INFORMS JOURNAL ON OPTIMIZATION
Vol. 00, No. 0, Xxxxx 2021, pp. 000–000
ISSN XXXX-XXXX | EISSN XXXX-XXXX | 00 | 0000 | 0001

INFORMS
DOI 10.1287/xxxx.0000.0000

A Solution Approach to Distributionally Robust Chance-Constrained Assignment Problems

Shanshan Wang
Department of Management Science and Engineering, Beijing Institute of Technology, Beijing, China 100081
shshwang_bit@163.com

Jinlin Li
Department of Management Science and Engineering, Beijing Institute of Technology, Beijing, China 100081
jinlinli@bit.edu.cn

Sanjay Mehrotra
Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL, 60208
mehrotra@iems.northwestern.edu

We study the assignment problem with chance constraints (CAP) and its distributionally robust counterpart (DR-CAP). We present a technique for estimating big-M in such a formulation that takes advantage of the ambiguity set. We consider a 0-1 bilinear knapsack set to develop valid inequalities for CAP and DR-CAP. This is generalized to the joint chance constraint problem. A probability cut framework is also developed to solve DR-CAP. A computational study on problem instances obtained from using real hospital surgery data shows that the developed techniques allow us to solve certain model instances, and reduce the computational time for others. The use of Wasserstein ambiguity set in the DR-CAP model improves the out-of-sample performance of satisfying the chance constraints more significantly than the one possible by increasing the sample size in the sample average approximation technique. The solution time for DR-CAP model instances is of the same order as that for solving the CAP instances. This finding is important because chance constrained optimization models are very difficult to solve when the coefficients in the constraints are random.

Key words: chance-constrained assignment problem, distributionally robust optimization, bilinear program, branch-and-cut, valid inequalities, operating room planning

History: Submitted in April 28, 2020; Revised in December 30, 2020 and March 25, 2021; Accepted in April 9, 2021

1. Introduction

In the chance-constrained assignment problem, we assign the items with random weights to available bins and minimize the assignment cost while satisfying the bin capacity constraints with probability at least $1-\varepsilon$. In a motivating example surgeries with random durations are assigned to available operating rooms, and we want to ensure that the assigned surgeries complete within a specified duration with a high probability. More specifically, we study the chance-constrained assignment problem:

$$\text{(CAP)} \qquad \underset{\boldsymbol{y}\in\{0,1\}^{|\mathcal{I}||\mathcal{J}|}}{\text{minimize}} \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} c_{ij}y_{ij} \tag{1a}$$

$$\text{subject to } \sum_{j\in\mathcal{J}} y_{ij} = 1, \qquad \forall i\in\mathcal{I}, \tag{1b}$$

$$\sum_{i\in\mathcal{I}} y_{ij} \le \rho_j, \qquad \forall j\in\mathcal{J}, \tag{1c}$$

$$\mathbb{P}\left\{\sum_{i\in\mathcal{I}} \xi_i y_{ij} \le t_j\right\} \ge 1-\varepsilon, \qquad \forall j\in\mathcal{J}, \tag{1d}$$

where $\mathcal{I} := \{1,\ldots,|\mathcal{I}|\}$ is the set of items, $\mathcal{J} := \{1,\ldots,|\mathcal{J}|\}$ is the set of bins, $|\cdot|$ is the cardinality of a set, c_{ij} is the nonnegative cost for assigning item i to bin j, ρ_j is the quantitative restriction of bin j, and t_j is the capacity of bin j. ξ_i is the random weight of item i. The binary decision variable y_{ij} indicates if item i is assigned to bin j. Let $\boldsymbol{y}_j := (y_{1j},\ldots,y_{|\mathcal{I}|j})^\top$ for $j\in\mathcal{J}$, and $\boldsymbol{y} := (\boldsymbol{y}_1,\ldots,\boldsymbol{y}_{|\mathcal{J}|})^\top$. The objective (1a) minimizes the total cost of assigning the items to the bins. Constraints (1b) ensure that item i is assigned to only one bin. Constraints (1c) ensure that at most ρ_j items are assigned to bin j. Constraints (1d) ensure that the capacity for bin j is satisfied with probability $1-\varepsilon$, where $\varepsilon\in[0,1]$. The chance-constrained assignment problem has a wide range of applications such as in healthcare (Zhang et al. 2020), facility location (Peng et al. 2020), and cloud computing (Cohen et al. 2019), among others.

There are several challenges in solving the chance-constrained assignment problem. (CAP) is not a convex optimization problem since the chance constraints (1d) might not induce a convex feasible region, and the variables in (CAP) are binary. The chance-constrained programming (CCP) literature commonly assumes that the probability distributions of the random weights ξ_i are known and finitely supported. Incomplete knowledge of the probability distribution of ξ_i can be addressed by using an ambiguity set $\mathscr{P}$ that allows a family of distributions. The chance constraints (1d) are satisfied over all probability distributions within the ambiguity set $\mathscr{P}$, resulting in the formulation:

$$\text{(DR-CAP)} \qquad \underset{\boldsymbol{y}\in\{0,1\}^{|\mathcal{I}||\mathcal{J}|}}{\text{minimize}} \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} c_{ij}y_{ij} \tag{2a}$$

$$\text{subject to (1b), (1c),}$$

$$\inf_{\mathbb{P}\in\mathscr{P}} \mathbb{P}\left\{\sum_{i\in\mathcal{I}} \xi_i y_{ij} \le t_j\right\} \ge 1-\varepsilon, \qquad \forall j\in\mathcal{J}. \tag{2b}$$

In this paper we assume that the probability distribution $\mathbb{P}$ has finite support $\boldsymbol{\xi} := (\boldsymbol{\xi}^1,\cdots,\boldsymbol{\xi}^N)^\top$, where $\boldsymbol{\xi}^\omega := (\xi_1^\omega,\cdots,\xi_{|\mathcal{I}|}^\omega)^\top$ for $\omega\in\Omega := \{1,\cdots,N\}$. ξ_i^ω denotes the weight of item i for scenario

$\omega \in \Omega$, and p_ω is the probability of scenario $\omega \in \Omega$ such that $p_\omega \geq 0$ and $\sum_{\omega \in \Omega} p_\omega = 1$. We further assume that ξ_i^ω and t_j are non-negative integers, and without loss of generality, $p_\omega \leq \varepsilon$ and $\xi_i^\omega \leq t_j$, for $i \in \mathcal{I}, j \in \mathcal{J}, \omega \in \Omega$.

The model framework corresponds to an approach where a sample average approximation replaces the original distribution of a random vector with a finite number of samples (Luedtke and Ahmed 2008, Pagnoncelli et al. 2009). The SAA approach may provide a good candidate solution for the 'true' chance-constrained program (Shapiro et al. 2009, Calafiore and Campi 2006). This has motivated a number of studies for solving CCPs by formulating it as a mixed-integer program (see, e.g., Luedtke et al. (2010), Küçükyavuz (2012), Abdi and Fukasawa (2016), Liu et al. (2019), Zhao et al. (2017), Peng et al. (2020)).

1.1. Chance-Constrained Programs with Random Technology Matrices

The model in (1) has randomness in the coefficients of the constraints, i.e., it has a random technology matrix. CCPs with random technology matrices are significantly more difficult to solve than the case where only the right-hand-side vector is random (Tanner and Ntaimo (2010)). Tanner and Ntaimo (2010) used irreducible infeasible subsystems to derive a class of valid inequalities for such problems. Luedtke (2014) used a technique similar to the one used for generating valid inequalities in mixed integer programming for CCPs with random right-hand side to develop strong valid inequalities, and proposed a branch-and-cut decomposition algorithm for CCPs. Qiu et al. (2014) proposed an iterative scheme to improve the coefficient estimation in a big-M formulation, and observed that the coefficient strengthening technique can significantly decrease the solution time. van Ackooij et al. (2016) investigated a generalized Benders decomposition approach with stabilization and inexact function computation to solve CCP. Liu et al. (2016) studied two-stage CCPs and developed a Benders decomposition approach with strengthened optimality cuts to solve the problem. More recently, Xie and Ahmed (2018) projected the mixing inequalities onto the original space to derive a family of quantile cuts for such problems.

1.2. Integer Chance-Constrained Programs

For the integer programming problem with chance constraints, Beraldi and Bruni (2010) formulated the problem as an integer program with knapsack constraints, and used the feasible solutions of the knapsack constraints to divide the feasible region of the problem within a branch and bound scheme. Song and Luedtke (2013) studied a chance-constrained reliable network design problem. They derived valid inequalities for this problem. Song et al. (2014) considered a chance-constrained packing problem. This problem is to select a subset of items that maximizes the total profit while satisfying a single chance constraint. The problem is viewed as a probabilistic cover problem, and the probabilistic cover inequalities are developed by using a lifting technique from Zemel (1989).

Deng and Shen (2016) investigated a chance-constrained appointment scheduling problem and used a decomposition algorithm with formulation strengthening strategies to solve this problem. Wu and Küçükyavuz (2017) studied a chance-constrained combinatorial optimization problem and presented an exact method for solving the problem under the assumption that the chance probability can be calculated.

1.3. Distributionally Robust Optimization

In the distributionally robust optimization (DRO) framework, the probability distribution of the random variables lies in an ambiguity set. Two widely used ambiguity sets are the moment-based ambiguity sets (see, e.g., Delage and Ye (2010), Wiesemann et al. (2014), Mehrotra and Papp (2014), and Bansal et al. (2018)) and the statistical distance-based sets (see, e.g., Ben-Tal et al. (2013), Jiang and Guan (2018), Esfahani and Kuhn (2018), and Luo and Mehrotra (2019)). For the distributionally robust chance-constrained programs, Chen et al. (2010) and Zymler et al. (2013) developed tractable approximations of ambiguous chance constraints under the moment-based ambiguity sets. Hanasusanto et al. (2017) studied the ambiguous joint chance constraints where the ambiguity set is characterized by the mean and an upper bound on the dispersion, and presented a convex reformulation under some conditions. Jiang and Guan (2016) studied a data-driven distributionally robust chance-constrained model using a ϕ-divergence measure-based set. They showed that this problem is equivalent to a classical chance-constrained problem with a perturbed risk level. As an important type of statistical distance, the Wasserstein metric is used to define an ambiguity set. In the finite support case the Wasserstein metric provides a polyhedral structure. Thus, several studies have investigated the use of distributionally robust chance-constrained problems with the Wasserstein ambiguity set (see, e.g., Xie (2019), Chen et al. (2018), Ji and Lejeune (2020)). Xie (2019) showed that the distributionally robust chance-constrained program (DRCCP) defined using the Wasserstein ambiguity set admits a conditional value-at-risk (CVaR) interpretation and it is mixed integer representable. The author also proposed inner and outer approximations based upon a CVaR reformulation. For DRCCP with pure binary decision variables, a big-M free mixed-integer linear reformulation is proposed by exploring the submodular structure of the problem. Chen et al. (2018) studied DRCCP with Wasserstein metric for the continuous support case. The authors proposed a mixed-integer conic reformulation of problems with individual and joint chance constraints with right-hand side uncertainty. For the Wasserstein ambiguity set defined using the 1-norm or the ∞-norm, they showed that DRCCP can be reformulated as a mixed-integer linear program. More recently, Ji and Lejeune (2020) studied DRCCP with Wasserstein ambiguity sets under finite support and continuum of realizations. For the case with finite support, they used duality to obtain a mixed-integer linear programming (MILP) reformulation with big-M coefficients, where the big-M coefficients are obtained from the bounds on the decision variables.

In this paper we study (DR-CAP) with a general ambiguity set under finite support, thus also allowing the framework to be applied to definitions of ambiguity sets different from the Wasserstein ambiguity set. For example, the probability cut algorithm developed in this paper is applicable for problems where the ambiguity sets are defined using ϕ-divergence. We used the Wasserstein ambiguity set as a specific example in our computations. The papers of Xie (2019), Chen et al. (2018), and Ji and Lejeune (2020) focus on Wasserstein ambiguity set only. Moreover, Wang et al. (2021) showed that the convex approximation of chance-constrained problems based on CVaR typically leaves a gap to satisfy the chance constraint and usually does not provide any computational benefit, suggesting that a CVaR approximation may not produce an optimal solution to the problem when compared with the exact reformulations. For the case with Wasserstein ambiguity set under finite support, the coefficient strengthening approach for the big-M coefficients proposed in this paper results in smaller values than the technique proposed by Ji and Lejeune (2020). Additionally, in our computational experiments we find that the MILP reformulation of (DR-CAP) with Wasserstein ambiguity set could be time-consuming, while the branch-and-cut algorithm with probability cuts proposed in this paper can solve our test problems to optimality more efficiently.

For the distributionally robust chance-constrained binary programs, Cheng et al. (2014) considered the distributionally robust chance-constrained quadratic knapsack problem and assumed that the first and second moments, and the joint support of random variables are known. They provided a semidefinite programming (SDP) relaxation for the binary constraints. Zhang et al. (2020) assumed that only the mean and the variance are known, and investigated the two-stage distributionally robust chance-constrained bin-packing problem with continuous bin extension decisions. They developed a branch-and-price approach based on a column generation reformulation to solve the mixed-integer reformulation. Wang et al. (2017) studied a distributionally robust chance-constrained surgery planning problem with uncertain service time and downstream resource requirements, and derived a second-order cone program (SOCP) reformulation under the mean-covariance ambiguity set. Deng et al. (2019) studied chance-constrained surgery planning by using a ϕ-divergence measure-based ambiguity set, and used a branch-and-cut algorithm to solve the mixed-integer linear reformulation of this problem. Zhang et al. (2018) considered the distributionally robust chance-constrained bin-packing problem in which only the mean and the covariance matrix are known. They reformulated the problem as a binary SOCP, and developed valid inequalities for the SOCP by using the submodularity and the bin-packing structure of the model. Finally, we refer interested readers to a recent survey by Rahimian and Mehrotra (2019) for more details about DRO.

1.4. Contributions of This Paper

Wang et al. (2021) studied a single chance-constrained bin packing problem. A binary bilinear reformulation of the problem was used to motivate the development of valid inequalities for this problem. Specifically, it was shown that the cover, clique, and projection inequalities can be adapted to generate inequalities for a bilinear knapsack set. It was also shown in this earlier work that the valid inequalities result in computational efficiencies within a branch-and-cut framework to solve the chance constraint bin packing problem. This paper makes several significant advancements to the work of Wang et al. (2021). Its emphasis is on studying chance constraints in the context of distributional robustness. Therefore, all results in this paper provide a contribution to the literature in that context. Specifically, it makes the following contributions:

- It shows that the big-M strengthening calculations in (DR-CAP) can directly take advantage of the ambiguity set in its computations. This result is applicable for problems more general than (DR-CAP).
- A new family of inequalities that are valid for (CAP) is obtained. The inequalities are shown to be facet defining under a condition, which seems to hold frequently in practice. About 50% of the cuts identified in the proposed approach satisfy this condition. It is also shown that the generation of these inequalities can take advantage of the distributional robustness as part of the inequality generation algorithm.
- A new family of valid inequalities is obtained for the set defined by the intersection of multiple binary bilinear knapsacks with a general 0-1 knapsack constraint and a cardinality constraint. Appropriate heuristics are developed to find these inequalities.
- The valid inequalities and solution schemes proposed in this paper are further developed for the joint chance constraint (CAP) and (DR-CAP).
- This paper also proposes a branch-and-cut algorithm with probability cuts, which uses a distribution separation procedure, the valid inequalities developed in this paper, and the feasibility/probability cuts, to solve the strengthened big-M semi-infinite reformulation of (DR-CAP). A convergence proof of this algorithm is provided.
- A computational study for an assignment problem based on real data from a hospital shows the benefits of the techniques developed in this paper. (CAP) instances with up to 1,500 scenarios are solved within ten hours when $\varepsilon = 0.08$, 0.1, 0.12. A smaller optimality gap is observed for instances with $\varepsilon = 0.06$. The lifted cover inequalities proposed in this paper outperform the single cover inequalities in (Wang et al. 2021). For (DR-CAP) using the Wasserstein metric, all instances with $N = 1,500$ are solved within two hours for $\varepsilon = 0.1$. An out-of-sample estimation of the chance constraint satisfaction for the solutions obtained from (CAP) and (DR-CAP) shows that the (DR-CAP) solutions achieve the desirable probability target more reliably, though we find that both

(CAP) and (DR-CAP) models may violate the chance constraint out-of-sample when the sample size and the radius of the Wasserstein set are small. The out-of-sample performance of the solution improves significantly when a moderate size sample and a Wasserstein ambiguity set is used. Additionally, (DR-CAP) instances are solved in about four times the time required to solve (CAP). This empirical finding of improved out-of-sample performance with a moderate increase in computational time on chance constraint problems is novel. It suggests that even with a moderate number of samples, the use of a distributionally robust framework allows one to significantly improve the out-of-sample performance of the obtained solution. In order to achieve a similar out-of-sample performance in the finite sample approach, the sample size will have to increase significantly, which makes the problems intractable.

1.5. Organization

The remainder of this paper is organized as follows. Section 2 formulates (CAP) as a binary integer program using the big-M technique. Subsequently, in this section, we formulate (DR-CAP) as a semi-infinite program and present a big-M coefficient strengthening procedure for this formulation. We then present alternative bilinear formulations for (CAP) and (DR-CAP), respectively. We exploit the structure of the bilinear formulations to develop two classes of valid inequalities in Section 3. Specifically, in Section 3.1 we utilize the sequential lifting technique to develop the lifted cover inequalities for the binary bilinear knapsack set and show that these inequalities are facet-defining under certain conditions. We then present stronger lifted cover inequalities for (CAP) and (DR-CAP) by restricting the feasible region of $\boldsymbol{y}$. We further analyze the multiple binary bilinear knapsack sets with a general 0-1 knapsack constraint and develop a class of valid inequalities in Section 3.2. In Section 4, we describe a branch-and-cut solution scheme for (CAP) and propose separation heuristics to obtain the violated valid inequalities. A branch-and-cut algorithm with probability cuts for solving (DR-CAP), and its convergence proof is provided in this section. In Section 5, we develop valid inequalities and solution schemes for the joint chance constraint (CAP) and (DRCAP) problems. Section 6 reports computational results on (CAP) and (DR-CAP) formulations using surgery duration for different types of surgeries in an operating room. Section 7 concludes the paper with a summary of the important findings. Appendix A provides coefficient calculations for the lifted cover inequalities that are valid for (CAP) and (DR-CAP). Appendix B presents proofs of propositions and theorems stated in the paper. Appendix C describes the pseudo-code of the algorithms implemented to perform our computations. Appendix D gives a dynamic programming based approach to compute the big-M coefficients. Appendix E presents the statistics of surgery duration for the real-life data. Appendix F provides additional computational results.

2. Model Reformulation

We formulate (CAP) as a binary linear program in Section 2.1. A semi-infinite reformulation for (DR-CAP) is presented in Section 2.2. We then present binary bilinear reformulations for (CAP) and (DR-CAP) in Section 2.3.

2.1. Binary Integer Reformulation for (CAP)

Let the binary variable $z_{j\omega}$ indicate if the capacity constraint is violated for $j \in \mathcal{J}$ and $\omega \in \Omega$. Namely, $z_{j\omega} = 1$ if the constraint $\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \le t_j$ is satisfied, and $z_{j\omega} = 0$, otherwise. For $j \in \mathcal{J}$, let $\boldsymbol{z}_j := (z_{j1}, \ldots, z_{jN})^\top$ and $\boldsymbol{z} := (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_{|\mathcal{J}|})^\top$. Constraints (1d) can be formulated as

$$\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} + (M_j^\omega - t_j) z_{j\omega} \le M_j^\omega, \qquad \forall j \in \mathcal{J}, \omega \in \Omega, \tag{3a}$$

$$\sum_{\omega\in\Omega} p_\omega z_{j\omega} \ge 1 - \varepsilon, \qquad \forall j \in \mathcal{J}, \tag{3b}$$

where M_j^ω is a constant which ensures that (3a) hold when $z_{j\omega} = 0$. Computation of a small valid value of M_j^ω gives a tighter formulation in (3). We first present a coefficient strengthening procedure, inspired from Song et al. (2014), to obtain a value of M_j^ω. For $j \in \mathcal{J}$ and $\omega \in \Omega$ let:

$$\bar{M}_j^\omega := \underset{\boldsymbol{y}_j \in \{0,1\}^{|\mathcal{I}|}}{\text{maximize}} \left\{ \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \middle| \mathbb{P}\left\{ \sum_{i\in\mathcal{I}} \xi_i y_{ij} \le t_j \right\} \ge 1 - \varepsilon,\ \boldsymbol{y}_j \in \mathcal{Y}_j \right\}, \tag{4}$$

where $\mathcal{Y}_j := \{\boldsymbol{y}_j | \sum_{i\in\mathcal{I}} y_{ij} \le \rho_j\}$. Note that $M_j^\omega \ge \bar{M}_j^\omega$. For $j \in \mathcal{J}$ and $\omega, k \in \Omega$, let

$$m_j^\omega(k) := \underset{\boldsymbol{y}_j \in \{0,1\}^{|\mathcal{I}|}}{\text{maximize}} \left\{ \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \middle| \sum_{i\in\mathcal{I}} \xi_i^k y_{ij} \le t_j,\ \boldsymbol{y}_j \in \mathcal{Y}_j \right\}. \tag{5}$$

We sort $m_j^\omega(k)$ such that $m_j^\omega(k_1) \le \ldots \le m_j^\omega(k_N)$. An upper bound for $\bar{M}_j^\omega$ is given in Proposition 1. A proof of this proposition is given in Appendix B.1.

Proposition 1. *$m_j^\omega(k_q)$ is an upper bound for $\bar{M}_j^\omega$, where $q := \min\left\{ l \middle| \sum_{j=1}^{l} p_{k_j} > \varepsilon \right\}$, and (CAP) can be equivalently reformulated as the binary integer program:*

$$\text{(IP)} \quad \underset{(\boldsymbol{y},\boldsymbol{z}) \in \{0,1\}^{|\mathcal{I}||\mathcal{J}|} \times \{0,1\}^{|\mathcal{J}|N}}{\text{minimize}} \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} c_{ij} y_{ij} \tag{6a}$$

$$\text{subject to (1b), (1c), (3b),}$$

$$\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} + \left(m_j^\omega(k_q) - m_j^\omega(\omega) \right) z_{j\omega} \le m_j^\omega(k_q), \quad \forall j \in \mathcal{J}, \omega \in \Omega. \ \square \tag{6b}$$

Remark 1. Note that (5) has a knapsack constraint and a cardinality constraint. We use a dynamic programming method for solving (5) in Section 6. The procedure uses the methodology in Bertsimas and Demir (2002). For $j \in \mathcal{J}$, if t_j and ρ_j are moderate, dynamic programming is an efficient approach for solving (5) to optimality (see more details about the dynamic programming in Appendix D).

Remark 2. Constraints (1b) and (1c) represent the assignment structure of model (1). In the corresponding statement of Proposition 1, they can be replaced with a general constraint set $\mathcal{Y}_j$.

2.2. Semi-Infinite Programming Reformulation for (DR-CAP)

In this section we study the chance-constrained models, where the distribution of random weights belongs to an ambiguity set. The results in this section are stated for any ambiguity set defined on a finite support (see Bansal et al. (2018)). However, in the computational results of this paper, we used the l_1-Wasserstein ambiguity set:

$$\mathscr{P}_W = \{\boldsymbol{p} \in \mathbb{R}^N_+ \mid \sum_{\omega\in\Omega} p_\omega = 1,\ \sum_{\omega\in\Omega}\sum_{k\in\Omega} \|\boldsymbol{\xi}^\omega - \boldsymbol{\xi}^k\| \nu_{\omega k} \le \eta,\ \sum_{k\in\Omega} \nu_{\omega k} = p_\omega,\ \forall \omega\in\Omega, \quad (7)$$
$$\sum_{\omega\in\Omega} \nu_{\omega k} = p^*_k,\ \forall k \in\Omega,\ \ \nu_{\omega k}\ge 0,\ \forall \omega, k\in\Omega\},$$

where $\eta \ge 0$ is the Wasserstein radius and $\{p^*_k\}_{k\in\Omega}$ is an empirical probability distribution of $\boldsymbol{\xi}$. Note that if $\eta = 0$, then $p_\omega = p^*_\omega$ for all $\omega\in\Omega$ and (DR-CAP) reduces to (CAP). Let $\mathbb{1}(\cdot)$ denote an indicator function. Using this notation constraint (2b) using the Wasserstein ambiguity set is given as follows:

$$\inf \left\{ \sum_{\omega\in\Omega} p_\omega \, \mathbb{1}\left(\sum_{i\in\mathcal{I}} \xi^\omega_i y_{ij} \le t_j \right) \middle| \, \boldsymbol{p}\in\mathscr{P}_W \right\} \ge 1-\varepsilon, \qquad \forall j \in \mathcal{J}.$$

Let $z_{j\omega}$ and $m^\omega_j(\cdot)$ be defined as in Section 2.1. The following theorem gives a reformulation of (DR-CAP) with a general ambiguity set $\mathscr{P}$. A proof is given in Appendix B.2. Note that this formulation is a semi-infinite program because of constraints (8b).

Theorem 1. *We sort $m^\omega_j(\cdot)$ in a non-decreasing order such that $m^\omega_j(k_1) \le \cdots \le m^\omega_j(k_N)$. Then, (DR-CAP) can be represented as the semi-infinite program:*

$$\text{(SIP)} \quad \underset{(\boldsymbol{y},\boldsymbol{z})\in\{0,1\}^{|\mathcal{I}||\mathcal{J}|}\times\{0,1\}^{|\mathcal{J}|N}}{\text{minimize}} \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} c_{ij} y_{ij} \quad (8a)$$
$$\text{subject to } (1b), (1c),$$
$$\inf_{\boldsymbol{p}\in\mathscr{P}} \sum_{\omega\in\Omega} p_\omega z_{j\omega} \ge 1-\varepsilon, \qquad \forall j\in\mathcal{J}, \quad (8b)$$
$$\sum_{i\in\mathcal{I}} \xi^\omega_i y_{ij} + \left(m^\omega_j(k_{\bar q}) - m^\omega_j(\omega)\right) z_{j\omega} \le m^\omega_j(k_{\bar q}), \quad \forall j\in\mathcal{J}, \omega\in\Omega, \quad (8c)$$

where $\bar q := \min\{l \mid \sup_{\boldsymbol{p}\in\mathscr{P}} \sum_{j=1}^{l} p_{k_j} > \varepsilon\}$. □

In the special case of Wasserstein ambiguity set (7), because of its polyhedral structure, constraints (8b) are equivalently representable by its extreme points, thus formulation (8) is finite. Note also that when $\mathscr{P} := \mathscr{P}_W$, $\bar{q}$ in Theorem 1 is obtained by solving a sequence of linear programs. Moreover, the left-hand side of (8b) is a linear program for a fixed $z_{j\omega}$. The use of an optimization problem in identifying $\bar{q}$ may provide a smaller value of $m_j^\omega(\cdot)$ used in the big-M formulation. Since solving linear programs for computing the index $\bar{q}$ for all $j \in \mathcal{J}$ and $\omega \in \Omega$ can be time-consuming, the following corollary shows that the use of any distribution in the set $\mathscr{P}$ is sufficient for the big-M estimation. Such distributions are available as the probability cut algorithm given in Section 4.3 progresses.

Corollary 1. *Let $\{\hat{p}_\omega\}_{\omega\in\Omega} \in \mathscr{P}$, and $\hat{q} = \min\{l \mid \sum_{j=1}^{l} \hat{p}_{k_j} > \varepsilon\}$. Then, $\bar{q} \le \hat{q}$ and $m_j^\omega(k_{\bar{q}}) \le m_j^\omega(k_{\hat{q}})$.*

Proof. Since $\sup_{\boldsymbol{p}\in\mathscr{P}} \sum_{j=1}^{\hat{q}} p_{k_j} \ge \sum_{j=1}^{\hat{q}} \hat{p}_{k_j} > \varepsilon$, we have $\hat{q} \ge \bar{q}$ and $m_j^\omega(k_{\bar{q}}) \le m_j^\omega(k_{\hat{q}})$. □

Remark 3. Theorem 1 and Corollary 1 remain valid for the case where the cardinality constraint in (4) is replaced by a more general constraint set $\mathcal{Y}_j$ for $j \in \mathcal{J}$.

2.3. Binary Bilinear Reformulations

In the previous section, we formulate the chance constraints as binary linear constraints using big-M coefficients. In this section, we present an alternative approach following Wang et al. (2021). Let $z_{j\omega}$ be defined as in Section 2.1. The constraints (6b) and (8c) can also be rewritten as

$$\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} z_{j\omega} \le m_j^\omega(\omega) z_{j\omega}, \qquad \forall j \in \mathcal{J}, \omega \in \Omega. \tag{9}$$

Thus, we can use (9) to obtain a binary bilinear reformulation and bilinear semi-infinite reformulation for (CAP) and (DR-CAP), respectively. In principle, a problem with binary bilinear constraints can be reformulated as a binary linear problem using the reformulation linearization technique (RLT). Let $\bar{m}_{j\omega} := \underset{\boldsymbol{y}_j\in[0,1]^{|\mathcal{I}|}}{\text{maximize}} \left\{ \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \middle| \mathbb{P}\left\{ \sum_{i\in\mathcal{I}} \xi_i y_{ij} \le t_j \right\} \ge 1-\varepsilon,\, \boldsymbol{y}_j \in \mathcal{Y}_j \right\}$, and $\bar{m}'_{j\omega} := \underset{\boldsymbol{y}_j\in[0,1]^{|\mathcal{I}|}}{\text{maximize}} \left\{ \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \middle| \inf_{\mathbb{P}\in\mathscr{P}} \mathbb{P}\left\{ \sum_{i\in\mathcal{I}} \xi_i y_{ij} \le t_j \right\} \ge 1-\varepsilon,\, \boldsymbol{y}_j \in \mathcal{Y}_j \right\}$. The following proposition shows a relationship between the bilinear reformulations with the formulations (8) and (6), respectively. A proof is given in Appendix B.3.

Proposition 2. *The relaxation of the binary bilinear reformulation for (CAP) obtained from relaxing the binary variables is stronger than the linear relaxation of (6) if $m_j^\omega(k_q) \ge \bar{m}_{j\omega}$. Similarly, the relaxation of the binary bilinear reformulation for (DR-CAP) obtained from relaxing the binary variables is stronger than the linear relaxation of (8) if $m_j^\omega(k_{\bar{q}}) \ge \bar{m}'_{j\omega}$.* □

Although $m_j^\omega(k_{\bar{q}}) \geq \bar{m}'_{j\omega}$ may not be true in general, in our test problems this condition is met for all instances with $N = 500$ or 1000 with $\varepsilon \geq 0.1$. This was also the case for (CAP).

We can also rewrite the bilinear constraints (9) using McCormick inequalities as follows:

$$\begin{aligned}
&\sum_{i\in\mathcal{I}} \xi_i^\omega u_{ij}^\omega \leq m_j^\omega(\omega) z_j^\omega, && \forall j \in \mathcal{J}, \omega \in \Omega,\\
&u_{ij}^\omega \leq y_{ij}, u_{ij}^\omega \leq z_j^\omega, && \forall i \in \mathcal{I}, j \in \mathcal{J}, \omega \in \Omega,\\
&y_{ij} + z_j^\omega - u_{ij}^\omega \leq 1, u_{ij}^\omega \geq 0, && \forall i \in \mathcal{I}, j \in \mathcal{J}, \omega \in \Omega.
\end{aligned}$$

We compared the relaxation of the above McCormick relaxation with that from (IP) (see Section 6 for a description of test instances). We found that the average solution time for the problem based on the McCormick reformulation is 23 seconds for $N = 500$ and 61 seconds for $N = 1000$, whereas the average time for solving the relaxation of (IP) is 1 second for $N = 500$ and 3 seconds for $N = 1000$. This is mainly because the use of McCormick inequalities significantly increase the problem size. Hence, in the computational experiments, we solve (IP) and generate valid inequalities based on the bilinear reformulation for (CAP). However, we observed that the instances resulting from using the McCormick relaxation have a higher objective value (60.4 versus 59.5 for $N = 500$ and 60.4 versus 59.6 for $N = 1000$). Similar small improvements in the lower bound were observed for other ε and N, suggesting that additional information is available in the inequalities present in McCormick relaxation.

Note that constraints (1c), (3b) and (9) give a key substructure of (CAP). Let $\mathcal{H} := \left\{(\boldsymbol{y}, \boldsymbol{z}) \in \{0,1\}^{|\mathcal{I}||\mathcal{J}|} \times \{0,1\}^{|\mathcal{J}|N} \,|\, (1c), (3b), (9)\right\}$. For $j \in \mathcal{J}$, let

$$\mathcal{G}_j := \left\{ (\boldsymbol{y}_j, \boldsymbol{z}_j) \in \{0,1\}^{|\mathcal{I}|} \times \{0,1\}^N \,\middle|\, \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} z_{j\omega} \leq m_j^\omega(\omega) z_{j\omega}, \forall \omega \in \Omega, \sum_{\omega\in\Omega} p_\omega z_{j\omega} \geq 1 - \varepsilon, \boldsymbol{y}_j \in \mathcal{Y}_j \right\}.$$

The set $\mathcal{G}_j$ is the intersection of multiple binary bilinear knapsacks with a general knapsack constraint, and a cardinality constraint. We have $\mathcal{H} = \bigcap_{j\in\mathcal{J}} \{(\boldsymbol{y}, \boldsymbol{z}) | (\boldsymbol{y}_j, \boldsymbol{z}_j) \in \mathcal{G}_j\}$.

Let us use $conv(\cdot)$ to denote the convex hull of a set. The following proposition shows that in order to identify strong valid inequalities for $conv(\mathcal{H})$, we can develop strong valid inequalities for $conv(\mathcal{G}_j)$. A proof can be found in Appendix B.4.

PROPOSITION 3. *If an inequality is valid for $conv(\mathcal{G}_j)$, this inequality is also valid for $conv(\mathcal{H})$. Moreover, if an inequality is facet-defining for $conv(\mathcal{G}_j)$, it is also facet-defining for $conv(\mathcal{H})$.* □

Proposition 3 gives a motivation to investigate the set $\mathcal{G}_j$. Hence, in the following, we develop a class of valid inequalities for $\mathcal{G}_j$. For $j \in \mathcal{J}$, let

$$\mathcal{G}'_j := \{(\boldsymbol{y}_j, \boldsymbol{z}_j) \in \{0,1\}^{|\mathcal{I}|} \times \{0,1\}^N \,\Big|\, \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} z_{j\omega} \leq m_j^\omega(\omega) z_{j\omega},\ \forall \omega \in \Omega, \inf_{p\in\mathscr{P}} \sum_{\omega\in\Omega} p_\omega z_{j\omega} \geq 1 - \varepsilon, \boldsymbol{y}_j \in \mathcal{Y}_j\}.$$

We also obtain valid inequalities using $\mathcal{G}'_j$.

3. Valid Inequalities for (CAP) and (DR-CAP)

We first apply the lifting technique for the knapsack problem to a binary bilinear knapsack set and develop a family of valid inequalities in Section 3.1. Section 3.2 further presents a family of valid inequalities for $\mathcal{G}_j$ and $\mathcal{G}'_j$.

3.1. Lifted Cover Inequalities

We assume that $j \in \mathcal{J}$, $\omega \in \Omega$ are fixed in this section. Let us consider the binary bilinear knapsack set $\mathcal{F}_{j\omega} := \left\{ (\boldsymbol{y}_j, \boldsymbol{z}_{j\omega}) \in \{0,1\}^{|\mathcal{I}|} \times \{0,1\} \middle| \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} z_{j\omega} \le m_j^\omega(\omega) z_{j\omega}, \boldsymbol{y}_j \in \mathcal{Y}_j \right\}$. Note that the inequalities valid for $conv(\mathcal{F}_{j\omega})$ are also valid for (CAP) and (DR-CAP). Note also that when compared to the development in Wang et al. (2021), we include the cardinality constraint in addition to the binary bilinear knapsack constraint in the description of $\mathcal{F}_{i\omega}$. When $z_{j\omega} = 1$, the set $\mathcal{F}_{j\omega}$ becomes the two-constraint 0-1 knapsack set $\mathcal{Q}_{j\omega} := \left\{ \boldsymbol{y}_j \in \{0,1\}^{|\mathcal{I}|} \middle| \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \le m_j^\omega(\omega), \boldsymbol{y}_j \in \mathcal{Y}_j \right\}$.

We now extend the results for the single binary knapsack set from Zemel (1989) and Gu et al. (1998) to develop valid inequalities that are facet-defining for the set $\mathcal{Q}_{j\omega}$ under a condition that is often satisfied in our computations. A lifted cover inequality that is valid for $conv(\mathcal{F}_{j\omega})$ obtained by rotating the valid inequalities provided below is given in Appendix A. Note that in obtaining the lifted cover inequality the restriction $\boldsymbol{y}_j \in \mathcal{Y}_j$ and the chance constraint are used to obtain a stronger inequality for (CAP) and (DR-CAP).

Definition 1. Set $\mathcal{C} \subseteq \mathcal{I}$ is a cover for $\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \le m_j^\omega(\omega)$ if $\sum_{i\in\mathcal{C}} \xi_i^\omega > m_j^\omega(\omega)$. The cover $\mathcal{C}$ is minimal if no subset of $\mathcal{C}$ is a cover for $\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \le m_j^\omega(\omega)$. □

In this section, we assume that $\mathcal{C}$ is a minimal cover for $\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \le m_j^\omega(\omega)$. Let $\mathcal{D} \subseteq \mathcal{C}$, and consider

$$conv\left(\left\{ \boldsymbol{y}_j \in \{0,1\}^{|\mathcal{I}|} \middle| \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \le m_j^\omega(\omega), \boldsymbol{y}_j \in \mathcal{Y}_j, y_{ij} = 0, \forall i \in \mathcal{I}\backslash\mathcal{C}, y_{ij} = 1, \forall i \in \mathcal{D} \right\}\right). \tag{10}$$

Note that polyhedron (10) is a restriction of $\mathcal{Q}_{j\omega}$. Proposition 4 provides a seed inequality that is valid for (10). The following proposition gives a valid inequality that is facet-defining under suitable cardinality conditions for the following convex hull. This inequality is lifted in Sections 3.1.1 and 3.1.2. A proof is given in Appendix B.5.

Proposition 4. *The inequality*

$$\sum_{i\in\mathcal{C}\backslash\mathcal{D}} y_{ij} \le |\mathcal{C}\backslash\mathcal{D}| - 1 \tag{11}$$

is valid for (10). *If* $|\mathcal{C}| \le \rho_j + 1$, *the inequality* (11) *is facet-defining for* (10). □

3.1.1. Up-Lifting In general, a cover inequality (11) does not induce a facet of a knapsack set. To obtain a facet-defining inequality of a knapsack set, we compute coefficients of variables in $\mathcal{I}\backslash\mathcal{C}$. This procedure is called up-lifting. By using the up-lifting technique, we obtain an inequality of the form

$$\sum_{i\in\mathcal{C}\backslash\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\backslash\mathcal{C}} \alpha_i y_{ij} \leq |\mathcal{C}\backslash\mathcal{D}| - 1, \tag{12}$$

where α_i is called an up-lifting coefficient. We now provide such an uplifting approach for our problem. Let $\{\pi_k\}_{k=1}^{|\mathcal{I}\backslash\mathcal{C}|}$ be a sequence of the set $\mathcal{I}\backslash\mathcal{C}$ and $\boldsymbol{\pi}(k)=\{\pi_1,\ldots,\pi_k\}$. For $k=1,\ldots,|\mathcal{I}\backslash\mathcal{C}|$, let

$$obj_{\pi_k} := \underset{\boldsymbol{y}_j\in\{0,1\}^{|(\mathcal{C}\backslash\mathcal{D})\cup\boldsymbol{\pi}(k-1)|}}{\text{maximize}} \sum_{i\in\mathcal{C}\backslash\mathcal{D}} y_{ij} + \sum_{i\in\boldsymbol{\pi}(k-1)} \alpha_i y_{ij} \tag{13a}$$

$$\text{subject to} \sum_{i\in\mathcal{C}\backslash\mathcal{D}} \xi_i^\omega y_{ij} + \sum_{i\in\boldsymbol{\pi}(k-1)} \xi_i^\omega y_{ij} \leq m_j^\omega(\omega) - \xi_{\pi_k}^\omega - \sum_{i\in\mathcal{D}} \xi_i^\omega, \tag{13b}$$

$$\sum_{i\in\mathcal{C}\backslash\mathcal{D}} y_{ij} + \sum_{i\in\boldsymbol{\pi}(k-1)} y_{ij} \leq \rho_j - 1 - |\mathcal{D}|. \tag{13c}$$

Note that different sequences of $\mathcal{I}\backslash\mathcal{C}$ might lead to different valid inequalities (Kaparis and Letchford 2008). The following lemma gives a sufficient condition under which inequality (12) is facet-defining for the convex hull of $\mathcal{Q}_{j\omega}$ when $y_{ij}=1$, $i\in\mathcal{D}$. A proof is given in Appendix B.6.

Lemma 1. *For $k=1,\ldots,|\mathcal{I}\backslash\mathcal{C}|$, let $\alpha_{\pi_k} = |\mathcal{C}\backslash\mathcal{D}| - 1 - obj_{\pi_k}$, where obj_{π_k} is defined in (13). Inequality (12) is valid for*

$$conv\left(\left\{\boldsymbol{y}_j \in \{0,1\}^{|\mathcal{I}|} \middle| \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \leq m_j^\omega(\omega), \boldsymbol{y}_j\in\mathcal{Y}_j, y_{ij}=1, \forall i\in\mathcal{D}\right\}\right). \tag{14}$$

If $|\mathcal{C}|\leq \rho_j+1$, inequality (12) is facet-defining for (14). □

We compute the lifting coefficient α_i using a dynamic programming based approach (see Zemel (1989)). This algorithm is given in Appendix C.1.

Remark 4. Note that the sufficient condition in Lemma 1 for ensuring that an inequality is facet defining requires us to start with covers with cardinality less than ρ_j+1. The inequality remains valid when this condition is not satisfied. However, it suggests a preference for identifying low cardinality covers.

3.1.2. Down-Lifting Similar to the up-lifing, down-lifting computes the coefficients for the variables y_{ij} in $\mathcal{D}$. We use this technique to obtain a valid inequality for $conv(\mathcal{Q}_{j\omega})$ of the form

$$\sum_{i\in\mathcal{C}\backslash\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\backslash\mathcal{C}} \alpha_i y_{ij} + \sum_{i\in\mathcal{D}} \beta_i y_{ij} \leq |\mathcal{C}\backslash\mathcal{D}| + \sum_{i\in\mathcal{D}} \beta_i - 1, \tag{15}$$

where for $i \in \mathcal{D}$, β_i is called a down-lifting coefficient. The coefficient β_i can be obtained by solving the following sequence of problems. Let $\{\kappa_l\}_{l=1}^{|\mathcal{D}|}$ be a sequence of the set $\mathcal{D}$ and $\boldsymbol{\kappa}(l) = \{\kappa_1, ..., \kappa_l\}$. For $l = 1, \cdots, |\mathcal{D}|$, let

$$\text{obj}_{\kappa_l} := \underset{\boldsymbol{y}_j \in \{0,1\}^{|(\mathcal{I}\setminus\mathcal{D})\cup\boldsymbol{\kappa}(l-1)|}}{\text{maximize}} \sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\setminus\mathcal{C}} \alpha_i y_{ij} + \sum_{i\in\boldsymbol{\kappa}(l-1)} \beta_i y_{ij} \tag{16a}$$

$$\text{subject to} \sum_{i\in\mathcal{I}\setminus\mathcal{D}} \xi_i^\omega y_{ij} + \sum_{i\in\boldsymbol{\kappa}(l-1)} \xi_i^\omega y_{ij} \le m_j^\omega(\omega) - \sum_{i=\kappa_{l+1}}^{\kappa_{|\mathcal{D}|}} \xi_i^\omega, \tag{16b}$$

$$\sum_{i\in\mathcal{I}\setminus\mathcal{D}} y_{ij} + \sum_{i\in\boldsymbol{\kappa}(l-1)} y_{ij} \le \rho_j - |\mathcal{D}| + l. \tag{16c}$$

Lemma 2. *For $l = 1, \ldots, |\mathcal{D}|$, let $\beta_{\kappa_l} = obj_{\kappa_l} - \sum_{i\in\boldsymbol{\kappa}(l-1)} \beta_i - |\mathcal{C}\setminus\mathcal{D}| + 1$, where obj_{κ_l} is defined in (16). The inequality (15) is valid for $conv(\mathcal{Q}_{j\omega})$. If $|\mathcal{C}| \le \rho_j + 1$, (15) is facet-defining for $conv(\mathcal{Q}_{j\omega})$.*

Proof See Appendix B.7. □

3.1.3. Examples of the Lifted Cover Inequalities We now provide an example to illustrate the lifted cover inequalities described in the previous sections and the advantage of using the cardinality constraint (i.e., solving a two-constrained dynamic program). In the second example, we use the family of valid inequalities referred to as single lifted cover inequality (obtained by ignoring the cardinality constraint in $\mathcal{F}_{j\omega}$) and show that it gets strengthened in the DR framework.

Example 1. Suppose $\mathcal{F}_{j\omega}$ is defined by $\rho_j = 3$, $m_j^\omega(\omega) = 40$, and $\boldsymbol{\xi}_\omega = (7, 8, 11, 10, 9, 14, 23)^\top$. Let $\hat{y} = \{0.6, 0.4, 0.4, 0.3, 1.0, 0.2, 0.1\}$. Then the set $\mathcal{C} = \{1, 2, 3, 4, 5\}$ is a minimal cover. Following the separation heuristic given in Section 4.1, we let $\mathcal{D} = \{i \in \mathcal{C} : \hat{y} = 1\} = \{5\}$. Suppose $N = 5$, $\varepsilon = 0.6$, and the other scenarios in the computation of lifted cover inequalities are $(8, 11, 7, 10, 7, 17, 23)^\top$, $(14, 7, 10, 11, 8, 13, 26)^\top$, $(21, 10, 7, 29, 16, 12, 23)^\top$, and $(15, 7, 8, 23, 12, 10, 5)^\top$, with $p_\omega = 1/N$ for all $\omega \in \Omega$. We get a lifted cover inequality by Theorem 8 as:

$$y_{1j} + y_{2j} + y_{3j} + y_{4j} + y_{5j} + 2y_{6j} + 2y_{7j} + z_{j\omega} \le 5. \tag{17}$$

If $p_\omega^* = 1/N$ for all $\omega \in \Omega$ and $\eta = 0.5$ in the Wasserstein set $\mathscr{P}_W$ in (7), then a lifted cover inequality for (DR-CAP) obtained from Theorem 9 is given as follows:

$$y_{1j} + y_{2j} + y_{3j} + y_{4j} + y_{5j} + 2y_{6j} + 2y_{7j} \le 4. \tag{18}$$

Example 2. (**Continued from Example 1**) Suppose that the cardinality constraint $\sum_{i\in\mathcal{I}} y_{ij} \le \rho_j$ is removed from $\mathcal{F}_{j\omega}$. Following a computation procedure similar to the one for the lifted cover inequality, we obtain a valid inequality of the following form:

$$y_{1j} + y_{2j} + y_{3j} + y_{4j} + y_{5j} + y_{6j} + 2y_{7j} + z_{j\omega} \le 5. \tag{19}$$

We call (19) single lifted cover inequality. The lifted cover inequality (17) is stronger than the single lifted cover inequality (19). The single lifted cover inequality for (DR-CAP) is

$$y_{1j}+y_{2j}+y_{3j}+y_{4j}+y_{5j}+y_{6j}+2y_{7j}\leq 4. \tag{20}$$

If $\hat{y}$ satisfies (18), it also satisfies (20), which implies that (18) is stronger than (20). Using a similar reasoning, we know that (18) is stronger than (17), and (20) is stronger than (19). Thus showing the possible benefit of using the cardinality constraint, and the ambiguity set in the coefficient calculations.

3.2. Global Lifted Cover Inequalities

In this section we develop a class of valid inequalities referred to as global lifted cover inequalities for $\mathcal{G}_j$ and $\mathcal{G}'_j$, which are valid for (CAP) and (DR-CAP), respectively. For (CAP), let $\bar{\Omega}$ be a set where each element $\Omega_k\in\bar{\Omega}$ is a subset of Ω such that $\sum_{\omega\in\Omega_k} p_\omega \geq 1-\varepsilon$, for $k=1,\ldots,|\bar{\Omega}|$. Without loss of generality, we reuse the notation set $\bar{\Omega}$ and Ω_k for (DR-CAP). For (DR-CAP), let $\bar{\Omega}$ be a set where each element $\Omega_k\in\bar{\Omega}$ is a subset of Ω such that $\inf_{\boldsymbol{p}\in\mathscr{P}}\sum_{\omega\in\Omega_k} p_\omega \geq 1-\varepsilon$, for $k=1,\ldots,|\bar{\Omega}|$. $\bar{\Omega}$ is maximal if it is not a proper subset of any other sets that satisfy the above condition. Let the global lifted cover inequalities be of the form

$$\sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij}+\sum_{i\in\mathcal{I}\setminus\mathcal{C}}\bar{\alpha}_i y_{ij}+\sum_{i\in\mathcal{D}}\bar{\beta}_i y_{ij}+\sum_{\omega\in\Omega_k}\bar{\gamma}_\omega(z_{j\omega}-1)\leq|\mathcal{C}\setminus\mathcal{D}|+\sum_{i\in\mathcal{D}}\bar{\beta}_i-1,\ \ k=1,\ldots,|\bar{\Omega}|, \tag{21}$$

where $\mathcal{C}$ is a cover for the set $\mathcal{Q}_{j\omega}$ for some $\omega\in\Omega$, and $\mathcal{D}\subseteq\mathcal{C}$. If $\bar{\Omega}$ is maximal, $\bar{\Omega}$ is unique and (21) includes all possible such type of inequalities. For $k\in\{1,\ldots,|\bar{\Omega}|\}$, when $z_{j\omega}=1$, $\omega\in\Omega_k$, (21) becomes

$$\sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij}+\sum_{i\in\mathcal{I}\setminus\mathcal{C}}\bar{\alpha}_i y_{ij}+\sum_{i\in\mathcal{D}}\bar{\beta}_i y_{ij}\leq|\mathcal{C}\setminus\mathcal{D}|+\sum_{i\in\mathcal{D}}\bar{\beta}_i-1. \tag{22}$$

Kaparis and Letchford (2008) developed a valid inequality for multi-constrained knapsack problems. In Section 3.2.1 and 3.2.2, we use the ideas from Kaparis and Letchford (2008) to calculate the coefficients $\bar{\alpha}_i$ and $\bar{\beta}_i$ in (22).

3.2.1. Up-Lifting

Let $\{\bar{\pi}_l\}_{l=1}^{|\mathcal{I}\setminus\mathcal{C}|}$ be a sequence of $\mathcal{I}\setminus\mathcal{C}$ and $\bar{\boldsymbol{\pi}}(l)=\{\bar{\pi}_1,\cdots,\bar{\pi}_l\}$. For $l=1,\ldots,|\mathcal{I}\setminus\mathcal{C}|$, the up-lifting problem is as follows:

$$obj_{\bar{\pi}_l}:=\underset{\boldsymbol{y}_j\in\{0,1\}^{|(\mathcal{C}\setminus\mathcal{D})\cup\bar{\boldsymbol{\pi}}(l-1)|}}{\text{maximize}}\ \sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij}+\sum_{i\in\bar{\boldsymbol{\pi}}(l-1)}\bar{\alpha}_i y_{ij} \tag{23a}$$

$$\text{subject to}\ \sum_{i\in\mathcal{C}\setminus\mathcal{D}}\xi_i^\omega y_{ij}+\sum_{i\in\bar{\boldsymbol{\pi}}(l-1)}\xi_i^\omega y_{ij}\leq m_j^\omega(\omega)-\xi_{\bar{\pi}_l}^\omega-\sum_{i\in\mathcal{D}}\xi_i^\omega,\qquad\forall\omega\in\Omega_k, \tag{23b}$$

$$\sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij}+\sum_{i\in\bar{\boldsymbol{\pi}}(l-1)} y_{ij}\leq\rho_j-1-|\mathcal{D}|. \tag{23c}$$

Then $\bar{\alpha}_{\bar{\pi}_l} = |\mathcal{C}\backslash\mathcal{D}| - 1 - obj_{\bar{\pi}_l}$. It is time-consuming to solve the up-lifting problem exactly. Dynamic programming is also not an efficient approach since its complexity grows with the number of constraints in (23). Kaparis and Letchford (2008) suggest relaxing $\boldsymbol{y}_j \in [0,1]^{|\mathcal{I}|}$ and solving the LP relaxation to compute an upper bound on $obj_{\bar{\pi}_l}$. The objective value is then rounded down to the nearest integer. We propose a heuristic to calculate $\bar{\alpha}_{\bar{\pi}_l}$ as follows. For each $\omega \in \Omega_k$, let

$$\begin{aligned} obj_{\bar{\pi}_l}(\omega) := \underset{\boldsymbol{y}_j \in \{0,1\}^{|(\mathcal{C}\backslash\mathcal{D})\cup\bar{\boldsymbol{\pi}}(l-1)|}}{\text{maximize}} \quad & \sum_{i\in\mathcal{C}\backslash\mathcal{D}} y_{ij} + \sum_{i\in\bar{\boldsymbol{\pi}}(l-1)} \bar{\alpha}_i y_{ij} \\ \text{subject to} \quad & \sum_{i\in\mathcal{C}\backslash\mathcal{D}} \xi_i^\omega y_{ij} + \sum_{i\in\bar{\boldsymbol{\pi}}(l-1)} \xi_i^\omega y_{ij} \le m_j^\omega(\omega) - \xi_{\bar{\pi}_l}^\omega - \sum_{i\in\mathcal{D}} \xi_i^\omega, \\ & \sum_{i\in\mathcal{C}\backslash\mathcal{D}} y_{ij} + \sum_{i\in\bar{\boldsymbol{\pi}}(l-1)} y_{ij} \le \rho_j - 1 - |\mathcal{D}|. \end{aligned}$$

Then, $obj_{\bar{\pi}_l}(\omega)$ is an upper bound for $obj_{\bar{\pi}_l}$. We use $\min_{\omega\in\Omega_k} obj_{\bar{\pi}_l}(\omega)$ to obtain a minimal upper bound for $obj_{\bar{\pi}_l}$ from among the values $\{obj_{\bar{\pi}_l}(\omega)\}_{\omega\in\Omega_k}$. Let $\bar{\alpha}_{\bar{\pi}_l} = |\mathcal{C}\backslash\mathcal{D}| - 1 - \min_{\omega\in\Omega_k} obj_{\bar{\pi}_l}(\omega)$, which implies $\bar{\alpha}_{\bar{\pi}_l} \le |\mathcal{C}\backslash\mathcal{D}| - 1 - obj_{\bar{\pi}_l}$. Thus, $\bar{\alpha}_{\bar{\pi}_l}$ is a valid lifting coefficient.

3.2.2. Down-Lifting Similar to up-lifting, we can obtain the down-lifting coefficient $\bar{\beta}_i$ for $i \in \mathcal{D}$. Let $\{\bar{\kappa}_l\}_{l=1}^{|\mathcal{D}|}$ be a sequence of $\mathcal{D}$ and $\bar{\boldsymbol{\kappa}}(l) = \{\bar{\kappa}_1, \cdots, \bar{\kappa}_l\}$. For $l = 1, \cdots, |\mathcal{D}|$, let

$$\text{obj}_{\bar{\kappa}_l} := \underset{\boldsymbol{y}_j \in \{0,1\}^{|(\mathcal{I}\backslash\mathcal{D})\cup\bar{\boldsymbol{\kappa}}(l-1)|}}{\text{maximize}} \sum_{i\in\mathcal{C}\backslash\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\backslash\mathcal{C}} \bar{\alpha}_i y_{ij} + \sum_{i\in\bar{\boldsymbol{\kappa}}(l-1)} \bar{\beta}_i y_{ij} \tag{24a}$$

$$\text{subject to} \sum_{i\in\mathcal{I}\backslash\mathcal{D}} \xi_i^\omega y_{ij} + \sum_{i\in\bar{\boldsymbol{\kappa}}(l-1)} \xi_i^\omega y_{ij} \le m_j^\omega(\omega) - \sum_{i=\bar{\kappa}_{l+1}}^{\bar{\kappa}_{|\mathcal{D}|}} \xi_i^\omega, \qquad \forall \omega \in \Omega_k, \tag{24b}$$

$$\sum_{i\in\mathcal{I}\backslash\mathcal{D}} y_{ij} + \sum_{i\in\bar{\boldsymbol{\kappa}}(l-1)} y_{ij} \le \rho_j - |\mathcal{D}| + l. \tag{24c}$$

Instead of computing $\text{obj}_{\bar{\kappa}_l}$, we use the method proposed in Section 3.2.1 to obtain an upper bound for $\text{obj}_{\bar{\kappa}_l}$. Specifically we use $\text{obj}_{\bar{\kappa}_l}(\omega)$ as the optimal objective value of the maximization problem that takes a single row ω of problem (24) for $\omega \in \Omega_k$ and let $\bar{\beta}_{\bar{\kappa}_l} = \min_{\omega\in\Omega_k} obj_{\bar{\kappa}_l}(\omega) - \sum_{i\in\bar{\boldsymbol{\kappa}}(l-1)} \bar{\beta}_i - |\mathcal{C}\backslash\mathcal{D}| + 1$ in the computations.

3.2.3. Global Lifted Cover Inequalities Finally, to calculate $\bar{\gamma}_\omega$ in sequence $\{\tau_1, \ldots, \tau_{|\Omega_k|}\}$, we consider the following problem for $\mathcal{G}_j$, for $l = 1, \ldots, |\Omega_k|$:

$$obj_{\tau_l} = \underset{(\boldsymbol{y}_j, \boldsymbol{z}_j)\in\{0,1\}^{|\mathcal{I}|}\times\{0,1\}^{|(\Omega\backslash\Omega_k)\cup\boldsymbol{\tau}(l-1)|}}{\text{maximize}} \sum_{i\in\mathcal{C}\backslash\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\backslash\mathcal{C}} \bar{\alpha}_i y_{ij} + \sum_{i\in\mathcal{D}} \bar{\beta}_i y_{ij} + \sum_{\omega\in\boldsymbol{\tau}(l-1)} \gamma_\omega z_{j\omega} \tag{25a}$$

$$\text{subject to} \sum_{\omega\in\Omega\backslash\Omega_k} p_\omega z_{j\omega} + \sum_{\omega\in\boldsymbol{\tau}(l-1)} p_\omega z_{j\omega} \ge 1 - \varepsilon - \sum_{\omega=\tau_{l+1}}^{\tau_{|\Omega_k|}} p_\omega, \tag{25b}$$

$$\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} + \left(m_j^\omega(k_q) - m_j^\omega(\omega)\right) z_{j\omega} \le m_j^\omega(k_q), \ \forall \omega \in \Omega\backslash\Omega_k \cup \boldsymbol{\tau}(l-1), \tag{25c}$$

$$\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \le m_j^\omega(\omega), \ \forall \omega \in \{\tau_{l+1}, \ldots, \tau_{|\Omega_k|}\}, \ \boldsymbol{y}_j \in \mathcal{Y}_j, \tag{25d}$$

where $\boldsymbol{\tau}(l-1)=\{\tau_1,\dots,\tau_{l-1}\}$. The calculation of obj_{τ_l} is a reformulation of a chance-constrained problem where some variables $z_{j\omega}$ are given. Instead of solving (25) exactly, we provide a heuristic to obtain an upper bound for obj_{τ_l}. We relax $\boldsymbol{y}_j\in[0,1]^{|\mathcal{I}|}$ and $\boldsymbol{z}_j\in[0,1]^{|\Omega_k|}$, and solve the LP relaxation of (25) to obtain an optimal solution $(\boldsymbol{y}_j^r,\boldsymbol{z}_j^r)$ and objective value $obj_{\tau_l}^r$ of the relaxed problem. Then, $obj_{\tau_l}^r$ gives an upper bound for obj_{τ_l}.

For $\mathcal{G}_j'$, for $l=1,\cdots,|\Omega_k|$, let

$$obj_{\tau_l}':=\underset{(\boldsymbol{y}_j,\boldsymbol{z}_j)\in\{0,1\}^{|\mathcal{I}|}\times\{0,1\}^{|(\Omega\setminus\Omega_k)\cup\boldsymbol{\tau}(l-1)|}}{\text{maximize}}\sum_{i\in\mathcal{C}\setminus\mathcal{D}}y_{ij}+\sum_{i\in\mathcal{I}\setminus\mathcal{C}}\bar{\alpha}_iy_{ij}+\sum_{i\in\mathcal{D}}\bar{\beta}_iy_{ij}+\sum_{\omega\in\boldsymbol{\tau}(l-1)}\gamma_\omega z_{j\omega} \tag{26a}$$

subject to (25d),

$$\inf_{\boldsymbol{p}\in\mathscr{P}}\sum_{\omega\in\Omega\setminus\Omega_k}p_\omega z_{j\omega}+\sum_{\omega\in\boldsymbol{\tau}(l-1)}p_\omega z_{j\omega}\geq 1-\varepsilon-\sum_{\omega=\tau_{l+1}}^{\tau_{|\Omega_k|}}p_\omega, \tag{26b}$$

$$\sum_{i\in\mathcal{I}}\xi_i^\omega y_{ij}+(m_j^\omega(k_{\bar{q}})-m_j^\omega(\omega))z_{j\omega}\leq m_j^\omega(k_{\bar{q}}),\quad\forall\omega\in\Omega\setminus\Omega_k\cup\boldsymbol{\tau}(l-1). \tag{26c}$$

We relax $\boldsymbol{y}_j\in[0,1]^{|\mathcal{I}|}$ and $\boldsymbol{z}_j\in[0,1]^{|\Omega_k|}$, and solve the relaxation problem using the method similar to Algorithm 1 in Section 4.3, and obtain the optimal objective value $obj_{\tau_l}^{r\prime}$, which is an upper bound on obj_{τ_l}'.

Theorem 2 gives valid inequalities for $conv(\mathcal{G}_j)$ and $conv(\mathcal{G}_j')$. A proof is given in Appendix B.8.

THEOREM 2. *Let $\{\bar{\alpha}_i\}_{i\in\mathcal{I}\setminus\mathcal{C}}$ and $\{\bar{\beta}\}_{i\in\mathcal{D}}$ be defined as in Section 3.2.1 and 3.2.2, respectively. For $l=1,\cdots,|\Omega_k|$, we set $\bar{\gamma}_{\tau_l}=\lfloor obj_{\tau_l}^r\rfloor-|\mathcal{C}\setminus\mathcal{D}|+1-\sum_{i\in\mathcal{D}}\bar{\beta}_i-\sum_{\omega\in\boldsymbol{\tau}(l-1)}\bar{\gamma}_\omega$, where $obj_{\tau_l}^r$ is the objective value of the LP relaxation of (25). Then, (21) is valid for $conv(\mathcal{G}_j)$. For $l=1,\cdots,|\Omega_k|$, we set $\bar{\gamma}_{\tau_l}=\lfloor obj_{\tau_l}^{r'}\rfloor-|\mathcal{C}\setminus\mathcal{D}|+1-\sum_{i\in\mathcal{D}}\bar{\beta}_i-\sum_{\omega\in\boldsymbol{\tau}(l-1)}\bar{\gamma}_\omega$, where $obj_{\tau_l}^{r'}$ is the objective value of the LP relaxation of (26). Then, (21) is valid for $conv(\mathcal{G}_j')$.* □

The following example gives a global lifted cover inequality.

EXAMPLE 3. (**Continued from Example 1**) We let $\mathcal{C}=\{1,2,3,4,5\}$ and $\mathcal{D}=\{5\}$ as before. Let $\Omega_k=\{1,2\}$. Then we can obtain a global lifted cover inequality (21) for (CAP) given as follows:

$$y_{1j}+y_{2j}+y_{3j}+y_{4j}+2y_{6j}+2y_{7j}+z_{j1}+z_{j2}\leq 5.$$

For (DR-CAP), $\Omega_k=\{1,2,3\}$ satisfies $\inf_{\boldsymbol{p}\in\mathscr{P}}\sum_{\omega\in\Omega_k}p_\omega\geq 1-\varepsilon$. A valid inequality (21) is given by

$$y_{1j}+y_{2j}+y_{3j}+y_{4j}+2y_{6j}+3y_{7j}+2z_{j1}+2z_{j2}+2z_{j3}\leq 9.$$

4. Solution Scheme

In Section 4.1, we present a heuristic sequential lifting procedure for separating the valid inequalities developed in Section 3. These valid inequalities are used within a branch-and-cut framework to solve the strengthened big-M binary reformulation (IP) of (CAP) in Section 4.2. A branch-and-cut algorithm with probability cuts to solve the strengthened big-M semi-infinite reformulation (SIP) of (DR-CAP) is given in Section 4.3.

4.1. Separation Problem

Separation problem finds valid inequalities that are violated by an LP relaxation solution $(\hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$. Klabjan et al. (1998) formulated the separation problem for the cover inequalities as a 0-1 knapsack problem, and showed that this separation problem is NP-hard. Hence, we use heuristics for computing inequalities (39) and (21). In this section, we adopt the ideas from Gu et al. (1998) and Kaparis and Letchford (2008) for the binary knapsack problem to separate (39) and (21), respectively.

4.1.1. Separation Problem for (39) To obtain the violated inequalities (39), we use a heuristic similar to the one in Gu et al. (1998) for the knapsack problem. This heuristic is provided as Algorithm 3 in Appendix C.2.

If $|\mathcal{D}| > \rho_j - 1$ or $m_j^\omega(\omega) - \sum_{i\in\mathcal{D}} \xi_i^\omega - \max_{i\in\mathcal{I}\setminus\mathcal{C}} \xi_i^\omega < 0$ for $\omega \in \Omega$, the down-lifting problems might be infeasible since the right hand side of the down-lifting problems might be negative. In this case, we remove items from $\mathcal{D}$ until $|\mathcal{D}| \le \rho_j - 1$ and $m_j^\omega(\omega) - \sum_{i\in\mathcal{D}} \xi_i^\omega - \max_{i\in\mathcal{I}\setminus\mathcal{C}} \xi_i^\omega \ge 0$ for $\omega \in \Omega$.

4.1.2. Separation Problem for (21) Gu et al. (1998) proposed a greedy heuristic for initializing the cover set by selecting items with the highest LP values. In Algorithms 3 and 4 of Appendix C.3, we use this greedy heuristic to build the initial cover set. Note that the generation of a valid inequality depends on the lifting sequence. The variables with an earlier lifting position are expected to have better coefficient values (i.e., resulting in stronger cuts for that variable) in the inequalities. Hence, the variables are first lifted from the set $\mathcal{I}\setminus(\mathcal{C}\cup\mathcal{I}_0)$, then the set $\mathcal{D}$, and finally the set $\mathcal{I}_0$ in Algorithm 3 and 4. As stated in Gu et al. (1998) and Kaparis and Letchford (2008), different lifting orders within set $\mathcal{I}\setminus(\mathcal{C}\cup\mathcal{I}_0)$, $\mathcal{I}_0$, and $\mathcal{D}$ have comparable computational performance. Thus the sequence of lifting variables chosen within the sets $\mathcal{I}\setminus(\mathcal{C}\cup\mathcal{I}_0)$, $\mathcal{I}_0$, and $\mathcal{D}$ is used in their order of indices.

In our computational experience, the conditions in line 4 of Algorithm 4 is met about 33% of the time. For example, for $N = 500$ (CAP) models with $\varepsilon = 0.1$, it is met in 37.4% cases. For the (DR-CAP) instances of the same size with $\eta = 1$, it is met in 25.2% cases at the nodes that are at depth ≤ 2. Similar to the discussion in Section 4.1.1, if $|\mathcal{D}| > \rho_j - 1$ or $m_j^\omega(\omega) - \sum_{i\in\mathcal{D}} \xi_i^\omega - \max_{i\in\mathcal{I}\setminus\mathcal{C}} \xi_i^\omega < 0$ for some $\omega \in \Omega$, we remove the items from $\mathcal{D}$ until $|\mathcal{D}| \le \rho_j - 1$ and $m_j^\omega(\omega) - \sum_{i\in\mathcal{D}} \xi_i^\omega - \max_{i\in\mathcal{I}\setminus\mathcal{C}} \xi_i^\omega \ge 0$ for all $\omega \in \Omega$.

4.2. Branch-and-Cut Algorithm for (CAP)

The valid inequalities in Section 3 are used within a branch-and-cut implementation to solve (CAP). An overview of the branch-and-cut framework is given in Algorithm 5 (Appendix C.4). The algorithm uses the violated inequalities described in Section 4.1.1 and 4.1.2 in line 9 (see Section 6.2 for further discussion). Let LB and UB denote the current lower and upper bound for the optimal objective value of (CAP), and $\mathcal{N}$ denote the set of remaining nodes in the branch-and-cut search tree.

4.3. Branch-and-Cut Algorithm with Probability Cuts for (DR-CAP)

We now investigate the probability cuts within a branch-and-cut framework for solving (DR-CAP). We define the master problem as follows:

$$\text{(MP)} \qquad \underset{(\boldsymbol{y},\boldsymbol{z})\in\left\{\{0,1\}^{|I||J|}\times\{0,1\}^{|J|N}\right\}\cap\mathcal{X}}{\text{minimize}} \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} c_{ij}y_{ij} \quad \text{subject to (1b), (1c), (8c),}$$

where the set $\mathcal{X}$ is a complementary set that defines the feasible region of (8). Set $\mathcal{X}$ is defined by a set of probability and feasibility cuts. Let $(\hat{\boldsymbol{y}},\hat{\boldsymbol{z}})$ be a feasible solution of (MP). For $j\in\mathcal{J}$, a distribution separation problem is given by:

$$(\text{SP}_j) \qquad \mathcal{S}_j(\hat{\boldsymbol{z}}) := \inf_{\boldsymbol{p}\in\mathscr{P}} \sum_{\omega\in\Omega} p_\omega \hat{z}_{j\omega}.$$

The problem (SP_j) is used to verify the feasibility of $(\hat{\boldsymbol{y}}_j,\hat{\boldsymbol{z}}_j)$ to (DR-CAP). If $\mathcal{S}_j(\hat{\boldsymbol{z}})\geq 1-\varepsilon$, $(\hat{\boldsymbol{y}}_j,\hat{\boldsymbol{z}}_j)$ is feasible to (DR-CAP). Otherwise, probability and feasibility cuts are added to (MP) as follows.

Let $\{\hat{p}_\omega\}_{\omega\in\Omega}\in\mathscr{P}$ be an optimal solution of (SP_j) corresponding to $\hat{\boldsymbol{z}}$, then the following inequality is called a probability cut:

$$\sum_{\omega\in\Omega}\hat{p}_\omega z_{j\omega}\geq 1-\varepsilon. \tag{28}$$

Let $\mathcal{I}_j^1=\{i\in\mathcal{I}|\hat{y}_{ij}=1\}$. The following feasibility cut in variable $\boldsymbol{y}$ is added to (MP):

$$\sum_{i\in\mathcal{I}_j^1} y_{ij}\leq |\mathcal{I}_j^1|-1. \tag{29}$$

In Algorithm 1, UB and LB denote the upper and lower bound, respectively. We initialize the algorithm by setting the iteration number k to 0, UB to positive infinity, and LB to negative infinity. We add a node o to the node list $\mathcal{N}$ and use (LMP) to denote the LP relaxation of (MP) (line 1-2). At the selected node o, we solve (LMP) and obtain the corresponding optimal solution $(\boldsymbol{y}^k,\boldsymbol{z}^k)$ and the objective value $lobj^k$ (line 4-6). If the objective value $lobj^k$ is smaller than the current upper bound, then we check if $(\boldsymbol{y}^k,\boldsymbol{z}^k)$ is binary (line 7). If $(\boldsymbol{y}^k,\boldsymbol{z}^k)$ is binary, we solve the distribution separation problem (SP_j) with $\mathscr{P}$ for all $j\in\mathcal{J}$, and obtain the optimal solution $\{p_\omega^k\}_{\omega\in\Omega}$ and the objective value $uobj^k$. We add probability and feasibility cuts to (LMP) if $uobj^k$ is smaller than $1-\varepsilon$ (line 8-14). If we find probability and feasibility cuts, we go to line 5, and resolve (LMP) at the current node o. Otherwise, $(\boldsymbol{y}^k,\boldsymbol{z}^k)$ is a feasible solution to (DR-CAP), we update the upper bound and record the corresponding solution $(\boldsymbol{y}^k, \boldsymbol{z}^k)$ (line 15-20). If $(\boldsymbol{y}^k, \boldsymbol{z}^k)$ is fractional, we add violated inequalities or continue branching (line 22-30). We terminate our algorithm when the node list is empty, and return the optimal value UB and the optimal solution $(\boldsymbol{y}^*, \boldsymbol{z}^*)$ (line 33). Algorithm 1 gives a pseudocode of the branch-and-cut algorithm with probability and feasibility cuts.

The following theorem shows that Algorithm 1 terminates in a finite number of iterations for solving (DR-CAP) to optimality under certain conditions.

THEOREM 3. *If there exists an oracle that solves (SP_j) to optimality, then Algorithm 1 terminates in finitely many iterations. If $UB < +\infty$, UB is the optimal value of (DR-CAP) and Algorithm 1 obtains an optimal solution $(\boldsymbol{y}^*, \boldsymbol{z}^*)$ at termination.*

Proof See Appendix B.9. □

Remark 5. Recall that $\mathscr{P}_W$ in (7) is a polyhedral set with a finite number of extreme points, thus in this case (SP_j) can be solved to optimality.

Remark 6. In the case of a polyhedral ambiguity set, such as the set $\mathscr{P}_W$, instead of using the probability cut approach discussed above, it is possible to dualize the problem in (8) and write this constraint explicitly (see Rahimian and Mehrotra (2019) and references therein). Such dualization introduces dual variables corresponding to the constraints specifying $\mathscr{P}_W$, and we can obtain the dual reformulation of (DR-CAP) (see Section 6.4).

5. Generalization to Joint Chance-Constrained Problems

The valid inequalities and solution schemes proposed in this paper are now developed for the joint chance constraint assignment problem (JCAP), and the joint distributionally robust chance constraint (DR-JCAP). Let us consider the joint chance constraint

$$\mathbb{P}\left\{\sum_{i\in\mathcal{I}} \xi_i y_{ij} \le t_j,\ \forall j\in\mathcal{J}\right\} \le 1-\varepsilon, \tag{30}$$

and the joint distributionally robust chance constraint

$$\inf_{\mathbb{P}\in\mathscr{P}} \mathbb{P}\left\{\sum_{i\in\mathcal{I}} \xi_i y_{ij} \le t_j,\ \forall j\in\mathcal{J}\right\} \le 1-\varepsilon. \tag{31}$$

Under the finite support assumption, constraint (30) can be rewritten as

$$\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} + \left(m_j^\omega(k_q) - m_j^\omega(\omega)\right) z'_\omega \le m_j^\omega(k_q), \qquad \forall j\in\mathcal{J}, \omega\in\Omega, \tag{32a}$$

$$\sum_{\omega\in\Omega} p_\omega z'_\omega \ge 1-\varepsilon, \qquad \forall j\in\mathcal{J}, \tag{32b}$$

Algorithm 1: Branch-and-Cut Algorithm with Probability Cuts

1 ***Initialize*** $\mathbb{P}^0 \in \mathscr{P}$, the number of iteration $k=0$, $\mathrm{UB}=+\infty$, $\mathrm{LB}=-\infty$, $\mathcal{N}=\{o\}$, o has no branching constraints.
2 Initialize the root node with the LP relaxation of (MP). Let the LP relaxation of (MP) be denoted by (LMP).
3 **while** *($\mathcal{N}$ is nonempty)* **do**
4   Select a node $o \in \mathcal{N}$, $\mathcal{N} \leftarrow \mathcal{N}/\{o\}$.
5   Solve (LMP) at the node o. $k=k+1$.
6   Obtain the optimal solution (y^k, z^k) and the optimal objective $lobj^k$ of (LMP).
7   **if** $lobj^k < UB$ **then**
8   **if** (y^k, z^k) *is an integer* **then**
9   **for** $j \in \mathcal{J}$ **do**
10   Solve (SP_j), and obtain an optimal solution (p^k) and objective value $uobj^k$
11   **if** $uobj^k < 1-\varepsilon$ **then**
12   Add the cuts (28) and (29) to (LMP).
13   **end**
14   **end**
15   **if** *Cuts (28) and (29) are found* **then**
16   Go to step 5.
17   **end**
18   **else**
19                 $\mathrm{UB} = lobj^k$, $(y^*, z^*) = (y^k, z^k)$.
20   **end**
21   **end**
22   **if** $(\boldsymbol{y}^k, \boldsymbol{z}^k)$ *is fractional* **then**
23   Use Algorithm 3 and 4 to find the violated inequalities.
24   **if** *Violated inequalities are found* **then**
25   Add the violated inequalities to (LMP). Go to line 5.
26   **end**
27   **else**
28   Branch, resulting in nodes o^* and o^{**}, $\mathcal{N} \leftarrow \mathcal{N} \cup \{o^*, o^{**}\}$.
29   **end**
30   **end**
31   **end**
32 **end**
33 **return** UB and its corresponding optimal solution (y^*, z^*).

where $m_j^\omega(\cdot)$ and q is defined in Section 2.1, and

$$z'_\omega = \begin{cases} 1, & \text{if } \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \le t_j, \ \forall j \in \mathcal{J}, \\ 0, & \text{otherwise.} \end{cases}$$

A similar formulation can be found in Deng and Shen (2016). For the distributionally robust joint constraint (31), we have the reformulation

$$\inf_{\boldsymbol{p}\in\mathscr{P}} \sum_{\omega\in\Omega} p_\omega z'_\omega \ge 1-\varepsilon, \qquad \forall j \in \mathcal{J}, \tag{33a}$$

$$\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} + \left(m_j^\omega(k_{\bar q}) - m_j^\omega(\omega)\right) z'_\omega \le m_j^\omega(k_{\bar q}), \qquad \forall j \in \mathcal{J}, \omega \in \Omega. \tag{33b}$$

Observe that the reformulations of (30) and (31) are obtained by replacing $z_{j\omega}$ in (IP) and (SIP) with z'_ω.

Let us define the distribution separation problem as

$$\text{(SP)} \qquad \mathcal{S}(\hat{\boldsymbol{z}}') := \inf_{\boldsymbol{p}\in\mathscr{P}} \sum_{\omega\in\Omega} p_\omega \hat{z}'_\omega,$$

and $\{\hat{p}_\omega\}_{\omega\in\Omega}$ be an optimal solution of (SP) corresponding to $\hat{\boldsymbol{z}}'$. We call the cut

$$\sum_{\omega\in\Omega} \hat{p}_\omega z'_\omega \geq 1-\varepsilon \tag{34}$$

a probability cut. Using the above distribution separation problem and probability cut, we can use the branch-and-cut algorithm with probability cut to solve (DR-JCAP). We state the following results for (DR-JCAP) and (JCAP), whose proofs are similar to those for (DR-CAP) and (CAP). More specifically, we attain the lifted cover inequalities for (JCAP) and (DR-JCAP) in Theorem 4.

Theorem 4. *For $j \in \mathcal{J}$, and $\omega\in\Omega$, the following inequality*

$$\sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\setminus\mathcal{C}} \alpha_i y_{ij} + \sum_{i\in\mathcal{D}} \beta_i y_{ij} + \gamma(z'_\omega - 1) \leq |\mathcal{C}\setminus\mathcal{D}| + \sum_{i\in\mathcal{D}} \beta_i - 1 \tag{35}$$

is valid for (JCAP) when $\gamma = \delta_{\bar{k}_{q}1}$, and is valid for (DR-JCAP) when $\gamma = \delta_{\bar{k}_{\bar{q}}1}$ or $\gamma = \delta_{\bar{k}_{\hat{q}}1}$, where $\mathcal{C}\subseteq\mathcal{I}$ is a cover for $\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \leq m_j^\omega(\omega)$, $\mathcal{D}\subseteq\mathcal{C}$, $\delta_{\bar{k}_{q}1}$, $\delta_{\bar{k}_{\bar{q}}1}$ and $\delta_{\bar{k}_{\hat{q}}1}$ are defined in Appendix A, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are up-lifting and down-lifting coefficients defined in Section 3.1.1 and 3.1.2, respectively. □

Let

$$o\bar{b}j_{\tau_l} = \underset{(\boldsymbol{y}_j,\boldsymbol{z}')\in\{0,1\}^{|\mathcal{I}|}\times\{0,1\}^{|(\Omega\setminus\Omega_k)\cup\boldsymbol{\tau}(l-1)|}}{\text{maximize}} \sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\setminus\mathcal{C}} \bar{\alpha}_i y_{ij} + \sum_{i\in\mathcal{D}} \bar{\beta}_i y_{ij} + \sum_{\omega\in\boldsymbol{\tau}(l-1)} \gamma_\omega z'_\omega \tag{36a}$$

$$\text{subject to} \quad \sum_{\omega\in\Omega\setminus\Omega_k} p_\omega z'_\omega + \sum_{\omega\in\boldsymbol{\tau}(l-1)} p_\omega z'_\omega \geq 1-\varepsilon - \sum_{\omega=\tau_{l+1}}^{\tau_{|\Omega_k|}} p_\omega, \tag{36b}$$

$$\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} + \left(m_j^\omega(k_q) - m_j^\omega(\omega)\right) z'_\omega \leq m_j^\omega(k_q), \ \ \forall \omega\in\Omega\setminus\Omega_k\cup\boldsymbol{\tau}(l-1), \tag{36c}$$

$$\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \leq m_j^\omega(\omega), \ \ \forall\omega\in\{\tau_{l+1},\ldots,\tau_{|\Omega_k|}\},\ \boldsymbol{y}_j\in\mathcal{Y}_j, \tag{36d}$$

and

$$o\bar{b}j'_{\tau_l} = \underset{(\boldsymbol{y}_j,\boldsymbol{z}')\in\{0,1\}^{|\mathcal{I}|}\times\{0,1\}^{|(\Omega\setminus\Omega_k)\cup\boldsymbol{\tau}(l-1)|}}{\text{maximize}} \sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\setminus\mathcal{C}} \bar{\alpha}_i y_{ij} + \sum_{i\in\mathcal{D}} \bar{\beta}_i y_{ij} + \sum_{\omega\in\boldsymbol{\tau}(l-1)} \gamma_\omega z'_\omega \tag{37a}$$

subject to (36d),

$$\inf_{\boldsymbol{p}\in\mathscr{P}} \sum_{\omega\in\Omega\setminus\Omega_k} p_\omega z'_\omega + \sum_{\omega\in\boldsymbol{\tau}(l-1)} p_\omega z'_\omega \geq 1-\varepsilon - \sum_{\omega=\tau_{l+1}}^{\tau_{|\Omega_k|}} p_\omega, \tag{37b}$$

$$\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} + (m_j^\omega(k_{\bar{q}}) - m_j^\omega(\omega)) z'_\omega \leq m_j^\omega(k_{\bar{q}}), \quad \forall\omega\in\Omega\setminus\Omega_k\cup\boldsymbol{\tau}(l-1), \tag{37c}$$

where Ω_k, $\mathcal{C}$ and $\mathcal{D}$ are defined in Section 3.2, $\{\tau_1,\ldots,\tau_{|\Omega_k|}\}$ is a sequence of Ω_k, and $\boldsymbol{\tau}(l-1)=\{\tau_1,\ldots,\tau_{l-1}\}$. Then the following theorem gives the global lifted cover inequalities that are valid for (JCAP) and (DR-JCAP).

THEOREM 5. *Let $\{\bar{\alpha}_i\}_{i\in\mathcal{I}\setminus\mathcal{C}}$ and $\{\bar{\beta}\}_{i\in\mathcal{D}}$ be defined as in Section 3.2.1 and 3.2.2, respectively. For $l=1,\cdots,|\Omega_k|$, we set $\bar{\gamma}_{\tau_l}=\lfloor \bar{obj}^r_{\tau_l}\rfloor-|\mathcal{C}\setminus\mathcal{D}|+1-\sum_{i\in\mathcal{D}}\bar{\beta}_i-\sum_{\omega\in\boldsymbol{\tau}(l-1)}\bar{\gamma}_\omega$, where $\bar{obj}^r_{\tau_l}$ is the objective value of the LP relaxation of (36). Then, (38) is valid for (JCAP).*

$$\sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij}+\sum_{i\in\mathcal{I}\setminus\mathcal{C}}\bar{\alpha}_i y_{ij}+\sum_{i\in\mathcal{D}}\bar{\beta}_i y_{ij}+\sum_{\omega\in\Omega_k}\bar{\gamma}_\omega(z'_\omega-1)\le|\mathcal{C}\setminus\mathcal{D}|+\sum_{i\in\mathcal{D}}\bar{\beta}_i-1,\ \ k=1,\ldots,|\bar{\Omega}|. \tag{38}$$

For $l=1,\cdots,|\Omega_k|$, we set $\bar{\gamma}_{\tau_l}=\lfloor \bar{obj}^{r'}_{\tau_l}\rfloor-|\mathcal{C}\setminus\mathcal{D}|+1-\sum_{i\in\mathcal{D}}\bar{\beta}_i-\sum_{\omega\in\boldsymbol{\tau}(l-1)}\bar{\gamma}_\omega$, where $\bar{obj}^{r'}_{\tau_l}$ is the objective value of the LP relaxation of (37). Then, (38) is valid for (DR-JCAP). □

Algorithm 6 in Appendix C.5 gives an overview of the branch-and-cut algorithm with probability cuts for (DR-JCAP). Theorem 6 shows that Algorithm 6 solves (DR-JCAP) in finitely many iterations under certain conditions.

THEOREM 6. *If there exists an oracle that solves (SP) to optimality, then Algorithm 6 terminates in finitely many iterations. If $UB<+\infty$, UB is the optimal value of (DR-CAP) and Algorithm 6 obtains an optimal solution $(\boldsymbol{y}^*,\boldsymbol{z}'^*)$ at termination.* □

6. Computational Experiments

We now present computational results for (CAP) and (DR-CAP). Computational experiments were performed using data from an operating room (OR) assignment problem, where a set of surgeries are assigned to operating rooms. Each surgery has a random duration, and each OR has a time limit determined by its work hours. Problem instance generation is discussed in Section 6.1. Section 6.2 provides additional implementation details. The performance of the branch-and-cut algorithm (Algorithm 5) for solving (CAP) is discussed in Section 6.3 and the branch-and-cut algorithm with probability cuts (Algorithm 1) for solving (DR-CAP) is discussed in Section 6.4. Section 6.5 presents the performance of strengthening big-M in (SIP). Section 6.6 compares the out-of-sample performance of the solutions generated from the (DR-CAP) instances with the corresponding (CAP) instances.

6.1. Instance Generation

We used historical surgery duration data from a large public hospital in Beijing, China from January 2015 to October 2015. 5,721 surgery durations for the nine major surgery types are available. For the problem instances, the log-normal distribution with the mean and the standard deviation of the surgery duration (see Appendix E) was used to generate surgery duration samples (i.e. Deng and

Shen (2016)). The samples generated from the log-normal distribution were rounded to the nearest 15 minutes and assigned equal probabilities as in sample average approximation. Eight ($|\mathcal{J}| = 8$) ORs are available to serve $|\mathcal{I}| = 27$ surgeries (close to the maximum number of surgeries in a day) a day. The daily time limit t_j is 10 hours, $\forall j \in \mathcal{J}$. Following Zhang et al. (2018), we let the assignment cost c_{ij} vary in $[0, 16]$, $\forall i \in \mathcal{I}, j \in \mathcal{J}$. The number of surgeries in an OR, ρ_j, is limited to $[3, 5]$, $\forall j \in \mathcal{J}$. We used the number of surgeries and the percentage for each surgery type to calculate the number of surgeries for each surgery type performed in a day. To ensure that (CAP) is always feasible, we added a pseudo OR j' to the set of ORs, which has no quantitative and capacity restrictions. We set the assignment cost $c_{ij'}$ for $i \in \mathcal{I}$ as 27. The sample size $N \in \{500, 1000, 1500\}$ and the level of chance satisfaction $\epsilon \in \{0.12, 0.1, 0.08, 0.06\}$ were used in the (CAP) instance generation. Five instances were generated for each sample size.

6.2. Implementation Details

In our implementation of the branch-and-cut algorithm, we added the violated valid inequalities generated from (39) at the nodes that are at a depth no more than 1. No limit was placed on the number of such inequalities added to the formulation. We observed that it is more time-consuming to find a violated inequality of the type (21). Therefore, we added the violated inequalities from (21) at the nodes that are at a depth no more than 2, and the number of violated inequalities of this type was limited to 15. The valid inequalities are generated until one of the following stopping criteria is met: no cut is available with the violation threshold 10^{-2}, or the number of iterations is up to 100 at the root node of the branch-and-cut tree. As suggested by (Gu et al. 1998) for the cover inequalities, in order to find a violated inequality (39) and (21), we let $\mathcal{D} = \{i \in \mathcal{C} : \hat{y}_{ij} = 1\}$, where $\hat{\boldsymbol{y}}$ is the current LP relaxation optimal solution. At each round of cut generation of the type (39), for each $j \in \mathcal{J}$, multiple violated inequalities might be found. We only added the inequality with the most violated value to the branch-and-cut tree.

The algorithm was implemented in the C programming language using IBM CPLEX solver, version 12.71 callable libraries. A laptop with Intel(R) 2.80 GHz processor and 16 GB RAM was used for computations on a 64-bit computer using the Windows operating system. We turned off the CPLEX presolve procedure and set the number of threads to one for all computations. We used CPLEX callback functions for adding the violated valid inequalities proposed in this paper. For all computations, a priority order for the binary variables in the node selection rule was used. The variables $\boldsymbol{y}$ were given a higher priority than $\boldsymbol{z}$. We used a runtime limit of 10 hours or an optimality tolerance of 1% as our stopping criteria. For instances that could not be solved to meet the stopping criteria, we give the average optimality gap, where the optimality gap is calculated as (UB − LB)/UB, and UB and LB are the upper and lower bound, respectively. We report the solution time (in seconds) for the instances that are solved to optimality within the runtime limit.

The computational results discussed below use the definition described in Section 2.1 to compute $m_j^\omega(k)$ for $j \in \mathcal{J}$ and $\omega, k \in \Omega$. An easier way to compute $m_j^\omega(k)$ is to let $m_j^\omega(k) = \underset{\boldsymbol{y}_j \in \{0,1\}^{|\mathcal{I}|}}{\text{maximize}} \{\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \Big| \sum_{i\in\mathcal{I}} \xi_i^k y_{ij} \le t_j\}$, i.e., ignoring the cardinality constraint in (5). This computation takes less time to compute the big-M coefficients, but lead to larger big-M coefficients. Computational results for this big-M coefficients based implementation of (CAP) are presented in Appendix F. Comparing results from this weaker upper bound with those in Table 1, we see the computational trade-offs resulting from using the weaker upper bound. For easier problems, the average total solution times (the sum of the average time for the big-M coefficients and the branch-and-cut algorithm) are less for the model with a weaker big-M. However, solution times for harder problems improve significantly with the strengthened big-M computation.

6.3. Computational Results for the Branch-and-Cut Algorithm for (CAP)

We now discuss the benefits of adding the valid inequalities proposed in Section 3 to the branch-and-cut algorithm when solving (CAP). The performance of the following four variants is compared:

- CPX: refers to using the branch-and-cut algorithm as implemented in CPLEX to solve (IP) of (CAP).
- Cover-1: refers to adding the single lifted cover inequalities defined in Example (2) to the branch-and-cut algorithm (Algorithm 5) for solving (IP) of (CAP). They are obtained by ignoring the cardinality constraint in the coefficient calculation procedures.
- Cover-2: refers to adding the lifted cover inequalities (39) to the branch-and-cut algorithm (Algorithm 5) for solving (IP) of (CAP).
- Cover-G: refers to adding the global lifted cover inequalities (21) to the branch-and-cut algorithm (Algorithm 5) for solving (IP) of (CAP).

Table 1 reports the average time for the big-M coefficient computations, the cut generation time, the branch-and-cut algorithm time, the average number of nodes, the average number of cuts, and the number of instances solved to optimality for the five generated instances. First we note from Table 1 that the problems become increasingly difficult as the value of ϵ reduces. A possible reason is that for these problems it is more difficult to find a feasible solution satisfying chance constraint. Note that for $\epsilon = 0.06$ to combinatorially explore chance constraint satisfaction for a 500 scenario problem we can violate 30 out of 500 scenarios. This suggests the possibility of requiring a large number of nodes in the branch-and-bound tree in proving infeasibility.

We see from Table 1 that adding the single cover and lifted cover inequalities reduce the average time for the branch-and-cut algorithm by about 55%. This decrease in the computation time can be associated with the reduction in the number of nodes explored in the branch-and-cut algorithm. For $\varepsilon = 0.08$ and $N = 1500$, adding the single and lifted cover inequalities can solve

Table 1 The average CPU time (in seconds) for strengthened big-M coefficients (AvT-M), branch-and-cut algorithm (AvT-B&C) and valid cut generation (AvT-cut), the average number of nodes (# of nodes) and cuts (# of cuts), and the number of solved instances from the five instances (solved) for (CAP) are reported.

ϵ	N	approach	AvT-M	AvT-B&C	AvT-cut	# of nodes	# of cuts	solved
0.12	500	CPX	165.0	52.8	-	1,725	-	5/5
		Cover-1		33.3	1.7	1,446	283	5/5
		Cover-2		47.1	14.7	1,076	300	5/5
		Cover-G		65.5	8.6	1,959	9	5/5
	1000	CPX	641.1	135.8	-	1,827	-	5/5
		Cover-1		66.5	4.4	1,557	348	5/5
		Cover-2		109.8	32.6	1,846	345	5/5
		Cover-G		219.5	37.7	2,847	9	5/5
	1500	CPX	1,439.3	781.4	-	5,094	-	5/5
		Cover-1		659.4	10.9	10,788	563	5/5
		Cover-2		502.4	76.0	3,756	561	5/5
		Cover-G		739.5	101.0	3,715	12	5/5
0.1	500	CPX	165.0	140.7	-	3,477	-	5/5
		Cover-1		122.5	1.2	4,638	210	5/5
		Cover-2		126.0	10.7	3,641	224	5/5
		Cover-G		136.7	10.6	2,926	12	5/5
	1000	CPX	641.1	523.5	-	6,492	-	5/5
		Cover-1		329.2	4.3	5,919	346	5/5
		Cover-2		305.3	29.2	5,832	320	5/5
		Cover-G		481.0	42.8	5,669	12	5/5
	1500	CPX	1,439.3	1,868.9	-	10,308	-	5/5
		Cover-1		995.7	11.7	9,983	657	5/5
		Cover-2		983.6	95.6	7,439	689	5/5
		Cover-G		1,713.0	140.4	8,831	14	5/5
0.08	500	CPX	165.0	816.6	-	29,710	-	5/5
		Cover-1		470.5	1.1	18,226	192	5/5
		Cover-2		360.2	9.1	14,614	201	5/5
		Cover-G		809.9	10.1	28,151	10	5/5
	1000	CPX	641.1	2,375.7	-	31,595	-	5/5
		Cover-1		2,024.8	4.3	29,594	307	5/5
		Cover-2		1,791.4	25.2	22,455	284	5/5
		Cover-G		2,166.4	37.4	28,596	9	5/5
	1500	CPX	1,439.3	4,600.4 [0.03]	-	65,740	-	4/5
		Cover-1		3,095.5+32,104.9*	7.6	76,573	402	5/5
		Cover-2		3,072.9+28,014.4*	55.2	69,798	413	5/5
		Cover-G		3,650.7+28,248.3*	77.4	54,969	10	5/5
0.06	500	CPX	165.0	32,178.1 [0.11]	-	1,588,803	-	1/5
		Cover-1		18,923.0 [0.07]	16.3	1,296,583	184	1/5
		Cover-2		20,497.6 [0.09]	13.3	1,780,324	193	1/5
		Cover-G		16,313.2 [0.11]	10.7	1,607,736	3	1/5
	1000	CPX	641.1	[0.19]	-	588,891	-	0/5
		Cover-1		[0.19]	32.2	514,576	288	0/5
		Cover-2		[0.17]	27.4	531,292	313	0/5
		Cover-G		[0.19]	37.1	562,600	10	0/5
	1500	CPX	1,439.3	[0.25]	-	267,632	-	0/5
		Cover-1		[0.22]	21.6	216,724	456	0/5
		Cover-2		[0.25]	54.9	247,320	431	0/5
		Cover-G		[0.31]	77.5	258,991	6	0/5

"-" in column of *AvT-Cut* and *# of cuts* indicates that no valid cut proposed in this paper is added.

"[·]" in column of *AvT-B&C* means the average optimality gap for instances that cannot be solved to optimality within 10 hours time limit.

"∗" in column of *AvT-B&C* means that *AvT-B&C* is the average time for the solved instances by CPX plus the average time for the other instances.

all instances to optimality within the runtime limit, whereas, CPX can only solve four of the five instances to optimality. We also observe that for $\varepsilon = 0.06$, most of the instances cannot be solved within the runtime limit by all variants. It seems that this level of chance requirement requires a pseudo OR, i.e., the original model for assigning 27 surgeries to the eight operating rooms with $\varepsilon = 0.06$ is infeasible. It makes it hard to decide how many and which surgeries are assigned to the pseudo OR while satisfying the chance constraint with $\varepsilon = 0.06$, and minimizing the total cost. Nevertheless, for these problems, the use of Cover-1 and Cover-2 result in a slightly smaller average optimality gap for most instances at termination. The results also show that Cover-2 has a better performance than Cover-1 in terms of the average time for the 1,500 scenario instances ($\varepsilon = 0.1, 0.08, 0.06$). We find that the big-M computation time is significant for the less difficult instances ($\varepsilon = 0.12, 0.10$). However, for the difficult instances ($\varepsilon = 0.08$), the time required in the branch-and-cut algorithm dominates. The benefits of adding Cover-1 and Cover-2 inequalities are more apparent for these instances, and here the use of Cover-2 saves computation time over Cover-1. For the easier problems ($\varepsilon = 0.10, 0.12$), we observe that typically the number of nodes in the branch-and-cut tree reduces due to the addition of Cover-2 inequalities. However, it does not always translate in a significant reduction of the solution time, and occasionally there is a modest increase in the solution time. Overall, adding Cover-2 inequalities outperforms other variants and yields a more stable performance for most instances.

The use of Cover-G yielded an unfavorable performance for easier instances ($\varepsilon \geq 0.08$). However, for the hardest instance (N=500, $\varepsilon = 0.06$) solved in our implementation, the use of Cover-G gives a slightly better performance when compared with Cover-1 and Cover-2. For some instances, it reduced the number of nodes significantly, while for other instances the number of nodes increased. Even for the hardest solved instance ($\varepsilon = 0.08, N = 1,500$), which took fewer number of nodes ($54,969$ versus $69,798$) when compared to Cover-2 variant, this reduction did not translate into a reduction in the overall solution time ($28,248$ versus $28,014$ seconds). It can be surmised that the linear programming relaxation problems resulting from the addition of these cuts are more time consuming to solve, hence offsetting the benefits from the reduction in branch-and-bound nodes. There are several instances where the use of Cover-G increased the number of nodes. This may be because the addition of these inequalities may be yielding a significantly different node selection path within CPLEX.

6.4. Computational Results for (DR-CAP)

We implemented Algorithm 1 to solve the semi-infinite reformulation (8) of (DR-CAP). Using the empirical probability distribution, we let $\bar{q} := \hat{q}$ (Corollary 1) for the big-M calculations in (8). For (39), we set the coefficient γ as $\delta_{\bar{k}_{\hat{q}^1}}$ (Theorem 9). We used the sample average distribution as the empirical probability distribution. The following variants are considered:

- CPX: refers to using the branch-and-cut algorithm with probability cuts (Algorithm 1) to solve (SIP) of (DR-CAP) without any valid inequalities proposed in this paper.
- Cover-1: refers to adding the Cover-1 inequalities from (CAP) to the branch-and-cut algorithm with probability cuts (Algorithm 1).
- Cover-2: refers to adding the valid inequalities (39) to the branch-and-cut algorithm with probability cuts (Algorithm 1).

We solved the instances generated in Section 6.3 with the Wasserstein set $\mathscr{P}_W$ as the ambiguity set to evaluate the performance of the variants. The sample size $N \in \{500, 1000, 1500\}$, the Wasserstein set radius parameter $\eta \in \{0.1, 0.5, 1\}$, and the level of chance satisfaction $\varepsilon = 0.1$ are used in these instances. Table 2 reports the average time for the branch-and-cut algorithm with probability cuts, the cut generation, the average number of nodes, the average number of cuts, and the number of instances that are solved to optimality from the five generated instances. The solution times for (DR-CAP) instances accounted for the preprocessing time to obtain $\hat{q}$ (Corollary 1). The reported results do not use $\bar{q}$ (Theorem 1) since we found that the additional time required to compute $\bar{q}$ is not offset by any time savings resulting from its use in our test problems. Specifically, we use $m_j^\omega(k_{\hat{q}})$ in all our tests. While developing the probability cut approach presented above, we also implemented the dualization reformulation (DSIP) to solve our test problems.

$$
\text{(DSIP)} \quad \underset{(\boldsymbol{y},\boldsymbol{z})\in\{0,1\}^{|\mathcal{I}||\mathcal{J}|}\times\{0,1\}^{|\mathcal{J}|N},\boldsymbol{\mu}^1,\boldsymbol{\mu}^2,\boldsymbol{\mu}^3,\boldsymbol{\mu}^4}{\text{minimize}} \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} c_{ij}y_{ij}
$$

$$
\begin{aligned}
\text{subject to } & (1b), (1c), (8c) \\
& \mu_j^1 - \eta\mu_j^2 + \sum_{k\in\Omega} p_k^* \mu_{jk}^4 \geq 1-\varepsilon, && \forall j \in \mathcal{J}, \\
& -\mu_j^1 + \mu_{j\omega}^3 + z_{j\omega} \geq 0, && \forall j \in \mathcal{J}, \omega \in \Omega, \\
& ||\boldsymbol{\xi}^\omega - \boldsymbol{\xi}^k||\mu_j^2 - \mu_{j\omega}^3 - \mu_{jk}^4 \geq 0, && \forall j \in \mathcal{J}, \omega, k \in \Omega, \\
& \mu_j^2 \geq 0, && \forall j \in \mathcal{J},
\end{aligned}
$$

where $\boldsymbol{\mu}^1$, $\boldsymbol{\mu}^2$, $\boldsymbol{\mu}^3$, and $\boldsymbol{\mu}^4$ are dual variables for the constraints in (7). We found that none of the test instances could be solved to optimality with a 10 hour CPU time limit. Specifically, the average optimality gap for the 500 scenario instances remained more than 80%, and for the larger instances ($N = 1500$) this approach could not even find a feasible solution. We conjecture that it is because introduction of continuous (dual) variables in a problem that is otherwise pure binary adds significantly to the difficulty in solving the resulting model.

Similar to the case of (CAP), the results in Table 2 show that Cover-2 yields a significant improvement over CPX and Cover-1 in two of the three harder instance sets ($\eta = 0.1$, and $\eta = 1$, $N = 1500$). However, the average performance of Cover-1 is better for the ($\eta = 0.5, N = 1500$)

Table 2 The average CPU time (in seconds) for the branch-and-cut algorithm with probability cuts (AvT-B&CP), valid cut generation (AvT-cut) and distribution separation problem (AvT-SP), the average number of nodes (# of nodes), valid cuts (# of cuts) and probability and feasibility cuts (# of p&f-cuts), and the number of solved instances from the five instances (solved) for (DR-CAP) are reported.

η	N	approach	AvT-B&CP	AvT-cut	AvT-SP	# of nodes	# of cuts	# of p&f-cuts	solved
0.1	500	CPX	272.7	–	61.9	6,277	–	3	5/5
		Cover-1	144.0	2.8	54.4	3,589	410	2	5/5
		Cover-2	147.0	11.5	45.9	3,379	250	2	5/5
	1000	CPX	889.6	–	273.9	9,476	–	2	5/5
		Cover-1	728.6	7.2	336.6	7,274	553	2	5/5
		Cover-2	723.5	32.1	285.1	7,606	349	2	5/5
	1500	CPX	3,051.7	–	880.9	14,650	–	2	5/5
		Cover-1	2,956.2	12.4	779.4	15,282	644	4	5/5
		Cover-2	1,658.0	96.0	700.6	7,343	716	2	5/5
0.5	500	CPX	648.6	–	101.0	18,426	–	18	5/5
		Cover-1	290.5	1.3	69.0	8,696	226	12	5/5
		Cover-2	319.5	10.8	69.4	12,446	250	14	5/5
	1000	CPX	1,390.0	–	403.4	12,095	–	12	5/5
		Cover-1	1,021.5	5.6	346.5	12,148	447	8	5/5
		Cover-2	884.0	33.0	397.5	10,720	373	9	5/5
	1500	CPX	4,957.9	–	1,080.9	37,824	–	12	5/5
		Cover-1	4,003.9	13.6	909.4	39,783	706	9	5/5
		Cover-2	4,598.5	100.9	1,088.9	38,227	759	14	5/5
1	500	CPX	826.9	–	104.0	31,989	–	34	5/5
		Cover-1	501.8	1.4	99.8	20,067	233	28	5/5
		Cover-2	775.8	11.1	105.2	32,536	247	30	5/5
	1000	CPX	2,987.3	–	502.2	47,221	–	30	5/5
		Cover-1	3,173.8	6.2	483.6	42,202	482	30	5/5
		Cover-2	2,173.9	36.3	484.7	33,143	414	28	5/5
	1500	CPX	8,091.2	–	1,268.2	61,039	–	28	5/5
		Cover-1	8,088.4	12.3	1,294.6	63,029	647	27	5/5
		Cover-2	4,962.4	98.9	1,167.3	44,377	716	26	5/5

"–" in column of *AvT-cut* and *# of cuts* indicates that no valid cut proposed in this paper is added.

instances. A comparison of the results in Table 1 and 2 shows that the time required to solve (DR-CAP) is approximately (at most) four times the time required to solve (CAP). Moreover, the average number of probability and feasibility cuts required to solve these models is typically less than 30, though this number grows with the Wasserstein radius. This is expected since with increasing radius, the Wasserstein ambiguity set increases in size, resulting in more solutions being generated in the algorithm that are infeasible with respect to the ambiguity set. The average number of nodes required to solve the models also increases with the Wasserstein radius (up to 5 times). Note that the branch-and-cut tree from the incumbent problem is used to warm-start the solution of the new problem after a probability cut is added.

Table 3 The CPU time (in seconds) for branch-and-cut algorithm with probability cuts (time), and distribution separation problem (time-SP), the number of nodes (# of nodes), and probability and feasibility cuts (# of p&f-cuts) for (DR-CAP) are reported.

instance	time		time-SP		# of nodes		# of p&f-cuts	
	CPX	CPX-UM	CPX	CPX-UM	CPX	CPX-UM	CPX	CPX-UM
1500-1	12,350.4	8,354.3	1,305.1	1,419.1	63,691	93,617	38	44
1500-2	9,490.7	8,710.0	1,515.7	1,309.2	63,699	85,382	30	24
1500-3	5,331.2	5,564.0	1,250.1	1,017.6	55,893	36,640	22	12
1500-4	5,453.2	5,272.6	898.2	885.2	39,918	25,362	10	12
1500-5	7,830.7	8,627.2	1,372.0	1,560.4	81,992	71,336	38	32
Average	8,091.2	7,305.6	1,268.2	1,238.3	61,039	62,467	28	25

6.5. Performance of Big-M Improvements from Ambiguity Set Information

The results in Table 2 were obtained by using the nominal distribution to compute the big-M coefficients. We now discuss our computational experience with the possibility of big-M tightening due to Theorem 1 and Corollary 1. While we found that the solution time required by the linear programs in Theorem 1 is not justified, we did find computational value in using Corollary 1 as part of our implementation. This is particularly true for the harder problems. In this section, we present the results for the harder problems that are generated for $\eta = 1$ and $N = 1500$. Five instances are considered. These instances are labeled as $N - \#$, where # denotes the instance number. We compare the performance of the following approaches:

- CPX: is described in Section 6.4.
- CPX-UM: refers to using new $\bar{q}$ as valid inequalities and adding these inequalities to CPX.

For CPX-UM, we update $\hat{q}$ defined in Corollary 1 as new $\{p_\omega\}_{\omega\in\Omega}$ becomes available in the probability cuts. We set $\bar{q} = \hat{q}$ and add constraints (8c) as valid inequalities. We needed to do this because CPLEX does not allow for changing in the coefficients of the original constraints once a branch-and-bound tree is built. We need to keep the original branch-and-bound tree when solving the problem. For each j, multiple violated inequalities might be found. We only added the inequality with the most violated value to the branch-and-cut tree. In the current implementation, it is done only once when a new probability distribution becomes available for each j. Table 3 reports the solution time for the branch-and-cut algorithm with probability cuts and the separation problem, the number of nodes, the number of probability and feasible cuts. Note that the time for the valid inequality generation was negligible, and therefore not included in this table.

Specially, for the model with the largest value of η ($\eta = 1$), where Algorithm 1 generates many probability cuts, we observe from Table 3 that CPX-UM provides better performance than CPX in the solution time in three of the five instances. The average solution time is decreased by about 10%. The solution time is significantly lower for one instance (1500-1), whereas that for other instances it is similar. Compared with CPX, CPX-UM has a reduced total number of nodes for three instances, whereas the number of nodes increases in the other two instances. The increase/decrease in the number of nodes does not necessarily imply a corresponding increase/decrease in solution time. This may be because the node linear programs may vary in difficulty. We could not find a setting for combining the valid inequalities discussed in this section with (39) to improve the performance of Cover-2 implementation discussed in Section 6.4. We attribute this to the fact that CPLEX does not allow us to change the coefficients of the original data with the progression of the algorithm.

6.6. Out-of-Sample Performance of (DR-CAP) Solutions

The chance constraints used to specify (CAP) and (DR-CAP) are generated using a finite number of samples drawn from a probability distribution. The goal of this section is to evaluate the 'true chance satisfaction' of the solution generated from this finite sample approximation. For this purpose, the integer solutions obtained from (CAP) and (DR-CAP) were evaluated using a large number (1,500,000) of scenarios generated from the log-normal distribution. We used five instances each for the sample sizes $N \in \{50, 100, 500, 1000, 1500\}$ for the (CAP) and (DR-CAP) solutions. The (DR-CAP) solutions were generated using the Wasserstein radius parameter $\eta \in \{0.1, 0.5, 1\}$. All evaluations were performed for $\varepsilon = 0.1$ in the chance constraint model. Table 4 gives the average total cost, the average overtime probability, the worst-case overtime probability, the average overtime (minutes), and 85%, 95%, 99% overtime quantiles (minutes) for (CAP) and (DR-CAP) solutions.

Table 4 The average total cost (Avg-cost), the average overtime probability (Avg-prob), the worst-case overtime probability (Worst-prob), the average overtime (Avg-over) (in minutes), and 85%, 95%, 99% quantiles (in minutes) for (CAP) (a) and (DR-CAP) (b) are reported.

(a)

N	Avg-cost	Avg-prob	Worst-prob	Avg-over	85%	95%	99%
50	68.6	0.075	0.144	6.8	0.0	42.4	150.8
100	69.9	0.071	0.123	6.2	0.0	36.4	150.0
500	69.9	0.070	0.122	6.1	0.0	36.4	150.4
1000	70.2	0.069	0.122	6.1	0.0	37.9	150.0
1500	70.7	0.067	0.117	5.8	0.0	34.5	148.1

The results in Table 4 show that the average and worst-case out-of-sample overtime probability decrease with increasing sample size in (CAP) and the radius of the Wasserstein set (η) in (DR-CAP). The same is observed for the average overtime, and the overtime 85% and 95% quantiles. Consequently, using the largest instance ($N = 1500$) and/or larger η solutions are viable alternatives when out-of-sample chance constraint satisfaction is of concern. We observe that the decrease in the worst-case out-of-sample chance constraint satisfaction probability is more modest with increasing sample sizes. For example, the solutions from the instances with $N = 1000$ give a worst

(b)

η	N	Avg-cost	Avg-prob	Worst-prob	Avg-over	85%	95%	99%
	50	69.6	0.071	0.144	6.4	0.0	37.9	149.3
	100	70.4	0.070	0.124	6.1	0.0	35.3	147.4
0.1	500	70.3	0.068	0.122	6.0	0.0	36.8	147.4
	1000	70.7	0.066	0.122	5.8	0.0	33.8	148.5
	1500	71.0	0.067	0.117	5.9	0.0	35.6	147.4
	50	70.9	0.071	0.121	6.3	0.0	39.8	148.9
	100	70.4	0.069	0.123	6.1	0.0	35.3	147.4
0.5	500	71.7	0.066	0.121	5.8	0.0	32.3	147.8
	1000	71.2	0.065	0.088	5.6	0.0	31.9	149.3
	1500	72.0	0.065	0.096	5.6	0.0	29.6	148.1
	50	71.5	0.068	0.121	5.9	0.0	34.1	147.4
	100	71.5	0.068	0.130	6.0	0.0	34.1	145.9
1	500	72.8	0.065	0.121	5.6	0.0	28.9	148.1
	1000	73.1	0.064	0.089	5.5	0.0	26.6	149.3
	1500	73.3	0.064	0.082	5.4	0.0	24.4	148.1

probability of 0.122, and the instances with $N = 1500$ have a worst probability of 0.117. However, this worst-case out-of-sample chance constraint satisfaction probability decreases more significantly with increasing η. For example, the instances with $N = 1000$ and $\eta = 0.1$ have the worst-case out-of-sample probability of 0.122, and the instances with $N = 1000$ and $\eta = 0.5$ have the worst-case probability of 0.088, i.e., in this case the solutions generated in all the instances satisfy the chance constraint with probability 0.1. We also note that when the sample size is smaller (N=50, 100, 500), even though the average out-of-sample overtime probability is less than 0.1 for the (CAP) solutions, the worst-case out-of-sample overtime probability can be significantly greater. For example even for $\eta = 0.1$ this probability is 0.144 for $N = 50$. The solutions for the (DR-CAP) models that satisfy the chance constraint have a modest increase in cost. This cost increases from 70.2 in the (CAP) model to 71.2 in the (DR-CAP) model when using $N = 1000$ and $\eta = 0.5$. Similar observations are made for (CAP) and (DR-CAP) problem instances with $N = 1500$. It is also interesting to observe that the worst-case probability for problem instances with $N = 500$ did not change significantly (0.122, 0.121, 0.121) for $\eta = 0.1, 0.5$ and 1.0, despite the solutions becoming costlier. Consequently, increasing both the sample size and the size of the ambiguity set may be important to ensure the worst-case probability satisfaction. However, it is important to note that for the chance constraint problems computational cost increases rapidly with the sample size, while the increase in the computational cost for the (DR-CAP) models is modest (only a constant factor). As observed, the average and worst-case overtime probability decrease with an increase in Wasserstein radius η. However, this comes at the expense of an increase in the objective value. Thus the choice of Wasserstein radius also plays a role in the model whose solution would finally be used. A data-driven bisection approach can be used to choose η. In this approach, one may start with $\eta \in \{0, \eta^0_{\max}\}$, where $\eta^0_{\max}$ is the largest Wasserstein radius possible on the finite support, a quantity that can be estimated from the available samples. Now on the given sample, assuming monotonicity in the out-of-sample performance, we may iteratively reduce $\eta^{k+1}_{\max} := \eta^k_{\max}/2$. The out-of-sample performance of the solution obtained for each value of $\eta^k_{\max}$ is evaluated, and the process stops when this performance does not meet the desired out-of-sample chance satisfaction.

7. Concluding Remarks

The use of big-M calculations and strong inequalities developed in this paper resulted in the chance-constrained assignment and distributionally robust chance-constrained assignment model solutions with a modest number ($N = 1500$) of scenarios. These models remain difficult to solve when they are infeasible or nearly feasible. The solution time for the models grows rapidly with increasing sample size. However, the solution time for the distributionally robust chance-constrained models appears to be only a constant factor of the time required to solve the chance constraint version. The use of a modest number of samples ($N = 1000$) and an appropriate choice of the radius of the Wasserstein set provide a solution that achieves an out-of-sample chance satisfaction. This out-of-sample performance is not possible for the solutions generated from solving the chance constraint problem specified using a modest number of samples. The use of the Wasserstein ambiguity set of an appropriate radius allows us to have the true probability distribution of the random parameters with a greater probability.

Acknowledgments

The authors thank three anonymous referees and the associate editor for their insightful and helpful comments to improve the manuscript. This research of the first two authors were partially supported by the National Natural Science Foundation of China (NSFC) grants 71972012, 91746210 and 71964034. The research of the last author was partially supported by the NSF grant CMMI-1763035. This paper also benefited from a discussion with Professor Simge Küçükyavuz and Hamed Rahimian.

References

Abdi, A., Fukasawa, R., 2016. On the mixing set with a knapsack constraint. Mathematical Programming 157 (1), 191–217.

Bansal, M., Huang, K.-L., Mehrotra, S., 2018. Decomposition algorithms for two-stage distributionally robust mixed binary programs. SIAM Journal on Optimization 28 (3), 2360–2383.

Ben-Tal, A., Den Hertog, D., De Waegenaere, A., Melenberg, B., Rennen, G., 2013. Robust solutions of optimization problems affected by uncertain probabilities. Management Science 59 (2), 341–357.

Beraldi, P., Bruni, M. E., 2010. An exact approach for solving integer problems under probabilistic constraints with random technology matrix. Annals of Operations Research 177 (1), 127–137.

Bertsimas, D., Demir, R., 2002. An approximate dynamic programming approach to multidimensional knapsack problems. Management Science 48 (4), 550–565.

Calafiore, G. C., Campi, M. C., 2006. The scenario approach to robust control design. IEEE Transactions on Automatic Control 51 (5), 742–753.

Chen, W., Sim, M., Sun, J., Teo, C.-P., 2010. From cvar to uncertainty set: Implications in joint chance-constrained optimization. Operations Research 58 (2), 470–485.

Chen, Z., Kuhn, D., Wiesemann, W., 2018. Data-driven chance constrained programs over wasserstein balls. arXiv preprint arXiv:1809.00210.

Cheng, J., Delage, E., Lisser, A., 2014. Distributionally robust stochastic knapsack problem. SIAM Journal on Optimization 24 (3), 1485–1506.

Cohen, M. C., Keller, P. W., Mirrokni, V., Zadimoghaddam, M., 2019. Overcommitment in cloud services: Bin packing with chance constraints. Management Science 65 (7), 3255–3271.

Delage, E., Ye, Y., 2010. Distributionally robust optimization under moment uncertainty with application to data-driven problems. Operations Research 58 (3), 595–612.

Deng, Y., Shen, S., 2016. Decomposition algorithms for optimizing multi-server appointment scheduling with chance constraints. Mathematical Programming 157 (1), 245–276.

Deng, Y., Shen, S., Denton, B., 2019. Chance-constrained surgery planning under conditions of limited and ambiguous data. INFORMS Journal on Computing 31 (3), 559–575.

Esfahani, P. M., Kuhn, D., 2018. Data-driven distributionally robust optimization using the wasserstein metric: Performance guarantees and tractable reformulations. Mathematical Programming 171 (1-2), 115–166.

Gu, Z., Nemhauser, G. L., Savelsbergh, M. W., 1998. Lifted cover inequalities for 0-1 integer programs: Computation. INFORMS Journal on Computing 10 (4), 427–437.

Hanasusanto, G. A., Roitch, V., Kuhn, D., Wiesemann, W., 2017. Ambiguous joint chance constraints under mean and dispersion information. Operations Research 65 (3), 751–767.

Ji, R., Lejeune, M. A., 2020. Data-driven distributionally robust chance-constrained optimization with wasserstein metric. Journal of Global Optimization, 1–33.

Jiang, R., Guan, Y., 2016. Data-driven chance constrained stochastic program. Mathematical Programming 158 (1-2), 291–327.

Jiang, R., Guan, Y., 2018. Risk-averse two-stage stochastic program with distributional ambiguity. Operations Research 66 (5), 1390–1405.

Kaparis, K., Letchford, A. N., 2008. Local and global lifted cover inequalities for the 0–1 multidimensional knapsack problem. European Journal of Operational Research 186 (1), 91–103.

Klabjan, D., Nemhauser, G. L., Tovey, C., 1998. The complexity of cover inequality separation. Operations Research Letters 23 (1-2), 35–40.

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

Liu, X., Kılınç-Karzan, F., Küçükyavuz, S., 2019. On intersection of two mixing sets with applications to joint chance-constrained programs. Mathematical Programming 175, 29–68.

Liu, X., Küçükyavuz, S., Luedtke, J., 2016. Decomposition algorithms for two-stage chance-constrained programs. Mathematical Programming 157 (1), 219–243.

Luedtke, J., 2014. A branch-and-cut decomposition algorithm for solving chance-constrained mathematical programs with finite support. Mathematical Programming 146 (1-2), 219–244.

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

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

Luo, F., Mehrotra, S., 2019. Decomposition algorithm for distributionally robust optimization using wasserstein metric with an application to a class of regression models. European Journal of Operational Research 278 (1), 20–35.

Mehrotra, S., Papp, D., 2014. A cutting surface algorithm for semi-infinite convex programming with an application to moment robust optimization. SIAM Journal on Optimization 24 (4), 1670–1697.

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

Peng, C., Delage, E., Li, J., 2020. Probabilistic envelope constrained multiperiod stochastic emergency medical services location model and decomposition scheme. Transportation Science 54 (6), 1471–1494.

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

Rahimian, H., Mehrotra, S., 2019. Distributionally robust optimization: A review. arXiv preprint arXiv:1908.05659.

Shapiro, A., Dentcheva, D., Ruszczyński, A., 2009. Lectures on stochastic programming: modeling and theory. SIAM.

Song, Y., Luedtke, J. R., 2013. Branch-and-cut approaches for chance-constrained formulations of reliable network design problems. Mathematical Programming Computation 5 (4), 397–432.

Song, Y., Luedtke, J. R., Küçükyavuz, S., 2014. Chance-constrained binary packing problems. INFORMS Journal on Computing 26 (4), 735–747.

Tanner, M. W., Ntaimo, L., 2010. 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.

van Ackooij, W., Frangioni, A., de Oliveira, W., 2016. Inexact stabilized benders decomposition approaches with application to chance-constrained problems with finite support. Computational Optimization and Applications 65 (3), 637–669.

Wang, S., Li, J., Mehrotra, S., 2021. Chance-constrained multiple bin packing problem with an application to operating room planning. INFORMS Journal on Computing.

Wang, S., Li, J., Peng, C., 2017. Distributionally robust chance-constrained program surgery planning with downstream resource. In: 2017 International Conference on Service Systems and Service Management. IEEE, pp. 1–6.

Wiesemann, W., Kuhn, D., Sim, M., 2014. Distributionally robust convex optimization. Operations Research 62 (6), 1358–1376.

Wu, H.-H., Küçükyavuz, S., 2017. Chance-constrained combinatorial optimization with a probability oracle and its application to probabilistic partial set covering. arXiv preprint arXiv:1708.02505.

Xie, W., 2019. On distributionally robust chance constrained programs with wasserstein distance. Mathematical Programming, 1–41.

Xie, W., Ahmed, S., 2018. On quantile cuts and their closure for chance constrained optimization problems. Mathematical Programming 172 (1-2), 621–646.

Zemel, E., 1989. Easily computable facets of the knapsack polytope. Mathematics of Operations Research 14 (4), 760–764.

Zhang, Y., Jiang, R., Shen, S., 2018. Ambiguous chance-constrained binary programs under mean-covariance information. SIAM Journal on Optimization 28 (4), 2922–2944.

Zhang, Z., Denton, B. T., Xie, X., 2020. Branch and price for chance-constrained bin packing. INFORMS Journal on Computing.

Zhao, M., Huang, K., Zeng, B., 2017. A polyhedral study on chance constrained program with random right-hand side. Mathematical Programming 166 (1-2), 19–64.

Zymler, S., Kuhn, D., Rustem, B., 2013. Distributionally robust joint chance constraints with second-order moment information. Mathematical Programming 137 (1-2), 167–198.

Appendix A: Lifted Cover Inequality

We now provide coefficient calculations for a lifted cover inequality that is valid for $conv(\mathcal{F}_{j\omega})$.

THEOREM 7. *The lifted cover inequality*

$$\sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\setminus\mathcal{C}} \alpha_i y_{ij} + \sum_{i\in\mathcal{D}} \beta_i y_{ij} + \gamma(z_{j\omega}-1) \le |\mathcal{C}\setminus\mathcal{D}| + \sum_{i\in\mathcal{D}} \beta_i - 1 \tag{39}$$

is valid for $conv(\mathcal{F}_{j\omega})$ *if*

$$\gamma = \underset{\boldsymbol{y}_j\in\{0,1\}^{|\mathcal{I}|}\cap\mathcal{Y}_j}{\text{maximize}} \sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\setminus\mathcal{C}} \alpha_i y_{ij} + \sum_{i\in\mathcal{D}} \beta_i y_{ij} - |\mathcal{C}\setminus\mathcal{D}| - \sum_{i\in\mathcal{D}} \beta_i + 1. \tag{40}$$

Furthermore, if $|\mathcal{C}| \le \rho_j + 1$*, (39) is facet-defining for* $conv(\mathcal{F}_{j\omega})$*.*

Proof. When $z_{j\omega}=1$, (39) is valid for $conv(\mathcal{F}_{j\omega})$ because of Lemma 2. When $z_{j\omega}=0$, due to the definition of γ, (39) is also valid for $conv(\mathcal{F}_{j\omega})$. Thus, (39) is valid for $conv(\mathcal{F}_{j\omega})$.

Consider the following $|\mathcal{I}|+1$ feasible points of $conv(\mathcal{F}_{j\omega})$: when $z_{j\omega}=1$, there exists $|\mathcal{I}|$ feasible points of $conv(\mathcal{F}_{j\omega})$ that are affinely independent and satisfy (39) at equality based on the Lemma 2; when $z_{j\omega}=0$, let $\boldsymbol{y}_j$ be the optimal solution of (40). These $|\mathcal{I}|+1$ feasible points satisfy (39) at equality and are affinely independent. Thus, (39) is facet-defining for $conv(\mathcal{F}_{j\omega})$. □

By restricting the feasible region of $\boldsymbol{y}_j$ in (40) using the chance constraints (1d), we obtain a stronger valid inequality for (CAP) in Theorem 8.

THEOREM 8. *For* $k\in\Omega\setminus\{\omega\}$*, let*

$$\delta_k = \underset{\boldsymbol{y}_j\in\{0,1\}^{|\mathcal{I}|}\cap\mathcal{Y}_j}{\text{maximize}} \sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\setminus\mathcal{C}} \alpha_i y_{ij} + \sum_{i\in\mathcal{D}} \beta_i y_{ij} - |\mathcal{C}\setminus\mathcal{D}| - \sum_{i\in\mathcal{D}} \beta_i + 1 \tag{41a}$$

$$\text{subject to } \sum_{i\in\mathcal{I}} \xi_i^k y_{ij} \le m_j^k(k). \tag{41b}$$

Sort δ_k *such that* $\delta_{\bar{k}_1} \le \ldots \le \delta_{\bar{k}_{|\Omega|-1}}$*. Let* $q^1 := \min\left\{l \,\middle|\, \sum_{j=1}^{l} p_{\bar{k}_j} > \varepsilon\right\}$*, then the inequality (39) is valid for (CAP), where* $\gamma = \delta_{\bar{k}_{q^1}}$*.*

Proof. Let

$$\gamma = \underset{\boldsymbol{y}_j\in\{0,1\}^{|\mathcal{I}|}\cap\mathcal{Y}_j}{\text{maximize}} \sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\setminus\mathcal{C}} \alpha_i y_{ij} + \sum_{i\in\mathcal{D}} \beta_i y_{ij} - |\mathcal{C}\setminus\mathcal{D}| - \sum_{i\in\mathcal{D}} \beta_i + 1 \tag{42a}$$

$$\text{subject to } \sum_{k\in\Omega\setminus\{\omega\}} p_k \mathbb{1}\left(\sum_{i\in\mathcal{I}} \xi_i^k y_{ij} \le t_j\right) \ge 1-\varepsilon. \tag{42b}$$

$\boldsymbol{y}_j$ satisfies the chance constraint (1d) and $z_\omega = 0$ for computing γ, the inequality (39) is valid for (CAP).

Let $\hat{\boldsymbol{y}}_j$ be an optimal solution of (42). Then, there exists at least one $k' \in \{\bar{k}_1,\ldots,\bar{k}_{q^1}\}$ such that $\sum_{i\in\mathcal{I}} \xi_i^{k'} \hat{y}_{ij} \le t_j$. Otherwise, if $\sum_{i\in\mathcal{I}} \xi_i^k \hat{y}_{ij} > t_j$ for all $k\in\{\bar{k}_1,\ldots,\bar{k}_{q^1}\}$, then $\sum_{k\in\{\bar{k}_1,\ldots,\bar{k}_{q^1}\}} p_k \mathbb{1}\left(\sum_{i\in\mathcal{I}} \xi_i^k \hat{y}_{ij} > t\right) > \varepsilon$, which indicates that (42b) is violated by $\hat{\boldsymbol{y}}_j$. Therefore, $\hat{\boldsymbol{y}}_j$ is a feasible solution of (41) for $k=k'$. We have $\delta_{\bar{k}_{q^1}} \ge \delta_{k'} \ge \gamma$, and (39) is a valid inequality for (CAP) when $\gamma = \delta_{\bar{k}_{q^1}}$. □

We further restrict the feasible region of $\boldsymbol{y}_j$ in (40) by using (2b) to obtain a stronger valid inequality for (DR-CAP) in the following theorem.

THEOREM 9. *For $k \in \Omega\backslash\{\omega\}$, let δ_k be defined as in Theorem 8, and sort δ_k such that $\delta_{\bar{k}_1} \leq \ldots \leq \delta_{\bar{k}_{|\Omega|-1}}$. Let $\bar{q}^1 := \min\{l \mid \sup_{\boldsymbol{p}\in\mathscr{P}} \sum_{j=1}^{l} p_{\bar{k}_j} > \varepsilon\}$. Then, the inequality (39) is valid for (DR-CAP) when $\gamma = \delta_{\bar{k}_{\bar{q}^1}}$. Moreover, if $\{\hat{p}_\omega\}_{\omega\in\Omega} \in \mathscr{P}$, let $\hat{q}^1 := \min\{l \mid \sum_{j=1}^{l} \hat{p}_{\bar{k}_j} > \varepsilon\}$. Then, $\hat{q}^1 \geq \bar{q}^1$ and the inequality (39) is valid for (DR-CAP) when $\gamma = \delta_{\bar{k}_{\hat{q}^1}}$.*

Proof. Let

$$\gamma = \underset{\boldsymbol{y}_j\in\{0,1\}^{|\mathcal{I}|}\cap\mathcal{Y}_j}{\text{maximize}} \sum_{i\in\mathcal{C}\backslash\mathcal{D}} y_{ij} + \sum_{i\in\mathcal{I}\backslash\mathcal{C}} \alpha_i y_{ij} + \sum_{i\in\mathcal{D}} \beta_i y_{ij} - |\mathcal{C}\backslash\mathcal{D}| - \sum_{i\in\mathcal{D}} \beta_i + 1 \tag{43a}$$

$$\text{subject to } \inf_{\boldsymbol{p}\in\mathscr{P}} \sum_{k\in\Omega\backslash\{\omega\}} p_k \mathbb{1}\left(\sum_{i\in\mathcal{I}} \xi_i^k y_{ij} \leq t_j\right) \geq 1-\varepsilon. \tag{43b}$$

$\boldsymbol{y}_j$ satisfies the chance constraint (2b) and $z_\omega = 0$ for computing γ, (39) is valid for (DR-CAP).

Let $\hat{\boldsymbol{y}}_j$ be an optimal solution of (43). Then, $\sum_{i\in\mathcal{I}} \xi_i^{k'} \hat{y}_{ij} \leq t_j$ for at least one $k' \in \{\bar{k}_1, \ldots, \bar{k}_{\bar{q}}\}$. Otherwise, if $\sum_{i\in\mathcal{I}} \xi_i^k \hat{y}_{ij} > t_j$ for all $k \in \{\bar{k}_1, \ldots, \bar{k}_{\bar{q}}\}$, we have $\sup_{\boldsymbol{p}\in\mathscr{P}} \sum_{k\in\{\bar{k}_1,\ldots,\bar{k}_{\bar{q}}\}} p_k \mathbb{1}\left(\sum_{i\in\mathcal{I}} \xi_i^k \hat{y}_{ij} > t\right) > \varepsilon$, which indicates that (43b) is violated by $\hat{\boldsymbol{y}}_j$. Therefore, $\hat{\boldsymbol{y}}_j$ is a feasible solution of (41) for $k = k'$. We have $\delta_{\bar{k}_{\bar{q}}} \geq \delta_{k'} \geq \gamma$, then (39) is a valid inequality for (DR-CAP) when $\gamma = \delta_{\bar{k}_{\bar{q}}}$. Since $\sup_{\boldsymbol{p}\in\mathscr{P}} \sum_{j=1}^{\hat{q}^1} p_{\bar{k}_j} \geq \sum_{j=1}^{\hat{q}^1} \hat{p}_{\bar{k}_j} > \varepsilon$, we have $\hat{q}^1 \geq \bar{q}^1$, which implies $\delta_{\bar{k}_{\hat{q}^1}} \geq \delta_{\bar{k}_{\bar{q}^1}} \geq \gamma$, and (39) is a valid inequality for (DR-CAP) when $\gamma = \delta_{\bar{k}_{\hat{q}^1}}$. □

Appendix B: Proof of Propositions and Theorems

B.1. Proof of Proposition 1

Let $\boldsymbol{y}_j^*$ be an optimal solution of (4). Then, there exists at least one $k' \in \{k_1, \ldots, k_q\}$ such that $\sum_{i\in\mathcal{I}} \xi_i^{k'} y_{ij}^* \leq t_j$. Otherwise, we have $\sum_{i\in\mathcal{I}} \xi_i^k y_{ij}^* > t_j$, for $k \in \{k_1, \ldots, k_q\}$. Since $\sum_{j=1}^{q} p_{k_j} > \varepsilon$, the inequality $\mathbb{P}\left\{\sum_{i\in\mathcal{I}} \xi_i y_{ij}^* \leq t_j\right\} \geq 1-\varepsilon$ is violated. This is a contradiction. Therefore, $\boldsymbol{y}_j^*$ is a feasible solution of (5) with $k = k'$. Then $m_j^\omega(k_{q+1}) \geq \bar{M}_j^\omega$, $m_j^\omega(k_{q+1})$ is an upper bound for $\bar{M}_j^\omega$. □

B.2. Proof of Theorem 1

For $j \in \mathcal{J}$ and $\omega \in \Omega$, let $\hat{M}_j^\omega := \underset{\boldsymbol{y}_j\in\{0,1\}^{|\mathcal{I}|}}{\text{maximize}}\{\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} \,\big|\, \inf_{\mathbb{P}\in\mathscr{P}} \mathbb{P}\{\sum_{i\in\mathcal{I}} \xi_i y_{ij} \leq t_j\} \geq 1-\varepsilon,\ \boldsymbol{y}_j \in \mathcal{Y}_j\}$. We show that $m_j^\omega(k_{\bar{q}})$ is an upper bound for $\hat{M}_j^\omega$. Let y_{ij}^* be an optimal solution of the above maximization problem, there exists at least one $k' \in \bar{\Omega} := \{1, \cdots, \bar{q}\}$ such that $\sum_{i\in\mathcal{I}} \xi_i^{k'} y_{ij}^* \leq t_j$. Otherwise, $\sum_{i\in\mathcal{I}} \xi_i^k y_{ij}^* > t_j$ for $k \in \bar{\Omega}$, we have $\inf_{\mathbb{P}\in\mathscr{P}} \mathbb{P}\left\{\sum_{i\in\mathcal{I}} \xi_i y_{ij}^* \leq t_j\right\} = \inf_{\boldsymbol{p}\in\mathscr{P}} \sum_{\omega\in\Omega\backslash\bar{\Omega}} p_\omega \mathbb{1}\left(\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij}^* \leq t_j\right) \leq \inf_{\boldsymbol{p}\in\mathscr{P}} \sum_{\omega\in\Omega\backslash\bar{\Omega}} p_\omega = \inf_{\boldsymbol{p}\in\mathscr{P}} \left(1 - \sum_{\omega\in\bar{\Omega}} p_\omega\right) = 1 - \sup_{\boldsymbol{p}\in\mathscr{P}} \sum_{\omega\in\bar{\Omega}} p_\omega < 1-\varepsilon$, which is a contradiction. Thus, $m_j^\omega(k_{\bar{q}}) \geq \hat{M}_j^\omega$. Therefore, (DR-CAP) can be rewritten as (8). □

B.3. Proof of Proposition 2

Let $(\boldsymbol{y}, \boldsymbol{z})$ be a feasible solution of the relaxation problem of the binary bilinear reformulation of (DR-CAP). We have $\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij}(z_j^\omega - 1) - m_j^\omega(k_{\bar{q}})(z_j^\omega - 1) = (z_j^\omega - 1)(\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} - m_j^\omega(k_{\bar{q}}))$. If $m_j^\omega(k_{\bar{q}}) \geq \bar{m}'_{j\omega}$, $\sum_{i\in\mathcal{I}} \xi_j^\omega y_{ij} m_j^\omega(k_{\bar{q}}) \leq 0$, which implies that $(z_{j\omega} - 1)\left(\sum_{i\in\mathcal{I}} \xi_j^\omega y_{ij} m_j^\omega(k_{\bar{q}})\right) \geq 0$. Consequently, $\sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} + m_j^\omega(k_{\bar{q}})(z_j^\omega - 1) \leq \sum_{i\in\mathcal{I}} \xi_i^\omega y_{ij} z_j^\omega \leq m_j^\omega(\omega) z_j^\omega$ holds. Therefore, $(\boldsymbol{y}, \boldsymbol{z})$ is a feasible solution of the relaxation problem of (8). The proof can be similarly extend to (CAP). □

B.4. Proof of Proposition 3

The set $\mathcal{H} = \bigcap_{j\in\mathcal{J}}\{(\boldsymbol{y},\boldsymbol{z})|(\boldsymbol{y}_j,\boldsymbol{z}_j)\in\mathcal{G}_j\}$ implies that $\mathcal{H}\subseteq\mathcal{G}_j$. Thus, if an inequality is valid for $conv(\mathcal{G}_j)$, then it is also valid for $conv(\mathcal{H})$. If an inequality is facet-defining for $conv(\mathcal{G}_j)$, then there exits $|\mathcal{I}|+N$ affinely independent points that satisfy this inequality at equality. Because this inequality does not have coefficients with respect to a pair of $(\boldsymbol{y}_{j_1},\boldsymbol{z}_{j_1})$ for $j_1\in\mathcal{J}$ and $j_1\neq j$, we can extend the $|\mathcal{I}|+N$ affinely independent points to a set of $|\mathcal{I}|\times|\mathcal{J}|+|\mathcal{J}|\times N$ affinely independent points by appropriately setting the values of $(\boldsymbol{y}_{j_1},\boldsymbol{z}_{j_1})$ for each $j_1\in\mathcal{J}$ and $j_1\neq j$. □

B.5. Proof of Proposition 4

The inequality (11) is valid for (10) based on the definition of $\mathcal{C}$.

Consider the following $|\mathcal{C}\backslash\mathcal{D}|$ feasible points of (10): for $k\in\mathcal{C}\backslash\mathcal{D}$, set $y_{ij}=1,\forall i\in\mathcal{C}\backslash\{\mathcal{D}\cup k\}$, $y_{ij}=0,\forall i\in k\bigcup(\mathcal{I}\backslash\mathcal{C})$, and $y_{ij}=1,\forall i\in\mathcal{D}$; These $|\mathcal{C}\backslash\mathcal{D}|$ points are affinely independent and satisfy (11) at equality. When $|\mathcal{C}|\le\rho_j+1$, these $|\mathcal{C}\backslash\mathcal{D}|$ points are feasible. □

B.6. Proof of Lemma 1

Suppose that there exists $\hat{\boldsymbol{y}}_j$ that serves as a member of the set $\{\boldsymbol{y}_j\in\{0,1\}^{|\mathcal{I}|}|\sum_{i\in\mathcal{I}}\xi_i^{\omega}y_{ij}\le m_j^{\omega}(\omega),\boldsymbol{y}_j\in\mathcal{Y}_j,y_{ij}=1,\forall i\in\mathcal{D}\}$ such that $\sum_{i\in\mathcal{C}\backslash\mathcal{D}}\hat{y}_{ij}\le|\mathcal{C}\backslash\mathcal{D}|-1$ and $\sum_{i\in\mathcal{C}\backslash\mathcal{D}}\hat{y}_{ij}+\sum_{i\in\mathcal{I}\backslash\mathcal{C}}\alpha_i\hat{y}_{ij}>|\mathcal{C}\backslash\mathcal{D}|-1$. Let $r:=\max\{k|\sum_{i\in\mathcal{C}\backslash\mathcal{D}}\hat{y}_{ij}+\sum_{i\in\boldsymbol{\pi}(k)}\alpha_i\hat{y}_{ij}\le|\mathcal{C}\backslash\mathcal{D}|-1\}$. We have

$$\sum_{i\in\mathcal{C}\backslash\mathcal{D}}\hat{y}_{ij}+\sum_{i\in\boldsymbol{\pi}(r+1)}\alpha_i\hat{y}_{ij}=\sum_{i\in\mathcal{C}\backslash\mathcal{D}}\hat{y}_{ij}+\sum_{i\in\boldsymbol{\pi}(r)}\alpha_i\hat{y}_{ij}+(|\mathcal{C}\backslash\mathcal{D}|-1-obj_{\pi_{r+1}})\hat{y}_{\pi_{r+1},j}\le|\mathcal{C}\backslash\mathcal{D}|-1,$$

which is a contradiction. Thus, (12) is valid for (14).

Consider the following $|\mathcal{I}\backslash\mathcal{D}|$ feasible points of (14): for $k\in\mathcal{C}\backslash\mathcal{D}$, set $y_{ij}=1,\forall i\in\mathcal{C}\backslash\{\mathcal{D}\cup k\}$, and $y_{ij}=0,\forall i\in k\bigcup(\mathcal{I}\backslash\mathcal{C})$; for $k=1,\cdots,|\mathcal{I}\backslash\mathcal{C}|$, set $y_{\pi_k j}=1$, $y_{ij}=0,\forall i\in\{\pi_{k+1},\cdots,\pi_{|\mathcal{I}\backslash\mathcal{C}|}\}$, and $\{y_{ij}\}_{i\in(\mathcal{C}\backslash\mathcal{D})\bigcup\{\pi_1,\ldots,\pi_{k-1}\}}$ are the optimal solutions of (13). All these points have $y_{ij}=1,\forall i\in\mathcal{D}$. When $|\mathcal{C}|\le\rho_j+1$, the above $|\mathcal{I}\backslash\mathcal{D}|$ points are feasible, satisfy (12) at equality and are affinely independent. □

B.7. Proof of Lemma 2

Suppose that we have $\hat{\boldsymbol{y}}_j\in\mathcal{Q}_{j\omega}$ that violates (15). $\boldsymbol{\kappa}$ can be partitioned into $\mathcal{D}^0:=\{i\in\boldsymbol{\kappa}|\hat{y}_{ij}=0\}$ and $\mathcal{D}^1:=\{i\in\boldsymbol{\kappa}|\hat{y}_{ij}=1\}$. We assume that the last element in the set $\mathcal{D}^0$ is κ_h where $h\le|\mathcal{D}|$. Then, we have $\sum_{i\in\mathcal{C}\backslash\mathcal{D}}\hat{y}_{ij}+\sum_{i\in\mathcal{I}\backslash\mathcal{C}}\alpha_i\hat{y}_{ij}>|\mathcal{C}\backslash\mathcal{D}|+\sum_{i\in\mathcal{D}^0}\beta_i-1$. Note that $|\mathcal{C}\backslash\mathcal{D}|+\sum_{i\in\mathcal{D}^0}\beta_i-1=|\mathcal{C}\backslash\mathcal{D}|+obj_{\kappa_h}-\sum_{i\in\boldsymbol{\kappa}(h-1)}\beta_i-|\mathcal{C}\backslash\mathcal{D}|+1+\sum_{i\in\mathcal{D}^0\backslash\kappa_h}\beta_i-1=obj_{\kappa_h}-\sum_{i\in\boldsymbol{\kappa}(h-1)}\beta_i+\sum_{i\in\mathcal{D}^0\backslash\kappa_h}\beta_i$. Based on the definition of obj_{κ_h}, we have that $\hat{\boldsymbol{y}}_j$ is a feasible solution of (16) with $l=h$. Then, $obj_{\kappa_h}-\sum_{i\in\boldsymbol{\kappa}(h-1)}\beta_i+\sum_{i\in\mathcal{D}^0\backslash\kappa_h}\beta_i\ge\sum_{i\in\mathcal{C}\backslash\mathcal{D}}\hat{y}_{ij}+\sum_{i\in\mathcal{I}\backslash\mathcal{C}}\alpha_i\hat{y}_{ij}$. This is a contradiction. Thus, (15) is valid for $conv(\mathcal{Q}_{j\omega})$.

Consider the following $|\mathcal{I}|$ feasible points of $conv(\mathcal{Q}_{j\omega})$: when $y_{ij}=1,\forall i\in\mathcal{D}$, then there exists $|\mathcal{I}\backslash\mathcal{C}|$ feasible points that are independent and satisfy the inequality (15) at equality based on Lemma 1; for $l\in\{1,\cdots,|\mathcal{D}|\}$, set $\boldsymbol{y}_j$ is the optimal solution of (16). When $|\mathcal{C}|\le\rho_j+1$, these $|\mathcal{I}|$ points are feasible, satisfy the inequality (15) at equality and are affinely independent. □

B.8. Proof of Theorem 2

We first prove that for (CAP) if the coefficients are described in Theorem 2, then, (21) is valid for $conv(\mathcal{G}_j)$. For $k \in \{1,\dots,|\bar{\Omega}|\}$, let $(\hat{\boldsymbol{y}}_j, \hat{\boldsymbol{z}}_j) \in \mathcal{G}_j$. If $\hat{z}_{j\omega} = 1$ for $\omega \in \Omega_k$, then (21) is valid for $conv(\mathcal{G}_j)$. Otherwise, let $\boldsymbol{\tau}$ be partitioned into $\Omega_k^0 = \{\omega \in \boldsymbol{\tau} | \hat{z}_{j\omega} = 0\}$ and $\Omega_k^1 = \{\omega \in \boldsymbol{\tau} | \hat{z}_{j\omega} = 1\}$. We assume that the last element of Ω_k^0 is τ_h where $h \le |\Omega_k|$. (21) becomes $\sum_{i\in\mathcal{C}\setminus\mathcal{D}} \hat{y}_{ij} + \sum_{i\in\mathcal{I}\setminus\mathcal{C}} \bar{\alpha}_i \hat{y}_{ij} + \sum_{i\in\mathcal{D}} \bar{\beta}_i \hat{y}_{ij} \le |\mathcal{C}\setminus\mathcal{D}| + \sum_{i\in\mathcal{D}} \bar{\beta}_i - 1 + \sum_{\omega\in\Omega_k^0} \bar{\gamma}_\omega$. Note that $|\mathcal{C}\setminus\mathcal{D}| + \sum_{i\in\mathcal{D}} \bar{\beta}_i - 1 + \sum_{\omega\in\Omega_k^0} \bar{\gamma}_\omega = obj_{\tau_h} - \sum_{\omega\in\boldsymbol{\tau}(h-1)} \bar{\gamma}_\omega + \sum_{\omega\in\{\Omega_k^0\setminus\tau_h\}} \bar{\gamma}_\omega$. Since $(\hat{\boldsymbol{y}}_j, \hat{\boldsymbol{z}}_j)$ satisfies (25) with $k = h$, we have $obj_{\tau_h} - \sum_{\omega\in\boldsymbol{\tau}(h-1)} \bar{\gamma}_\omega + \sum_{\omega\in\{\Omega_k^0\setminus\tau_h\}} \bar{\gamma}_\omega \ge \sum_{i\in\mathcal{C}\setminus\mathcal{D}} \hat{y}_{ij} + \sum_{i\in\mathcal{I}\setminus\mathcal{C}} \bar{\alpha}_i \hat{y}_{ij} + \sum_{i\in\mathcal{D}} \bar{\beta}_i \hat{y}_{ij}$. Thus, (21) is valid for $conv(\mathcal{G}_j)$ when $\bar{\gamma}_{\tau_l} = obj_{\tau_l} - |\mathcal{C}\setminus\mathcal{D}| + 1 - \sum_{i\in\mathcal{D}} \bar{\beta}_i - \sum_{\omega\in\boldsymbol{\tau}(l-1)} \bar{\gamma}_\omega$ for $l = 1,\cdots,|\Omega_k|$. Note that $\bar{\alpha}_{\bar{\pi}_l} = |\mathcal{C}\setminus\mathcal{D}| - 1 - \min_{\omega\in\Omega_k} obj_{\bar{\pi}_l}(\omega)$, based on the definition of $obj_{\bar{\pi}_l}(\omega)$, it is easy to see that $obj_{\bar{\pi}_1}(\omega)$ is integer, and consequently $\bar{\alpha}_{\bar{\pi}_1}$ is integer. If $\bar{\alpha}_{\bar{\pi}_1}$ is integer, then $obj_{\bar{\pi}_2}(\omega)$ is integer, which implies $\bar{\alpha}_{\bar{\pi}_2}$ is integer. Using these arguments we know that $\bar{\boldsymbol{\alpha}}$ is integer. Similarly, $\bar{\boldsymbol{\beta}}$ is also integer. Since the coefficients in (21) are integers, and $\boldsymbol{y}$ and $\boldsymbol{z}$ are binary, obj_{τ_l} is integer. $obj^r_{\tau_l}$ is an upper bound on obj_{τ_l} and obj_{τ_l} is integer, thus $\lfloor obj^r_{\tau_l} \rfloor$ is also an upper bound on obj_{τ_l}. Therefore, (21) is valid for $conv(\mathcal{G}_j)$ when $\bar{\gamma}_{\tau_l} = \lfloor obj^r_{\tau_l} \rfloor - |\mathcal{C}\setminus\mathcal{D}| + 1 - \sum_{i\in\mathcal{D}} \bar{\beta}_i - \sum_{\omega\in\boldsymbol{\tau}(l-1)} \bar{\gamma}_\omega$ for $l = 1,\cdots,|\Omega_k|$. The proof is similarly extended to $\mathcal{G}'_j$. □

B.9. Poof of Theorem 3

The algorithm processes a finite number of nodes as it is based on branching on a finite number of binary variables. When there exists an oracle that solve (SP$_j$) to optimality, we can obtain an optimal solution of (SP$_j$) and verify the feasibility of $(\boldsymbol{y}^k, \boldsymbol{z}^k)$ from (MP) to (DR-CAP). In addition, since a finite number of integer solutions are obtained from (MP), (SP$_j$) is solved finite times and the set of feasibility cuts generated in line 12 is finite. Thus, Algorithm 1 terminates in finitely many iterations. Next, we show that the cuts (28) and (29) can remove the current infeasible solution and never cut off any feasible solutions of (DR-CAP). It can be verified that (28) and (29) can remove the current infeasible solution. Also, $\sum_{\omega\in\Omega} p^k_\omega z_{j\omega} \ge \inf_{\boldsymbol{p}\in\mathscr{P}} \sum_{\omega\in\Omega} p_\omega z_{j\omega} \ge 1-\varepsilon$. Thus, (28) never cuts off any feasible solutions of (DR-CAP). We assume that $\tilde{\boldsymbol{y}}$ is a new future solution from (MP) and the corresponding set $\tilde{\mathcal{I}}^1_j$. Let $y_{ij} = \tilde{y}_{ij}$, for $i \in \mathcal{I}$. Then the feasibility cut (29) becomes $\sum_{i\in\mathcal{I}^1_j} \tilde{y}_{ij} \le |\mathcal{I}^1_j| - 1$, which is decomposed to $\sum_{i\in\mathcal{I}^1_j\cap\tilde{\mathcal{I}}^1_j} \tilde{y}_{ij} + \sum_{i\in\mathcal{I}^1_j\setminus\tilde{\mathcal{I}}^1_j} \tilde{y}_{ij} \le |\mathcal{I}^1_j \cap \tilde{\mathcal{I}}^1_j| + |\mathcal{I}^1_j\setminus\tilde{\mathcal{I}}^1_j| - 1 \iff \sum_{i\in\mathcal{I}^1_j\setminus\tilde{\mathcal{I}}^1_j} \tilde{y}_{ij} \le |\mathcal{I}^1_j\setminus\tilde{\mathcal{I}}^1_j| - 1$. If $\mathcal{I}^1_j \subseteq \tilde{\mathcal{I}}^1_j$, $\tilde{\boldsymbol{y}}$ is not a feasible solution, and does not satisfy the feasibility cut. Otherwise, $\sum_{i\in\mathcal{I}^1_j\setminus\tilde{\mathcal{I}}^1_j} \tilde{y}_{ij} = 0$ and $|\mathcal{I}^1_j\setminus\tilde{\mathcal{I}}^1_j| - 1 \ge 0$. □

Appendix C: Algorithm Details

C.1. Dynamic Programming for Up-lifting Coefficient

For $k = 1,\dots,|\mathcal{I}\setminus\mathcal{C}|$, $\lambda_1 = 0,\dots,|\mathcal{C}\setminus\mathcal{D}| - 1$, and $\lambda_2 = 0,\dots,\rho_j - 1 - |\mathcal{D}|$, let $A_{\pi_k}(\lambda_1,\lambda_2) = \underset{\boldsymbol{y}_j\in\{0,1\}^{|(\mathcal{C}\setminus\mathcal{D})\cup\boldsymbol{\pi}(k-1)|}}{\text{minimize}} \{ \sum_{i\in\mathcal{C}\setminus\mathcal{D}} \xi^\omega_i y_{ij} + \sum_{i\in\boldsymbol{\pi}(k-1)} \xi^\omega_i y_{ij} | \sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i\in\boldsymbol{\pi}(k-1)} \alpha_i y_{ij} \ge \lambda_1, \sum_{i\in\mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i\in\boldsymbol{\pi}(k-1)} y_{ij} \le \lambda_2 \}$ and l_t, $t = 0,\dots,|\mathcal{C}\setminus\mathcal{D}| - 1$ be the sum of the t smallest ξ^ω_i, $i \in \mathcal{C}\setminus\mathcal{D}$. Algorithm 2 gives an outline of our dynamic programming framework. Since Algorithm 2 is a dynamic programming based approach, it is easy to see that it has the complexity $O\left(|\mathcal{I}\setminus\mathcal{C}| \cdot (\rho - |\mathcal{D}|) \cdot |\mathcal{C}\setminus\mathcal{D}|\right)$ for calculating the up-lifting coefficients exactly.

Algorithm 2: Dynamic Programming for the Lifting Coefficients

for $\lambda_2 = 0, \ldots, \rho_j - 1 - |\mathcal{D}|$ **do**
  for $\lambda_1 = 0, \ldots, |\mathcal{C}\backslash\mathcal{D}| - 1$ **do**
    if $\lambda_1 \leq \lambda_2$ **then**
      $A_{\pi_1}(\lambda_1, \lambda_2) = l_{\lambda_1}$.
    end
    else
      $A_{\pi_1}(\lambda_1, \lambda_2) = +\infty$.
    end
  end
end
for $k = 1, \ldots, |\mathcal{I}\backslash\mathcal{C}|$ **do**
  $obj_{\pi_k} = \max\left\{\lambda_1 : A_{\pi_k}(\lambda_1, \rho_j - 1 - |\mathcal{D}|) \leq m_j^{\omega}(\omega) - \xi_{\pi_k}^{\omega} - \sum_{i\in\mathcal{D}} \xi_i^{\omega}\}\right\}$, $\alpha_{\pi_k} = |\mathcal{C}\backslash\mathcal{D}| - 1 - obj_{\pi_k}$.
  for $\lambda_2 = 0, \ldots, \rho_j - 1 - |\mathcal{D}|$ **do**
    for $\lambda_1 = 0, \ldots, |\mathcal{C}\backslash\mathcal{D}| - 1$ **do**
      if $\lambda_1 \geq \alpha_{\pi_k}$ *and* $\lambda_2 \geq 1$ **then**
        $A_{\pi_{k+1}}(\lambda_1, \lambda_2) = \min\left\{A_{\pi_k}(\lambda_1, \lambda_2), A_{\pi_k}(\lambda_1 - \alpha_{\pi_k}, \lambda_2 - 1) + \zeta_{\pi_k}^{\omega}\right\}$.
      end
      else
        $A_{\pi_{k+1}}(\lambda_1, \lambda_2) = A_{\pi_k}(\lambda_1, \lambda_2)$.
      end
    end
  end
end

C.2. Separation Heuristic for (39)

Algorithm 3 gives an overview of separation heuristic for (39).

C.3. Separation Heuristic for (21)

Algorithm 4 gives an overview of separation heuristic for (21).

Algorithm 3: Separation Heuristic for (39)

1 Given the LP relaxation optimal solution $(\hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$.
2 **for** $j = 1, \ldots, |\mathcal{J}|$ **do**
3   **for** $\omega = 1, \ldots, N$ **do**
4   **if** $z_{j\omega} = 1$ **then**
5   Sort $\hat{\boldsymbol{y}}_j$: $\hat{y}_{i_1 j} \geq \ldots \geq \hat{y}_{i_{|\mathcal{I}|} j}$. Let $\mathcal{C} = \{i_1, \ldots, i_o\}$ where $o \leq |\mathcal{I}|$ is a smallest number such that $\mathcal{C}$ is a cover.
6   Delete elements from $\mathcal{C}$ in non-decreasing order of $\hat{y}_j$ to get a minimal cover $\mathcal{C}$.
7   Let $\mathcal{D} = \{i \in \mathcal{C} : \hat{y}_{ij} = 1\}$ and $\mathcal{I}_0 = \{i \in \mathcal{I} \backslash \mathcal{C} | \hat{y}_{ij} = 0\}$. Calculate α_i for $i \in \mathcal{I} \backslash (\mathcal{C} \cup \mathcal{I}_0)$.
8   **if** $\sum_{i \in \mathcal{C} \backslash \mathcal{D}} \hat{y}_{ij} + \sum_{i \in \mathcal{I} \backslash (\mathcal{C} \cup \mathcal{I}_0)} \alpha_i \hat{y}_{ij} > |\mathcal{C} \backslash \mathcal{D}| - 1$ **then**
9   Calculate β_i for $i \in \mathcal{D}$, and α_i for $i \in \mathcal{I}_0$.
10   Calculate δ_k, $k \in \Omega \backslash \omega$, set $\gamma = \delta_{\bar{k}_q 1}$ for (CAP), $\gamma = \delta_{\bar{k}_{\bar{q}} 1}$ for (DR-CAP). Obtain the inequality (39).
11   **end**
12   **end**
13   **end**
14 **end**

Algorithm 4: Separation Heuristic for (21)

1 Given the LP relaxation optimal solution $(\hat{\boldsymbol{y}}, \hat{\boldsymbol{z}})$.
2 **for** $j = 1, \ldots, |\mathcal{J}|$ **do**
3   Let $\Omega_1 = \{\omega \in \Omega | \hat{z}_{j\omega} = 1\}$.
4   **if** $\sum_{\omega \in \Omega_1} p_\omega \hat{z}_{j\omega} \geq 1 - \varepsilon$ *(for (CAP)) or* $\inf_{\boldsymbol{p} \in \mathscr{P}} \sum_{\omega \in \Omega_1} p_\omega \hat{z}_{j\omega} \geq 1 - \varepsilon$ *(for (DR-CAP))* **then**
5   Sort $\hat{\boldsymbol{y}}_j$ in non-increasing order: $\hat{y}_{i_1 j} \geq \ldots \geq \hat{y}_{i_{|\mathcal{I}|} j}$.
6   **for** $\omega \in \Omega_1$ **do**
7   Let $\mathcal{C} = \{i_1, \ldots, i_o\}$ where $o \leq |\mathcal{I}|$ is a smallest number such that $\mathcal{C}$ is a cover for ω.
8   Delete elements from $\mathcal{C}$ in non-decreasing order of $\hat{y}_j$ to get a minimal cover $\mathcal{C}$.
9   Let set $\mathcal{D} = \{i \in \mathcal{C} | \hat{y}_{ij} = 1\}$ and $\mathcal{I}_0 = \{i \in \mathcal{I} \backslash \mathcal{C} | \hat{y}_{ij} = 0\}$. Calculate $\bar{\alpha}_i$ for $i \in \mathcal{I} \backslash (\mathcal{C} \cup \mathcal{I}_0)$.
10   **if** $\sum_{i \in \mathcal{C} \backslash \mathcal{D}} \hat{y}_{ij} + \sum_{i \in \mathcal{I} \backslash (\mathcal{C} \cup \mathcal{I}_0)} \bar{\alpha}_i \hat{y}_{ij} > |\mathcal{C} \backslash \mathcal{D}| - 1$ **then**
11   Calculate $\bar{\beta}_i$ for $i \in \mathcal{D}$, $\bar{\alpha}_i$ for $i \in \mathcal{I}_0$, and γ_ω for $\omega \in \Omega_1$. Obtain the violated inequality (21).
12   **end**
13   If (21) is obtained, go to step 2.
14   **end**
15   **end**
16 **end**

C.4. Branch-and-Cut Algorithm

The branch-and-cut algorithm is provided in Algorithm 5.

Algorithm 5: Branch-and-Cut Implementation

```
1  Initialize UB = +∞, LB = −∞, k = 0. Node list 𝒩 = {o}, o is a branching node without constraints.
2  while (𝒩 is nonempty) do
3      Select a node o ∈ 𝒩, 𝒩 ← 𝒩/{o}.
4      At the node o, solve the LP relaxation problem of (IP). k = k + 1.
5      Obtain an optimal solution (y^k, z^k) and objective value obj^k.
6      if obj^k < UB then
7          if (y^k, z^k) is fractional then
8              if Violated inequalities are found then
9                  Add the violated inequalities to the LP relaxation problem. Go to line 5.
10             end
11             else
12                 Branch, resulting in nodes o* and o**, 𝒩 ← 𝒩 ∪ {o*, o**}.
13             end
14         end
15         else
16             Update UB, UB = obj^k, (y*, z*)=(y^k, z^k).
17         end
18     end
19 end
20 return UB and its corresponding optimal solution (y*, z*).
```

C.5. Branch-and-Cut with Probability Cuts Algorithm for (DR-JCAP)

Algorithm 6 gives an overview of the branch-and-cut with probability cuts algorithm for (DR-JCAP).

Appendix D: Dynamic Programming Approach for Computing Big-M values

In this appendix, we use the *dynamic programming* approach proposed by Bertsimas and Demir (2002) to compute the Big-M values in the model reformulation. For $j \in J$, let $D(|\mathcal{I}|, t_j, \rho_j)$ represents (5), where $|\mathcal{I}|$ denotes the $|\mathcal{I}|$ variables of $\boldsymbol{y}_j$. Let $D(n, t_j, \rho_j)$ be a subproblem of $D(|\mathcal{I}|, t_j, \rho_j)$, where n denotes the first n variables of $\boldsymbol{y}_j$ in (5). Let $S(n, t_j, \rho_j)$ be the optimal objective value of $D(n, t_j, \rho_j)$. If $D(n, t_j, \rho_j)$ is infeasible, we set $S(n, t_j, \rho_j) = -\infty$. Note that if $y_{nj} = 0$, $S(n, t_j, \rho_j)$ is equal to $S(n-1, t_j, \rho_j)$. If $y_{nj} = 1$, $S(n, t_j, \rho_j)$ is equal to $S(n-1, t_j - \xi_n^k, \rho_j - 1) + \xi_n^\omega$, Thus, we have

$$S(n, t_j, \rho_j) = \max\{S(n-1, t_j, \rho_j), S(n-1, t_j - \xi_n^k, \rho_j - 1) + \xi_n^\omega\},$$

where $n = 2, \ldots, |\mathcal{I}|$, with an initial condition $S(1, t_j, \rho_j)$. Hence,

$$m_j^\omega(k) = S(|\mathcal{I}|, t_j, \rho_j).$$

Appendix E: Statistics of Surgery Duration

Table 5 presents the statistics of surgery duration for the real-life data, i.e. mean, standard deviation and the percentage for each surgery type.

Appendix F: Computational Results using Weaker Big-M in (CAP)

Table 6 reports computational results for the weaker big-M of (CAP)

Algorithm 6: Branch-and-Cut Algorithm with Probability Cuts for (DR-JCAP)

1 ***Initialize*** $\mathbb{P}^0 \in \mathscr{P}$, the number of iteration $k=0$, UB $=+\infty$, LB $=-\infty$, $\mathcal{N}=\{o\}$, o has no branching constraints.

2 Initialize the root node with the LP relaxation of (MP). Let the LP relaxation of (MP) be denoted by (LMP).

$$\text{(MP)} \qquad \underset{(\boldsymbol{y},\boldsymbol{z}')\in\left\{\{0,1\}^{|I||J|}\times\{0,1\}^N\right\}\cap\mathcal{X}}{\text{minimize}} \sum_{i\in\mathcal{I}}\sum_{j\in\mathcal{J}} c_{ij}y_{ij}$$
$$\text{subject to } (1b), (1c), (33b),$$

3 **while** *($\mathcal{N}$ is nonempty)* **do**

4   Select a node $o\in\mathcal{N}$, $\mathcal{N}\leftarrow\mathcal{N}/\{o\}$.

5   Solve (LMP) at the node o. $k=k+1$.

6   Obtain the optimal solution (y^k, z'^k) and the optimal objective $lobj^k$ of (LMP).

7   **if** $lobj^k < UB$ **then**

8   **if** (y^k, z'^k) *is an integer* **then**

9   Solve (SP), and obtain an optimal solution (p^k) and objective value $uobj^k$

10   **if** $uobj^k < 1-\varepsilon$ **then**

11   Add the cuts (34) and $\sum_{j\in\mathcal{J}}\sum_{i\in\mathcal{I}_j^1} y_{ij} \le |\mathcal{I}|-1$ to (LMP).

12   **end**

13   **if** *The cuts in Step 11 are found* **then**

14   Go to step 5.

15   **end**

16   **else**

17   UB $= lobj^k$, $(y^*, z'^*) = (y^k, z'^k)$.

18   **end**

19   **end**

20   **if** $(\boldsymbol{y}^k, \boldsymbol{z}'^k)$ *is fractional* **then**

21   Use the algorithms that are similar to Algorithm 3 and 4 to find the violated inequalities (35) and (38).

22   **if** *Violated inequalities are found* **then**

23   Add the violated inequalities to (LMP). Go to line 5.

24   **end**

25   **else**

26   Branch, resulting in nodes o^* and o^{**}, $\mathcal{N}\leftarrow\mathcal{N}\cup\{o^*, o^{**}\}$.

27   **end**

28   **end**

29   **end**

30 **end**

31 **return** UB and its corresponding optimal solution (y^*, z'^*).

Table 5 For each surgery type, the mean (mean), standard deviation (std) in hours, and the percentage for each surgery type (percentage) are reported

surgery type	mean (hrs)	std (hrs)	percentage
Gynaecology	1.1	1.3	0.29
Galactophore	1.6	1.0	0.15
Lymphatic	3.2	1.1	0.14
Ear	2.8	1.7	0.13
Urology	2.3	1.7	0.07
Vascular	2.6	1.5	0.07
Obstetrics	1.5	0.5	0.06
Joint	2.8	1.3	0.06
Orthopeadic	3.2	1.8	0.03

Table 6 The average time (in seconds) for the weaker big-M computations (AvT-M), the branch-and-cut algorithm (AvT-B&C), the average number of nodes (# of nodes), and the number of instances solved to optimality (solved).

ε	N	AvT-M	AvT-B&C	# of nodes	solved
0.12	500	11.4	122.6	1,798	5/5
	1000	43.8	219.7	2,088	5/5
	1500	98.7	771.0	5,090	5/5
0.1	500	11.4	164.9	3,914	5/5
	1000	43.8	604.7	7,192	5/5
	1500	98.7	2,298.8	11,049	5/5
0.08	500	11.4	1,290.8	42,876	5/5
	1000	43.8	2,777.8	25,874	5/5
	1500	98.7	8,459.9[0.03]	103,689	4/5
0.06	500	11.4	[0.11]	2,232,748	0/5
	1000	43.8	[0.21]	632,822	0/5
	1500	98.7	[0.28]	362,215	0/5

"[·]" in column of *AvT-B&C* means the average sub-optimality gap for instances that cannot be solved to optimality within 10 hours time limit.