Distributionally Robust Chance-Constrained Assignment Problem with an Application to Operating Room Planning

Shanshan Wang * 1, 2, Jinlin Li † 1, and Sanjay Mehrotra ‡ 2

1Department of Management Science and Engineering, Beijing Institute of Technology, Beijing, China 100081
2Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL, 60208

Abstract: We study an assignment problem with chance constraints (CAP) and its distributionally robust counterpart (DR-CAP). For a big-M formulation of DR-CAP, we present a technique that takes advantage of DR chance constraints in the big-M estimations. Next, for a bilinear formulation of CAP, we develop a class of valid inequalities for a 0-1 bilinear knapsack set, as well as multiple bilinear knapsack sets with a binary linear knapsack constraint. These are subsequently extended to DR-CAP. The branch-and-cut approaches combined with the valid inequalities are proposed to solve CAP and DR-CAP, respectively. A computational study for CAP and DR-CAP using data from a hospital operating room is conducted. We find that the incorporation of the big-M calculations and the proposed cuts allow us to solve certain model instances, and reduce computational time for others. The instances requiring a high probability of chance satisfaction remain challenging. We also find that the use of the Wasserstein ambiguity set improves the out-of-sample performance of satisfying the chance constraints. This improvement is more significant that the one possible by increasing the sample size. The DR-CAP model instances can be solved in approximately four times the time required to solve CAP instances.

Keywords: Chance-constrained assignment problem; distributionally robust optimization; bilinear program; branch-and-cut; valid inequalities; operating room planning

1 Introduction

In the chance-constrained assignment problem, we assign 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 with a specified duration with a high
probability. More specifically, we study the chance-constrained assignment problem:

\[
\text{(CAP)} \quad \begin{align*}
\text{minimize} & \quad y \in \{0, 1\}^{|I||J|} \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij} \\
\text{subject to} & \quad \sum_{j \in J} y_{ij} = 1, \quad \forall i \in I, \\
& \quad \sum_{i \in I} y_{ij} \leq \rho_j, \quad \forall j \in J, \\
& \quad P \left\{ \sum_{i \in I} \xi_i y_{ij} \leq t_j \right\} \geq 1 - \epsilon, \quad \forall j \in J,
\end{align*}
\]

where \( I := \{1, \ldots, |I|\} \) is the set of items, \( J := \{1, \ldots, |J|\} \) is the set of bins, \(| \cdot |\) is the cardinality of a set, \( c_{ij} \) is the nonnegative cost for assigning item \( i \) to bin \( j \), \( \rho_j \) is the quantitative restriction of bin \( j \), and \( t_j \) is the capacity of bin \( j \). \( \xi_i \) is the random weight of item \( i \). The binary decision variable \( y_{ij} \) indicates if item \( i \) is assigned to bin \( j \). Let \( y_j := (y_{1j}, \ldots, y_{|I|j})^T \) for \( j \in J \), and \( y := (y_1, \ldots, y_{|J|})^T \).

The objective (1a) minimizes the total cost of assigning the items to the bins. Constraints (1b) ensure that each item \( i \) is assigned to only one bin. Constraints (1c) ensure that at most \( \rho_j \) items are assigned to bin \( j \). Constraints (1d) ensure that the capacity for bin \( j \) is satisfied with probability \( 1 - \epsilon \) with respect to distribution \( P \), where \( \epsilon \in [0, 1] \) and \( P \) is a known joint distribution of \( (\xi_1, \cdots, \xi_{|I|})^T \). The chance-constrained assignment problem has a wide range of real-life applications in resource allocation (Lamas and Demeulemeester, 2016), healthcare (Zhang et al., 2015), facility location (Peng et al., 2018), and cloud computing (Cohen et al., 2019), among others.

There are several challenges in solving the chance-constrained assignment problem. First, (CAP) is not a convex optimization problem, given that the variables in (CAP) are binary and chance constraints (1d) might not induce a convex set. Moreover, a chance constraint does not necessarily preserve the smoothness of the original constraints (Hu et al., 2013). In the chance-constrained program (CCP) literature, it is commonly assumed that the probability distributions of the random weights \( \xi_i \) are known and finitely supported. Incomplete knowledge of the probability distribution of \( \xi_i \) can be addressed by using an ambiguity set \( \mathcal{P} \) that allows a family of distributions. The chance constraints (1d) are satisfied over all probability distributions within the ambiguity set \( \mathcal{P} \), resulting in the formulation:

\[
\text{(DR-CAP)} \quad \begin{align*}
\text{minimize} & \quad y \in \{0, 1\}^{|I||J|} \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij} \\
\text{subject to} & \quad (1b), (1c), \\
& \quad \inf_{P \in \mathcal{P}} P \left\{ \sum_{i \in I} \xi_i y_{ij} \leq t_j \right\} \geq 1 - \epsilon, \quad \forall j \in J.
\end{align*}
\]

In this paper we assume that the probability distribution \( P \) has finite support \( \xi := (\xi^1, \cdots, \xi^N)^T \), where \( \xi^\omega := (\xi^\omega_1, \cdots, \xi^\omega_{|I|})^T \) for \( \omega \in \Omega := \{1, \cdots, N\} \). \( \xi^\omega_i \) denotes the weight of item \( i \) for scenario \( \omega \in \Omega \), and \( p_\omega \) 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 \( \xi^\omega_i \) and \( t_j \) are non-negative integers, and without loss of generality, \( p_\omega \leq \epsilon \) and \( \xi^\omega_i \leq t_j \), for
The model framework in 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 chance-constrained programs (Shapiro et al., 2009; Calafiore and Campi, 2006). This has motivated a number of studies to solve 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. (2018)).

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 when only the right-hand-side vector is random (Tanner and Ntaiimo (2010)). Tanner and Ntaiimo (2010) used irreducibly infeasible subsystems to derive a class of valid inequalities for such problems. Luedtke (2014) used a technique similar to the one for generating valid inequalities 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. They applied this new version of Benders decomposition 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 problems with chance constraints, Beraldi and Bruni (2010) formulated the problem as 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 binary 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 an approximate lifting technique for the probabilistic cover inequalities is proposed. Song et al. (2014) also provided a coefficient strengthening procedure, and a family of projection cuts and local cuts to strengthen the probabilistic cover formulation. 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 chance-constrained combinatorial optimization problem and presented an exact method for solving the problem under the assumption that the chance probability can be calculated. For the special case where the distribution is represented by samples, they developed a class of
valid inequalities for a probabilistic partial set covering problem.

In a companion paper, Wang et al. (2019) studied the chance-constrained bin-packing problem and use the lifting technique to develop a family of valid inequalities for the set obtained from only one bilinear knapsack constraint (see Example 2). The difference in the structures of the bin-packing problems and the assignment problems motivate us to generalize the applicability of the approach developed in Wang et al. (2019) and obtain stronger valid inequalities for (CAP).

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., Bertsimas et al. (2010), 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), Zhao and Guan (2018) and Luo and Mehrotra (2019)). For the distributionally robust chance-constrained programs, Chen et al. (2010), Zymler et al. (2013), and Postek et al. (2018) developed tractable approximations of ambiguous chance constraints under the moment-based ambiguity sets. Hansusanto 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 convex reformulation under some conditions. Jiang and Guan (2016) studied a data-driven distributionally robust chance constraint with $\phi$-divergence measure-based set. They showed that this problem is equivalent to a classical chance constraint with a perturbed risk level. As an important type of statistical distance, the Wasserstein metric can be used to define an ambiguity set, which ensures that the true probability distribution belongs to the ambiguity set with a prescribed level of confidence for an appropriate choice of its radius parameter (Esfahani and Kuhn (2018)). Thus, several studies have investigated the distributionally robust chance-constrained problem with the Wasserstein ambiguity set (see, e.g., Xie (2018), Chen et al. (2018)). 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. Deng et al. (2016) studied chance-constrained surgery planning by using a $\phi$-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 covariance matrix are known. They reformulated the problem as a binary second-order cone (SOC) program and developed valid inequalities for the SOC program by using the submodularity and the bin-packing structure. Zhang et al. (2015) investigated the stochastic and distributionally robust chance-constrained bin-packing problem with the bin extension decisions. They developed a branch-and-price approach based on a column generation reformulation for these two problems.
1.4 Contributions of This Paper

This paper makes the following specific contributions:

- We first use a big-M approach to formulate (CAP) and (DR-CAP) as the binary and semi-infinite integer programs, respectively. We present a coefficient strengthening approach for these reformulations and provide alternative bilinear reformulations for (CAP) and (DR-CAP).

- We develop a new family of valid inequalities for the binary bilinear knapsack set from a single row and scenario in the bilinear constraints and the constraint (1c). More specifically, we use the lifting technique for the binary bilinear knapsack set to derive lifted cover inequalities and show that these inequalities are facet-defining under certain conditions. Furthermore, we present stronger valid inequalities for (CAP) and (DR-CAP) by further restricting the feasible region of $y$ in a lifting problem.

- We consider the intersection of multiple binary bilinear knapsacks with a general 0-1 knapsack constraint, and a cardinality constraint. By using the lifting technique and a heuristic procedure, we obtain another family of valid inequalities for this intersection set. These valid inequalities are a generalization of the cover inequalities.

- We develop separation heuristics that efficiently obtain the violated inequalities and incorporate the inequalities in a branch-and-cut framework to solve (CAP). We then propose a branch-and-cut algorithm with probability cuts, which uses a distribution separation procedure, the valid inequalities and the feasibility cuts, to solve (DR-CAP). A convergence proof of this algorithm is provided.

- We perform a computational study for the operating room assignment problem based on real data from a hospital to show the benefits of the techniques developed in this paper. Using the techniques developed in this paper, we solve (CAP) instances with up to 1,500 scenarios within ten hours when $\varepsilon = 0.08, 0.1, 0.12$, and obtain a smaller optimality gap for instances with $\varepsilon = 0.06$. For (DR-CAP) using Wasserstein metric, we solved all instances with $N = 1,500$ within two hours for $\varepsilon = 0.1$. We performed an out-of-sample estimation of the chance constraint satisfaction for the solutions obtained from (CAP) and (DR-CAP) models. The (DR-CAP) solutions achieve the desirable probability target more reliably, though we also 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. As expected, the (DR-CAP) solutions are more ‘costly’. We also find that (DR-CAP) instances are solved in about four times the time required to solve (CAP).

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 \( y \). We further analyze the multiple binary bilinear knapsack set with a general 0-1 knapsack constraint and develop a second 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 cut for (DR-CAP) and its convergence proof are also provided in this section. Section 5 reports computational results on (CAP) and (DR-CAP) formulations of the operating room assignment problem. Section 6 concludes the paper with a summary of the important findings. Appendix A gives the proofs of some propositions. Appendix B provides pseudo-code of some of the algorithms.

2 Model Reformulation

We formulate (CAP) as a binary linear program in Section 2.1 and present a dynamic programming based approach to estimate the big-M coefficients in Appendix B.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 \( j \in J \) and \( \omega \in \Omega \). Namely, \( z_{j\omega} = 1 \) if the constraint \( \sum_{i \in I} \xi_{i\omega} y_{ij} \leq t_j \) is satisfied, and \( z_{j\omega} = 0 \) otherwise. For \( j \in J \), let \( z_j := (z_{j1}, \ldots, z_{jN})^T \) and \( z := (z_1, \ldots, z_{|J|})^T \). The constraints (1d) can be formulated as

\[
\begin{align*}
\sum_{i \in I} \xi_{i\omega} y_{ij} + (M^\omega_j - t_j) z_{j\omega} &\leq M^\omega_j, \quad \forall j \in J, \omega \in \Omega, \quad (3a) \\
\sum_{\omega \in \Omega} p_{\omega} z_{j\omega} &\geq 1 - \varepsilon, \quad \forall j \in J, \quad (3b)
\end{align*}
\]

where \( M^\omega_j \) is a constant that ensures that the constraints (3a) hold when \( z_{j\omega} = 0 \). Computation of a small valid value of big-M gives a tighter formulation in (3). We further develop the big-M coefficient strengthening procedure from Song et al. (2014) to obtain a value of \( M^\omega_j \). Note that for \( j \in J \) and \( \omega \in \Omega \):

\[
M^\omega_j \geq \bar{M}^\omega_j := \maximize_{y_j \in \{0,1\}^{|I|}} \left\{ \sum_{i \in I} \xi_{i\omega} y_{ij} \right\} \mathbb{P} \left\{ \sum_{i \in I} \xi_i y_{ij} \leq t_j \right\} \geq 1 - \varepsilon, \sum_{i \in I} y_{ij} \leq \rho_j \right\}. \quad (4)
\]
For $j \in J$ and $\omega, k \in \Omega$, let

$$m_{\omega j}^{\omega}(k) := \max_{y_j \in \{0,1\}^{|I|}} \left\{ \sum_{i \in I} \xi_{\omega i} y_{ij} \left| \sum_{i \in I} \xi_{k i} y_{ij} \leq t_j, \sum_{i \in I} y_{ij} \leq \rho_j \right. \right\}. \tag{5}$$

We sort $m_{\omega j}^{\omega}(k)$ in a non-decreasing order such that $m_{\omega j}^{\omega}(k_1) \leq \ldots \leq m_{\omega j}^{\omega}(k_N)$. An upper bound for $\bar{M}_{\omega j}^{\omega}$ is provided in Proposition 1. The proof of Proposition 1 is given in Appendix A.1.

**Proposition 1** $m_{\omega j}^{\omega}(k_q)$ is an upper bound for $\bar{M}_{\omega j}^{\omega}$, where

$$q := \min \left\{ l \left| \sum_{j = 1}^{l} p_{kj} > \varepsilon \right. \right\},$$

and (CAP) can be equivalently reformulated as the following binary integer program

**(IP)**

minimize $\sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij}$ \hspace{1cm} (6a)

subject to \hspace{1cm} (1b), (1c), (3b),

$$\sum_{i \in I} \xi_{\omega i} y_{ij} + (m_{\omega j}^{\omega}(k_q) - m_{\omega j}^{\omega}(\omega)) z_{j\omega} \leq m_{\omega j}^{\omega}(k_q), \quad \forall j \in J, \omega \in \Omega; \tag{6b}$$

$$y_{ij}, z_{j\omega} \in \{0,1\}, \quad \forall i \in I, j \in J, \omega \in \Omega. \tag{6c}$$

Note that (5) has a knapsack constraint and a cardinality constraint. We describe a dynamic programming method for solving (5) in Appendix B.1. The procedure applies the technique in Bertsimas and Demir (2002). For $j \in J$, if $t_j$ and $\rho_j$ are moderate, dynamic programming is an efficient approach for solving (5) to optimality.

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

In this section, we study the chance constraints where the distribution $\mathbb{P}$ of random weights belongs to an ambiguity set. The results in this section are stated for any ambiguity set defined on a finite support. However, in the computational results of this paper, we use the $l_1$-Wasserstein ambiguity set:

$$\mathcal{P}_W = \{ p \in \mathbb{R}_+^N \mid \sum_{\omega \in \Omega} p_{\omega} = 1, \sum_{\omega \in \Omega} \sum_{k \in \Omega} \| \xi^\omega - \xi^k \| \nu_{\omega k} \leq \eta, \sum_{k \in \Omega} \nu_{\omega k} = p_{\omega}, \forall \omega \in \Omega, \sum_{\omega \in \Omega} \nu_{\omega k} = p_{k}^*, \forall k \in \Omega, \nu_{\omega k} \geq 0, \forall \omega, k \in \Omega \},$$

where $\eta \geq 0$ is the Wasserstein radius and $\{ p_{k}^* \}_{k \in \Omega}$ is an empirical probability distribution of $\xi$. Note that if $\eta = 0$, then $p_{\omega} = p_{\omega}^*$ for all $\omega \in \Omega$ and (DR-CAP) reduces to (CAP). (2a) with respect to the Wasserstein ambiguity set is

$$\inf \left\{ \sum_{\omega \in \Omega} p_{\omega} \mathbb{I} \left( \sum_{i \in I} \xi_{\omega i} y_{ij} \leq t_j \right) \left| p \in \mathcal{P}_W \right. \right\} \geq 1 - \varepsilon, \quad \forall j \in J,$$

where $\mathbb{I}(\cdot)$ is an indication function.
Let $z_{j \omega}$ and $m^\omega_j(\cdot)$ be defined as in the Section 2.1. The following theorem gives the semi-infinite reformulation of (DR-CAP) with a general ambiguity set $\mathcal{P}$.

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

\[
\begin{align*}
(SIP) \quad & \text{minimize} & & \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij} \\
& \text{subject to} & & \inf_{p \in \mathcal{P}} \sum_{\omega \in \Omega} p_{\omega} z_{j \omega} \geq 1 - \varepsilon, & \forall j \in J, \\
& & & \sum_{i \in I} \xi_i^\omega y_{ij} + (m^\omega_j(k_q) - m^\omega_j(\omega)) z_{j \omega} \leq m^\omega_j(k_q), & \forall j \in J, \omega \in \Omega, \\
& & & y_{ij}, z_{j \omega} \in \{0, 1\}, & \forall i \in I, j \in J, \omega \in \Omega,
\end{align*}
\]

where $q := \min \{1 \mid \sup_{p \in \mathcal{P}} \sum_{j = 1}^q p_{k_j} > \varepsilon\}$.

**Proof** We first show that $m^\omega_j(k_q)$ is an upper bound for $M^\omega_j$, where

\[
\begin{align*}
\hat{M}^\omega_j := & \max_{y} \sum_{i \in I} \xi_i^\omega y_{ij} \\
& \text{subject to} \sum_{i \in I} y_{ij} \leq \rho_j, & \forall j \in J, \\
& \inf_{p \in \mathcal{P}} \mathcal{P} \left\{ \sum_{i \in I} \xi_i y_{ij} \leq t_j \right\} \geq 1 - \varepsilon, & \forall j \in J.
\end{align*}
\]

Let $y_{ij}^*$ be an optimal solution of (8), then there exist at least one $k' \in \bar{\Omega} := \{1, \cdots, q\}$ such that $\sum_{i \in I} \xi_i^k y_{ij}^* \leq t_j$. Otherwise, $\sum_{i \in I} \xi_i^k y_{ij}^* > t_j$ for all $k \in \{1, \cdots, \bar{q}\}$. We have

\[
\begin{align*}
\inf_{p \in \mathcal{P}} \mathcal{P} \left\{ \sum_{i \in I} \xi_i y_{ij}^* \leq t_j \right\} &= \inf_{p \in \mathcal{P}} \sum_{\omega \in \Omega} p_{\omega} \mathcal{P} \left( \sum_{i \in I} \xi_i^\omega y_{ij}^* \leq t_j \right) \\
&= \inf_{p \in \mathcal{P}} \sum_{\omega \in \Omega} p_{\omega} \left( \sum_{i \in I} \xi_i^\omega y_{ij}^* \leq t_j \right) \\
&\leq \inf_{p \in \mathcal{P}} \sum_{\omega \in \Omega} p_{\omega} = \inf_{p \in \mathcal{P}} (1 - \sum_{\omega \in \Omega} p_{\omega}) \\
&= 1 - \sup_{p \in \mathcal{P}} \sum_{\omega \in \Omega} p_{\omega} < 1 - \varepsilon,
\end{align*}
\]

which is a contradiction. Thus, $m^\omega_j(k_q) \geq m^\omega_j(k'_q) \geq \hat{M}^\omega_j$, which implies that (7c) hold. Therefore, (DR-CAP) can be rewritten as (7). \(\square\)

Note that $\bar{q}$ in Theorem 1 is obtained by solving a sequence of linear programs when $\mathcal{P} := \mathcal{P}_W$. Moreover, the lift-hand side of (7b) 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 $m^\omega_j(\cdot)$ used in the big-M formulation. The following
Corollary 1 Let \( \{p_\omega\}_{\omega \in \Omega} \in \mathcal{P} \), and \( q = \min\{l | \sum_{j=1}^l p_{k_j} > \varepsilon\} \). Then, \( q \leq \bar{q} \) and \( m_j^{\omega}(k_q) \leq m_j^{\omega}(k_q) \).

Proof. Since \( \sup_{p \in \mathcal{P}} \sum_{j=1}^q p_{k_j} = \sum_{j=1}^q \hat{p}_{k_j} > \varepsilon \), we have \( \hat{q} \geq \bar{q} \) and \( m_j^{\omega}(k_{\bar{q}}) \leq m_j^{\omega}(k_q) \).

2.3 Binary Bilinear Reformulations

In the previous section, we calculated the strengthened big-M coefficients to formulate the chance constraints as binary linear constraints. In this section, we present an alternative approach following Wang et al. (2019). Let \( z_{j\omega} \) be defined as in Section 2.1. The constraints (6b) and (7c) can also be rewritten as

\[
\sum_{i \in I} \xi_i^\omega y_{ij} z_{j\omega} \leq m_j^{\omega}(\omega) z_{j\omega}, \quad \forall j \in J, \omega \in \Omega. \tag{9}
\]

Thus, we can use (9) to obtain a binary bilinear reformulation and bilinear semi-infinite reformulation for (CAP) and (DR-CAP), respectively. The following proposition shows the relationship between the bilinear reformulations with (7) and (6) for (DR-CAP) and (CAP), respectively. A proof of Proposition 2 is given in Appendix A.2.

Proposition 2 The relaxation problem of the binary bilinear reformulation for (DR-CAP) is stronger than the linear relaxation problem of (7). Similarly, the relaxation problem of the binary bilinear reformulation for (CAP) is stronger than the linear relaxation problem of (6).

Note that constraints (1c), (3b) and (9) give a key substructure of the binary bilinear reformulation of (CAP). Let

\[
\mathcal{H} := \left\{ (y, z) \in \{0,1\}^{|I||J|} \times \{0,1\}^{|N||J|} | (1c), (3b), (9) \right\}.
\]

For \( j \in J \), let

\[
\mathcal{G}_j := \left\{ (y_j, z_j) \in \{0,1\}^{|I|} \times \{0,1\}^N | \sum_{i \in I} y_{ij} \leq \rho_j \sum_{i \in I} \xi_i^\omega y_{ij} z_{j\omega} \leq m_j^{\omega}(\omega) z_{j\omega}, \forall \omega \in \Omega, \sum_{\omega \in \Omega} p_\omega z_{j\omega} \geq 1 - \varepsilon \right\}.
\]

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

Let us use \( \text{conv}(\cdot) \) to denote the convex hull of a set. The following proposition shows that in order to identify strong valid inequalities for \( \text{conv}(\mathcal{H}) \), we can develop strong valid inequalities for \( \text{conv}(\mathcal{G}_j) \). A proof can be found in Appendix A.3.

Proposition 3 If an inequality is valid for \( \text{conv}(\mathcal{G}_j) \), this inequality is also valid for \( \text{conv}(\mathcal{H}) \). Moreover, if an inequality is facet-defining for \( \text{conv}(\mathcal{G}_j) \), it is also facet-defining for \( \text{conv}(\mathcal{H}) \).
(DR-CAP) and obtain valid inequalities for $G'_j$. For $j \in J$, let

$$G'_j := \left\{ (y_j, z_j) \in \{0,1\}^{|I|} \times \{0,1\}^N \bigg| \sum_{i \in I} y_{ij} \leq \rho_j, \sum_{i \in I} \xi_i^\omega y_{ij} z_{j\omega} \leq m_j^\omega(\omega) z_{j\omega}, \forall \omega \in \Omega, \inf_{p \in \mathcal{P}} \sum_{\omega \in \Omega} p_\omega z_{j\omega} \geq 1 - \varepsilon \right\}. $$

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. Due to the flexibility of our approach, Section 3.2 further presents a family of valid inequalities for $G_j$ and $G'_j$.

3.1 Lifted Cover Inequalities

We assume that $j \in J$ and $\omega \in \Omega$ are fixed in this section. Let the binary bilinear knapsack set

$$F_{j\omega} := \left\{ (y_j, z_{j\omega}) \in \{0,1\}^{|I|} \times \{0,1\} \bigg| \sum_{i \in I} y_{ij} \leq \rho_j, \sum_{i \in I} \xi_i^\omega y_{ij} z_{j\omega} \leq m_j^\omega(\omega) z_{j\omega} \right\}. $$

Note that the inequalities valid for $\text{conv}(F_{j\omega})$ are also valid for (CAP) and (DR-CAP). Note also that when compared to the development in Wang et al. (2019), we include the cardinality constraint in addition to the binary bilinear knapsack constraint in the description of $F_{j\omega}$. When $z_{j\omega} = 1$, the set $F_{j\omega}$ becomes the two-constraint 0-1 knapsack set $Q_{j\omega}$:

$$Q_{j\omega} := \left\{ y \in \{0,1\}^{|I|} \bigg| \sum_{i \in I} y_{ij} \leq \rho_j, \sum_{i \in I} \xi_i^\omega y_{ij} \leq m_j^\omega(\omega) \right\}. $$

We now extend the results for the single binary knapsack set from Zemel (1989) and Gu et al. (1998) to develop valid inequality that is facet-defining for the set $Q_{j\omega}$. We also provide a lifted cover inequality that is valid for $\text{conv}(F_{j\omega})$ by rotating this valid inequality. Then the restriction of the feasible region of $y$ is used to obtain a stronger valid inequality for (CAP) and (DR-CAP).

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

In this section, we assume that $C$ is a minimal cover for $\sum_{i \in I} \xi_i^\omega y_{ij} \leq m_j^\omega(\omega)$. Let set $D \subseteq C$. The following proposition gives a valid inequality that is facet-defining for the following convex hull

$$\text{conv} \left( \left\{ y_j \in \{0,1\}^{|I|} \bigg| \sum_{i \in I} y_{ij} \leq \rho_j, \sum_{i \in I} \xi_i^\omega y_{ij} \leq m_j^\omega(\omega), y_{ij} = 0, \forall i \in I \setminus C, y_{ij} = 1, \forall i \in D \right\} \right). $$

**Proposition 4** The inequality

$$\sum_{i \in C \setminus D} y_{ij} \leq |C \setminus D| - 1$$

(11)
is valid for (10). If \(|C| \leq \rho_j + 1\), the inequality (11) is facet-defining for (10).

**Proof** The proof is given in Appendix A.4. \(\square\)

### 3.1.1 Up-Lifting

In general, (11) does not induce the facets of \(Q_{j\omega}\). To obtain a facet-defining inequality for \(Q_{j\omega}\), we first compute the coefficients of variables in \(I \setminus C\). This procedure is called up-lifting. By using the up-lifting technique, we obtain an inequality of the form

\[
\sum_{i \in C \setminus D} y_{ij} + \sum_{i \in I \setminus C} \alpha_i y_{ij} \leq |C \setminus D| - 1,
\]  

(12)

where \(\alpha_i\) is called the up-lifting coefficient. Let \(\pi = \{\pi_1, \ldots, \pi_{|I \setminus C|}\}\) be a sequence of the set \(I \setminus C\). For \(k = 1, \ldots, |I \setminus C|\), let

\[
\text{obj}_{\pi_k} := \maximize_{y_j} \sum_{i \in C \setminus D} y_{ij} + \sum_{i = \pi_1}^{\pi_{k-1}} \alpha_i y_{ij}
\]  

subject to

\[
\sum_{i \in C \setminus D} \xi_i^\omega y_{ij} + \sum_{i = \pi_1}^{\pi_{k-1}} \xi_i^\omega y_{ij} \leq m_j^\omega(\omega) - \sum_{i \in D} \xi_i^\omega,
\]  

(13b)

\[
\sum_{i \in C \setminus D} y_{ij} + \sum_{i = \pi_1}^{\pi_{k-1}} y_{ij} \leq \rho_j - 1 - |D|,
\]  

(13c)

\[
y_{ij} \in \{0, 1\}, \quad \forall i \in (C \setminus D) \cup \{\pi_1, \ldots, \pi_{k-1}\}.
\]  

(13d)

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

**Lemma 1** For \(k = 1, \ldots, |I \setminus C|\), let \(\alpha_{\pi_k} = |C \setminus D| - 1 - \text{obj}_{\pi_k}\), where \(\text{obj}_{\pi_k}\) is defined in (13). The inequality (12) is valid for

\[
\text{conv} \left( \left\{ y_j \in \{0, 1\}^{|I|} \mid \sum_{i \in I} y_{ij} \leq \rho_j, \sum_{i \in I} \xi_i^\omega y_{ij} \leq m_j^\omega(\omega), y_{ij} = 1, \forall i \in D \right\} \right).
\]  

(14)

If \(|C| \leq \rho_j + 1\), the inequality (12) is facet-defining for (14).

**Proof** Suppose that there exists \(\tilde{y}_j\) that serves as a member of the set \(\{y_j \in \{0, 1\}^{|I|} \mid \sum_{i \in I} y_{ij} \leq \rho_j, \sum_{i \in I} \xi_i^\omega y_{ij} \leq m_j^\omega(\omega), y_{ij} = 1, \forall i \in D\}\) such that \(\sum_{i \in C \setminus D} \tilde{y}_{ij} \leq |C \setminus D| - 1\) and \(\sum_{i \in C \setminus D} \alpha_i \tilde{y}_{ij} > \).

11
\[ |\mathcal{C}\setminus\mathcal{D}| - 1. \] Let \( r := \max\{k | \sum_{i \in \mathcal{C}\setminus\mathcal{D}} \hat{y}_{ij} + \sum_{i=\pi_1}^{\pi_k} \alpha_i \hat{y}_{ij} \leq |\mathcal{C}\setminus\mathcal{D}| - 1\} \). We have

\[
\sum_{i \in \mathcal{C}\setminus\mathcal{D}} \hat{y}_{ij} + \sum_{i=\pi_1}^{\pi_{r+1}} \alpha_i \hat{y}_{ij} = \sum_{i \in \mathcal{C}\setminus\mathcal{D}} \hat{y}_{ij} + \sum_{i=\pi_1}^{\pi_r} \alpha_i \hat{y}_{ij} + \alpha_{\pi_{r+1}} \hat{y}_{\pi_{r+1},j} = \sum_{i \in \mathcal{C}\setminus\mathcal{D}} \hat{y}_{ij} + \sum_{i=\pi_1}^{\pi_r} \alpha_i \hat{y}_{ij} + (|\mathcal{C}\setminus\mathcal{D}| - 1 - \text{obj}_{\pi_{r+1}}) \hat{y}_{\pi_{r+1},j} \leq |\mathcal{C}\setminus\mathcal{D}| - 1,
\]

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

Consider the following \(|\mathcal{T}\setminus\mathcal{D}|\) feasible points of (14): for \( k \in \mathcal{C}\setminus\mathcal{D} \), set \( y_{ij} = 1, \forall i \in \mathcal{C}\setminus(\mathcal{T}\setminus\mathcal{C}) \); for \( k = 1, \cdots, |\mathcal{T}\setminus\mathcal{C}| \), set \( y_{\pi_k,j} = 1, \ y_{ij} = 0, \forall i \in \{\pi_{k+1}, \cdots, \pi_{|\mathcal{T}\setminus\mathcal{C}|}\} \), and \( \{y_{ij}\}_{i \in (\mathcal{C}\setminus\mathcal{D}) \cup (\pi_1, \cdots, \pi_{k-1})} \) are the optimal solutions of (13). All these feasible points of (14) have \( y_{ij} = 1, \forall i \in \mathcal{D} \). It can be verified that the above \(|\mathcal{T}\setminus\mathcal{D}|\) feasible points satisfy the inequality (12) at equality and are affinely independent. □

We apply the dynamic programming, which has been used to calculate the up-lifting coefficients in the binary single knapsack problem (see Zemel (1989)) to obtain the lifting coefficient \( \alpha_i \). This algorithm is given in Appendix B.2.

3.1.2 Down-Lifting

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

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

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

\[
\begin{align*}
\text{obj}_{\kappa_l} &:= \underset{y_j \in \{0,1\}^{\mathcal{I}}} \text{maximize} \sum_{i \in \mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i \in \mathcal{I}\setminus\mathcal{C}} \alpha_i y_{ij} + \sum_{i=\kappa_1}^{\kappa_{l-1}} \beta_i y_{ij} \tag{16a} \\
&\text{subject to} \sum_{i \in \mathcal{I}} \xi_j y_{ij} \leq m_j^\omega(\omega), \tag{16b} \\
&\sum_{i \in \mathcal{I}} y_{ij} \leq \rho_j, \tag{16c} \\
&y_{\kappa_{l},j} = 0, \ y_{ij} = 1, \quad \forall i \in \{\kappa_{l+1}, \cdots, \kappa_{|\mathcal{D}|}\}. \tag{16d}
\end{align*}
\]

**Lemma 2** For \( l = 1, \cdots, |\mathcal{D}| \), let \( \beta_{\kappa_l} = \text{obj}_{\kappa_l} - \sum_{i=\kappa_1}^{\kappa_{l-1}} \beta_i - |\mathcal{C}\setminus\mathcal{D}| + 1 \), where \( \text{obj}_{\kappa_l} \) is defined in (16). The inequality (15) is valid for \( \text{conv}(\mathcal{Q}_{j\omega}) \). If \(|\mathcal{C}| \leq \rho_j + 1\), (15) is facet-defining for \( \text{conv}(\mathcal{Q}_{j\omega}) \).

**Proof** Suppose that we have \( \bar{y}_j \in \mathcal{Q}_{j\omega} \) that violates (15). \( \kappa \) can be partitioned into \( \mathcal{D}^0 := \{i \in \kappa | \bar{y}_{ij} = 0\} \) and \( \mathcal{D}^1 := \{i \in \kappa | \bar{y}_{ij} = 1\} \). We assume that the last element in the set \( \mathcal{D}^0 \) is \( \kappa_h \) where \( h \leq |\mathcal{D}| \). Then,
we have
\[ \sum_{i \in \mathcal{C} \setminus \mathcal{D}} \tilde{y}_{ij} + \sum_{i \in \mathcal{I} \setminus \mathcal{C}} \alpha_i \tilde{y}_{ij} > |\mathcal{C} \setminus \mathcal{D}| + \sum_{i \in \mathcal{D}} \beta_i - 1. \]

Note that
\[ |\mathcal{C} \setminus \mathcal{D}| + \sum_{i \in \mathcal{D}} \beta_i - 1 = \text{obj}_{\mathcal{h}} - \sum_{i=\mathcal{h}}^{\mathcal{h}-1} \beta_i + \sum_{i \in \mathcal{D} \setminus \mathcal{C}} \beta_i, \]

Based on the definition of \( \text{obj}_{\mathcal{h}} \), we have that \( \tilde{y}_j \) is a feasible solution of (16) with \( l = \mathcal{h} \). Then,
\[ \text{obj}_{\mathcal{h}} - \sum_{i=\mathcal{h}}^{\mathcal{h}-1} \beta_i + \sum_{i \in \mathcal{D} \setminus \mathcal{C}} \beta_i \geq \sum_{i \in \mathcal{C} \setminus \mathcal{D}} \tilde{y}_{ij} + \sum_{i \in \mathcal{I} \setminus \mathcal{C}} \alpha_i \tilde{y}_{ij} + \sum_{i=\mathcal{h}}^{\mathcal{h}-1} \beta_i \tilde{y}_{ij} - \sum_{i=\mathcal{h}}^{\mathcal{h}-1} \beta_i + \sum_{i \in \mathcal{D} \setminus \mathcal{C}} \beta_i \]

This is a contradiction. Thus, (15) is valid for \( \text{conv}(\mathcal{Q}_{j\omega}) \).

Consider the following \(|\mathcal{I}| + 1\) feasible points of \( \text{conv}(\mathcal{Q}_{j\omega}) \): when \( y_{ij} = 1 \), \( \forall i \in \mathcal{D} \), then there exists \(|\mathcal{I} \setminus \mathcal{C}|\) feasible points that are independent and satisfy the inequality (15) at equality based on Lemma 1; for \( l \in \{1, \ldots, |\mathcal{D}|\} \), set \( y_{\mathcal{h}ij} = 0 \), and the other \( y_{ij} \) are optimal solutions of (16). These \(|\mathcal{I}| \) feasible points satisfy the inequality (15) at equality and are affinely independent. □

3.1.3 Lifted Cover Inequality

In the following, we provide coefficient calculations for a lifted cover inequality that is valid for \( \text{conv}(\mathcal{F}_{j\omega}) \).

**Theorem 2** The lifted cover inequality
\[ \sum_{i \in \mathcal{C} \setminus \mathcal{D}} y_{ij} + \sum_{i \in \mathcal{I} \setminus \mathcal{C}} \alpha_i y_{ij} + \sum_{i \in \mathcal{D}} \beta_i y_{ij} + \gamma(z_\omega - 1) \leq |\mathcal{C} \setminus \mathcal{D}| + \sum_{i \in \mathcal{D}} \beta_i - 1 \tag{17} \]

is valid for \( \text{conv}(\mathcal{F}_{j\omega}) \), if
\[ \gamma = \max_{y_j \in \{0,1\}^{|\mathcal{I}|}} \sum_{i \in \mathcal{C} \setminus \mathcal{D}} y_{ij} + \sum_{i \in \mathcal{I} \setminus \mathcal{C}} \alpha_i y_{ij} + \sum_{i \in \mathcal{D}} \beta_i y_{ij} - |\mathcal{C} \setminus \mathcal{D}| - \sum_{i \in \mathcal{D}} \beta_i + 1 \tag{18a} \]

subject to \( \sum_{i \in \mathcal{I}} y_{ij} \leq \rho_j \). \tag{18b}

Furthermore, if \( |\mathcal{C}| \leq \rho_j + 1 \), (17) is facet-defining for \( \text{conv}(\mathcal{F}_{j\omega}) \).

**Proof** When \( z_\omega = 1 \), (17) is valid for \( \text{conv}(\mathcal{F}_{j\omega}) \) because of Lemma 2. When \( z_\omega = 0 \), due to the definition of \( \gamma \), (17) is also valid for \( \text{conv}(\mathcal{F}_{j\omega}) \). Thus, (17) is valid for \( \text{conv}(\mathcal{F}_{j\omega}) \).

Consider the following \(|\mathcal{I}| + 1\) feasible points of \( \text{conv}(\mathcal{F}_{j\omega}) \): when \( z_\omega = 1 \), there exists \(|\mathcal{I}| \) feasible
points of $\text{conv}(\mathcal{F}_{j \omega})$ that are affinely independent and satisfy (17) at equality based on the Lemma 2; when $z_{\omega} = 0$, let $y_j$ be the optimal solution of (18). These $|I| + 1$ feasible points satisfy (17) at equality and are affinely independent. Thus, (17) is facet-defining for $\text{conv}(\mathcal{F}_{j \omega})$. □

By restricting the feasible region of $y_j$ in (18) using the chance constraints (1d), we obtain a stronger valid inequality for (CAP) in Theorem 3.

**Theorem 3** For $k \in \Omega \setminus \{\omega\}$, let

$$
\delta_k = \max_{y_j \in (0,1)^{|I|}} \sum_{i \in C \setminus D} y_{ij} + \sum_{i \in I \setminus C} \alpha_i y_{ij} + \sum_{i \in D} \beta_i y_{ij} - |C \setminus D| - \sum_{i \in D} \beta_i + 1 
$$

subject to

$$
\sum_{i \in I} y_{ij} \leq m_j^k(k), 
$$

$$
\sum_{i \in I} y_{ij} \leq \rho_j. 
$$

Sort $\delta_k$ such that $\delta_{k_1} \leq \ldots \leq \delta_{k_{|I|-1}}$. Let $q^1 := \min \{ l | \sum_{j=1}^l p_k \beta_j > \varepsilon \}$, then the inequality (17) is valid for (CAP), where $\gamma = \delta_{k_1}$.

**Proof** Let

$$
\gamma = \max_{y_j \in (0,1)^{|I|}} \sum_{i \in C \setminus D} y_{ij} + \sum_{i \in I \setminus C} \alpha_i y_{ij} + \sum_{i \in D} \beta_i y_{ij} - |C \setminus D| - \sum_{i \in D} \beta_i + 1 
$$

subject to

$$
\sum_{k \in \Omega \setminus \{\omega\}} p_k \| \left( \sum_{i \in I} \xi_i y_{ij} \leq t_j \right) \| \geq 1 - \varepsilon, 
$$

$$
\sum_{i \in I} y_{ij} \leq \rho_j. 
$$

Because $y_j$ satisfies the chance constraint (1d) and $z_{\omega} = 0$ for computing $\gamma$, the inequality (17) is valid for (CAP).

Let $\hat{y}_j$ be an optimal solution of (20). Then, there exists at least one $k' \in \{k_1, \ldots, k_{|I|}\}$ such that $\sum_{i \in I} \xi_i y_{ij} \leq t_j$. Otherwise, if $\sum_{i \in I} \xi_i y_{ij} > t$ for all $k \in \{k_1, \ldots, k_{|I|}\}$, we have $\sum_{k \in \{k_1, \ldots, k_{|I|}\}} p_k \| (\sum_{i \in I} \xi_i y_{ij} > t) > \varepsilon$, which indicates that (20b) is violated by $\hat{y}_j$. Therefore, $\hat{y}_j$ is a feasible solution of (19) for $k = k'$. We have $\delta_{k_{|I|}} \geq \delta_{k'} \geq \gamma$, and (17) is a valid inequality for (CAP) when $\gamma = \delta_{k_{|I|}}$. □

We further restrict the feasible region of $y_j$ in (18) by using the distributionally robust chance constraints (2a) to obtain a stronger valid inequality for (DR-CAP) in the following theorem.

**Theorem 4** For $k \in \Omega \setminus \{\omega\}$, let $\delta_k$ be defined as in Theorem 3, and sort $\delta_k$ such that $\delta_{k_1} \leq \ldots \leq \delta_{k_{|I|-1}}$. Let $\hat{q}^1 := \min \{ l | \sup_{p \in \mathcal{P}} \sum_{j=1}^l p_k \beta_j > \varepsilon \}$. Then, the inequality (17) is valid for (DR-CAP) when $\gamma = \delta_{k_{|I|}}$.

Moreover, if $\{\tilde{p}_\omega\}_{\omega \in \mathcal{P}} \in \mathcal{P}$, let $\hat{q}^1 := \min \{ l | \sum_{j=1}^l \tilde{p}_k \beta_j > \varepsilon \}$. Then, $\hat{q}^1 \geq \hat{q}$ and the inequality (17) is valid for (DR-CAP) when $\gamma = \delta_{k_{|I|}}$. □
Proof Let

\[
\gamma = \max_{y_j \in \{0,1\}^{|I|}} \sum_{i \in C \setminus D} y_{ij} + \sum_{i \in I \setminus C} \alpha_i y_{ij} + \sum_{i \in D} \beta_i y_{ij} - |C \setminus D| - \sum \beta_i + 1
\]  
(21a)

subject to

\[
\inf_{p \in \mathcal{P}} \sum_{k \in \Omega(\omega)} p_k \| \left( \sum_{i \in I} \xi_{ij} y_{ij} > t \right) \| \geq 1 - \epsilon,
\]  
(21b)

\[
\sum_{i \in I} y_{ij} \leq \rho_j.
\]  
(21c)

Because \( y_j \) satisfies the chance constraint (2a) and \( z_\omega = 0 \) for computing \( \gamma \), the inequality (17) is valid for (DR-CAP).

Let \( \tilde{y}_j \) be an optimal solution of (21). Then, \( \sum_{i \in I} \xi_{ij} \tilde{y}_{ij} \leq t_j \) for at least one \( k' \in \{k_1, \ldots, k_q\} \). Otherwise, if \( \sum_{i \in I} \xi_{ij} \tilde{y}_{ij} > t_j \) for all \( k \in \{k_1, \ldots, k_q\} \), we have \( \sup_{p \in \mathcal{P}} \sum_{k \in \{k_1, \ldots, k_q\}} p_k \| \left( \sum_{i \in I} \xi_{ij} \tilde{y}_{ij} > t \right) \| > \epsilon \), which indicates that (21b) is violated by \( \tilde{y}_j \). Therefore, \( \tilde{y}_j \) is a feasible solution of (19) for \( k = k' \). We have \( \delta_{k_1} \geq \delta_{k' j} \geq \gamma \), then (17) is a valid inequality for (DR-CAP) when \( \gamma = \delta_{k_1} \).

Since \( \sup_{p \in \mathcal{P}} \sum_{j=1}^{q} p_{k_j} \geq \sum_{j=1}^{q} \tilde{p}_{k_j} > \epsilon \), we have \( \tilde{q}^1 \geq q^1 \), which implies \( \delta_{k_1} \geq \delta_{k_1} \geq \gamma \), and (17) is a valid inequality for (DR-CAP) when \( \gamma = \delta_{k_1} \). \( \square \)

We use the dynamic programming approach similar to the one described in Appendix B.1 to solve the down-lifting sequence of problems (16), and (19).

### 3.1.4 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 from Wang et al. (2019) and show that it gets strengthened in the DR framework.

**Example 1** Suppose \( F_{j,\omega} \) is defined by \( \rho_j = 3 \), \( m_j^*(\omega) = 40 \), and \( \xi_\omega = (7, 8, 11, 10, 9, 14, 23)^T \). Then the set \( C = \{1, 2, 3, 4, 5\} \) is a minimal cover. Let \( D = \{5\} \), Suppose \( N = 5 \), \( \epsilon = 0.6 \), and the other scenarios in the computation of lifted cover inequalities are \( (8, 11, 7, 10, 7, 17, 23)^T \), \( (14, 7, 10, 11, 8, 13, 26)^T \), \( (21, 10, 7, 29, 16, 12, 23)^T \), and \( (15, 7, 8, 23, 12, 10, 5)^T \), with \( p_\omega = 1/N \) for all \( \omega \in \Omega \). We get a lifted cover inequality by Theorem 3 as:

\[
y_{1j} + y_{2j} + y_{3j} + y_{4j} + y_{5j} + 2y_{6j} + 2y_{7j} + z_{j,\omega} \leq 5.
\]  
(22)

If \( p_\omega^* = 1/N \) for all \( \omega \in \Omega \) and \( \eta = 0.5 \) in the Wasserstein set, then a lifted cover inequality for (DR-CAP) obtained from Theorem 4 is

\[
y_{1j} + y_{2j} + y_{3j} + y_{4j} + y_{5j} + 2y_{6j} + 2y_{7j} \leq 4.
\]  
(23)
Example 2 (Continued) Suppose that the cardinality constraint \( \sum_{i \in I} y_{ij} \leq \rho_j \) is removed from \( F_j \).

Following a computation procedure similar to the one for the lifted cover inequality, we can have a valid inequality of the following form:

\[
y_{1j} + y_{2j} + y_{3j} + y_{4j} + y_{5j} + y_{6j} + 2y_{7j} + z_{j\omega} \leq 5.
\]

We call the inequality (24) single lifted cover inequality. Obviously, the lifted cover inequality (22) is stronger than the single lifted cover inequality (24). The single lifted cover inequality for (DR-CAP) is

\[
y_{1j} + y_{2j} + y_{3j} + y_{4j} + y_{5j} + y_{6j} + 2y_{7j} \leq 4.
\]

(25) (23) is also stronger than (25). Thus showing the possible benefit of using the cardinality constraint 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 \( G_j \) and \( G'_j \), which are valid for (CAP) and (DR-CAP), respectively. For (CAP), let \( \Omega \) be a set where each element \( \Omega_k \in \tilde{\Omega} \) is a subset of \( \Omega \) such that \( \sum_{\omega \in \Omega_k} p_\omega \geq 1 - \epsilon \), for \( k = 1, \ldots, |\Omega| \). Without loss of generality, we abuse the notation and re-use the set \( \tilde{\Omega} \) and \( \Omega_k \) for (DR-CAP). For (DR-CAP), let \( \Omega \) be a set where each element \( \Omega_k \in \tilde{\Omega} \) is a subset of \( \Omega \) such that \( \inf_{p \in \mathcal{P}} \sum_{\omega \in \Omega_k} p_\omega \geq 1 - \epsilon \), for \( k = 1, \ldots, |\Omega| \). \( \tilde{\Omega} \) is maximal if it is not a proper subset of any other sets that satisfy the above condition. For maximal \( \tilde{\Omega} \), let the global lifted cover inequalities be of the form

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

where \( \mathcal{C} \) is a cover for the set \( Q_{j\omega} \) for some \( \omega \in \Omega \), and \( \mathcal{D} \subseteq \mathcal{C} \). For \( k \in \{1, \ldots, |\Omega|\} \), when \( z_{j\omega} = 1 \), \( \omega \in \Omega_k \), (26) becomes

\[
\sum_{i \in \mathcal{C} \setminus \mathcal{D}} y_{ij} + \sum_{i \in \mathcal{I} \setminus \mathcal{C}} \bar{\alpha}_i y_{ij} + \sum_{i \in \mathcal{D}} \bar{\beta}_i y_{ij} \leq |\mathcal{C} \setminus \mathcal{D}| + \sum_{i \in \mathcal{D}} \bar{\beta}_i - 1,
\]

which can be obtained from the multidimensional knapsack problems. Kaparis and Letchford (2008) developed a valid inequality for multidimensional knapsack problems. In Section 3.2.1 and 3.2.2, we use the idea from Kaparis and Letchford (2008) to calculate the coefficients \( \bar{\alpha}_i \) and \( \bar{\beta}_i \) in (27).

3.2.1 Up-Lifting

For \( l = 1, \ldots, |\mathcal{I} \setminus \mathcal{C}| \), let \( \tilde{\pi}_l \) be a sequence of \( \mathcal{I} \setminus \mathcal{C} \). The up-lifting problem is as follows:
\[ \text{obj}_{\pi_i} := \maximize_{y_j} \sum_{i \in C \setminus D} y_{ij} + \sum_{i = \pi_i}^{\bar{\pi}_{l-1}} \bar{\alpha}_i y_{ij} \]  

subject to \[ \sum_{i \in C \setminus D} \xi^\omega_{ij} y_{ij} + \sum_{i = \pi_i}^{\bar{\pi}_{l-1}} \xi^\omega_i y_{ij} \leq m^\omega_j(\omega) - \xi^\omega_{\bar{\pi}_l} - \sum_{i \in D} \xi^\omega_i, \quad \forall \omega \in \Omega_k, \]  

\[ \sum_{i \in C} y_{ij} + \sum_{i = \pi_i}^{\bar{\pi}_{l-1}} y_{ij} \leq \rho_j - 1 - |D|, \]  

\[ y_{ij} \in \{0, 1\}, \quad \forall i \in C \bigcup \{\bar{\pi}_1, \ldots, \bar{\pi}_{l-1}\}. \]  

Then \( \bar{\alpha}_{\pi_i} = |C \setminus D| - 1 - \text{obj}_{\pi_i} \). 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 (28). Kaparis and Letchford (2008) suggest relaxing \( y_j \in [0, 1]^{|I|} \) and solving the LP relaxation to compute an upper bound on \( \text{obj}_{\pi_i} \). The objective value is then rounded down to the nearest integer. In order to make use of Algorithm 3 in Appendix B.2, we propose a heuristic to calculate \( \bar{\alpha}_{\pi_i} \). For each \( \omega \in \Omega_k \), let

\[ \text{obj}_{\pi_i}(\omega) := \maximize_{y_j} \sum_{i \in C \setminus D} y_{ij} + \sum_{i = \pi_i}^{\bar{\pi}_{l-1}} \bar{\alpha}_i y_{ij} \]  

subject to \[ \sum_{i \in C \setminus D} \xi^\omega_{ij} y_{ij} + \sum_{i = \pi_i}^{\bar{\pi}_{l-1}} \xi^\omega_i y_{ij} \leq m^\omega_j(\omega) - \xi^\omega_{\bar{\pi}_l} - \sum_{i \in D} \xi^\omega_i, \]  

\[ \sum_{i \in C} y_{ij} + \sum_{i = \pi_i}^{\bar{\pi}_{l-1}} y_{ij} \leq \rho_j - 1 - |D|, \]  

\[ y_{ij} \in \{0, 1\}, \quad \forall i \in C \bigcup \{\pi_1, \ldots, \pi_{l-1}\}. \]  

Then, \( \text{obj}_{\pi_i}(\omega) \) is an upper bound for \( \text{obj}_{\pi_i} \). Algorithm 3 in Appendix B.2 is used to compute \( \text{obj}_{\pi_i}(\omega) \), for \( \omega \in \Omega_k \). We use \( \min_{\omega \in \Omega_k} \text{obj}_{\pi_i}(\omega) \) to obtain a minimal upper bound for \( \text{obj}_{\pi_i} \), from among the values \( \{\text{obj}_{\pi_i}(\omega)\}_{\omega \in \Omega_k} \). Let \( \bar{\alpha}_{\pi_i} = |C \setminus D| - 1 - \min_{\omega \in \Omega_k} \text{obj}_{\pi_i}(\omega) \), which implies \( \bar{\alpha}_{\pi_i} \leq |C \setminus D| - 1 - \text{obj}_{\pi_i} \). Thus, \( \bar{\alpha}_{\pi_i} \) is a valid lifting coefficient.

### 3.2.2 Down-Lifting

Similarly, we can obtain the down-lifting coefficient \( \bar{\beta}_i \), \( i \in D \). For \( l = 1, \ldots, |D| \), let \( \bar{\kappa}_l \) be a sequence of \( D \), and
Appendix B.1 to compute obj of problem (29) for ω
Finally, for calculating 3.2.3 Global Lifted Cover Inequalities
Instead of computing obj given. Instead of solving (30) exactly, we provide a simple heuristic to get an upper bound for The calculation of y, and objective value
κ|l, z, ...|. Let obj(29) and objective value
τj l= maximize , for
l= 1, j=1
z=1
κ|l ∈{κ_{t+1}, ..., κ_{|D|}}.

Instead of computing obj, we use the heuristic proposed in Section 3.2.1 to obtain an upper bound for obj. Let obj(ω) be the optimal objective value of the maximization problem that takes a single row ω of problem (29) for ω ∈ Ωk. We use a dynamic programming approach similar to the one proposed in Appendix B.1 to compute obj(ω), and let \( \hat{\beta}_{k_l} = \min_{\omega \in \Omega_k} \text{obj}_{k_l}(\omega) - \sum_{i=k_{l+1}}^{k_{l-1}} \beta_i - |C\setminus D| + 1. \)

3.2.3 Global Lifted Cover Inequalities

Finally, for calculating \( \tau_{j_l} \) in a sequence \( \tau = \{\tau_1, \ldots, \tau_{|\Omega|}\} \), we solve the following problem for \( G_j \), for \( l = 1, \ldots, |\Omega_k| \):

\[
\text{obj}_{\tau_l} = \max_{(y_j, z_j) \in \{0,1\}^{|I|} \times \{0,1\}^{|K|}} \sum_{i \in C \setminus D} y_{ij} + \sum_{i \in I \setminus C} \alpha_i y_{ij} + \sum_{i \in D} \beta_i y_{ij} + \sum_{\omega = \tau_1}^{\tau_{l-1}} \gamma_{\omega} z_{j\omega}
\]

subject to \sum_{\omega \in \Omega} \sum_{i \in I} \xi_{ij} y_{ij} \leq m^\omega_j(\omega), \quad \forall \omega \in \Omega_k,

\sum_{i \in I} y_{ij} \leq \rho_j, \quad y_{k_{l+1}} = 0, \quad y_{ij} = 1, \quad \forall i \in \{k_{l+1}, \ldots, k_{|D|}\}. \quad (29c)

The calculation of obj is a reformulation of a chance-constrained problem where some variables \( z_{j \omega} \) are given. Instead of solving (30) exactly, we provide a simple heuristic to get an upper bound for obj. We relax \( y_j \in [0,1]^{|I|} \) and \( z_j \in [0,1]^{|\Omega_k|} \), and solve the LP relaxation of (30) to obtain the optimal solution \((y^*_j, z^*_j)\) and objective value \( \text{obj}^*_\tau_l \) of the relaxed problem. Then, \( \text{obj}^\tau_l \) gives an upper bound for \( \text{obj}_{\tau_l} \).

For \( G'_j \), for \( l = 1, \ldots, |\Omega_k| \), let

\[
\text{obj}^\tau_l := \max_{(y_j, z_j) \in \{0,1\}^{|I|} \times \{0,1\}^{|K|}} \sum_{i \in C \setminus D} y_{ij} + \sum_{i \in I \setminus C} \alpha_i y_{ij} + \sum_{i \in D} \beta_i y_{ij} + \sum_{\omega = \tau_1}^{\tau_{l-1}} \gamma_{\omega} z_{j\omega}
\]

subject to \inf_{p \in \Omega} \sum_{\omega \in \Omega} \sum_{i \in I} \xi_{ij} y_{ij} \leq m^\omega_j(\omega) - m^\omega_j(k_q), \quad \forall \omega \in \Omega. \quad (31c)

18
We also provide a heuristic to get an upper bound on $obj_{\bar{r}_l}$. We relax $y_j \in [0,1]^{|D|}$ and $z_j \in [0,1]^{|\Omega_k|}$, and use probability cut Algorithm 4, presented in Appendix B.3, to solve the relaxation problem of (31) and obtain the optimal objective value $obj_{\bar{r}_l}'$, which is an upper bound on $obj_{r_l}$.

The following theorem gives valid inequalities for $\text{conv}(G_j)$ and $\text{conv}(G'_j)$.

**Theorem 5** Let $\{\bar{\alpha}_i\}_{i \in I \setminus C}$ and $\{\bar{\beta}_i\}_{i \in D}$ be defined as in Section 3.2.1 and 3.2.2, respectively. For $l = 1, \cdots, \vert \Omega_k \vert$, we set $\bar{\tau}_{r_l} = \lceil obj_{\bar{r}_l} \rceil - |C \setminus D| + 1 - \sum_{i \in D} \bar{\beta}_i - \sum_{\omega = r_l}^{\tau_{r_l} - 1} \bar{\gamma}_\omega$, where $obj_{\bar{r}_l}$ is the objective value of the LP relaxation of (30). Then, (26) is valid for $\text{conv}(G_j)$. For $l = 1, \cdots, \vert \Omega_k \vert$, we set $\bar{\tau}_{r_l} = \lceil obj_{\bar{r}_l} \rceil - |C \setminus D| + 1 - \sum_{i \in D} \bar{\beta}_i - \sum_{\omega = r_l}^{\tau_{r_l} - 1} \bar{\gamma}_\omega$, where $obj_{\bar{r}_l}'$ is the objective value of the LP relaxation of (31). Then, (26) is valid for $\text{conv}(G'_j)$.

**Proof** We first prove that for (CAP) if the coefficients are described in Theorem 5, then, (26) is valid for $\text{conv}(G_j)$. For $k \in \{1, \cdots, \bar{\Omega}\}$, let $(\bar{y}_j, \bar{z}_j) \in G_j$. If $\bar{z}_{j\omega} = 1$ for $\omega \in \Omega_k$, then (26) is valid for $\text{conv}(G_j)$. Otherwise, let $\hat{\tau}$ be partitioned into $\Omega_k^0 = \{\omega \in \hat{\tau} | \bar{z}_{j\omega} = 0\}$ and $\Omega_k^1 = \{\omega \in \hat{\tau} | \bar{z}_{j\omega} = 1\}$. We assume that the last element of $\Omega_k^0$ is $\bar{\tau}_h$ where $h \leq \vert \Omega_k \vert$. (26) becomes

$$\sum_{i \in \Omega_k^0} \bar{y}_i + \sum_{i \in \Omega_k^1} \bar{\alpha}_i \bar{y}_i + \sum_{i \in D} \bar{\beta}_i \bar{y}_i \leq |C \setminus D| + \sum_{i \in D} \bar{\beta}_i - 1 + \sum_{\omega \in \Omega_k^0} \bar{\gamma}_\omega.$$ 

Note that

$$|C \setminus D| + \sum_{i \in D} \bar{\beta}_i - 1 + \sum_{\omega \in \Omega_k^0} \bar{\gamma}_\omega = obj_{\bar{r}_h} - \sum_{\omega = r_1}^{\tau_{r_h} - 1} \bar{\gamma}_\omega + \sum_{\omega \in \Omega_k^0 \setminus \{r_h\}} \bar{\gamma}_\omega.$$ 

Since $(\bar{y}_j, \bar{z}_j)$ satisfies (30) with $k = h$, we have

$$\begin{align*} 
obj_{\bar{r}_h} - \sum_{\omega = r_1}^{\tau_{r_h} - 1} \bar{\gamma}_\omega + \sum_{\omega \in \Omega_k^0 \setminus \{r_h\}} \bar{\gamma}_\omega \\
\geq \sum_{i \in C \setminus D} \bar{y}_i + \sum_{i \in I \setminus C} \bar{\alpha}_i \bar{y}_i + \sum_{i \in D} \bar{\beta}_i \bar{y}_i - \sum_{\omega = r_1}^{\tau_{r_h} - 1} \bar{\gamma}_\omega + \sum_{\omega \in \Omega_k^0 \setminus \{r_h\}} \bar{\gamma}_\omega \\
= \sum_{i \in C \setminus D} \bar{y}_i + \sum_{i \in I \setminus C} \bar{\alpha}_i \bar{y}_i + \sum_{i \in D} \bar{\beta}_i \bar{y}_i. 
\end{align*}$$ 

Thus, (26) is valid for $\text{conv}(G_j)$ when $\bar{\tau}_{r_l} = \lceil obj_{\bar{r}_h} \rceil - |C \setminus D| + 1 - \sum_{i \in D} \bar{\beta}_i - \sum_{\omega = r_1}^{\tau_{r_h} - 1} \bar{\gamma}_\omega$ for $l = 1, \cdots, \vert \Omega_k \vert$. Since $obj_{\bar{r}_l}'$ is an upper bound on $obj_{\bar{r}_l}$ and all the coefficients in (26) are integers, $\lceil obj_{\bar{r}_l}' \rceil$ is also an upper bound on $obj_{\bar{r}_l}$. Therefore, (26) is valid for $\text{conv}(G_j)$ when $\bar{\tau}_{r_l} = \lceil obj_{\bar{r}_l}' \rceil - |C \setminus D| + 1 - \sum_{i \in D} \bar{\beta}_i - \sum_{\omega = r_1}^{\tau_{r_l} - 1} \bar{\gamma}_\omega$ for $l = 1, \cdots, \vert \Omega_k \vert$. The proof can be similarly extended to $G'_j$. This completes the proof. □

The following example gives a global lifted cover inequality.

**Example 3 (Continued from Example 1)** We set cover $C = \{1, 2, 3, 4, 5\}$ and $D = \{5\}$ as before. Let $\Omega_k = \{1, 2\}$. Then we can obtain a global lifted cover inequality (26) for (CAP) as follows

$$y_{1j} + y_{2j} + y_{3j} + y_{4j} + 2y_{6j} + 2y_{7j} + z_{1j} + z_{2j} \leq 5.$$
For (DR-CAP), the set $\Omega_k = \{1, 2, 3\}$ satisfies $\inf_{p \in \mathcal{P}} \sum_{\omega \in \Omega_k} p_\omega \geq 1 - \varepsilon$. A global lifted cover inequality (26) is given by

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

4 Solution Scheme

In Section 4.1, we present a heuristic sequential lifting procedure for separating the valid inequalities developed in Section 3, and show how the valid inequalities can be used within a branch-and-cut framework to solve (CAP) in Section 4.2. A branch-and-cut algorithm with probability cuts to solve (DR-CAP) is given in Section 4.3.

4.1 Separation Problem

Separation problem is to find the inequalities that are violated by a LP relaxation solution $(\hat{y}, \hat{z})$. In this section, we adopt the ideas from Gu et al. (1998) and Kaparis and Letchford (2008) for the binary knapsack problem to separate the valid inequalities (17) and (26), respectively.

4.1.1 Separation Problem for (17)

For obtaining the violated inequalities (17), we use a heuristic similar to the one in Gu et al. (1998) for the knapsack problem to solve the separation problem. This heuristic is provided in Algorithm 1.

Algorithm 1: Separation Heuristic for (17)

1. Given the LP relaxation optimal solution $(\hat{y}, \hat{z})$.
2. for $j = 1, \ldots, |J|$ do
3.   for $\omega = 1, \ldots, N$ do
4.     if $z_{j\omega} = 1$ then
5.       Sort $\hat{y}_{ij}$ in non-increasing order: $\hat{y}_{i1} \geq \ldots \geq \hat{y}_{i|I|}$.
6.       Let $C = \{i_1, \ldots, i_o\}$ where $o \leq |I|$ is a smallest number such that $C$ is a cover.
7.       Delete elements from the tail of $C$ to get a minimal cover $C$.
8.       Let $D = \{i \in C : \hat{y}_{ij} = 1\}$ and $I_0 = \{i \in \mathcal{T}_C[\hat{y}_{ij} = 0]\}$.
9.       Calculate the up-lifting coefficient $\alpha_i$ for $i \in \mathcal{I} \setminus (C \cup I_0)$.
10.      if $\sum_{i \in C \setminus D} \hat{y}_{ij} + \sum_{i \in \mathcal{I} \setminus (C \cup I_0)} \alpha_i \hat{y}_{ij} > |C \setminus D| - 1$ then
11.         Calculate the down-lifting coefficient $\beta_i$ for $i \in D$.
12.         Calculate the up-lifting coefficient $\alpha_i$ for $i \in I_0$.
13.         Calculate $\delta_k$ for $k \in \Omega \setminus \omega$, set $\gamma = \delta_{kq}$ for (CAP) and $\gamma = \delta_{kq}$ for (DR-CAP).
14.      Obtain the violated inequality (17).
15. end
16. end
17. end
18. end

Note that

$$\sum_{i \in C \setminus D} \hat{y}_{ij} + \sum_{i \in \mathcal{I} \setminus C} \alpha_i \hat{y}_{ij} + \sum_{i \in D} \beta_i (\hat{y}_{ij} - 1) + \gamma (\hat{z}_{j\omega} - 1) = \sum_{i \in C \setminus D} \hat{y}_{ij} + \sum_{i \in \mathcal{I} \setminus (C \cup I_0)} \alpha_i \hat{y}_{ij} > |C \setminus D| - 1.$$
Hence, we obtain an inequality that is violated by the LP relaxation solution.

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

4.1.2 Separation Problem for (26)

We develop a heuristic procedure similar to the one in Kaparis and Letchford (2008) for the multidimensional knapsack problem, to obtain the violated inequalities (26). Assume that we are given a LP relaxation solution $(\hat{y}, \hat{z})$ of (IP), for $j \in J$, let $\Omega_1 = \{ \omega \in \Omega | \hat{z}_{j\omega} = 1 \}$ such that $\sum_{\omega \in \Omega_1} \rho_\omega \hat{z}_{j\omega} \geq 1 - \epsilon$ for (CAP) and $\inf_{\rho \in \mathbb{R}^{|D|}} \sum_{\omega \in \Omega_1} \rho_\omega \hat{z}_{j\omega} \geq 1 - \epsilon$ for (DR-CAP), which indicates that $\Omega_1 \in \bar{\Omega}$. Then, we sort $\hat{y}_j$ in a nonincreasing order: $\hat{y}_{i_1j} \geq \ldots \geq \hat{y}_{i_oj}$. Let $C = \{ i_1, \ldots, i_o \}$ be such that $C$ is a cover for some $\omega \in \Omega_1$, and we delete elements from the tail of the set $C$ until the cover $C$ is minimal. Let $D = \{ i \in C | \hat{y}_{ij} = 1 \}$ and $I_0 = \{ i \in I \setminus C | \hat{y}_{ij} = 0 \}$. We first use the up-lifting procedure on the variables $y_j$ in $I \setminus (C \cup I_0)$. If the inequality $\sum_{i \in C \setminus D} \hat{y}_{ij} + \sum_{i \in I \setminus (C \cup I_0)} \alpha_i \hat{y}_{ij} > |C \setminus D| - 1$, then, we use the down-lifting procedure for variables $y_j$ in $D$ and the up-lifting procedure for variables $y_j$ in $I_0$. Finally, we compute the coefficients of $z_{j\omega}$ for $\omega \in \Omega_1$. Algorithm 5 in Appendix B.4 gives an overview of this heuristic.

Similar to Section 4.1.1, if $|D| > \rho_j - 1$ or $m_j^\omega(\omega) - \sum_{i \in D} \xi_i^\omega - \max_{i \in I \setminus C} \xi_i^\omega < 0$ for some $\omega \in \Omega$, we remove the items from $D$ until $|D| \leq \rho_j - 1$ and $m_j^\omega(\omega) - \sum_{i \in D} \xi_i^\omega - \max_{i \in I \setminus C} \xi_i^\omega \geq 0$ 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). Let LB and UB denote the current lower and upper bound of (CAP), and $N$ denote the set of remaining nodes in the branch-and-cut search tree. An overview of the branch-and-cut framework is given in Algorithm 2. The algorithm uses the violated inequalities described in Section 4.1.1 and 4.1.2 in step 9 (see Section 5.2 for further discussion).

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:

\[(MP) \quad \minimize_{y,z} \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij} \]

subject to (1b), (1c), (7c), (7d)

\((y,z) \in Y,\)
Algorithm 2: Branch-and-Cut Implementation

1. Initialize UB = +∞, LB = −∞, the iteration number k = 0.
2. Initialize Node list \( N = \{o\} \), where o is a branching node without constraints.

while \( N \) is nonempty do

3. Select a node o \( \in N \), \( N \leftarrow N / \{o\} \).
4. At the node o, solve the LP relaxation problem of (IP). \( k = k + 1 \).
5. Obtain the optimal solution \((y^k, z^k)\) and the objective value \( \text{obj}^k \).

6. if \( \text{obj}^k < UB \) then

7. if \((y^k, z^k)\) is fractional then

8. Use Algorithm 1 and Algorithm 5 to find the violated inequalities.
9. if Violated inequalities are found then
10. Add the violated inequalities to the LP relaxation problem.
11. Go to line 5.
12. end
13. else
14. Branch, resulting in nodes \( o^* \) and \( o^{**} \), \( N \leftarrow N \cup \{o^*, o^{**}\} \).
15. end
16. end
17. else
18. Update UB, \( UB = \text{obj}^k \), \((y^*, z^*) = (y^k, z^k)\).
19. end
20. end
21. end
22. end
23. return UB and its corresponding optimal solution \((y^*, z^*)\).

where set \( Y \) is complementary set for defining the feasible region of (7). Set \( Y \) is defined by a set of probability and feasibility cuts. Let \((\hat{y}, \hat{z})\) be a feasible solution of (MP). For \( j \in J \), a distribution separation problem is given by:

\[
(S_{j}) \quad S_{j}(\hat{z}) := \min_{p \in \mathcal{P}} \sum_{\omega \in \Omega} p_{\omega} \hat{z}_{j\omega}
\]

The problem \((S_{j})\) is used to verify that whether \((\hat{y}_j, \hat{z}_j)\) is feasible to (DR-CAP). If \( S_{j}(\hat{z}) \geq 1 - \varepsilon \), \((\hat{y}_j, \hat{z}_j)\) is feasible to (DR-CAP). Otherwise, the probability and feasibility cuts are added to (MP).

Let \( \{\hat{p}_\omega\}_{\omega \in \Omega} \) be an optimal solution of \((S_{j})\) corresponding to \( \hat{z} \), a probability cut is given by

\[
\sum_{\omega \in \Omega} \hat{p}_\omega z_{j\omega} \geq 1 - \varepsilon. \tag{33}
\]

Let \( I^1_j = \{i \in I | y_{ij} = 1\} \). We also add the following feasibility cut in \( y \) variables to (MP):

\[
\sum_{i \in I^1_j} y_{ij} \leq |I^1_j| - 1. \tag{34}
\]

Algorithm 6 in Appendix B.5 gives a pseudocode of the branch-and-cut algorithm with probability cuts.

In Algorithm 6, 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 \( N \) 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 $(y^k, z^k)$ 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 whether $(y^k, z^k)$ is binary (line 7). If $(y^k, z^k)$ is binary, we solve the distribution separation problem (SP$_j$) with the ambiguity set $\mathcal{P}$ for all $j \in J$, and obtain the optimal solution $\{p^k_{\omega}\}_{\omega \in \Omega}$ 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, $(y^k, z^k)$ is a feasible solution to (DR-CAP), we update the upper bound and record the corresponding solution $(y^k, z^k)$ (line 15-20). If $(y^k, z^k)$ is fractional, we continue branching. New nodes are obtained and added to the node list $N$ (line 22-26). We terminate our algorithm when the node list is empty, and return the optimal value UB and the optimal solution $(y^*, z^*)$ (line 29).

The following theorem shows that Algorithm 6 terminates in a finite number of iterations for solving (DR-CAP) to optimality under certain condition.

**Theorem 6** Let the ambiguity set $\mathcal{P}$ be defined by a polytope with a finite number of extreme points, Algorithm 6, presented in Appendix B.5, terminates in finitely many iterations. If $UB < +\infty$, UB is the optimal value of (DR-CAP) and obtain an optimal solution $(y^*, z^*)$ at termination.

**Proof** The algorithm processes a finite number of nodes as it is based on branching on a finite number of binary variables, and given that $z^k$ is obtained from (MP), we solve $|J|$ distribution separation problems with $\mathcal{P}$. When $\mathcal{P}$ is defined by a polytope with a finite number of extreme points, the distribution separation problems are finitely convergent. In addition, the set of feasibility cuts generated in line 12 is finite. Thus, Algorithm 6 terminates in finitely many iterations.

Next, we show that the cuts (33) and (34) can remove the current infeasible solution and never cut off any feasible solutions of (DR-CAP). It can be verified that (33) and (34) can remove the current infeasible solution. Also,

$$\sum_{\omega \in \Omega} p^k_{\omega} z_{j\omega} \geq \inf_{p \in \mathcal{P}} \sum_{\omega \in \Omega} p_{\omega} z_{j\omega} \geq 1 - \varepsilon.$$ 

Thus, (33) never cuts off any feasible solutions of (DR-CAP). We assume that $\tilde{y}$ is a new future solution from (MP) and the corresponding set $\tilde{I}^1_j$. Let $y_{ij} = \tilde{y}_{ij}$, for $i \in I$. Then the feasibility cut (34) becomes

$$\sum_{i \in \tilde{I}^1_j} \tilde{y}_{ij} \leq |\tilde{I}^1_j| - 1,$$

which is decomposed to

$$\sum_{i \in \tilde{I}^1_j \cap \tilde{I}^1_j} \tilde{y}_{ij} + \sum_{i \in \tilde{I}^1_j \setminus \tilde{I}^1_j} \tilde{y}_{ij} \leq |\tilde{I}^1_j \cap \tilde{I}^1_j| + |\tilde{I}^1_j \setminus \tilde{I}^1_j| - 1 \iff \sum_{i \in \tilde{I}^1_j \setminus \tilde{I}^1_j} \tilde{y}_{ij} \leq |\tilde{I}^1_j \setminus \tilde{I}^1_j| - 1,$$
If $I_j^1 \subseteq \tilde{I}_j^1$, $\tilde{y}$ is not a feasible solution, and does not satisfy the feasibility cut. Otherwise, $\sum_{i \in I_j^1 \setminus \tilde{I}_j^1} \tilde{y}_{ij} = 0$ and $|I_j^1 \setminus \tilde{I}_j^1| - 1 \geq 0$. Thus, the feasibility cut holds. This completes the proof. □

5 Computational Experiments

We now present computational results for (CAP) and (DR-CAP). Computational experiments were performed using data from 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 5.1. Section 5.2 provides additional implementation details. Performance of the branch-and-cut algorithm (Algorithm 2) for solving (CAP) is discussed in Section 5.3 and that of the branch-and-cut algorithm with probability cuts (Algorithm 6) for solving (DR-CAP) is discussed in Section 5.4. Section 5.5 compares the out-of-sample performance of the solutions generated from the (DR-CAP) instances with the corresponding (CAP) instances.

5.1 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. Table 1 provides the mean and standard deviation of the surgery duration, and the percentage for each surgery type. For the problem instances the log-normal distribution with the mean, and the standard deviation provided in Table 1 was used to generate surgery duration samples (see 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 ($|J| = 8$) ORs are available to serve $|I| = 27$ surgeries (close to the maximum number of surgeries in a day) a day. The daily time limit $t_j$ is 10 hours, $\forall j \in J$. Following Zhang et al. (2018), we let the assignment cost $c_{ij}$ vary in [0,16]. The number of surgeries in an OR $\rho_j$ is limited to [3,5], $\forall i \in I, j \in J$. We used the number of surgeries and the percentage for each surgery type (given in Table 1) 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 $i \in I$ 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 and chance level.

5.2 Implementation Details

In our implementation of the branch-and-cut algorithm, we add the violated valid inequalities generated from (17) at the nodes that are at a depth no more than 1. There was no limit to the number of such inequalities added to the formulation. We observed that it is more time-consuming to find a violated
Table 1: For each surgery type, the mean (mean), standard deviation (std) in hours, and the percentage for each surgery type (percentage) are reported

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

Inequality of the type (26). Therefore, we added the violated inequalities from (26) 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. At each round of the cut generation of the type (17), for each $j \in J$, 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 programming language C using IBM CPLEX, 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 CPLEX presolve procedure and set the number of threads to one for all computations because 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 $y$ were given a higher priority than $z$. 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 this 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 bounds, 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 dynamic programming approach described in Appendix B.1 to compute $m^ω_j(k)$ for $j \in J$ and $ω, k \in Ω$ (defined in problem (5)). An easier way to compute $m^ω_j(k)$ is to let $m^ω_j(k) = \maximize{y_j \in \{0,1\}^{|I|}, \sum_{i \in I} \xi^ω_i y_{ij} \sum_{i \in I} \xi^k_i y_{ij} \leq t_j}$, 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. These results demonstrate the computational trade-offs resulting from using this weaker upper bound.
5.3 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 (Algorithm 2) to solve (CAP) without adding valid inequalities proposed in this paper.
- Cover-1: refers to adding the single lifted cover inequalities to the branch-and-cut algorithm (Algorithm 2). They are obtained by ignoring the cardinality constraint in the coefficient calculation procedures.
- Cover-2: refers to adding the lifted cover inequalities (17) to the branch-and-cut algorithm (Algorithm 2).
- Cover-G: refers to adding the global lifted cover inequalities (26) to the branch-and-cut algorithm (Algorithm 2).

Table 2 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. Note that the average total time spent to solve these instances is the sum of the average times for the branch-and-cut algorithm and the big-M coefficients.

We see from Table 2 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 all instances to optimality within the runtime limit, whereas, CPX can only solve four of 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 $\varepsilon = 0.06$, while minimizing the total cost. Nevertheless, for these problems, adding the single cover and lifted cover inequalities 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 single cover and lifted cover inequalities are more apparent for these instances, and here the use of lifted cover inequalities saves computation time over the single cover inequalities. 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 lifted cover 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 the lifted cover inequalities outperforms other variants and yields more stable performance for most instances.

The use of global cover inequalities 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 a fewer number of nodes (54,969 versus 69,798) when compared to the lifted cover inequality 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 node reduction. There are several instances where the use of the global cover inequalities 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.

### 5.4 Computational Results for (DR-CAP)

We implemented Algorithm 6 to solve the semi-infinite reformulation (17) of (DR-CAP). Using the sample average distribution, we let $\overline{q} := \hat{q}$ (Corollary 1) for the big-M calculations in (7). For (17), we set the coefficient $\gamma'$ as $\delta_{k_{i1}}$. We update $\hat{q}$ as new $\{p_{\omega}\}_{\omega \in \Omega}$ becomes available in the probability cuts. The following variants of Algorithm 6 are considered:

- **CPX**: refers to using the branch-and-cut algorithm with probability cuts (Algorithm 6) to solve (DR-CAP) without any valid inequalities proposed in this paper.
- **Cover-1**: refers to adding the single lifted cover inequalities to the branch-and-cut algorithm with probability cuts (Algorithm 6).
- **Cover-2**: refers to adding the valid inequalities (17) to the branch-and-cut algorithm with probability cuts (Algorithm 6).

We solved the instances generated in Section 5.3 with the Wasserstein set $\mathcal{P}_W$ 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 3 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.

Similar to the case of (CAP), the results in Table 3 show that adding the lifted cover inequalities yields significant improvement over CPX and the single lifted cover inequality variants in two of the three harder instance sets ($\eta = 0.1$, and $\eta = 1$, $N = 1500$). However, the average performance was worse for the instances ($\eta = 0.5, N = 1500$). A comparison of the results in Tables 2 and 3 show that the time required to solve (DR-CAP) is approximately (at most) four times the time required to solve (CAP).
Table 2: 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.

<table>
<thead>
<tr>
<th>$\epsilon$</th>
<th>N</th>
<th>approach</th>
<th>AvT-M</th>
<th>AvT-B&amp;C</th>
<th>AvT-Cut</th>
<th># of nodes</th>
<th># of cuts</th>
<th>solved</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.12</td>
<td>500</td>
<td>CPX</td>
<td>165.0</td>
<td>52.8</td>
<td>-</td>
<td>1,725</td>
<td>-</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-1</td>
<td>165.0</td>
<td>33.3</td>
<td>1.7</td>
<td>1,446</td>
<td>283</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-2</td>
<td>165.0</td>
<td>47.1</td>
<td>14.7</td>
<td>1,076</td>
<td>300</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-G</td>
<td>165.0</td>
<td>65.5</td>
<td>8.6</td>
<td>1,959</td>
<td>9</td>
<td>5/5</td>
</tr>
<tr>
<td>0.12</td>
<td>1000</td>
<td>CPX</td>
<td>641.1</td>
<td>135.8</td>
<td>-</td>
<td>1,827</td>
<td>-</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-1</td>
<td>641.1</td>
<td>66.5</td>
<td>4.4</td>
<td>1,557</td>
<td>348</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-2</td>
<td>641.1</td>
<td>109.8</td>
<td>32.6</td>
<td>1,846</td>
<td>345</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-G</td>
<td>641.1</td>
<td>219.5</td>
<td>37.7</td>
<td>2,847</td>
<td>9</td>
<td>5/5</td>
</tr>
<tr>
<td>0.12</td>
<td>1500</td>
<td>CPX</td>
<td>1,439.3</td>
<td>781.4</td>
<td>-</td>
<td>5,094</td>
<td>-</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-1</td>
<td>1,439.3</td>
<td>659.4</td>
<td>10.9</td>
<td>10,788</td>
<td>563</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-2</td>
<td>1,439.3</td>
<td>502.4</td>
<td>76.0</td>
<td>3,756</td>
<td>561</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-G</td>
<td>1,439.3</td>
<td>739.5</td>
<td>101.0</td>
<td>3,715</td>
<td>12</td>
<td>5/5</td>
</tr>
<tr>
<td>0.1</td>
<td>500</td>
<td>CPX</td>
<td>165.0</td>
<td>140.7</td>
<td>-</td>
<td>3,477</td>
<td>-</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-1</td>
<td>165.0</td>
<td>122.5</td>
<td>1.2</td>
<td>4,638</td>
<td>210</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-2</td>
<td>165.0</td>
<td>126.0</td>
<td>10.7</td>
<td>3,641</td>
<td>224</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-G</td>
<td>165.0</td>
<td>136.7</td>
<td>10.6</td>
<td>2,926</td>
<td>12</td>
<td>5/5</td>
</tr>
<tr>
<td>0.1</td>
<td>1000</td>
<td>CPX</td>
<td>641.1</td>
<td>523.5</td>
<td>-</td>
<td>6,492</td>
<td>-</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-1</td>
<td>641.1</td>
<td>329.2</td>
<td>4.3</td>
<td>5,919</td>
<td>346</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-2</td>
<td>641.1</td>
<td>305.3</td>
<td>29.2</td>
<td>5,882</td>
<td>320</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-G</td>
<td>641.1</td>
<td>481.0</td>
<td>42.8</td>
<td>5,669</td>
<td>12</td>
<td>5/5</td>
</tr>
<tr>
<td>0.1</td>
<td>1500</td>
<td>CPX</td>
<td>1,439.3</td>
<td>1,868.9</td>
<td>-</td>
<td>10,308</td>
<td>-</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-1</td>
<td>1,439.3</td>
<td>995.7</td>
<td>11.7</td>
<td>9,983</td>
<td>657</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-2</td>
<td>1,439.3</td>
<td>983.6</td>
<td>95.6</td>
<td>7,439</td>
<td>689</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-G</td>
<td>1,439.3</td>
<td>1,713.0</td>
<td>140.4</td>
<td>8,831</td>
<td>14</td>
<td>5/5</td>
</tr>
<tr>
<td>$\epsilon$</td>
<td>N</td>
<td>approach</td>
<td>AvT-M</td>
<td>AvT-B&amp;C</td>
<td>AvT-Cut</td>
<td># of nodes</td>
<td># of cuts</td>
<td>solved</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>500</td>
<td>CPX</td>
<td>165.0</td>
<td>816.6</td>
<td>-</td>
<td>29,710</td>
<td>-</td>
<td>5/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-1</td>
<td>165.0</td>
<td>470.5</td>
<td>1.1</td>
<td>18,226</td>
<td>192</td>
<td>5/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-2</td>
<td>165.0</td>
<td>360.2</td>
<td>9.1</td>
<td>14,614</td>
<td>201</td>
<td>5/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-G</td>
<td>165.0</td>
<td>809.9</td>
<td>10.1</td>
<td>28,151</td>
<td>10</td>
<td>5/5</td>
<td></td>
</tr>
<tr>
<td>0.08</td>
<td>CPX</td>
<td>641.1</td>
<td>2,375.7</td>
<td>-</td>
<td>31,595</td>
<td>-</td>
<td>5/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-1</td>
<td>641.1</td>
<td>2,024.8</td>
<td>4.3</td>
<td>29,594</td>
<td>307</td>
<td>5/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-2</td>
<td>641.1</td>
<td>1,791.4</td>
<td>25.2</td>
<td>22,455</td>
<td>284</td>
<td>5/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-G</td>
<td>641.1</td>
<td>2,166.4</td>
<td>37.4</td>
<td>28,596</td>
<td>9</td>
<td>5/5</td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td>CPX</td>
<td>1,439.3</td>
<td>4,600.4</td>
<td>0.03</td>
<td>65,740</td>
<td>-</td>
<td>4/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-1</td>
<td>1,439.3</td>
<td>3,095.5+32,104.9*</td>
<td>7.6</td>
<td>76,573</td>
<td>402</td>
<td>5/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-2</td>
<td>1,439.3</td>
<td>3,072.9+28,014.4*</td>
<td>55.2</td>
<td>69,798</td>
<td>413</td>
<td>5/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-G</td>
<td>1,439.3</td>
<td>3,650.7+28,248.3*</td>
<td>77.4</td>
<td>54,969</td>
<td>10</td>
<td>5/5</td>
<td></td>
</tr>
<tr>
<td>1500</td>
<td>CPX</td>
<td>165.0</td>
<td>32,178.1</td>
<td>0.11</td>
<td>1,588,803</td>
<td>-</td>
<td>1/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-1</td>
<td>165.0</td>
<td>18,923.0</td>
<td>0.07</td>
<td>1,296,583</td>
<td>184</td>
<td>1/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-2</td>
<td>165.0</td>
<td>20,497.6</td>
<td>0.09</td>
<td>1,780,324</td>
<td>193</td>
<td>1/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-G</td>
<td>165.0</td>
<td>16,313.2</td>
<td>0.11</td>
<td>1,607,736</td>
<td>3</td>
<td>1/5</td>
<td></td>
</tr>
<tr>
<td>0.06</td>
<td>CPX</td>
<td>641.1</td>
<td>32.178.1</td>
<td>0.19</td>
<td>-</td>
<td>588,891</td>
<td>-</td>
<td>0/5</td>
</tr>
<tr>
<td></td>
<td>Cover-1</td>
<td>641.1</td>
<td>18,923.0</td>
<td>0.19</td>
<td>514,576</td>
<td>288</td>
<td>0/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-2</td>
<td>641.1</td>
<td>20,497.6</td>
<td>0.17</td>
<td>531,292</td>
<td>313</td>
<td>0/5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-G</td>
<td>641.1</td>
<td>16,313.2</td>
<td>0.19</td>
<td>562,600</td>
<td>10</td>
<td>0/5</td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td>CPX</td>
<td>1,439.3</td>
<td>4,600.4</td>
<td>0.25</td>
<td>-</td>
<td>267,632</td>
<td>-</td>
<td>0/5</td>
</tr>
<tr>
<td></td>
<td>Cover-1</td>
<td>1,439.3</td>
<td>21.6</td>
<td>216,724</td>
<td>456</td>
<td>0/5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-2</td>
<td>1,439.3</td>
<td>21.6</td>
<td>247,320</td>
<td>431</td>
<td>0/5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cover-G</td>
<td>1,439.3</td>
<td>21.6</td>
<td>258,991</td>
<td>6</td>
<td>0/5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

"*" 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.
Table 3: The average CPU time (in seconds) for branch-and-cut algorithm with probability cuts (AvT-B&CP), probability and feasibility cut generation (AvT-fea-cut) and valid cut generation (AvT-cut), the average number of nodes (# of nodes), probability and feasibility cuts (# of fea-cuts) and valid cuts (# of cuts), and the number of solved instances from the five instances (solved) for (CAP) are reported.

<table>
<thead>
<tr>
<th>$\eta$</th>
<th>N</th>
<th>approach</th>
<th>AvT-B&amp;CP</th>
<th>AvT-cut</th>
<th>AvT-fea-cut</th>
<th># of nodes</th>
<th># of cuts</th>
<th># of fea-cuts</th>
<th>solved</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.5</td>
<td>1000</td>
<td>CPX</td>
<td>1,390.0</td>
<td>–</td>
<td>403.4</td>
<td>12,095</td>
<td>–</td>
<td>12</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-1</td>
<td>1,021.5</td>
<td>5.6</td>
<td>346.5</td>
<td>12,148</td>
<td>447</td>
<td>8</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-2</td>
<td>884.0</td>
<td>33.0</td>
<td>397.5</td>
<td>10,720</td>
<td>373</td>
<td>9</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1000</td>
<td>CPX</td>
<td>826.9</td>
<td>–</td>
<td>104.0</td>
<td>31,989</td>
<td>–</td>
<td>34</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-1</td>
<td>501.8</td>
<td>1.4</td>
<td>99.8</td>
<td>20,067</td>
<td>233</td>
<td>28</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-2</td>
<td>775.8</td>
<td>11.1</td>
<td>105.2</td>
<td>32,536</td>
<td>247</td>
<td>30</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>1000</td>
<td>CPX</td>
<td>2,987.3</td>
<td>–</td>
<td>502.2</td>
<td>47,221</td>
<td>–</td>
<td>30</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-1</td>
<td>3,173.8</td>
<td>6.2</td>
<td>483.6</td>
<td>42,202</td>
<td>482</td>
<td>30</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cover-2</td>
<td>2,173.9</td>
<td>36.3</td>
<td>484.7</td>
<td>33,143</td>
<td>414</td>
<td>28</td>
<td>5/5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

"–" in column of AvT-cut and # of cuts indicates that no valid cut proposed in this paper is added.
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-overtime) (in minutes), and 85%, 95%, 99% quantiles (in minutes) for (CAP) and (DR-CAP) are reported.

<table>
<thead>
<tr>
<th>η</th>
<th>N</th>
<th>model</th>
<th>Avg-cost</th>
<th>Avg-prob</th>
<th>Worst-prob</th>
<th>Avg-overtime</th>
<th>85%</th>
<th>95%</th>
<th>99%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>500</td>
<td>(CAP)</td>
<td>69.9</td>
<td>0.070</td>
<td>0.122</td>
<td>6.1</td>
<td>0.0</td>
<td>36.4</td>
<td>150.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DR-CAP)</td>
<td>70.3</td>
<td>0.068</td>
<td>0.122</td>
<td>6.0</td>
<td>0.0</td>
<td>36.8</td>
<td>147.4</td>
</tr>
<tr>
<td>0.1</td>
<td>1000</td>
<td>(CAP)</td>
<td>70.2</td>
<td>0.069</td>
<td>0.122</td>
<td>6.1</td>
<td>0.0</td>
<td>37.9</td>
<td>150.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DR-CAP)</td>
<td>70.7</td>
<td>0.066</td>
<td>0.122</td>
<td>5.8</td>
<td>0.0</td>
<td>33.8</td>
<td>148.5</td>
</tr>
<tr>
<td>0.5</td>
<td>1500</td>
<td>(CAP)</td>
<td>70.7</td>
<td>0.067</td>
<td>0.117</td>
<td>5.8</td>
<td>0.0</td>
<td>34.5</td>
<td>148.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DR-CAP)</td>
<td>71.0</td>
<td>0.067</td>
<td>0.117</td>
<td>5.9</td>
<td>0.0</td>
<td>35.6</td>
<td>147.4</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>(CAP)</td>
<td>69.9</td>
<td>0.070</td>
<td>0.122</td>
<td>6.1</td>
<td>0.0</td>
<td>36.4</td>
<td>150.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DR-CAP)</td>
<td>71.7</td>
<td>0.066</td>
<td>0.121</td>
<td>5.8</td>
<td>0.0</td>
<td>32.3</td>
<td>147.8</td>
</tr>
<tr>
<td>0.5</td>
<td>1000</td>
<td>(CAP)</td>
<td>70.2</td>
<td>0.069</td>
<td>0.122</td>
<td>6.1</td>
<td>0.0</td>
<td>37.9</td>
<td>150.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DR-CAP)</td>
<td>71.2</td>
<td>0.065</td>
<td>0.088</td>
<td>5.6</td>
<td>0.0</td>
<td>31.9</td>
<td>149.3</td>
</tr>
<tr>
<td>1</td>
<td>1500</td>
<td>(CAP)</td>
<td>70.7</td>
<td>0.067</td>
<td>0.117</td>
<td>5.8</td>
<td>0.0</td>
<td>34.5</td>
<td>148.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DR-CAP)</td>
<td>72.0</td>
<td>0.065</td>
<td>0.096</td>
<td>5.6</td>
<td>0.0</td>
<td>29.6</td>
<td>148.1</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>(CAP)</td>
<td>69.9</td>
<td>0.070</td>
<td>0.122</td>
<td>6.1</td>
<td>0.0</td>
<td>36.4</td>
<td>150.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DR-CAP)</td>
<td>72.8</td>
<td>0.065</td>
<td>0.121</td>
<td>5.6</td>
<td>0.0</td>
<td>28.9</td>
<td>148.1</td>
</tr>
<tr>
<td>1</td>
<td>1000</td>
<td>(CAP)</td>
<td>70.2</td>
<td>0.069</td>
<td>0.122</td>
<td>6.1</td>
<td>0.0</td>
<td>37.9</td>
<td>150.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DR-CAP)</td>
<td>73.1</td>
<td>0.064</td>
<td>0.089</td>
<td>5.5</td>
<td>0.0</td>
<td>26.6</td>
<td>149.3</td>
</tr>
<tr>
<td></td>
<td>1500</td>
<td>(CAP)</td>
<td>70.7</td>
<td>0.067</td>
<td>0.117</td>
<td>5.8</td>
<td>0.0</td>
<td>34.5</td>
<td>148.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(DR-CAP)</td>
<td>73.3</td>
<td>0.064</td>
<td>0.082</td>
<td>5.4</td>
<td>0.0</td>
<td>24.4</td>
<td>148.1</td>
</tr>
</tbody>
</table>

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-overtime) (in minutes), and 85%, 95%, 99% quantiles (in minutes) for (CAP) and (DR-CAP) are reported.

### 5.5 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 $N \in \{500, 1000, 1500\}$ for the (CAP) and (DR-CAP) solutions, and the (DR-CAP) solutions were generated using the Wasserstein set radius parameter $\eta \in \{0.1, 0.5, 1\}$. All evaluations were performed for $\epsilon = 0.1$ in the chance constraint. 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.

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 ($\eta$) 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 $\eta$ 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 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 $\eta$. 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., the solutions generated in all the instances satisfy the chance constraint with probability 0.1. 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 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).

6 Concluding Remarks

The use of big-M calculations and strong inequalities developed in this paper resulted in 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 an increasing sample size. However, the solution time for the distributionally robust chance-constrained models appear 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 allows one to have the true probability distribution of the random parameters with a greater probability.

Acknowledgement

This research of the first two authors were partially supported by the National Natural Science Foundation of China (NSFC) grants 71432002, 91746210. The research of the last author was partially supported by the NSF grant CMMI-1763035 and the ONR grant N00014-18-1-2097. This paper also benefited from a discussion with Professor Simge Küçükyavuz.
References


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

Peng, C., Delage, E., Li, J., 2018. Probabilistic envelope constrained multiperiod stochastic emer-
gency medical services location model and decomposition scheme. Available at Optimization-Online

Postek, K., Ben-Tal, A., Den Hertog, D., Melenberg, B., 2018. Robust optimization with ambiguous


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


Tanner, M. W., Ntaimo, L., 2010. Iis branch-and-cut for joint chance-constrained stochastic programs and

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

Wang, S., Li, J., Mehrotra, S., 2019. Chance-constrained bin packing problem with an applica-
tion to operating room planning. Available at Optimization-Online http://www.optimization-
online.org/DB_HTML/2019/02/7053.html.

Research 62 (6), 1358–1376.


Xie, W., 2018. On distributionally robust chance constrained program with wasserstein distance. arXiv

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


Appendix

A Proof of Propositions

A.1 Proof of Proposition 1

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

A.2 Proof of Proposition 2

Let $(y, z)$ be a feasible solution of the relaxation problem of the binary bilinear reformulation of (DR-CAP). We have

$$\sum_{i \in I} \xi_i^j y_{ij}^* z_j^\omega - \sum_{i \in I} \xi_i^j y_{ij}^* - m_j^\omega(k_q)(z_j^\omega - 1) = (z_j^\omega - 1) \left( \sum_{i \in I} \xi_i^j y_{ij}^* - m_j^\omega(k_q) \right) \geq 0.$$ 

Consequently, the following inequality

$$\sum_{i \in I} \xi_i^j y_{ij}^* + m_j^\omega(k_q)(z_j^\omega - 1) \leq \sum_{i \in I} \xi_i^j y_{ij}^* z_j^\omega \leq m_j^\omega(\omega) z_j^\omega$$

holds. Therefore, $(y, z)$ is a feasible solution of the relaxation problem of (7). The proof can be similarly extend to (CAP). □

A.3 Proof of Proposition 3

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

A.4 Proof of Proposition 4

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

37
Consider the following $|\mathcal{C}\setminus\mathcal{D}|$ feasible points of \eqref{eq:10}: for $k \in \mathcal{C}\setminus\mathcal{D}$, set $y_{ij} = 1, \forall i \in \mathcal{C}\setminus\mathcal{D} \cup k$, $y_{ij} = 0, \forall i \in k \cup \mathcal{Z}(\mathcal{C})$, and $y_{ij} = 1, \forall i \in \mathcal{D}$; These $|\mathcal{C}\setminus\mathcal{D}|$ points are affinely independent and satisfy \eqref{eq:11} at equality. \hfill $\square$

\section*{B \ Algorithm Details}

\subsection*{B.1 Dynamic Programming for Estimating big-M}

Let $D(|\mathcal{I}|, t_j, \rho_j)$ represents \eqref{eq:5}, where $|\mathcal{I}|$ denotes the $|\mathcal{I}|$ variables of $y_j$. Let us consider the subproblem $D(n, t^0_j, \rho^0_j)$ of $D(|\mathcal{I}|, t_j, \rho_j)$ which includes the first $n$ variables of $y_j$ and the right-hand side values of constraints in \eqref{eq:5} respectively. Let $S(n, t^0_j, \rho^0_j)$ be the optimal objective value of $D(n, t^0_j, \rho^0_j)$. If $D(n, t^0_j, \rho^0_j)$ is infeasible, we set $S(n, t^0_j, \rho^0_j) = -\infty$. Since $y_{nj}$ is binary, if $y_{nj} = 0$, $S(n, t^0_j, \rho^0_j)$ is equal to $S(n-1, t^0_j, \rho^0_j)$, which is the optimal objective value of the subproblem $D(n-1, t^0_j, \rho^0_j)$. If $y_{nj} = 1$, $S(n, t^0_j, \rho^0_j)$ is equal to $S(n-1, t^0_j - \xi^k, \rho^0_j - 1) + \xi^\omega$, which is the optimal objective value of the subproblem $D(n-1, t^0_j - \xi^k, \rho^0_j)$ plus $\xi^\omega$. Thus, we have

$$S(n, t^0_j, \rho^0_j) = \max\{S(n-1, t^0_j, \rho^0_j), S(n-1, t^0_j - \xi^k, \rho^0_j - 1) + \xi^\omega\},$$

where $n = 2, \ldots, |\mathcal{I}|$, and $t^0_j \leq t_j, \rho^0_j \leq \rho_j$. It is easy to verify that the dynamic programming procedure has $O(|\mathcal{I}|(\max\{t_j, \rho_j\})^2)$ time complexity.

\subsection*{B.2 Dynamic Programming for Up-lifting Coefficient}

Dynamic programming has been used to calculate the up-lifting coefficients in the binary single knapsack problem (see, Zemel (1989)). We use this technique to obtain the lifting coefficient $\alpha_i$. For each $k = 1, \ldots, |\mathcal{I}|\setminus\mathcal{C}|$, we solve the following problem for $\lambda_1 = 0, \ldots, |\mathcal{C}\setminus\mathcal{D}| - 1$, and $\lambda_2 = 0, \ldots, \rho_j - 1 - |\mathcal{D}|$:

$$A_{\pi_k}(\lambda_1, \lambda_2) = \min_{y_j} \sum_{i \in \mathcal{C}\setminus\mathcal{D}} \xi^\omega y_{ij} + \sum_{i = \pi_1}^{\pi_k - 1} \xi^\omega y_{ij}$$

subject to \(\sum_{i \in \mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i = \pi_1}^{\pi_k - 1} \alpha_i y_{ij} \geq \lambda_1\)

\(\sum_{i \in \mathcal{C}\setminus\mathcal{D}} y_{ij} + \sum_{i = \pi_1}^{\pi_k - 1} y_{ij} \leq \lambda_2\),

\(y_{ij} \in \{0, 1\}, \quad \forall i \in \mathcal{C}\setminus\{\pi_1, \ldots, \pi_{k-1}\}\).

\textbf{Proposition 5} Let $\text{obj}_{\pi_k}$ be defined as in \eqref{eq:13}. Then, $\text{obj}_{\pi_k} = \max\{\lambda_1 : A_{\pi_k}(\lambda_1, \rho_j - 1 - |\mathcal{D}|) \leq \lambda_2\}$
$$m_\pi^w(\omega) - \xi_\pi k - \sum_{i \in D} \xi_\pi i.$$  

Let $l_t$, $t = 0, \ldots, |C\setminus D| - 1$ be the sum of the $t$ smallest $\xi_i$, $i \in C\setminus D$. Algorithm 3 gives an outline of our dynamic programming framework.

### Algorithm 3: Dynamic Programming for the Lifting Coefficients

1. for $\lambda_2 = 0, \ldots, \rho_j - 1 - |D|$ do
2.   for $\lambda_1 = 0, \ldots, |C\setminus D| - 1$ do
3.     if $\lambda_1 \leq \lambda_2$ then
4.       $A_{\pi_1}(\lambda_1, \lambda_2) = l_{\lambda_1}$.
5.     end
6.     else
7.       $A_{\pi_1}(\lambda_1, \lambda_2) = +\infty$.
8.     end
9.   end
10. end
11. for $k = 1, \ldots, |C\setminus C|$ do
12.   $\text{obj}_{\pi k} = \max \left\{ \lambda_1 : A_{\pi k}(\lambda_1, \rho_j - 1 - |D|) \leq m_\pi^w(\omega) - \xi_\pi k - \sum_{i \in D} \xi_\pi i \right\}$.
13. $\alpha_{\pi k} = |C\setminus D| - 1 - \text{obj}_{\pi k}$.
14. for $\lambda_2 = 0, \ldots, \rho_j - 1 - |D|$ do
15.   for $\lambda_1 = 0, \ldots, |C\setminus D| - 1$ do
16.     if $\lambda_1 \geq \alpha_{\pi k}$ and $\lambda_2 \geq 1$ then
17.       $A_{\pi k+1}(\lambda_1, \lambda_2) = \min \left\{ A_{\pi k}(\lambda_1, \lambda_2), A_{\pi k}(\lambda_1 - \alpha_{\pi k}, \lambda_2 - 1) + \xi_\pi k \right\}$.
18.     end
19.     else
20.       $A_{\pi k+1}(\lambda_1, \lambda_2) = A_{\pi k}(\lambda_1, \lambda_2)$.
21.     end
22.   end
23. end
24. end

### B.3 Probability Cut Algorithm

Algorithm 4 gives an outline of the probability cut algorithm for obtaining the optimal objective value $\text{obj}_{\pi l}'$ defined in Section 3.2.3.

### B.4 Separation Heuristic for (26)

Algorithm 5 gives an overview of the separation heuristic for (26).

### B.5 Branch-and-Cut Algorithm with Probability Cuts for (DR-CAP)

The branch-and-cut algorithm with probability cuts for (DR-CAP) is provided in Algorithm 6.
Algorithm 4: Probability Cut Algorithm

1. **Initialize** \( \text{Obj} = 0 \), the number of iterations \( k = 1 \), and the maximal number of iterations \( K = 100 \).

2. **while** \((\text{Obj} < 1 - \varepsilon \&\& k < K)\) **do**

3.   Solve the following problem.

\[
\begin{align*}
\text{maximize} & \quad (y_j, z_j) \in [0, 1]^{|I| \times N} \\
\text{subject to} & \quad \sum_{i \in \mathcal{C}\backslash \mathcal{D}} y_{ij} + \sum_{i \in \mathcal{I} \backslash \mathcal{C}} \bar{\alpha}_i y_{ij} + \sum_{\omega=\tau_l}^{\tau_{l-1}} \gamma_\omega z_{j\omega} \\
& \quad \quad \sum_{\omega \in \Omega} p_\omega z_{j\omega} \geq 1 - \varepsilon, \quad l = 1, \ldots, k-1.
\end{align*}
\]

4. Record the optimal solution \((y_k, z_k)\) and objective value \(\text{obj}_k\).

5. **Fix** \(z\) to be \(z_k\), solve the distribution separation problem.

\[
\inf_{p \in \mathcal{P}} \sum_{\omega \in \Omega} p_\omega z^k_{j\omega}
\]

6. Obtain the optimal solution \(p^k\) and objective value \(\text{Obj}^k\).

7. \(k = k + 1\);

8. **end**

9. **return** \(\text{obj}^{k-1}\).

---

Algorithm 5: Separation Heuristic for (26)

1. Given the LP relaxation optimal solution \((\hat{y}, \hat{z})\).

2. **for** \(j = 1, \ldots, |J|\) **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** \(\inf_{p \in \mathcal{P}} \sum_{\omega \in \Omega_1} p_\omega \hat{z}_{j\omega} \geq 1 - \varepsilon\) (for (DR-CAP)) **then**

5.     Sort \(\hat{y}_j\) in non-increasing order: \(\hat{y}_{i_1} \geq \ldots \geq \hat{y}_{i_{|I|}}\).

6.     **for** \(\omega \in \Omega_1\) **do**

7.       Let \(\mathcal{C} = \{i_1, \ldots, i_o\}\) where \(o \leq |I|\) is a smallest number such that \(\mathcal{C}\) is a cover for \(\omega\).

8.       Delete elements from the tail of \(\mathcal{C}\) to get a minimal cover \(\mathcal{C}\).

9.       Let set \(\mathcal{D} = \{i \in \mathcal{C} | \hat{y}_{i1} = 1\}\) and \(\mathcal{I}_0 = \{i \in \mathcal{I} \backslash \mathcal{C} | \hat{y}_{i1} = 0\}\).

10. Calculate the up-lifting coefficient \(\bar{\alpha}_i\) for \(i \in \mathcal{I} \backslash (\mathcal{C} \cup \mathcal{I}_0)\).

11. **if** \(\sum_{i \in \mathcal{C}\backslash \mathcal{D}} \bar{\alpha}_i \hat{y}_{i1} + \sum_{i \in \mathcal{I} \backslash (\mathcal{C} \cup \mathcal{I}_0)} \bar{\alpha}_i \hat{y}_{i1} > |\mathcal{C}\backslash \mathcal{D}| - 1\) **then**

12.       Calculate the down-lifting coefficient \(\bar{\beta}_i\) for \(i \in \mathcal{D}\).

13.       Calculate the up-lifting coefficient \(\bar{\alpha}_i\) for \(i \in \mathcal{I}_0\).

14.       Calculate the coefficient \(\gamma_\omega\) for \(\omega \in \Omega_1\).

15. Obtain the violated inequality (26).

16. **end**

17. **if** (26) **is obtained** **then**

18.       Go to step 2.

19. **end**

20. **end**

21. **end**

22. **end**
Algorithm 6: Branch-and-Cut Algorithm with Probability Cuts

1 Initialize $P^0 \in \mathcal{P}$, the number of iteration $k = 0$, $UB = +\infty$, $LB = -\infty$, $\mathcal{N} = \{o\}$ where $o$ 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 $j \in J$ 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 (33) and (34) to (LMP).

13 end

14 end

15 if Cuts (33) and (34) are found then

16 Go to step 5.

17 end

18 else

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

20 end

21 end

22 if $(\hat{y}, \hat{z})$ is fractional then

23 Use Algorithm 1 and 5 to find the violated inequalities.

24 if Violated inequalities are found then

25 Add the violated inequalities to (LMP).

26 Go to line 5.

27 end

28 else

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

30 end

31 end

32 end

33 return UB and its corresponding optimal solution $(y^*, z^*)$. 

41