefficients, but develop scenario-based subproblems to generate cuts or objective bounds.

- `B&C+SP`: Solve the branch-and-cut algorithm with lifted valid inequalities discussed in Section 3.3.1.
- `Pbnd+SP`: Solve $\texttt{MP}^*$ by Algorithm 1, which is a bounding method with subproblems derived from decomposing the primal formulation of $\texttt{MP}^*$.
- `Dbnd+SP`: Solve $\texttt{MP}^*$ by another bounding method in Algorithm 2 where subproblems are derived based on a Lagrangian relaxation of $\texttt{MP}^*$. We discuss some implementation details of the subgradient method in the following. To update the dual multiplier λ, we compute the step-length to be used in iteration $n+1$ based on the Polyak rule (cf. Bertsekas 1999):

  $$\tau_{n+1} = (\overline{B} - \overline{\ell}(\lambda_n, \mathcal{S}))/\|\alpha(\lambda_n)\|_2^2$$

  where $\overline{\ell}(\lambda_n, \mathcal{S})$ is the dual objective value, and $\alpha(\lambda_n)$ is the subgradient of the current iteration n. We terminate the subgradient-method iteration when the improvement of the optimal dual objective is adequately small, i.e., $0 < \overline{\ell}(\lambda_n, \mathcal{S}) - \overline{\ell}(\lambda_{n-1}, \mathcal{S}) < 0.05$.

In all the three schemes of solving $\texttt{MP}^*$, we directly solve the separation problem $\texttt{SP}(\hat{x}, \hat{y})$ as its MILP reformulation by using the default choice of big-M coefficients.

Table 3 presents the total solution time ("total"), the number of scenario-based subproblems solved ("#sub") and the solution time per subproblem ("sub (sec)"). All the three schemes manage to solve all the instances with $|J| = 5$ and $|I| = 10$ within time limit. When solving the instances with $|J| = 10$ and $|I| = 20$, `Pbnd+SP` and `Dbnd+SP` benefit from the high efficiency of solving the scenario-based subproblems. We observe little time increase in

Table 3: Total solution time, number of scenario-based subproblems, and average solution time per subproblem

Instance	$\|\Omega\|$	B&C+SP			Pbnd+SP			Dbnd+SP		
		total (sec)	#sub	sub (sec)	total (sec)	#sub	sub (sec)	total (sec)	#sub	sub (sec)
$\|J\| = 5$	20	2.5	35	0.06	3.2	146	0.02	1.6	248	0.008
$\|I\| = 10$	200	173.2	388	0.44	10.0	568	0.02	13.8	2480	0.007
	2000	2494.9	3754	0.66	78.5	2100	0.03	185.2	29500	0.007
$\|J\| = 10$	20	6535.8	2672	2.40	115.3	751	0.12	46.8	656	0.07
$\|I\| = 20$	200	-	-	-	410.5	1136	0.16	347.8	2080	0.09
	2000	-	-	-	1332.4	4000	0.13	1053.4	8324	0.10

solving each individual subproblem as we increase the sample size. Most of the instances are solved within half an hour. Overall, Pbnd+SP is the most efficient, and in the following we test whether adding projection cuts can make the procedure faster.

- Pbnd$^{\perp}$+SP: We add projection cuts through a two-phase procedure suggested by Song et al. (2014): First, we solve the LP relaxation of MP* (using big-M coefficients attained from the iterative strengthening approach) by a cutting-plane method. In particular, we enforce the chance constraint (9) through projection cuts (18), which are gradually identified and added until no more inequalities passing a parallel check[5] are found. We then add all these cuts to the initial formulation of each subproblem (15), and start Algorithm 1. We test each infeasible solution on (18) and add the inequality as a cut again if it passes the parallel check.

We compare Pbnd$^{\perp}$+SP with Pbnd+SP in Table 4, where we also report the number of iterations for bounding the objective ("#bnd") in both schemes, as well as the number of projection cuts generated ("#proj") and the time spent on constructing them ("proj (sec)") for Pbnd$^{\perp}$+SP.

Table 4: Total solution time, number of bounding iterations, number of scenario-based subproblems, solution time per subproblem, time on deriving projection cuts and number of projection cuts

$\|\Omega\|$	Pbnd$^{\perp}$+SP						Pbnd+SP			
	time	#bnd	#sub	sub (sec)	#proj	proj (sec)	time	#bnd	#sub	sub (sec)
20	2.7	6	105	0.02	17	0.002	3.2	7	146	0.02
200	20.3	2	531	0.03	66	0.006	10.0	3	568	0.02
2000	76.7	2	1456	0.04	148	0.040	78.5	2	2100	0.03

According to the table, the addition of projection cuts reduces the number of bounding iterations, and thus fewer scenario subproblems are solved. However, these additional cuts increase the time for solving each scenario-based subproblem in the instances we tested. Altogether, the improvement in the total solution time is not obvious.

[5]Parallel check for a tentative cut: Denote the coefficient vector of the cut as ϑ. For any inequality with coefficient vector ϑ' in the formulation, we compute $\vartheta^{\mathsf{T}}\vartheta'/(\|\vartheta\|\|\vartheta'\|)$. If any of these values > 0.99, we abandon the cut.

Server-wise Decomposition: All the results reported above are based on directly solving the extended formulation of $\texttt{SP}(\hat{x}, \hat{y})$ with the default big-M coefficients. Table 5 presents the time on solving $\texttt{SP}(\hat{x}, \hat{y})$ ("sp (sec)"), and that as a percentage of the total solution time ("sp%") from some previous computational schemes. As the table indicates, $\texttt{SP}(\hat{x}, \hat{y})$ is generally solved quite efficiently, under different methods used for solving $\texttt{MP}^*$. Comparably, the two bounding methods tend to produce better solutions $(\hat{x}, \hat{y})$ from $\texttt{MP}^*$ and lead to shorter time of solving $\texttt{SP}(\hat{x}, \hat{y})$.

Table 5: Solution time on directly solving the separation problem

Instance	$\lvert\Omega\rvert$	$\texttt{MP}^*_{\texttt{iter}}$+SP		B&C+SP		Pbnd+SP		Dbnd+SP	
		sp (sec)	sp%	sp (sec)	sp%	sp (sec)	sp%	sp (sec)	sp%
$\lvert J\rvert = 5$	20	0.3	18.3%	0.4	16.0%	0.8	12.8%	0.1	3.8%
$\lvert I\rvert = 10$	200	3.4	9.8%	0.2	0.1%	0.2	4.5%	0.1	2.5%
	2000	42.0	4.0%	27.7	1.1%	2.4	3.7%	1.6	2.1%
$\lvert J\rvert = 10$	20	4.1	6.7%	23.0	0.3%	0.1	0.1%	0.9	3.8%
$\lvert I\rvert = 20$	200	-	-	-	-	2.4	1.3%	13.8	3.5%
	2000	-	-	-	-	25.3	1.9%	22.1	2.1%

In a comparison, we apply the server-based decomposition method discussed in Section 3.4 to solve $\texttt{SP}(\hat{x}, \hat{y})$ and report the results in Table 6. We see from the table that this method does not help solving the previously unsolved instances. The improvement in computational time for instances with $|J| = 10$ and $|I| = 20$ is not obvious. For instances with $|J| = 5$ and $|I| = 10$, it increases the solution time in some cases. On the contrary, the server-wise decomposition method is crucial for enhancing the computational efficiency of solving instances of the variant model that penalizes overtime in the objective. We present the results shortly.

Table 6: Results of using the server-based decomposition method for the separation problem

Instance	$\lvert\Omega\rvert$	$\texttt{MP}^*_{\texttt{iter}}$+B&C'		B&C+B&C'		Pbnd+B&C'		Dbnd+B&C'	
		sp (sec)	sp%	sp (sec)	sp%	sp (sec)	sp%	sp (sec)	sp%
$\lvert J\rvert = 5$	20	3.4	70.3%	0.3	9.9%	0.5	7.9%	4.3	84.0%
$\lvert I\rvert = 10$	200	13.3	27.9%	3.9	1.7%	3.4	37.2%	2.3	26.0%
	2000	39.7	3.6%	21.6	0.8%	11.7	14.5%	9.4	9.9%
$\lvert J\rvert = 10$	20	27.0	30.5%	27.1	0.3%	12.3	54.3%	2.0	7.3%
$\lvert I\rvert = 20$	200	-	-	-	-	14.0	6.4%	5.7	1.4%
	2000	-	-	-	-	13.2	0.9%	12.1	1.1%

Overtime cost in the objective: For simplicity, we use a homogenous overtime unit penalty cost $c^3_j = 1,\ \forall j \in J,\ w \in \Omega$. We solve the CC-MAS penalty variant for all instances used above under the two-stage framework. For the master problem, we alternate between the two bounding methods Pbnd and Dbnd. For the separation problem, we alternate between

directly solving the extended formulation (SP) and the server-wise decomposition method with branch-and-cut (B&C').

Table 7: Results of the CC-MAS variant with overtime penalty cost

Instance	$\|\Omega\|$	Pbnd+SP		Dbnd+SP		Pbnd+B&C'		Dbnd+B&C'	
		mp (sec)	sp (sec)	mp (sec)	sp (sec)	mp (sec)	sp (sec)	mp (sec)	sp (sec)
$\|J\| = 5$	20	32.4	20.3	29.7	1.5	18.9	34.5	65.5	35.7
$\|I\| = 10$	200	50.7	248.6	110.6	449.3	64.3	104.8	47.5	76.3
penal.	2000	361.1	1242.7	249.1	3032.6	130.9	523.8	117.8	700.7
$\|J\| = 10$	20	369.8	198.5	248.4	104.8	465.3	127.4	388.1	333.0
$\|I\| = 20$	200	842.3	2036.1	502.3	3571.7	535.7	369.6	476.1	629.2
penal.	2000	1567.5	5413.7	1098.4	2499.0	795.4	749.9	731.5	166.9

Table 7 presents the time of solving the master problem ("mp (sec)") and the separation problem ("sp (sec)"), which is substantially longer than solving the generic CC-MAS problem without penalty. On the other hand, using the server-wise decomposition method to solve the separation problem saves the solution time significantly.

Multiple individual chance constraints: Lastly, we compare the CC-MAS variant with individual chance constraints on server overtime (discussed in Section 4.1, referred to as MCC in the following) using the two-stage framework. Here the master problem is a binary integer program with multiple individual chance constraints, while the separation problem solves $|J(\hat{x})|$ smaller-scale mixed-integer chance-constrained problems. We set $\epsilon_j = 0.1,\ \forall j \in J$, and test instances with $|J| = 5$ and $|I| = 10$. We reformulate all individual chance constraints using the default choice of big-M coefficients and solve them directly.

Table 8 compares the total solution time and the number of nodes branched for solving the generic problem with a joint overtime constraint and MCC. In most of the instances, MCC requires fewer branching nodes and shorter computational time.

Table 8: Total solution time and number of nodes

Instance	$\|\Omega\|$	MP*+SP (MCC)		MP*+SP	
		total	#node	total	#node
$\|J\| = 5$	20	0.3	311	1.3	421
$\|I\| = 10$	200	31.9	751	54.6	6503
	2000	4474.5	9701	-	13199*

$*$: not all tested cases are solved within the time limit. We report the average number of nodes and cuts counted up to terminating each instance regardless of their solution status.

6 Conclusions

In this paper, we consider multi-server appointment scheduling problems with chance constraints to bound the risk of server overtime due to random service durations. Each appoint-

ment arrives at an appointed start time in a requested time window, and we minimize the total cost of operating servers and serving all appointments. We present a generic model to impose a sufficiently small risk for satisfying a joint chance constraint on server overtime. We also investigate several variants of the problem to take into account (i) individual chance constraints, (ii) bounded appointment waiting time, and (iii) expected linear penalty cost in the length of overtime.

We propose a two-stage programming framework different from the traditional decomposition method of separating decisions that are made before realizing the uncertainty and those after. (Instead, we keep cross-server decisions and constraints at the first stage, so that the remaining problem features a server-wise decomposable structure.) Decomposition, coefficient strengthening, bounding, and cutting-plane approaches are heavily used to accelerate computing both master and subproblems. Furthermore, we generalize the decomposition algorithms to solving the three variants, by modifying valid inequalities and bounds depending on problem structures. The computational results demonstrate the effectiveness of the proposed approaches. We also show how the CPU time, and the number of nodes and cuts contributed by each part of our algorithm depend on the problem size as well as sample size.

Future research directions include adapting the discussed approaches to mixed-integer programs with multiple chance constraints and exploring parallel computing schemes to implement the decomposition algorithms.

References

S. Ahmed. A scenario decomposition algorithm for 0-1 stochastic programs. *Operations Research Letters*, 41(6):565–569, 2013.

S. Ahmed, J. Luedtke, Y. Song, and W. Xie. Nonanticipative duality and mixed-integer programming formulations for chance-constrained stochastic programs. Available at Optimization-Online `http://www.optimization-online.org/DB_FILE/2014/07/4447.pdf`, 2014.

D. P. Bertsekas. *Nonlinear programming*. Athena Scientific, 1999.

J. T. Blake and J. Donald. Mount sinai hospital uses integer programming to allocate operating room time. *Interfaces*, 32(2):63–73, 2002.

M. Brahimi and D. Worthington. Queueing models for out-patient appointment systems–a case study. *Journal of the Operational Research Society*, 42(9):733–746, 1991.

G. Codato and M. Fischetti. Combinatorial Benders' cuts for mixed-integer linear programming. *Operations Research*, 54(4):756–766, 2006.

Y. Deng, S. Shen, and B. Denton. Chance-constrained surgery planning under uncertain or ambiguous surgery duration. Available at SSRN: `http://ssrn.com/abstract=2432375`, 2014.

D. Dentcheva, A. Prékopa, and A. Ruszczyński. Concavity and efficient points of discrete distributions in probabilistic programming. *Mathematical Programming*, 89(1):55–77, 2000.

B. Denton and D. Gupta. A sequential bounding approach for optimal appointment scheduling. *IIE Transactions*, 35(11):1003–1016, 2003.

B. T. Denton, A. J. Miller, H. J. Balasubramanian, and T. R. Huschka. Optimal allocation of surgery blocks to operating rooms under uncertainty. *Operations Research*, 58(4):802–816, 2010.

S. A. Erdogan and B. Denton. Dynamic appointment scheduling of a stochastic server with uncertain demand. *INFORMS Journal on Computing*, 25(1):116–132, 2013.

J. Gleeson and J. Ryan. Identifying minimally infeasible subsystems of inequalities. *ORSA Journal on Computing*, 2(1):61–63, 1990.

S. Gul, B. T. Denton, J. W. Fowler, and T. Huschka. Bi-criteria scheduling of surgical services for an outpatient procedure center. *Production and Operations Management*, 20(3):406–417, 2011.

R. Hassin and S. Mendel. Scheduling arrivals to queues: A single-server model with no-shows. *Management Science*, 54(3):565–572, 2008.

A. Jebali, A. B. Hadj Alouane, and P. Ladet. Operating rooms scheduling. *International Journal of Production Economics*, 99(1):52–62, 2006.

S. Küçükyavuz. On mixing sets arising in chance-constrained programming. *Mathematical Programming*, 132(1–2):31–56, 2012.

G. Laporte and F. V. Louveaux. The integer L-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13(3):133–142, 1993.

X. Liu, S. Küçükyavuz, and J. Luedtke. Decomposition algorithms for two-stage chance-constrained programs. Available at Optimization-Online `http://www.optimization-online.org/DB_HTML/2014/03/4262.html`, 2014.

J. Luedtke. A branch-and-cut decomposition algorithm for solving chance-constrained mathematical programs with finite support. Online first, *Mathematical Programming*, 2013.

J. Luedtke and S. Ahmed. A sample approximation approach for optimization with probabilistic constraints. *SIAM Journal on Optimization*, 19(2):674–699, 2008.

J. Luedtke, S. Ahmed, and G. Nemhauser. An integer programming approach for linear programs with probabilistic constraints. *Mathematical Programming*, 122(2):247–272, 2010.

A. Nemirovski and A. Shapiro. Scenario approximations of chance constraints. In G. Calafiore and F. Dabbene, editors, *Probabilistic and Randomized Methods for Design Under Uncertainty*, pages 3–47. Springer, London, UK, 2006.

A. Nemirovski and A. Shapiro. Convex approximations of chance constrained programs. *SIAM Journal on Optimization*, 17(4):969–996, 2007.

I. Ozkarahan. Allocation of surgeries to operating rooms by goal programing. *Journal of Medical Systems*, 24(6):339–378, 2000.

B. Pagnoncelli, S. Ahmed, and A. Shapiro. Sample average approximation method for chance constrained programming: Theory and applications. *Journal of Optimization Theory and Applications*, 142(2):399–416, 2009.

F. Qiu, S. Ahmed, S. S. Dey, and L. A. Wolsey. Covering linear programming with violations. *INFORMS Journal on Computing*, 26(3):531–546, 2014.

A. Ruszczyński. Probabilistic programming with discrete distributions and precedence constrained knapsack polyhedra. *Mathematical Programming*, 93(2):195–215, 2002.

S. C. Sarin, H. D. Sherali, and L. Liao. Minimizing conditional-value-at-risk for stochastic scheduling problems. *Journal of Scheduling*, 17(1):5–15, 2014.

S. Sen. Relaxations for probabilistically constrained programs with discrete random variables. *Operations Research Letters*, 11(2):81–86, 1992.

O. V. Shylo, O. A. Prokopyev, and A. J. Schaefer. Stochastic operating room scheduling for high-volume specialties under block booking. *INFORMS Journal on Computing*, 25(4):682–692, 2012.

Y. Song, J. R. Luedtke, and S. Küçükyavuz. Chance-constrained binary packing problems. To appear in *INFORMS Journal on Computing*, 2014.

M. Tanner and L. Ntaimo. IIS branch-and-cut for joint chance-constrained stochastic programs and application to optimal vaccine allocation. *European Journal of Operational Research*, 207 (1):290–296, 2010.

J.-P. Watson, R. J. Wets, and D. L. Woodruff. Scalable heuristics for a class of chance-constrained stochastic programs. *INFORMS Journal on Computing*, 22(4):543–554, 2010.

APPENDIX

A Derivation for the Strengthened No-good Cut (8)

Note that in `CC-MAS`, x is a vector of dependent variables and given any fixed solution $\hat{y}$, the best $\hat{x}$ is uniquely given by: $\hat{x}_j = \max_{i \in I} \hat{y}_{ij}$, under the assumption that $c^1 > 0$. We thus formulate no-good cuts only in y variables which takes the form of:

$$\sum\nolimits_{i\in I, j\in J:\hat{y}_{ij}=1} y_{ij} - \sum\nolimits_{i\in I, j\in J:\hat{y}_{ij}=0} y_{ij} \leq \sum\nolimits_{i\in I, j\in J:\hat{y}_{ij}=1} \hat{y}_{ij} - 1.$$

According to (3a), we have a valid equality: $\sum_{i\in I, j\in J} y_{ij} = |I|$, which if added to the inequality above will yield:

$$\begin{aligned}
&\sum\nolimits_{i\in I, j\in J} y_{ij} + \sum\nolimits_{i\in I, j\in J:\hat{y}_{ij}=1} y_{ij} - \sum\nolimits_{i\in I, j\in J:\hat{y}_{ij}=0} y_{ij} \leq |I| + \sum\nolimits_{i\in I, j\in J:\hat{y}_{ij}=1} \hat{y}_{ij} - 1 \\
\Rightarrow\ & 2\sum\nolimits_{j\in J(\hat{x})} \sum\nolimits_{i\in I_j(\hat{y})} y_{ij} \leq 2|I| - 1 \\
\Rightarrow\ & \sum\nolimits_{j\in J(\hat{x})} \sum\nolimits_{i\in I_j(\hat{y})} y_{ij} \leq |I| - 1
\end{aligned}$$

where we repeatedly apply (3a).

B Proof for Proposition 2

Proof. Consider some $(x^*, y^*) \in \mathcal{P}^* \setminus \mathcal{S}$. There must exist some $\kappa^* \in \{0,1\}^{|\Omega|}$ such that (x^*, y^*, κ^*) is feasible to `MP*`. Now construct a feasible solution $(\hat{x}^w, \hat{y}^w, \hat{\kappa}_w, \forall w \in \Omega)$ for the Lagrangian relaxation by letting $\hat{x}^w = x^*,\ \hat{y}^w = y^*,\ \hat{\kappa}_w = \kappa^*_w,\ \forall w \in \Omega$. This solution yields an objective value of

$$\begin{aligned}
&\sum\nolimits_{w\in\Omega} \frac{1}{|\Omega|} \left(c^1\hat{x}^w + c^2\hat{y}^w\right) + \rho\left(\sum\nolimits_{w\in\Omega} \hat{\kappa}_w - \theta\right) + \lambda\left(\sum\nolimits_{w\in\Omega} A_w\hat{y}^w - b\right) \\
&= c^1x^* + c^2y^* + \rho\left(\sum\nolimits_{w\in\Omega} \kappa^*_w - \theta\right) \\
&\leq c^1x^* + c^2y^*
\end{aligned}$$

which leads to $\ell(\rho, \lambda, \mathcal{S}) \leq c^1x^* + c^2y^*$. □

C Example for CC-MAS with Waiting Time Restriction

Suppose that appointments A and B with waiting tolerances $W_{\mathsf{A}} = 1$ and $W_{\mathsf{B}} = 3$, respectively, are allocated to a server that can operate from time 0 to time 6, Realizations of their service durations are shown in Table C, which satisfy $\xi^1_{\mathsf{A}} + \xi^1_{\mathsf{B}} \leq 6$ and $\xi^2_{\mathsf{A}} + \xi^2_{\mathsf{B}} \leq 6$, and thus

	Scenario 1	Scenario 2
A	$\xi^1_{\mathsf{A}} = 4$ hours	$\xi^2_{\mathsf{A}} = 2$ hours
B	$\xi^1_{\mathsf{B}} = 2$ hours	$\xi^2_{\mathsf{B}} = 4$ hours

Table 9: Realization of service durations of A and B in Example 1

by allocating them on the same server, we satisfy the "packing" chance constraint (9) given any risk tolerance ϵ. Now consider the original chance constraint (2) which accounts for the waiting-time restriction. Using the assumed waiting-time tolerances, we will serve A before B to yield probability 1 of keeping the server not running overtime. Otherwise, if starting appointment B first at time 0, due to $W_{\mathsf{A}} = 1$, the earliest start time for A is time 3 and we incur one-hour overtime in scenario 1 and potentially violate the chance constraint. Now suppose that $W_{\mathsf{B}} = 1$ instead of 3 and $W_{\mathsf{A}} = 1$. Let p_1 and p_2 represent the probabilities of scenario 1 and scenario 2, respectively. Then serving B before A yields probability p_1 of running the server overtime and by symmetry, serving A before B leads to p_2 overtime probability. Therefore, if the chance constraint (2) has a risk tolerance $\epsilon < \min\{p_1, p_2\}$, we have to open another server to process the two appointments.