Almost Robust Optimization with Binary Variables

Opher Baron, Oded Berman, Mohammad M. Fazel-Zarandi
Rotman School of Management, University of Toronto, Toronto, Ontario M5S 3E6, Canada,
Opher.Baron@Rotman.utoronto.ca, Berman@Rotman.utoronto.ca, M.FazelZarandi10@Rotman.utoronto.ca

Abstract

The main goal of this paper is to develop a simple and tractable methodology (both theoretical and computational) for incorporating data uncertainty into optimization problems in general and into discrete (binary decision variables) optimization problems in particular. We present the *Almost* Robust Optimization (*A*RO) model that addresses data uncertainty for discrete optimization models. The *A*RO model trade-offs the objective function value with robustness, to find optimal solutions that are *almost* robust (feasible under most realizations). The proposed model is attractive due to its simple structure, its ability to model dependence among uncertain parameters, and its ability to incorporate the decision maker's attitude towards risk by controlling the degree of conservatism of the optimal solution. Specifically, we define the Robustness Index that enables the decision maker to adjust the *A*RO to better suit her risk preference. We also demonstrate the potential advantages of being almost robust as opposed to being fully robust. To solve the *A*RO model with discrete decision variables efficiently, we develop a decomposition approach that decomposes the model into a deterministic master problem and a single subproblem that checks the master problem solution under different realizations and generates cuts if needed. Computational experiments demonstrate that the decomposition approach is able to find the optimal solutions up to three orders-of-magnitude faster than the original formulation. We demonstrate our methodology on the knapsack problem with uncertain weights, the capacitated facility location problem with uncertain demands, and the vehicle routing problem with uncertain demands and travel times.

1. Introduction

Binary requirements on decision variables are essential in many operational and planning models. A variety of applications with such requirements include: facility location, vehicle routing, production planning and scheduling. Traditional optimization models for these

applications assume that the parameters and data are known and certain. In practice, however, such assumptions are often unrealistic, e.g., the data may be noisy or incomplete.

Two of the main approaches used to incorporate uncertainty are stochastic programming (see Ruszczynski and Shapiro (2003) and references therein) and chance constrained programming (Charnes and Cooper (1959)). In stochastic programming (which are usually modeled in two stages) the decision maker optimizes an expected value of an objective function that involves random parameters. In the Chance Constrained Programming, the decision maker wants to ensure that the constraints hold with at least a specified level of probability. Both approaches are typically not effective when the inclusion of discrete decision variables is required, then these problem becomes very challenging to solve.

Another approach to handle uncertainty is robust mathematical programming, Mulvey et al. (1995), which presents an approach that integrates goal programming formulations with scenario-based description of the problem data. The approach seeks to generate solutions that are less sensitive to the specific realization of the data. Mulvey et al. (1995) only consider convex optimization models and compare the advantages of their model to parameter sensitivity analysis and to solutions obtained from stochastic programming.

Another method to incorporate uncertainty into optimization problems is robust optimization. Unlike stochastic programming, chance constrained programming, and robust mathematical programming that use probabilistic information, in robust optimization the uncertainty is set-based, requiring no specific information about the distribution functions. In such models, the decision-maker seeks to find solutions that are feasible for any realization of the uncertainty in a given set (see Ben-Tal and Nemirovski (1998, 1999, 2000, 2001), El-Ghaoui et al. (1998), Bertsimas and Sim (2004), Chen et al. (2007), Natarajan et al. (2008)). The majority of these papers, however, focus on convex optimization and limited work is done on discrete optimization problems (exceptions include: Kouvelis and Yu (1997), Averbakh (2001), Bertsimas and Sim (2003), Atamturk (2006), Atamturk and Zhang (2007), and Baron et al. (2011)).

In this paper, we present the *Almost* Robust Optimization (*A*RO) model that incorporates data uncertainty, into optimization models. The *A*RO, trade-offs the objective function value with robustness of the solution (robustness is measured by the amount of violation of

the uncertain constraints under different realizations) to find optimal solutions that are *almost* robust (feasible for most data realizations). The *A*RO also incorporates the decision maker's attitude towards risk (control over the degree of conservatism of the optimal solution and the degree of the robustness of the solution). An important advantage of our model is its tractability for problems with discrete (binary) decision variables. Thus, our exposition is focused on such problems, modeling the data uncertainty using a set of scenarios. But our model can also be utilized for optimization problems with general uncertainty sets.

The *A*RO approach combines ideas from stochastic programming, chance constrained programming, robust mathematical programming, and robust optimization. A comparison of these approaches with ARO is presented in Table 1 and further discussed below.

- *Robust optimization vs.* A*RO:* Both robust optimization and *A*RO seek to find solutions that are robust given an uncertainty set and can incorporate the decision makers' risk attitude (Bertsimas and Sim (2004), Natarajan et al. (2009), and Ben-Tal et al. (2010)); there are two main differences between them. First, unlike robust optimization, *A*RO can use probabilistic knowledge of the uncertain data. In particular, *A*RO can be setup to suit a wide spectrum of available probabilistic information such as: full probabilistic information, no probabilistic information, and partial probabilistic information. Second, in robust optimization the constraints are usually hard, i.e., the constraints must be satisfied for all possible realizations of the data, (exceptions include Ben-Tal et al. (2006) and Ben-Tal et al. (2010))) while in *A*RO, both hard and soft constraints can be accommodated, i.e., the decision maker has the choice to decide whether or not to allow infeasibility under some realizations. Note that even the framework of generalized robust optimization that allows large data variations, which can be thought of as rare scenarios, treats the constraints as hard (see chapter 3 of Ben-Tal et al. (2009)).

- *Stochastic programming vs.* A*RO:* The main similarity between the two models is that they use probabilistic knowledge of the uncertain parameters; but *A*RO also applies to situations with limited probabilistic information. The main difference is that stochastic programming models seek to minimize the expected cost (dis-utility) of a decision made

in the first stage, while *A*RO seeks to find solutions that are almost robust (feasible under most realizations). Furthermore, unlike *A*RO which accommodates both hard and soft constraints, stochastic programming only considers soft constraints. Finally, while discrete variables can make stochastic programming hard and computationally intractable, the discrete variable version of the *A*RO (as will be discussed shortly) is much more tractable.

- *Chance constrained programming vs.* A*RO:* As we show in Section 2.2, chance constraints are a special case of ARO. Conceptually, however, there are two main difference between these models: first, we believe that the ARO is more suitable for cases where the decision maker faces quantitative risk (i.e., cases where measuring the amount of constraint violation is important), whereas chance-contraints are more suited for cases with qualitative risk (i.e., cases where instead of the amount of violation, including violation with its probability is important); secondly, while chance constraints require probabilistic knowledge of the uncertain parameters, *A*RO can be applied to setting with limited (or no) probabilistic information.

- *Robust mathematical programming vs.* A*RO:* The *A*RO retains the advantages of robust mathematical programming while offering the following additional advantages: first, *A*RO allows control of the degree of conservatism for every constraint (i.e., gives the decision maker control over the amount of violation allowed for each uncertain constraint), thus, fully integrating the decision makers' risk attitude in the optimization model. Second, while robust mathematical programming only considers a scenario-based description of the uncertain data, the *A*RO can also address general uncertainty structures. Third, while robust mathematical programming assumes that the probability distribution of the scenarios are known, *A*RO can also be used for setting in which the probabilistic information of the uncertain data is limited. Finally, while robust mathematical programming focuses on convex optimization, we focus on discrete optimization models. This generalization makes the problems much more difficult to solve and thus the issue of tractability arises. To overcome this issue, we develop a tractable decomposition approach for solving such problems.

Model	Constraint Type	Uncertainty Type	Incorporate Risk	Risk Type
Stochastic Programming	Soft	Probabilistic	Indirectly	Quantitative
Chance Constrained Programming	Soft / Hard	Probabilistic	Directly	Qualitative
Robust Mathematical Programming	Soft	Probabilistic	Indirectly	Quantitative
Robust Programming	Usually Hard	Non-Probabilistic	Directly	Qualitative
ARO	Soft / Hard	Probabilistic / Non-Probabilistic	Directly	Quantitative (accommodates Qualitative)

Table 1: Comparison of ARO with Other Approaches.

The main goal of this paper is to develop a simple, practical, and tractable methodology for incorporating data uncertainty into discrete optimization problems. Therefore, the exposition is focused on the discrete optimization version of the *A*RO, which we refer to as the *Almost Robust Discrete Optimization model* (*A*RDO).

Due to both its combinatorial and uncertain nature, *A*RDO can be difficult to solve. To overcome such difficulties, we develop a decomposition approach, in which the *A*RDO is decomposed into a deterministic master problem and a single subproblem which checks the master problem solution under different realizations and generates cuts if needed. Our computational experiments demonstrate that the decomposition approach is very effective, i.e., it is able to find an optimal solution up to three orders-of-magnitude faster than the original formulation. Furthermore, unlike the original formulation, in which the computational difficulty drastically increases with the number of scenarios, the number of scenarios does not substantially affect the computational difficulty of the decomposition approach. Another important upside of our approach is that since the master problem is a deterministic discrete optimization problem, existing approaches for solving such problems can be incorporated into the decomposition model to speed up the solution procedure. Our decomposition approach also overcomes the main drawbacks of the integer L-shape method (Laporte and Louveaux (1993)), which is the main tool used to solve stochastic integer programs. Specifically, unlike the L-shaped method, in which the subproblems are integer programs and are hard to solve, the subproblem in our model is very simple to solve; furthermore, in the integer L-shape method, the cuts are generated from the relaxed LP dual of the subproblems (this relaxation can substantially decreases the efficiency of the approach), while our cuts are uniquely designed for integer programs.

In summary, the main contributions of this paper are threefold. First, we introduce

the Almost Robust Optimization model and formulation that is simple, intuitive, flexible, and easy to communicate to managers. This formulation allows uncertainty in the decision making process and bridges stochastic programming and robust optimization. Second, we develop a decomposition approach to efficiently solve the Almost Robust Optimization model with discrete (binary) decision variables. We show that our methodology is practical and computationally tractable for discrete optimization problems that are subject to uncertainty. Third, by defining the Robustness Index we demonstrate the potential advantages of being almost robust as opposed to being fully robust.

The paper is organized as follows. Section 2 presents the general framework and formulation of the *A*RO. The details of the decomposition approach are described in Section 3. Some applications are provided in Section 4. Computational results are presented in Section 5. Section 6 concludes the paper. Proofs and extensions are included in the Appendices.

2. Almost Robust Optimization

In this section, we present models that incorporate data uncertainty in the decision making process using a scenario-based description of the data (as we demonstrate in Section 6, our approach also works when the uncertainty has other structures). Scenarios are used since they are very powerful, convenient and practical in representing uncertainty, and they allow the decision maker considerable flexibility in the specification of uncertainty (e.g., uncertain data may be highly correlated, may be samples from simulation, and/or may be obtained from historical data).

This section is divided into two parts. We first present some prior formulations in which scenario-based data are incrementally incorporated into a deterministic optimization model, and discuss their advantages and disadvantages. Then, we present our Almost Robust Optimization model.

2.1 Prior Optimization Models

Deterministic Model: Consider a deterministic optimization problem of the form

$$\begin{array}{lll} \min & z = c^T x & \\ \text{s.t.} & Ax \leq b & , \\ & x \in X, & \end{array} \qquad (Deterministic\ Model)$$

where x, c, and b are $N \times 1$ vectors and A is a $(M + J) \times N$ matrix.

The classical paradigm in mathematical programming assumes that the input data of the above formulation is known. In practice, however, portions of the data may be uncertain.

Without loss of generality, we assume that the data uncertainty affects only a partition of A. As is standard (see Ben-Tal et al. (2009)), we assume that both c and b are deterministic. (These can be achieved by incorporating them into the uncertain partition of A, by adding the constraint $z - c^T x \leq 0$, introducing a new variables $1 \leq x_{n+1} \leq 1$, and transforming the constraints to $Ax - bx_{n+1} \leq 0$.) We discuss a treatment on uncertain objectives in a more general manner in Section 2.2. We partition A into a deterministic $M \times N$ matrix, D, and an uncertain $J \times N$ matrix U. Similarly, we partition the vector b into a $M \times 1$ vector b^D corresponding to the deterministic partition of A, and a $J \times 1$ vector b^U corresponding to the uncertain partition.

As discussed in the introduction, we focus on discrete optimization problems in which $x \in X \subseteq \{0, 1\}^N$. Such problems are often difficult to solve due to the their integrality and combinatorial nature, and indeed many of them are NP-hard. In particular, convexity, which often exists in problems with continuous variables, no longer holds in problems with integrality constraints. For a comprehensive discussion of the theory of optimization problems with integrality constraints, and the general tools developed for solving them efficiently see Nemhauser and Wolsey (1988), Wolsey (1998), and Bertsimas and Weismantel (2005).

As stated earlier, in the body of the paper (except in Section 6) we assume that the uncertain data is represented by a set of scenarios $S = \{1, 2, ..., |S|\}$, with probabilities p_s for each $s \in S$. Each scenario is associated with a particular realization of the uncertain matrix, U^s with corresponding entries u^s_{ij}. Note that later in this section, we show that our Almost Robust Optimization model can also be setup to suit a wide spectrum of available probabilistic information such as: full probabilistic information, no probabilistic information, and partial probabilistic information.

We next introduce two existing models that incorporate scenarios into the deterministic model and discuss their advantages.

Robust Model: Consider the following robust formulation

$$\begin{array}{lll} \min & c^T x & \\ \text{s.t.} & Dx \leq b^D, & \\ & U^s x \leq b^U, & \forall s \in S, \\ & x \in X. & \end{array} \qquad (Robust\ Model)$$

The robust model follows the formulation of robust optimization (see Ben-Tal et al. (2009)). This formulation ensures that the optimal solution does not violate any of the constraints under any possible scenario realization.

This formulation has three main drawbacks. First, due to the scenarios, it may no longer have a feasible solution. Second, even when feasible solutions exist, they may be over conservative, i.e., the optimal objective function value may be very high. The overconservatism is magnified when a scenario with a low probability of occurrence has a large effect on the optimal outcome. Specifically, a major flaw of the robust model is that it completely ignores available information regarding the probabilities of the different scenarios. Indeed, robust optimization was developed to be effective when no specific probabilistic knowledge on the different realizations of U exists. Finally, the size of the robust formulation grows with the number of scenarios and therefore this formulation becomes less tractable as the number of scenarios increases. Recall that when the deterministic problem is a discrete optimization problem, i.e., $x \in \{0,1\}^N$, it may itself be very difficult to solve and by introducing scenarios the difficulty increases substantially.

Worst-Case Model: A more conservative formulation in comparison to the robust model is to optimize the worst-case problem. Let each element of the matrix U' be equal to the maximum value of the corresponding element over all scenarios, i.e., $u'_{ij} = \max_s u^s_{ij}$ (note that this is the most conservative value of u^s_{ij}, even if it is negative, since our constraints are of the smaller-equal type). The worst-case formulation is

$$
\begin{array}{ll}
\min & c^T x \\
\text{s.t.} & Dx \leq b^D, \\
& U'x \leq b^U, \\
& x \in X.
\end{array}
\qquad (Worst-Case\ Model)
$$

As can be seen, the worst-case model, which is a special case of Soyster (1973) in which uncertain matrix belong to a convex set of scenarios, has all the drawbacks of the robust model (its solution is even more conservative). The main upside of this model compared to the robust model is its smaller size that makes it more tractable in general (its complexity is the same as that of the deterministic model independent of the number of scenarios).

2.2 The Almost Robust Optimization Model

We next introduce a model that overcomes the issues of infeasibility and overconservatism, and which integrates the decision maker's risk attitude into the optimization process. This model, which we refer to as the *Almost Robust Optimization* (*A*RO) model trade-offs the solution value with robustness to find solutions that are almost robust (feasible under most realizations). In such a model infeasibility of the uncertain constraints under some of the scenarios may be allowed at a penalty.

Consider the jth uncertain constraint, $U_j^s x \leq b_j^U$. For every $j \in 1, .., J$, the amount of infeasibility of solution x in scenario s can be measured by

$$Q_j^s(x) = max\{0, (U_j^s x - b_j^U)\} \equiv (U_j^s x - b_j^U)^+, \quad (1).$$

Our approach allows other measures of infeasibility such as the absolute value of infeasibility, $Q_j^s(x) = |U_j^s x - b_j^U|$, which is applicable for equality constraints. In the body of the paper, we focus on the infeasibility measure (1), and defer the explanation of the incorporation of the absolute value measure to Appendix C.

We now introduce penalty functions to penalize uncertain constraint violations. Three alternative penalty functions (based on the available probabilistic information) are:

- *Expected Penalty Function:* this penalty function is suitable for situations where the decision maker has full probabilistic knowledge of the scenarios. This penalty function has the form:

$$\overline{Q}_j(x) = \sum_{s \in S} p_s \alpha_s Q_j^s(x), \quad (2),$$

where the α_s's is the per unit penalty paid for each unit of constraint violation under the different scenarios.

- *Max Penalty Function:* this penalty function is mostly applicable when the decision maker does not have knowledge of the probabilities (which is one of the main assumption in robust optimization). This penalty function has the form:

$$\overline{Q}_j(x) = max_{s \in S} \alpha_s Q_j^s(x), \quad (3).$$

This penalty function determines the maximum (worst-case) penalty over all scenarios.

- *Distributionally-Robust Penalty Function:* this penalty function can be applied in situations where the decision maker has some but limited information regarding the probability distribution of the scenarios, i.e., the decision maker knows that the probability distribution of the scenarios belongs to a known set of probability distributions, F. Let p_s^f be the probability of scenario s under distribution $f \in F$, and $\overline{Q}_j^f(x) = \sum_{s \in S} p_s^f \alpha_s Q_j^s(x)$ be the expected penalty of uncertain constraint j under distribution f. The distributionally-robust penalty function has the form:

$$\overline{Q}_j(x) = max_{f \in F} \overline{Q}_j^f(x), \quad (4).$$

The penalty function (4) determines the worst-case expected penalty over distributions in F. Therefore, this type of penalty can be used to enforce robustness with respect to the set of possible distributions.

For simplicity of the exposition, throughout the paper we focus on the expected penalty function as in (2) and assume that $\alpha_s = 1$ so that $\overline{Q}_j(x) = \sum_{s \in S} p_s Q_j^s(x)$ is the expected penalty. The inclusion of different α_s can be easily incorporated into the solution methodology and the proofs. In Appendix B, we extend the results to the max penalty (3) and distributionally-robust penalty (4) cases. Furthermore, in Section 6, we extend the summation used in (2) to

represent an expectation with respect to any well defined random variable. Note that chance-constraints (Charnes and Cooper (1959)) and integrated chance constraints (Klein Haneveld (1986)) are special case of the almost robust optimization model, e.g., by modifying the penalty function to $\overline{Q}_j(x) = \sum_{s\in S} p_s I\{Q_j^s > 0\}$ where $I\{Q_j^s(x) > 0\} = 1$ ($I\{Q_j^s > 0\} = 0$) if $Q_j^s > 0$ ($Q_j^s \leq 0$), the model can be converted into a chance-constrained model.

We further define $\overline{Q}(x) = [\overline{Q}_1(x), \overline{Q}_2(x), ..., \overline{Q}_J(x)]^T$. Let l_j denote the maximum penalty accepted by the decision maker for violating constraint j, so that the $J \times 1$ vector l denotes the penalty limits. Specifically, the vector l incorporates the decision maker attitude towards risk, i.e., a decrease in the values of l implies that the decision maker is less tolerant and is more conservative. Later in this section, we introduce the Robustness Index that the decision maker can use to determine an appropriate vector l based on her risk attitude.

Using the above notation, the ARO formulation is

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Dx \leq b^D, \\ & \overline{Q}(x) \leq l, \qquad\qquad (ARO) \\ & x \in X. \end{array}$$

As can be seen, this model allows the decision maker to control the magnitude of the penalties by changing l. For example, the robust model is a special instance of the ARO where $l_j = 0,\ j = 1, ..., J$, i.e., when the decision maker is not willing to risk infeasibility of any of the constraints under any scenario. Note that the model with $l_j = 0,\ j = 1, ..., J$ is equivalent to the robust counterpart (Ben-Tal and Nemirovski (1998), Ben-Tal et al. (2009)) in which the uncertain set is a convex hull of the scenarios.

As stated earlier, when the coefficients of the objective function are uncertain, we augment them into the uncertain constraints by using the objective function $min\ z$ and adding constraint $z - c^T x \leq 0$. In the almost robust optimization model, this augmentation would result in optimizing the worst-case objective over all scenarios, $z \leq c^{sT} x\ \ \forall s$, i.e, the almost robust model with uncertain objective coefficient is equivalent to the problem: $min\ max_s c^{sT} x$ subject to the ARO constraints. In general, other objective functions can also be considered, e.g., the mean value objective, $min\ \sum_{s\in S} p_s c^{sT} x \equiv min\ \overline{c}^T x$, where $\overline{c} = \sum_{s\in S} p_s c$.

Such a mean value objective function can be extended to the distributionally-robust model (discussed in Section 6) in which the objective is: $min\ max_{f \in F} \sum_{s \in S} p_s^f c^{sT} x$.

We now focus on the ARO with integrality constraints $x \in X \subseteq \{0,1\}^N$, which we denote as the *Almost Robust Discrete Optimization* (ARDO) model. The ARDO incorporates data uncertainty into discrete optimization problems (unlike the deterministic model), while trading-off the solution value and the model robustness in view of the total penalty allowed by the decision maker. As perviously discussed, the ARDO produces solutions that are almost robust, and that are less conservative than those of the robust or worst-case models. Furthermore, the ARDO is flexible in incorporating the decision maker's risk attitude.

The main drawback of the ARDO is that similar to the robust model, the combination of integer variables and scenarios can make the ARDO computationally intractable. Specifically, the intractability is caused by the integrality of the decision variables and the need to linearize the penalty constraints by introducing new variables and adding new inequality constraints that significantly increases the problem's size. Therefore, in Section 3 we propose a decomposition approach that overcomes this difficulty.

2.3 Robustness Index

As stated earlier, in order to aid the decision maker in determining the appropriate penalty limits, l, and to show a connection between the decision maker's risk preference and the penalty limit, we introduce the Robustness Index. Specifically, the Robustness Index can be used to investigate the trade-off between the changes in the objective function and in the penalty as the penalty limit changes. To formally introduce the Robustness Index, let x_l^* denote the optimal solution of the ARO model given the penalty limits l, and let x_0^* be the optimal solution when violation is not allowed. Recall that the objective function value of the problem where infeasibility is not allowed is equal to the objective function of the (fully) robust model.

Let $I_{1\times J}$ denote a $J \times 1$ vector of 1s. We introduce the following Robustness Index:

$$Robustness\ Index = \frac{improvement\ (decrease)\ in\ objective\ function\ value}{increase\ in\ total\ penalty} = \frac{c^T x_0^* - c^T x_l^*}{\overline{Q}(x_l^*)^T I_{1\times J}}, \quad (5).$$

The Robustness Index indicates the relative improvement in the objective function value,

$c^T x_0^* - c^T x_l^*$, (the difference between the objective function with penalty limits, l, and the objective function values where infeasibility is not allowed, i.e., $l = 0$) over the increase in the total penalty ($\overline{Q}(x_l^*)^T I_{1\times J}$).

Similar to "the price of robustness" (Bertsimas and Sim (2004)) we expect that allowing some infeasibility while paying a penalty should improve the objective function. Moreover, it is reasonable to expect that the benefit from allowing a higher deviation alongside an increase in the corresponding penalty will be less effective as the allowed penalty increases. That is we expect some decreasing returns for allowing a higher penalty. As the Robustness Index in (5) measures the benefit of allowing a higher expected penalty, we expect that it will be a convex decreasing function of the penalty, l. Indeed our computational results in Section 5.2 agrees with this intuition. Moreover, our results suggest that often a little increase in the expected penalty is all that is required in order to substantially improve the objective function.

3. The Decomposition Approach

As stated in the previous section, the *A*RDO generally becomes computationally intractable as the number of scenarios increases. To overcome the issue of tractability and to solve the *A*RDO efficiently, we decompose it into a deterministic master problem and a simple subproblem. As in most decomposition methods (see, for example, Benders (1962), Geoffrion and Graves (1974), Blankenship and Falk (1976), and Hooker and Ottosson (2003)), we start by solving a relaxation of the original problem (the master problem) and iteratively adding cuts (derived from solving the subproblem) until a feasible (optimal) solution is found. Specifically, in the master problem, the uncertain parameters are replaced with their expected values (the master problem is similar to the deterministic problem). The single subproblem calculates the penalty of the different scenarios given the master problem solution and generates cuts when this solution is infeasible in the original *A*RDO (i.e., when the master problem solution violates at least one penalty constraint). The cuts, when added to the master problem, eliminate *at least* the current infeasible master solution. The decomposition approach iterates between solving the master problem and the subproblem until the optimal

solution to the ARDO is found. Note

In order for the decomposition model to generate an optimal solution of the ARDO, it must satisfy the following two conditions:

- *Condition 1- Lower Bound:* In any iteration, the master problem solution must be a lower bound on that of the ARDO.
- *Condition 2 - Validity of the Cuts:* A cut is valid if it: (i) excludes the current master problem solution, which is not feasible to the ARDO, and (ii) does not remove any feasible solutions of the ARDO.

Condition 1 ensures that any feasible solution of the ARDO is also feasible to the master problem. Condition 2(i) ensures that the decomposition converges in a finite number of steps, i.e., this condition guarantees that the decomposition does not get stuck in a loop, and since the decision variables have finite domains (each decision variable can only take the value of 1 or 0), the decomposition has a finite number of iterations. Condition 2(ii) guarantees optimality, i.e., because the cuts never remove feasible solutions of ARDO, while removing infeasible solutions of ARDO (condition 2(i)), and since all feasible solutions of ARDO are also feasible for the master problem (condition 1), the decomposition is guaranteed to find an optimal solution to the ARDO, if one exists.

3.1 Master Problem

Let $\overline{U} = E_s[U^s] = [\sum_{s\in S} p_s U_1^s, ..., \sum_{s\in S} p_s U_J^s]^T$ be a $J \times N$ matrix where each element is equal to the expectation of the corresponding element over all possible realization, and let the $J \times 1$ vector $l' = l + b^U$. The deterministic discrete optimization formulation of the master problem is:

$$
\begin{array}{ll}
\min & c^T x \\
\text{s.t.} & Dx \leq b^D, \\
& \overline{U}x \leq l', \qquad\qquad (Master\ Problem) \\
& cuts, \\
& x \in \{0,1\}.
\end{array}
$$

As stated earlier, *cuts* are constraints that are added to the master problem when its current solution is not feasible for the ARDO. As can be seen, the master problem formulation is equivalent to a deterministic formulation with an extra set of cuts. Note that the cuts that we introduce later are both linear and deterministic.

3.2 Subproblem

The subproblem focuses on checking the master problem solution under different realizations and creating cuts for the master problem when the current solution is not feasible to the ARDO. These cuts prevent the current infeasible master problem solution from being considered in future iterations without removing any feasible solution of the original ARDO.

The subproblem is comprised of three steps: (i) penalty calculation, (ii) feasibility check, and (iii) cuts generation. To discuss these steps we let x^t be the optimal master problem solution in iteration t of the decomposition approach, $\circ$ denote the entrywise product of two vectors (i.e., this binary operation takes two vectors of the same dimension and produces another vector where each element i is the product of elements i of the original two vectors), and I a vector of 1s with the same dimension as x (i.e., I is a $N \times 1$ vector of 1s).

Step 1 - Penalty Calculation: For any scenario $s \in S$ and uncertain constraint $j = 1, .., J$ calculate $Q_j^s(x^t)$. Then, calculate the penalty of each uncertain constraint given the master solution, $\overline{Q}_j(x^t) = \sum_{s \in S} p_s Q_j^s(x^t),\ j = 1, .., J$.

Step 2 - Feasibility Check: Check if $\overline{Q}(x^t) \leq l$. The feasibility check has two possible outcomes:

- *No Violation:* If the optimal solution of the master problem does not violate any penalty constraints, the decomposition approach is terminated with the current solution, which is optimal.
- *Violation:* If the optimal master solution violates at least one of the penalty constraints, determine the set of penalty constraints that are violated in iteration t, denoted by V^t, and the set of scenarios where uncertain constraint $j \in V^t$ has a non-zero penalty, denoted by S_j^t where $S_j^t = \{s | Q_j^s(x^t) > 0\}$, and continue to step 3.

Step 3 - Cuts Generation: Generate and add to the master problem the cuts:

$$\overline{Q}_j(x^t) - \sum_{s \in S_j^t} p_s((U_j^s \circ x^{t^T})(I - x \circ x^t) - U_j^s((I - x^t) \circ x)) \leq l_j, \quad \forall j \in V^t. \quad (6)$$

As can be seen, the cuts are linear and deterministic. Note that the first term in the left hand side of the cuts (6) is the penalty of the master problem solution at iteration t (x^t) as calculated in step 1. The term $(U_j^s \circ x^{t^T})(I - x \circ x^t)$ measures the total reduction/increase in penalties of uncertain constraint j when some of the elements of x^t are switched from 1 to 0 in x. That is, by removing the elements with positive coefficients the total penalty decreases and removing elements with negative coefficient increases the total penalty. Similarly, the term $U_j^s((I - x^t) \circ x)$ measures the total increase/reduction in penalties of uncertain constraint j when some of the elements of x^t are switched from 0 to 1 in x. That is, by adding the elements with negative coefficients (of U_j^s) the total penalty decreases, and adding elements with possible coefficient increases the total penalty. Therefore, for any future master problem solution, x, the summation measures the maximum change in the penalty of constraints j. The cut ensures that the changes in the current infeasible solution must reduce the current penalty to a level below the allowable limit.

The pseudo code of the implementation of the decomposition approach for solving the ARDO is given in Algorithm 1.

Algorithm 1: The Decomposition Approach for Solving the *A*RDO

```
1   cuts ← ∅, t ← 1;
2   terminate ← false;
3   while terminate ≠ true do
4       terminate ← true;
5       Solve master problem to obtain x^t;
6       Solve subproblem given x^t:
6.1     for each uncertain constraint j do
6.2         Run penalty calculation to obtain Q̄_j(x^t) ;
6.3         Run feasibility check to check Q̄_j(x^t) ≤ l_j;
6.4         if Q̄_j(x^t) > l_j then
6.5             Run cut generation to generate new cuts;
6.6             cuts ← cuts + new cuts;
6.7             terminate ← false;
6.8     return cuts ;
7       t ← t + 1;
```

The decomposition model generates an optimal solution of the *A*RDO since it satisfies the two conditions stated earlier, i.e., the lower bound and cut validity conditions. Theorem 1, whose proof appears in Appendix A, summarizes this discussion.

Theorem 1. (*Convergence of the decomposition approach*) *If the* A*RDO has an optimal solution, the decomposition approach finds the optimal solution in a finite number of iterations.*

In theory, for some problems, the number of iterations of the decomposition can be large, however, as will be shown in the computational experiments, in practice, the average number of iterations of the decomposition is small; in our experiments the average number of iterations was about 3.

Note that the cuts in (6) are the most general cuts, however, other problem specific cuts can also be used. For example, if all coefficients of U are non-negative (e.g., units of time or demand), an alternative simpler valid cut would not include the term $U_j^s((I - x^t) \circ x)$ in the summation in (6).

4. Applications

In this section, we illustrate our methodology, both the modeling and the decomposition approaches, on three well studied discrete optimization problems: the knapsack problem with uncertain weights, the capacitated facility location problem with uncertain demands, and the vehicle routing problem with uncertain demands and travel times.

4.1 The Knapsack Problem with Uncertain Weights

We first apply our methodology to the knapsack problem. The knapsack problem seeks to select a subset of items from a set of items K to maximize the profit under the condition that the total weight of the selected items does not exceed the knapsack's capacity. Let r denote the knapsack's capacity, w_k denotes the weight of item $k \in K$, π_k denotes the profit of item k, and let the binary variable x_k indicate whether product k is selected. The deterministic integer programming formulation of the knapsack problem is:

$$\begin{aligned} \max \quad & \sum_{k \in K} \pi_k x_k \\ s.t. \quad & \sum_{k} w_{k \in K} x_k \leq r, \\ & x_k \in \{0,1\}, \qquad k \in K. \end{aligned}$$

This deterministic problem is known to be NP-hard, see e.g., Papadimitriou (2003). In the above formulation, it is assumed that the weights are known. In practice, however, weights may be uncertain. To model the uncertainty, we assume that the weights are uncertain, w_k^s, and take values from the set of scenarios $s \in S$, with known probabilities p_s. Since weights are uncertain, under some scenarios, the (single) capacity constraint may be violated, i.e., the total weight of the selected items may exceed the capacity of the knapsack.

We use the ARDO model presented in Section 2 to take this uncertainty into account. Specifically, in some scenarios, violations are allowed, $Q^s(x) = (\sum_k w_k^s x_k - r)^+$, at a penalty, $\overline{Q}(x) = \sum_{s \in S} p_s Q^s(x)$. Note that the penalties due to these violations are dealt with outside the optimization problem (e.g., additional capacity may be bought at a known price). The ARDO model, maximizes the total profit such that the total penalty, $\overline{Q}(x)$, does not exceed a given threshold value, l. The threshold value is used to tradeoff robustness with profit,

and its value reflects the decision maker's tolerance level towards exceeding the knapsack's capacity.

In this problem, we do not have deterministic items, i.e., the matrix D and vector b^D do not exist, $c = (-\pi_1, -\pi_2, ..., -\pi_k)$, $U^s = (w_1^s, w_2^s, ..., w_K^s)\ \forall s$, $b^U = r$, and $x = [x_1, ..., x_k]$. Note that there is only one constraint in U, i.e., $J = 1$.

The ARDO formulation of the knapsack problem with uncertain weights is

$$\begin{array}{ll} \max & \sum_{k\in K} \pi_k x_k \\ s.t. & \overline{Q}(x) \leq l, \\ & x_k \in \{0,1\}, \quad k = 1, ..., K. \end{array}$$

We next demonstrate the use of the decomposition approach for solving this problem.

Master Problem: Let $\overline{w}_k = \sum_s p_s w_k^s$, and $l' = l + r$. The master problem formulation is:

$$\begin{array}{ll} \max & \sum_{k\in K} \pi_k x_k \\ s.t. & \sum_{k\in K} \overline{w}_k x_k \leq l', \\ & cuts, \\ & x_k \in \{0,1\}, \qquad k = 1, ..., K. \end{array}$$

Subproblem:

- *Step 1 - Penalty Calculation:* Given the set of items selected in iteration t of the master problem, $K^t = \{k | x_k^t = 1\}$, calculate the capacity violation of scenario s, $Q^s(x^t) = (\sum_{k\in K^t} (w_k^s) - r)^+$, and the penalty of the master solution, $\overline{Q}(x^t) = \sum_{s\in S} p_s Q^s(x^t)$.
- *Step 2 - Feasibility Check:* If $\overline{Q}(x^t) \leq l$, terminate. Otherwise, go to step 3
- *Step 3 - Cut Generation:* Generate and add the cuts:

$$\overline{Q}(x^t) - \sum_{\{s|Q^s(x^t)>0\}} p_s (\sum_{k\in K^t} w_k^s (1 - x_k) - \sum_{k\in K - K^t} w_k^s x_k) \leq l.$$

The inner summation term is the change in the penalty, given that the set of selected items is changed: the maximum amount of reduction/increase of the penalty of scenario

s in removing/adding one item is the weight of that item in scenario s. As can be seen, the cut prevents the set of items in K^t or a superset of it from being selected in future iterations.

As discussed in Section 3, since the weights are non-negative, another valid cut would be to remove the second term in the summation, $\sum_{k \in K - K^t} w_k^s x_k$.

4.2 The Capacitated Facility Location Problem with Uncertain Demand

Consider the single-source capacitated facility location problem. In this problem, there is a set of potential facilities, H, and a set of customers, G. For each facility h there is an opening cost, f_h, and a capacity on the total demand that can be served, b_h. Furthermore, each customer has demand, d_g, and providing service to customer g from facility h has an associated cost, c_{gh}. The goal is to select the set of facilities to open and assign customers to facilities in order to minimize the total cost, under the conditions that facility capacities are not exceeded.

Let the binary variable y_h indicate whether facility h is selected and x_{gh} indicate whether customer g is assigned to facility h. The integer programming formulation of the single-source capacitated facility location problem is:

$$
\begin{aligned}
\min \quad & \sum_{h \in H} f_h y_h + \sum_{g \in G} \sum_{h \in H} c_{gh} x_{gh} & & \\
s.t. \quad & \sum_{h \in H} x_{gh} = 1, & g \in G & \quad (7) \\
& x_{gh} \leq y_h, & g \in G, h \in H, & \quad (8) \\
& \sum_{g \in G} d_g x_{gh} \leq b_h y_h, & h \in H, & \quad (9) \\
& x_{gh}, y_h \in \{0,1\}, & g \in G, h \in H. & \quad (10)
\end{aligned}
$$

This deterministic problem is NP-hard, see e.g., Mirchandani and Francis (1990). The above formulation assumes that the customer demand data is perfectly known. In practice, however, this assumption may not hold. We use the ARDO model to consider such uncertainties. Assume that the customer demands are uncertain, and that different demand conditions are considered as different scenarios, d_g^s, with known probability p_s. Due to this uncertainty, the capacity constrains in (9) may no longer hold under all scenarios. We will

allow violations of constraints $h \in H$ under some scenarios, $Q_h^s(x) = (\sum_g d_g^s x_{gh} - b_h y_h)^+$, at a penalty, $\overline{Q}_h(x) = \sum_s p_s Q_h^s(x)$. The objective is to minimize the total cost such that the penalty of each facility, $\overline{Q}_h(x)$, does not exceed a given threshold value, l_h. Recall that the threshold value for each facility reflects the decision makers willingness to have less than the required capacity at this potential facility. Note that if there is no facility located at location h, we have $y_h = x_{gh} = 0 \ \forall g$, and therefore $\overline{Q}_h(x) = 0$.

In this problem, the deterministic part of the formulation ($Dx \leq b^D$) is comprised of constraints (7) and (8), while the uncertain part ($Ux \leq b^U$) is presented by constraint (9).

The ARDO formulation of the single-source capacitated facility location problem with uncertain demand is

$$\begin{array}{lll} \min & \sum_{h\in H} f_h y_h + \sum_{g\in G}\sum_{h\in H} c_{gh}x_{gh} & \\ s.t. & (7), (8), (10), & \\ & \overline{Q}_h(x) \leq l_h, & h \in H. \end{array}$$

We apply the decomposition approach to solve this problem as follows:

Master Problem: Let $\overline{d_g} = \sum_s p_s d_g^s$. The formulation of the master problem is:

$$\begin{array}{lll} \min & \sum_{h\in H} f_h y_h + \sum_{g\in G}\sum_{h\in H} c_{gh}x_{gh} & \\ s.t. & (7), (8), (10), & \\ & \sum_{g\in G} \overline{d_g} x_{gh} \leq l_h + b_h y_h, & h \in H, \\ & cuts. & \end{array}$$

Subproblem:

- *Step 1 - Penalty Calculation:* Given the set of facilities opened in the solution of the master problem in iteration t, $H^t = \{h | y_h^t = 1\}$, and the set of customers assigned to facility $h \in H^t$, $G_h^t = \{g | x_{gh}^t = 1\}$, calculate the penalty cost of all facilities $h \in H^t$ in scenario s, $Q_h^s(x^t) = (\sum_{g\in G_h^t} d_g^s - b_h)^+$, and the expected penalty cost of each facility over all the scenarios, $\overline{Q}_h(x^t) = \sum_s p_s Q_h^s(x^t), \quad h \in H^t$.
- *Step 2 - Feasibility Check:* If $\overline{Q}_h(x^t) \leq l_h, \quad \forall h \in H$, terminate. Otherwise, determine the set of facilities that violate the penalty constraints, V^t, and go to step 3

- *Step 3 - Cut Generation:* Generate and add the cuts:

$$\overline{Q}_h(x^t) - \sum_{\{s|Q_h^s(x^t)>0\}} p_s\Big(\sum_{g\in G_h^t} d_g^s(1-x_{gh}) - \sum_{g\in G-G_h^t} d_g^s x_{gh}\Big) \leq l_h, \quad h \in V^t.$$

The inner summation term is the decrease/increase in the penalty cost of the corresponding facility, given that some of the customers are reassigned. As can be seen, the cut prevents the customers in G_h^t or a super-set of them from being assigned to facility h in future iterations.

Similar to the knapsack problem, since the demands are non-negative, by removing the second term in the summation, $\sum_{g\in G-G_h^t} d_g^s x_{gh}$, the cut would still be valid.

4.3 Vehicle Routing Problem with Uncertain Demands and Travel-Times

The last application we consider is the vehicle routing problem. Let $G = (N, A)$ be an undirected graph, where N is the set of nodes and A is the set of arcs. Node 0 is a depot of a fleet of K identical vehicles each with capacity q, and all routes must start and end at the depot. All other nodes represent customers with an associate demand $d_u,\ u \in N\backslash\{0\}$. Each arc (u, v) has a distance or cost c_{uv}, and using arc (u, v) generates travel time z_{uv} for the vehicle performing the service. The objective is to design K vehicle routes of minimum total cost/distance, such that (i) the total demand of all customers in any route does not exceed the vehicle capacity, q, and (ii) the total travel time of any vehicle k does not exceed it's maximum travel time limit, r_k (e.g., due to union work time regulation).

Let the binary variable x_{uvk} indicate if arc (u, v) appears on the route of vehicle k, and let y_{uk} indicate whether customer u is served by vehicle k. The integer programming formulation of the vehicle routing problem is:

$$\min \sum_{k} \sum_{(u,v)\in A} c_{uv} x_{uvk}$$

$$s.t. \sum_{v\in N\setminus\{0\}} x_{uvk} = y_{uk}, \qquad u \in N, \forall k, \tag{11}$$

$$\sum_{k} y_{uk} \leq 1, \qquad u \in N, \tag{12}$$

$$\sum_{u\in N\setminus\{0\}} x_{iok} - \sum_{v\in N\setminus\{0\}} x_{ovk} = 0, \quad o \in N, \forall k, \tag{13}$$

$$\sum_{v\in N\setminus\{0\}} x_{0vk} = 1, \qquad \forall k, \tag{14}$$

$$\sum_{u\in N\setminus\{0\}} x_{u0k} = 1, \qquad \forall k, \tag{15}$$

$$\sum_{u\in N\setminus\{0\}} d_u y_{uk} \leq q, \qquad \forall k, \tag{16}$$

$$\sum_{(u,v)\in A} z_{uv} x_{uvk} \leq r_k, \qquad \forall k, \tag{17}$$

$$x_{uvk}, y_{uk} \in \{0,1\}, \qquad (u,v) \in A, u \in N, \forall k. \tag{18}$$

This deterministic problem is NP-hard, see e.g., Lenstra and Kan (1981). Unlike the previous two applications in which only one aspect was uncertain, we now assume that two aspects of the problem are affected by uncertainly, i.e., the decision maker faces uncertainty in both the demands and the travel times. Specifically, we use the ARDO model to incorporate uncertainties in demands and travel times. Assume different demands and travel times are represented by a set of demand scenarios (S^{DE}) and a set of travel time scenarios (S^{TT}), i.e., serving customer u in scenario $s \in S^{DE}$ generates demand d_u^s and using arc (u,v) in scenario $s' \in S^{TT}$ generates a travel time $z_{uv}^{s'}$. Furthermore, each demand scenario $s \in S^{DE}$ occurs with probability p_s^{DE}, while each travel time scenario $s' \in S^{TT}$ has a known probability $p_{s'}^{TT}$. Since demands and travel times are uncertain, constraints (16) and (17) may be violated under some scenarios. The ARDO model will allow such infeasibility at a penalty. Since two aspects of the problem are uncertain, we need to determine the infeasibility for both the demands, $Q_k^{s,DE}(x) = (\sum_{u\in N\setminus\{0\}} d_u^s y_{uk} - q)^+ \ \forall s \in S^{DE}$, and the travel times, $Q_k^{s',TT}(x) = (\sum_{(u,v)\in A} z_{uv}^{s'} x_{uvk} - r_k)^+ \ \forall s' \in S^{TT}$. The objective of the ARDO model is to minimize total cost/distance, such that the demand penalty, $\overline{Q}_k^{DE}(x) = \sum_{s\in S^{DE}} p_s^{DE} Q_k^{s,DE}(x)$, and travel time penalty, $\overline{Q}_k^{TT}(x) = \sum_{s'\in S^{TT}} p_{s'}^{TT} Q_k^{s',TT}(x)$, of each vehicle do not exceed threshold values l_k^{DE} and l_k^{TT} respectively.

In this problem, the deterministic part of the formulation ($Dx \leq b^D$) is comprised of constraints (11)-(15), while the uncertain part ($Ux \leq b^U$) is represented by constraints (16) and (17).

The *A*RDO formulation of the vehicle routing problem with uncertain travel times is:

$$\begin{array}{ll} \min & \displaystyle\sum_{k\in K}\sum_{(u,v)\in A} c_{uv}x_{uvk} \\ s.t. & (11)-(15),(18), \\ & \overline{Q}_k^{DE}(x) \leq l_k^{DE}, \qquad \forall k, \\ & \overline{Q}_k^{TT}(x) \leq l_k^{TT}, \qquad \forall k. \end{array}$$

As can be seen, the above formulation produces solutions that are almost robust with respect to the different realizations of demands and travel times.

We next apply the decomposition approach to solve this problem.

Master Problem: Let $\overline{d_u} = \displaystyle\sum_{s\in S^{DE}} p_s d_u^s$ be the mean demand of customer u and $\overline{z_{uv}} = \displaystyle\sum_{s'\in S^{TT}} p_{s'}^{TT} z_{uv}^{s'}$ be the mean travel time of arc (u,v). Furthermore, let $l_k^{DE\prime} = l_k^{DE} + q$ and $l_k^{TT\prime} = l_k^{TT} + r_k$. The master problem is:

$$\begin{array}{ll} \min & \displaystyle\sum_{k\in K}\sum_{(u,v)\in A} c_{uv}x_{uvk} \\ s.t. & (11)-(15),(18), \\ & \displaystyle\sum_{u\in N\setminus\{0\}} \overline{d_u} y_{uk} \leq l_k^{DE\prime}, \quad \forall k, \\ & \displaystyle\sum_{(u,v)\in A} \overline{z_{uv}} x_{uvk} \leq l_k^{TT\prime}, \quad \forall k, \\ & cuts. \end{array}$$

Subproblem:

- *Step 1 - Penalty Calculation:* Given the set of nodes ($N_k^t = \{u | y_{uk}^t = 1\}$) and the set of arcs ($A_k^t = \{(u,v) | x_{uvk}^t = 1\}$) assigned to vehicle k in iteration t, calculate the demand penalties, $\overline{Q}_k^{DE}(x^t) = \displaystyle\sum_{s\in S^{DE}} p_s^{DE}(\sum_{u\in N_k^t} d_u^s - q)^+ \; \forall k$, and the travel time penalties, $\overline{Q}_k^{TT}(x^t) = \displaystyle\sum_{s'\in S^{TT}} p_{s'}^{TT}(\sum_{(u,v)\in A_k^t} z_{uv}^{s'} - r_k)^+ \; \forall k$.

- *Step 2 - Feasibility Check:* If $\overline{Q}_k^{DE}(x^t) \leq l_k^{DE}$ and $\overline{Q}_k^{TT}(x^t) \leq l_k^{TT} \quad \forall k$, terminate. Otherwise, determine the set of vehicles that violate the demand penalty constraints, V_t^{DE}, and the travel time penalty constraints, V_t^{TT}. Go to step 3.

- *Step 3 - Cut Generation:* Generate and add the following two sets of cuts (one for the demands and one for the travel times):

$$\overline{Q}_k^{DE}(x^t) \; - \sum_{\{s|Q_k^{s,DE}(x^t)>0\}} p_s^{DE}\Big(\sum_{u\in N_k^t} d_u^s(1-y_{uk}) - \sum_{u\in N\setminus\{0\}-N_k^t} d_u^s y_{uk}\Big) \leq l_k^{DE}, \quad \forall k \in V_t^{DE},$$

$$\overline{Q}_k^{TT}(x^t) \; - \sum_{\{s'|Q_k^{s,TT}(x^t)>0\}} p_{s'}^{TT}\Big(\sum_{(u,v)\in A_k^t} z_{uv}^{s'}(1-x_{uvk}) - \sum_{(u,v)\in A-A_k^t} z_{uv}^{s'} x_{uvk}\Big) \leq l_k^{TT}, \quad \forall k \in V_t^{TT}.$$

  The inner summation terms of the two cuts are the decrease/increase in the demand and travel time penalties of the corresponding vehicle, given that the nodes/arcs are reassigned. Note that in iterations where only one type of uncertain penalty constraint (either the demand or the travel time penalty constraint) is violated, the decomposition approach only generates cuts for the violated type, i.e., when only demand penalty constraints are violated, the decomposition approach solely generates demand cuts, while in iterations with only travel time penalty constraint violations, only travel time cuts are generated.

5. Computational Experiments

In previous sections we claimed that our decomposition approach is efficient in solving the *A*RO. In this section, we present experimental results for the capacitated facility location problem with uncertain demand presented in the previous section to support our claim. Similar observations of these experiments were detected in the other two applications discussed and therefore are not shown here.

We conducted two experiments to address the following:

1. *Efficiency:* Our initial experiment addresses the efficiency (solvability and tractability) of the decomposition approach by comparing the performance of the ARDO with and without the decomposition approach. (Throughout this section, we use the abbreviation ARDO to denote the model without the decomposition.)

2. *Value of Almost Robustness:* In the second experiment, we use the Robustness Index (defined in Section 2.3) to show the connection between the decision-maker's risk preference and the penalty limit. Furthermore, by using this index, we highlight the potential benefits of being almost robust as opposed to being fully robust.

Problem Instances: The problem instances were randomly generated as follows: the facility capacities are uniformly drawn from an integer interval $b_h \in [50, 200]$; the fixed facility opening costs, f_h, are randomly generated based on the capacities b_h and are calculated as: $f_h = b_h(10 + rand[1; 15])$, where 10 is the per unit capacity cost. We have added a random integer multiplier from $[1, 15]$ to take into account differences in property costs. The assignment costs are uniformly generated from an interval $c_{gh} \in [1, 100]$. The demands for each scenario is uniformly generated from the interval $d_g^s \in [1, 30]$. To generate the probabilities of the different scenarios we assigned each one of them a random number in the range $[1, 100]$ in a uniform manner. We then normalized these numbers to probabilities by dividing by the sum of the numbers assigned to all of the scenarios.

Overall, the problem instances consist of three sizes: 30×15 (i.e., 30 customers, 15 possible facility sites), 40×20, and 50×25; four number of scenarios: 5, 15, 30, and 60; four different penalty limits (l) 10, 20, 30, and 40. Finally, six instances of each size-scenario-limit combination are generated for a total of $3 \times 4 \times 4 \times 6 = 288$ instances.

The tests were performed on a Dual Core AMD 270 CPU with 1 MB cache, 4 GB of main memory, running Red Hat Enterprise Linux 4. The models are implemented in ILOG CPLEX. A time limit of three hours was used, and for unsolved instances (instance that where timed-out) the time-limit is used in the calculations (we also report the percentage of unsolved instances).

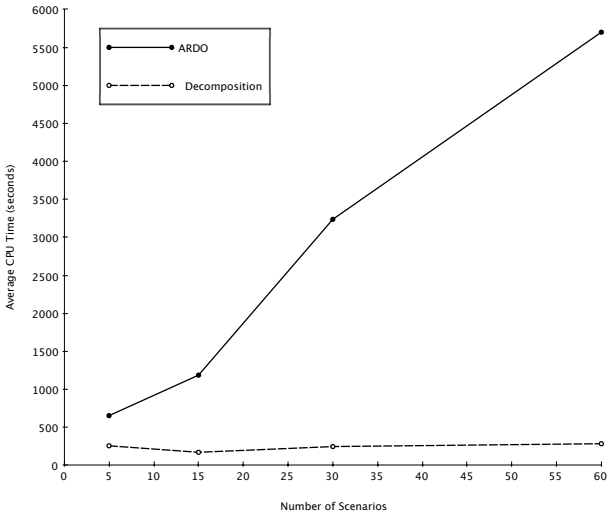

Figure 1: **Average** Run-Time of *A*RDO and Decomposition Approach as a Function of the Number of Scenarios.

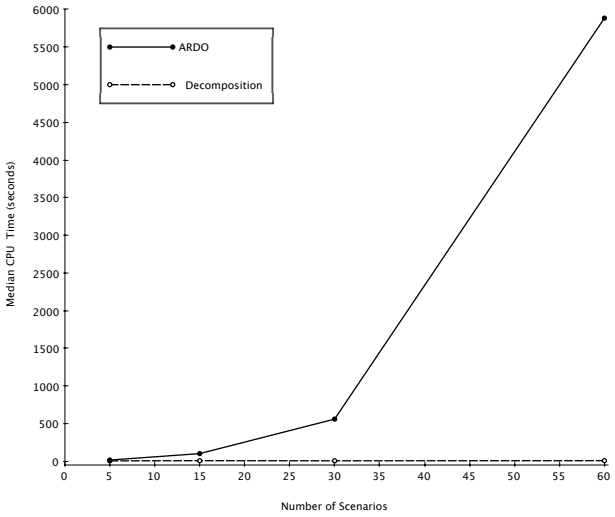

Figure 2: **Median** Run-Time of *A*RDO and Decomposition Approach as a Function of the Number of Scenarios.

5.1 Efficiency of the Decomposition Approach

Figures 1 and 2 illustrate the effects of the number of scenarios on the run-time performance of the *A*RDO and the decomposition approach. As can be seen, the mean and median run time of the *A*RDO drastically increases with the number of scenarios. However, the number of scenarios has only a minor effect if any on the run time of the decomposition approach.

Table 2 gives the mean, median, and the 10th and 90th percentiles of the CPU time in seconds required to solve each problem instance for each scenario size for the *A*RDO and the decomposition approach. The column labeled "% Uns." indicates the percentage of the problems for which the models were terminated because of the time limit (of 3 hours). For the decomposition approach, the column labeled "Iter." indicates the average number of iterations needed to converge to optimality. The "Time ratio" for a given instance is calculated as the *A*RDO runtime divided by the decomposition run-time. The mean over each instance in each subset was then calculated.

As Table 2 indicates the decomposition model is very efficient. Specifically, the Decomposition approach is unable to find the optimal solution within the time limit in only 1.0% of instances, while the *A*RDO is unable to find the optimal solution in 19.5% of the instances. For the *A*RDO, we see an increasing trend in the number of unsolved instance as

Number of Scenarios	*A*RDO					Decomposition						Time Ratio
	Time				% Uns.	Time				% Uns.	% Iter.	
	Mean	Median	Percentile			Mean	Median	Percentile				
			10	90				10	90			
5	655	13.2	1.13	309.41	4.2	258	2.1	0.24	61.77	1.3	3.2	18
15	1188	98.9	4.17	1187.98	6.9	133	4.9	0.28	72.59	0	2.4	91
30	3238	557.4	18.39	10800	23.6	248	3.2	0.33	65.63	1.3	3.4	240
60	5524	5883.7	70.67	10800	43.1	286	5.1	0.28	77.78	1.3	3.1	566
Overall	2651	222.8	5.03	10800	19.5	231	3.7	0.27	70.73	1.0	3.0	229

Table 2: The mean, median, and the 10th and 90th percentiles of the CPU time (in Seconds) and percentage of unsolved problem instances ("%*Uns.*") for the *A*RDO and decomposition approaches, and the mean number of iterations ("Iter.") for the decomposition approach. "Overall" indicates the mean results over all problem instances.

the number of scenarios increases (up to 43.1%). Furthermore, as the time ratios indicate, on average, the decomposition approach is 229 times faster than *A*RDO. We again see an increasing trend of the time ratios as the number scenarios increase (up to 566). Finally, the table demonstrates that the decomposition approach converges to optimality on average in 3 iterations.

Figure 3 shows a scatterplot of the run-times for both the *A*RDO and the decomposition approach. Both axes are log-scaled, and the points below the $x = y$ line indicate a lower runtime for the decomposition approach. On all but 8 of the 288 instances (over 97%), the decomposition achieves equivalent or better run-time (Note that 6 out of the 8 instances that are dominated are for the problem instances with 5 scenarios).

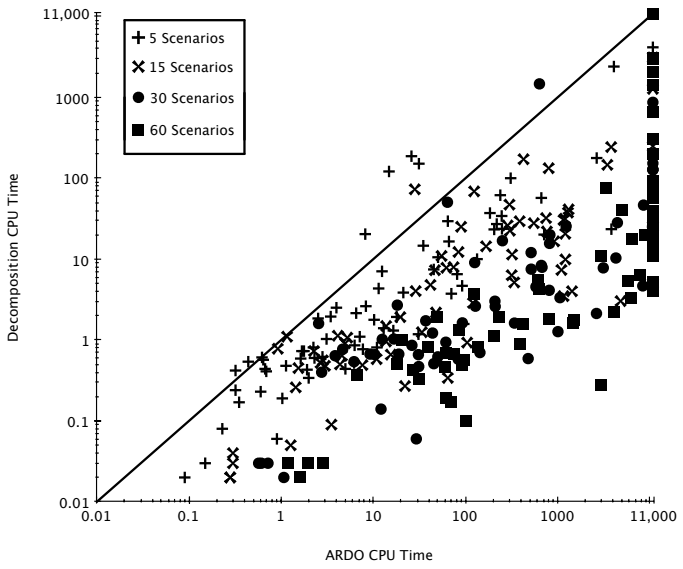

Figure 3: Runtime of the *A*RDO (x Axis, Log-Scale) vs. decomposition model (y Axis, Log-Scale)

We summarize this discussion with the following observation:

Observation 1. (*Tractability of the decomposition approach*) The decomposition approach is computationally more tractable than the *A*RDO, i.e., it is up to three orders-of-magnitude faster, and can find the optimal solution for many more problems within the time limit.

5.2 Value of Almost Robustness

In this section, by using the Robustness Index in (5), we computationally investigate the trade-off between the changes in the objective function and in the penalty as the penalty limit changes. This investigation demonstrates how the Robustness Index can be used by the decision makers to choose an appropriate penalty limit to better suit her risk preferences. It also highlights the benefits of being almost robust as opposed to fully robust.

To study the Robustness Index, we calculated it for all the problem instances discussed earlier for the capacitated facility location problem with uncertain demand. We summarized the results in Table 3 and Figure 4. In Table 3, we show the mean, median, max, and the 10th and 90th percentiles of the Robustness Index over all problems instances with a particular l value. Figure 4 depicts the changes in the average Robustness Index (the average is taken over all the instances with a given l) as a function the penalty limit. As expected from the discussion in Section 2.3, as the penalty limit increases the average incremental improvement in the objective function value increases slower than the average incremental increases in the penalty. Therefore, it may not be necessary for the decision maker to choose a very high penalty limit since her relative gain (in terms of improvement in the solution value) may not grow as fast as the loss (in terms of the penalty).

This experiment also demonstrates that allowing a small amount of infeasibility can substantially improve the solution value. Specifically, from Table 3 we can see that as we go from $l = 0$ to $l = 5$ the Robustness Index can be as high as 1397, which indicates that the improvement in the objective function value can be up to 1397 times more than the expected penalty. This suggests that the solution of the *A*RO model can be (substantially) less conservative than that of the robust model, and therefore, highlights the potential benefits

Penalty Limit (l)	Robustness Index				
	Mean	Median	Max	Percentile	
				10	90
5	105.4	100.2	1397.8	6.4	306.6
10	43.3	40.8	146.3	5.3	71.3
20	30.6	27.7	103.1	4.5	50.3
30	23.1	22.8	103.1	4.1	31.5
40	21.7	19.3	103.1	3.8	26.9

Table 3: The mean, median, max, and the 10th and 90th percentiles of the Robustness Index as a function of the penalty limit, l.

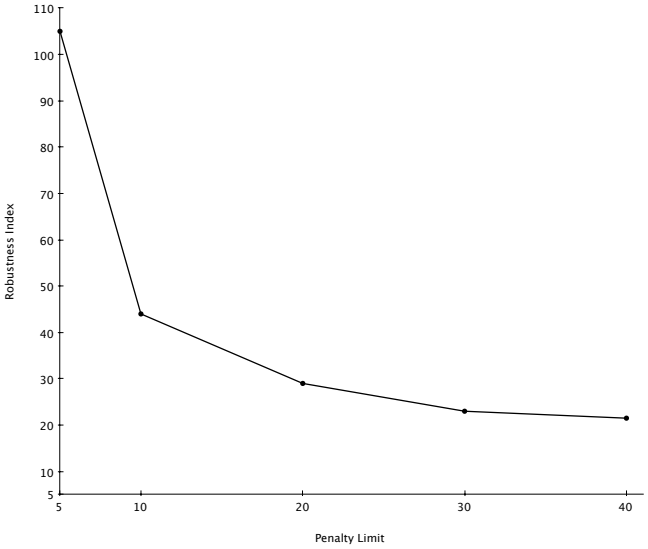

Figure 4: The mean robustness index as a function of the penalty limit.

of being almost robust (instead of being fully robust). Moreover, Table 3 also emphasizes that a little increase in the expected penalty is all that is required in order to substantially improve the objective function.

To summarize:

Observation 2. (*The Robustness Index*) The decision maker can use the Robustness Index to determine the appropriate penalty limit based on her risk attitude. Furthermore, by using the Robustness Index the decision maker can not only weigh the potential benefits of being almost robust as opposed to being fully robust, but to also choose a (possibly small) amount of penalty to (possibly substantially) improve the objective function.

6. General Uncertainty Structure

In the previous sections, we focused on the case where the uncertainty is represented by a set of scenarios with known probabilities. However, our approach is also applicable when uncertainty has a more general structure.

We assume that the uncertain matrix U has a multivariate cumulative distribution, f (representing the joint distribution of u_{ij}, for each i, j), with mean μ, and covariance ma-

trix Σ, and support $\Omega(U)$. Under such uncertainty, based on the available probabilistic information, the penalty function of constraint j, can take the following forms:

- *Expected Penalty Function:* $\overline{Q}_j(x) = E[(U_j x - b_j^U)^+] = \int_{\Omega(U)} (U_j x - b_j^U)^+ dF, \qquad (19),$

- *Distributionally-Robust Penalty Function:* Let F denote the of set of probability distributions with mean μ and covariance matrix Σ that is assumed to include the true distribution f. The distributionally-robust penalty function has the form:

$$\overline{Q}_j(x) = max_{f \in F} E_f[(U_j x - b_j^U)^+], \qquad (20).$$

Therefore, given the appropriate penalty function, the almost robust model with general uncertainty is:

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Dx \leq b^D, \\ & \overline{Q}_j(x) \leq l_j, \quad j = 1, .., J, \qquad (Generalized\ ARO) \\ & x \in X. \end{array}$$

As an example, when $l = 0$ (i.e., when no violation is allowed) and $\Omega(U)$ is bounded, the model is equivalent to the robust counterpart, see e.g., Ben-Tal and Nemirovski (1998) and Ben-Tal et al. (2009), where the uncertain parameter belonged to the uncertainty set $\Omega(U)$.

Due to the multidimensional integral, which are computationally prohibitive when the dimension exceeds four (see Chen and Sim (2009)), the Generalized ARO (even with continues variables) may be computationally intractable. Therefore, it may be essential to find tractable approximation for the penalty function. For each constraint j, let vector $\overline{U}_j$ denotes the mean of U_j and Σ_j denotes the corresponding covariance matrix. The next lemma provides a distribution-free bound denoted by $B_j(x)$ on the penalty function:

Lemma 1. (*Distribution free bound on the penalty function*)

$$E_f[(U_j x - b_j^U)^+] \leq \tfrac{1}{2}((\overline{U}_j x - b_j^U) + \sqrt{\Sigma_j x x^T + (\overline{U}_j x - b_j^U)^2}) = B_j(x), \quad \forall f \in F. \qquad (21).$$

Proof. It is easy to see:

$$(U_j x - b_j^U)^+ = \frac{|U_j x - b_j^U| + (U_j x - b_j^U)}{2}, \qquad (22).$$

By taking the expectation of both sides of equation (22), and applying the Cauchy-Schwarz inequality to $E_f|U_j x - b_j^U|$, we obtain the bound. □

Using the above bound, we can present the following second-order cone programming approximation of the Generalized ARO model,

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Dx \leq b^D, \\ & B_j(x) \leq l_j, \quad j = 1, .., J, \qquad (\textit{Approximate ARO}) \\ & x \in X. \end{array}$$

It is easy to see that since $\overline{Q}_j(x) \leq B_j(x)$, the optimal solution of the Approximate ARO model is a feasible (and conservative) solution of the Generalized ARO model.

Generalized ARO with discrete variables: If the decision variables of the Generalized ARO model are discrete, the model becomes even more intractable. Fortunately, the decomposition model can be modified to solve such problems. As before, we use the first moment matrix, $\overline{U}$, in the mater problem. Furthermore, the subproblem follows the same steps as in Section 3: (i)In the penalty calculation step of the subproblem, we calculate $\overline{Q}_j(x^t) = E[(U_j x^t - b_j^U)^+]$ (if we have full probabilistic knowledge) or $\overline{Q}_j(x^t) = max_f E_f[(U_j x^t - b_j^U)^+]$ (if we have partial probabilistic knowledge). Note that in circumstances where calculating $\overline{Q}_j(x^t)$ is not computationally feasible, we can instead calculate $B_j(x^t)$. Doing so however, can result in finding more conservative solutions. (ii) In the feasibility check step we examine if any penalty constraints is violated. (iii) Finally, in the cut generation step, the following modified cuts are generated and added to the master problem:

$$\overline{Q}_j(x^t) - ((\overline{U}_j \circ x^{t^T})(I - x \circ x^t) - \overline{U}_j((I - x^t) \circ x)) \leq l_j, \quad \forall j \in V^t. \qquad (23)$$

**Corollary 1. (*Optimality of the decomposition approach with general uncertainty* *If the Generalized* ARO *with discrete variable has an optimal solution, the decomposition approach produces an optimal solution for it in a finite number of iterations.*

7. Conclusions

In this paper, we presented the Almost Robust Optimization model that is simple, intuitive, flexible, and easy to communicate to managers. This model allows uncertainty in the decision making process and bridges stochastic programming and robust optimization. We also developed a practical and computationally tractable decomposition approach to efficiently solve the Almost Robust Optimization model with discrete (binary) decision variables. Furthermore, we defined and studied the Robustness Index that can be used by decision makers to adjust the ARO model to better suit her risk preferences. By using this index we also demonstrated the potential advantages of being almost robust as opposed to being fully robust.

We believe that the approach of Almost Robust Optimization is applicable in many settings and for many problems. We hope that this approach will be used in many future studies. We also intend to establish additional theoretical results on the solvability of ARO problems and the Robustness Index for different families of optimization models.

Appendix A - Proofs

Proof. of Theorem1. As explained in Section 3, conditions 1 and 2 are sufficient to establish Theorem 1. We first prove condition 2, that the cuts (6) are valid. We start by establishing 2(i), i.e., that the cut excludes the current infeasible master problem solution from all subsequent solutions. Suppose the optimal master problem solution is infeasible in iteration t. Then there exist at least one penalty constraint, i, such that:

$$\overline{Q}_i(x^t) > l_i. \quad \text{(A1)}$$

Then, the generated cut for this constraint, from (6), is

$$\overline{Q}_i(x^t) - \sum_{s \in S_i^t} p_s((U_i^s \circ x^{t^T})(I - x \circ x^t) - U_i^s((I - x^t) \circ x)) \leq l_i. \quad \text{(A2)}$$

If the same solution x^t is found in future iterations, $(U_i^s \circ x^{t^T})(I - x^t \circ x^t) - U_i^s((I - x^t) \circ x^t) = 0$, so that the left hand side of (A2) equal to $\overline{Q}_i(x^t)$ that from (A1) is greater than l_i, creating a contradiction. Thus, x^t is infeasible in future iterations of the master problem.

We now prove 2(ii), that is, the proposed cuts do not remove any feasible solution of the ARDO: let x^w be a feasible solution of ARDO found in iteration $w > t$,

$$\overline{Q}_j(x^w) \leq l_j, \quad j = 1, .., J. \quad \text{(A3)}$$

We define $x^{10} = x^t \circ (I - x^w)$, $x^{11} = x^t \circ x^w$, and $x^{01} = (I - x^t) \circ x^w$. Thus, x^{10} is a vector with elements equal to 1 if these elements are 1 in x^t and 0 in x^w, 0 otherwise. Similarly, x^{11} is a vector with elements equal to 1 if these elements are 1 in both x^t and x^w, 0 otherwise. Finally, x^{01} is a vector with elements equal to 1 if these elements are 0 in x^t and 1 in x^w, 0 otherwise.

We present a proof by contradiction. Assume that x^w does not satisfy a cut formed in iteration t for constraint i. Since the elements that are 1 in x^{11} are not included in the cut, they do not contribute to the LHS of (A2). Thus, from the LHS of (A2) when $x = x^w$,

$$\overline{Q}_i(x^t) - \sum_{s \in S_i^t} p_s(U_i^s x^{10} - U_i^s x^{01}) > l_i. \quad \text{(A4)}$$

Furthermore, since $U_i^s x^t = U_i^s x^{10} + U_i^s x^{11}$, $\overline{Q}_i(x^t)$ can be rewritten as,

$$\overline{Q}_i(x^t) = \sum_{s \in S} p_s(U_i^s x^{10} + U_i^s x^{11} - b_i^U)^+ = \sum_{s \in S_i^t} p_s(U_i^s x^{10} + U_i^s x^{11} - b_i^U), \quad \text{(A5)}$$

where the last equality follows by only summing over the scenarios with non-negative penalties, S_i^t. Therefore, by replacing $\overline{Q}_i(x^t)$ in the LHS of (A4) by (A5) we have

$$\sum_{s \in S_i^t} p_s(U_i^s x^{11} + U_i^s x^{01} - b_i^U) > l_i. \quad \text{(A6)}$$

Since $U_i^s x^w - b_i^U = U_i^s x^{11} + U_i^s x^{01} - b_i^U$, thus, $\overline{Q}_i(x^w) = \sum_{s \in S} p_s(U_i^s x^w - b_i^U)^+ = \sum_{s \in S} p_s(U_i^s x^{11} + U_i^s x^{01} - b_i^U)^+$. Therefore,

$$\begin{aligned}
\overline{Q}_i(x^w) &= \sum_{s \in S} p_s(U_i^s x^{11} + U_i^s x^{01} - b_i^U)^+ \\
&= \sum_{s \in S_i^t} p_s(U_i^s x^{11} + U_i^s x^{01} - b_i^U)^+ + \sum_{s \in S - S_i^t} p_s(U_i^s x^{11} + U_i^s x^{01} - b_i^U)^+ \\
&\geq \sum_{s \in S_i^t} p_s(U_i^s x^{11} + U_i^s x^{01} - b_i^U)^+ \\
&\geq \sum_{s \in S_i^t} p_s(U_i^s x^{11} + U_i^s x^{01} - b_i^U) \\
&> l_i. \qquad \text{(A7)}
\end{aligned}$$

where the first inequality is due to $\sum_{s \in S - S_i^t} p_s (U_i^s x^{11} + U_i^s x^{01} - b_i^U)^+ \geq 0$, the second due to $(U_i^s x^{11} + U_i^s x^{01} - b_i^U) \leq max\{0, (U_i^s x^{11} + U_i^s x^{01} - b_i^U)\}$, and the last from (A6). Therefore, from (A7) we have $\overline{Q}_i(x^w) > l_i$ which contradicts (A3) and thus, our assumption that the feasible solution x^w does not satisfy the cut. This establishes that the cut does not remove any feasible solution of ARDO.

Since both conditions 2(i) and 2(ii) are satisfied, we conclude that the cut is valid.

We now prove condition 1, that the optimal objective value of the master problem is a lower bound on that of the ARDO. We first note that because the cuts are valid, they do not remove any feasible solution of the ARDO at any iterations of the decomposition. Moreover, since the objective function and the deterministic constraints of the ARDO also appear in the master problem, we must only take into consideration the penalty constraints of the ARDO, $\overline{Q}_j(x) = \sum_s p_s (U_j^s x - b_j^U)^+ \leq l_j \ \ \forall j$, and the expected-value constraints of the master problem, $\overline{U}_j x \leq l'_j \equiv \sum_s p_s (U_j^s x - b_j^U) \leq l_j \ \ \forall j$. In particular, we must show that any feasible solution of the ARDO is also feasible to the master problem, i.e., if $\overline{Q}_j(x) \leq l_j$ then $\overline{U}_j x \leq l'_j = l_j + b_j^U$. Indeed,

$$\overline{U}_j x - b_j^U = \sum_s p_s (U_j^s x - b_j^U) \leq \sum_s p_s (U_j^s x - b_j^U)^+ = \overline{Q}_j(x), \quad \forall j. \quad \text{(A8)}$$

Thus, we conclude that any feasible solution of ARDO is also feasible for the master problem. Therefore, the optimal objective value of master problem is a lower bound on that of the ARDO.

Since both conditions are satisfied, and since the decision variables have a finite domain, the decomposition model produces an optimal solution to the ARDO in a finite number of iterations. □

Appendix B - Extension of Results to Different Penalty Functions

As discussed in Section 2, based on the available probabilistic information, the almost robust model accommodate various penalty functions. The exposition of the paper focuses on the expected penalty function. In this section, we modify the decomposition approach to capture both the worst-case and the distributionally-robust penalty functions.

Worst-Case Penalty Function: The following modifications in the decomposition approach are required: (i) in the master problem, replace the expected value matrix $\overline{U} = E_s[U^s]$ by a minimum matrix $\overline{U}'$ with elements equal to the minimum of the corresponding element over all scenarios; (ii) in Step 1 of the subproblem calculate the maximum penalty ($\overline{Q}_j(x^t) = \max_s Q_j^s$) instead of the expected penalty; (iii) in Step 2 of the subproblem let s_j^t denote the scenario with the maximum penalty; and (iv) in Step 3 use the modified cut:

$$\overline{Q}_j(x^t) - (U_j^{s_j^t} \circ x^{t^T})(I - x \circ x^t) - U_j^{s_j^t}((I - x^t) \circ x) \leq l_j, \quad \forall j \in V^t. \quad (A16)$$

Corollary 2. (*Convergence of the decomposition approach with the worst-case penalty function*) *Cuts of the form (A16) are valid, and the modified decomposition approach produces an optimal solution to the* A*RDO with the worst-case penalty function in a finite number of iterations.*

The proof of Theorem 1 directly applies here, we thus omit the proof of Corollary 2.

Distributionally-Robust Penalty Function: Recall that this penalty function is applicable when the decision maker has limited probabilistic information. Specifically, as is common in the literature (see Scarf (1958), Bertsimas and Popescu (2005), Yue et al. (2006), Popescu (2007)), we assume that the decision maker knows that the probability distribution of the scenarios belongs to a known set of probability distributions, F, and has an estimate of the first moment of the uncertain matrix, $\overline{U}$.

The following modifications in the decomposition approach are required: (i) as in the decomposition approach use the first moment matrix, $\overline{U}$, in the mater problem (if no information on first moment is available, use $\overline{U}'$, as described for the worst-case penalty function); (ii) in Step 1 of the subproblem, for each uncertain constraint $j = 1, .., J$ calculate the expected penalty under distribution f, $\overline{Q}_j^f(x^t) \quad j = 1, .., J, \ \forall f \in F$, determine the distribution with the maximum expected penalty $f_j^t = argmax_{f \in F} \overline{Q}_j^f(x^t)$ and let $\overline{Q}_j = \overline{Q}_j^{f_j^t}(x^t)$; (iii) in Step 3 use the modified cuts:

$$\overline{Q}_j(x^t) - \sum_{s \in S_j^t} p_s^{f_j^t}[(U_j^s \circ x^{t^T})(I - x \circ x^t) - U_j^s((I - x^t) \circ x)] \leq l_j, \quad \forall j \in V^t. \quad (A17)$$

Corollary 3. (*Convergence of the decomposition approach with the distributionally-robust penalty function*) *Cuts of the form (A17) are valid, and the modified decomposition approach produces an optimal solution to the* ARDO *with the distributionally-robust penalty function in a finite number of iterations.*

Proof. The proof of condition 2(i) is similar to that in Theorem 1 and is omitted. Similar to Theorem 1, we use contradiction to prove condition 2(ii): we assume x^w is a feasible solution of the distributionally-robust model, i.e.,

$$max_{f\in F}\overline{Q}_j^f(x^w) \leq l_j, \quad j \in 1,..,J, \text{ (A18)},$$

that does not satisfy the cut (A17) of constraint i. Therefore, given $\overline{Q}_i(x^t) = \overline{Q}_i^{f_i^t}(x^t)$ the cut when $x = x^w$ is

$$\overline{Q}_i^{f_i^t}(x^t) - \sum_{s\in S_i^t} p_s^{f_i^t}[U_i^s x^{10} - U_i^s x^{01}] > l_i. \quad \text{(A19)}$$

Since $\overline{Q}_i^{f_i^t}(x^t) = \sum_{s\in S_i^t} p_s^{f_i^t}[U_i^s x^{10} + U_i^s x^{11} - b_i^U]$ (as in Theorem 1 we removed superscript $^+$ by summing over S_i^t), (A19) is reduced to

$$\sum_{s\in S_i^t} p_s^{f_i^t}[U_i^s x^{11} + U_i^s x^{01} - b_i^U] > l_i. \quad \text{(A20)}$$

We now have two possible outcomes:

- $f_i^w = argmax_{f\in F}\overline{Q}_i^f(x^w) \equiv f_i^t$, i.e., the distribution in iteration w with the maximum penalty is the same as that of iteration t. Under this condition, $max_{f\in F}\overline{Q}_i^f(x^w) = \sum_{s\in S} p_s^{f_i^w}[U_i^s x^{11} + U_i^s x^{01} - b_i^U]^+ = \sum_{s\in S} p_s^{f_i^t}[U_i^s x^{11} + U_i^s x^{01} - b_i^U]^+$. From this and following the argument in (A7) we get a contradiction, thus under this condition, the cut is valid.

- $f_i^w \neq f_i^t$. In this case:

$$max_{f\in F}\overline{Q}_i^f(x^w) = \sum_{s\in S} p_s^{f_i^w}[U_i^s x^{11} + U_i^s x^{01} - b_i^U]^+ \geq \sum_{s\in S} p_s^{f_i^t}[U_i^s x^{11} + U_i^s x^{01} - b_i^U]^+. \quad \text{(A21)}$$

where the inequality is due to that f_i^w has the maximum expected penalty in iteration w. From (A21) and following the argument in (A7) we again get a contradiction, thus, under this condition, the cut is valid.

From the above two conditions we conclude that the cut is always valid. Furthermore,

$$E_f[U_j^s x - b_j^U] = \overline{U}_j x - b_j^U \leq E_f[U_j^s x - b_j^U]^+, \ \forall f \in F, \ j = 1, .., J. \quad \text{(A22)}$$

From (A22) and using the same argument as the proof of condition 1 of Theorem 1, we can conclude that the master problem solution is a lower bound on that of the distributionally-robust model. □

Appendix C - Absolute Value of Infeasibility Measure

An alternative measure of uncertainty is the absolute value of infeasibility, $|U_j^s x - b_j^U|$, which is suitable for equality constraints where neither upward nor downward infeasibility is desired. For such measure, we consider the expected penalty function $\overline{Q}_j(x) = \sum_{s \in S} p_s |U_j^s x - b_j^U|$. Note that the max or the distributionally-robust penalty functions introduced in Section 2 could also be used if full probabilistic information is not available.

The decomposition approach now requires two major modification. First, in step 1 of the subproblem, we calculate the absolute infeasibility, $Q_j^s(x^t) = |U_j^s x^t - b_j^U|, \ j = 1, .., J$. Second, in step 3, the cuts are modified to take into account both downward and upward deviations. In order to introduce the modified cuts, let $n^{s,t} = 0$ if $U_j^s x - b_j^U > 0$ and $n^{s,t} = 1$ if $U_j^s x - b_j^U < 0$. The cuts are:

$$\overline{Q}_j(x^t) - \sum_{s \in S_j^t} p_s (-1)^{n^{s,t}} ((U_j^s \circ x^{t^T})(I - x \circ x^t) - U_j^s((I - x^t) \circ x)) \leq l_j, \quad \forall j \in V^t. \quad (A9)$$

As can be seen, the cuts are identical to the cuts presented in (6) with an addition term $(-1)^{n^{s,t}}$. $n^{s,t}$ is defined such that in scenarios with downward deviation ($n^{s,t} = 1$), removing an element with a positive (negative) coefficients from x^t increases (decreases) the penalty of the scenario (reflected by $U_j^s((I - x^t) \circ x)$), while adding an element with negative (positive) coefficient decreases (increases) the penalty (enforced through $(U_j^s \circ x^{t^T})(I - x \circ x^t)$). For scenarios with upward deviation the effects are reversed.

Corollary 4. (***Convergence of the decomposition approach with the absolute value measure***) *Cuts of the form (A9) are valid, and the modified decomposition approach produces an optimal solution to the* A*RDO with the absolute value measure in a finite number of iterations.*

Proof. Since the proof of condition 1 and condition 2(i) (see Section 3) is similar to the proof presented for Theorem 1, we omit it.

We again use contradiction to prove condition 2(ii), thus assume x^w is a feasible solution of the *A*RDO with the absolute value measure, i.e.,

$$\overline{Q}_j(x^w) \leq l_j \quad j \in 1, .., J, \text{ (A10)}$$

and it does not satisfy the cut of constraint i (of the form (A9)). Let x^{10}, x^{11}, and x^{01} be as defined in Appendix A. From (A9) when $x = x^w$:

$$\overline{Q}_i(x^t) - \sum_{s \in S_i^t} p_s(-1)^{n^{s,t}}(U_i^s x^{10} - U_i^s x^{01}) > l_i. \text{ (A11)}$$

Furthermore, since $U_i^s x^t = U_i^s x^{10} + U_i^s x^{11}$ and $\overline{Q}_i(x^t) = \sum_{s \in S_i^t} p_s |U_i^s x^t - b_i^U|$, (A11) can be rewritten as,

$$\sum_{s \in S_i^t} p_s(|U_i^s x^{10} + U_i^s x^{11} - b_i^U| - (-1)^{n^{s,t}}(U_i^s x^{10} - U_i^s x^{01})) > l_i. \text{ (A12)}$$

We define $S_i^{t+} = \{s \in S_i^t | n^{s,t} = 0\}$ to be the set of scenarios with upward deviation, i.e., $U_i^s x^t - b_i^U > 0 \quad \forall s \in S_i^{t+}$, and $S_i^{t-} = \{s \in S_i^t | n^{s,t} = 1\}$ to be the set of scenarios with downward deviation, i.e., $U_i^s x^t - b_i^U < 0 \quad \forall s \in S_i^{t-}$. Since $U_i^s x^w = U_i^s x^{11} + U_i^s x^{01}$, thus $\overline{Q}_i(x^w) = \sum_{s \in S} p_s |U_i^s x^w - b_i^U| = \sum_{s \in S} p_s |U_i^s x^{11} + U_i^s x^{01} - b_i^U|$. Therefore,

$$\begin{aligned}
\overline{Q}_i(x^w) &= \sum_{s \in S} p_s |U_i^s x^{11} + U_i^s x^{01} - b_i^U| \\
&= \sum_{s \in S_i^t} p_s |U_i^s x^{11} + U_i^s x^{01} - b_i^U| + \sum_{s \in S - S_i^t} p_s |U_i^s x^{11} + U_i^s x^{01} - b_i^U| \\
&\geq \sum_{s \in S_i^t} p_s |U_i^s x^{11} + U_i^s x^{01} - b_i^U| \\
&= \sum_{s \in S_i^{t+}} p_s |U_i^s x^{11} + U_i^s x^{01} - b_i^U| + \sum_{s \in S_i^{t-}} p_s |U_i^s x^{11} + U_i^s x^{01} - b_i^U| \\
&\geq \sum_{s \in S_i^{t+}} p_s (U_i^s x^{11} + U_i^s x^{01} - b_i^U) + \sum_{s \in S_i^{t-}} p_s (-(U_i^s x^{11} + U_i^s x^{01} - b_i^U)) \\
&= \sum_{s \in S_i^t} p_s (|U_i^s x^{10} + U_i^s x^{11} - b_i^U| - (-1)^{n^{s,t}}(U_i^s x^{10} - U_i^s x^{01}))
\end{aligned}$$

$$> l_i. \tag{A13}$$

where the first inequality is due to $\sum_{s \in S - S_i^t} p_s |U_i^s x^{11} + U_i^s x^{01} - b_i^U| \geq 0$, the second due to $|U_i^s x^{11} + U_i^s x^{01} - b_i^U| \geq (-(U_i^s x^{11} + U_i^s x^{01} - b_i^U))$, the last equality is by the definition of $n^{s,t}$ and S_i^{t+} and S_i^{t-}, and the last inequality is from (A12).

From (A13) we have $\overline{Q}_i(x^w) > l_i$, which contradicts (A10). Therefore, the cut does not remove any feasible solution of ARDO with the absolute value measure (condition 2(ii)). □

References

Atamturk, A. 2006. Strong formulation of robust mixed 0-1 programming. *Mathematical programming* **108** 235–250.

Atamturk, A., M. Zhang. 2007. Two-stage robust netwrok flow and design under uncertainty. *Operations Research* **55, No. 4** 662–673.

Averbakh, I. 2001. On the complexity of a class of combinatorial optimization problems with uncertainty. *Mathematical Programming* **90** 263–272.

Baron, O., J. Milner, H. Naseraldin. 2011. Facility location: A robust optimization approach. *Production and Operations Management* **20**(5) 772–785.

Ben-Tal, A., D. Bertsimas, D. B. Brown. 2010. A soft robust model for optimization under ambiguity. *Operations research* **58**(4-part-2) 1220–1234.

Ben-Tal, A., S. Boyd, A. Nemirovski. 2006. Extending scope of robust optimization: Comprehensive robust counterparts of uncertain problems. *Mathematical Programming* **107**(1-2) 63–89.

Ben-Tal, A., L. El Ghaoui, A. Nemirovski. 2009. *Robust Optimization*. Princeton University Press.

Ben-Tal, A., A. Nemirovski. 1998. Robust convex optimization. *Mathematics of operations research* **23(4)** 769–805.

Ben-Tal, A., A. Nemirovski. 1999. Robust solutions to uncertain programs. *Operations Research Letters* **25** 1–13.

Ben-Tal, A., A. Nemirovski. 2000. Robust solutions of linear programming problems contaminated with uncertain data. *Mathematical Programming* **88(3)** 411–424.

Ben-Tal, A., A. Nemirovski. 2001. Robust optimization-methodology and applications. *Mathematical programming* **92(3)** 453–480.

Benders, J. 1962. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik* **4**(1) 238–252.

Bertsimas, D., I. Popescu. 2005. Optimal inequalities in probability theory: A convex optimization approach. *SIAM Journal on Optimization* **15**(3) 780–804.

Bertsimas, D., M. Sim. 2003. Robust discerete optimization and network flow. *Mathematical programming* **98** 49–71.

Bertsimas, D., M. Sim. 2004. The price of robustness. *Operations Research* **52(1)** 35–53.

Bertsimas, D., R. Weismantel. 2005. *Optimization over integers*. Athena Scientific.

Blankenship, J. W., J.E. Falk. 1976. Infinitely constrained optimization problems. *Journal of Optimization Theory and Applications* **19**(2) 261–281.

Charnes, A., W.W. Cooper. 1959. Chance-constrained programming. *Management Science* **5** 73–79.

Chen, W., M. Sim. 2009. Goal-driven optimization. *Operations Research* **57(2)** 342–357.

Chen, X., M. Sim, P. Sun. 2007. A robust optimization perspective to stochastic programming. *Operations Research* **55(6)** 1058–1071.

El-Ghaoui, L., F. Oustry, H. Lebret. 1998. Robust solutions to uncertain semidefinite programs. *SIAM Journal on Optimization* **9** 33–52.

Fazel-Zarandi, M. M., O. Berman, J.C. Beck. 2012. Solving a stochastic facility/location fleet management problem with logic-based benders de. *IIE Transactions, Forthcoming* .

Geoffrion, A. M., G. W. Graves. 1974. Multicommodity distribution system design by Benders decomposition. *Management Science* **20**(5) 822–844.

Hooker, J. N., G. Ottosson. 2003. Logic-based Benders decomposition. *Mathematical Programming* **96** 33–60.

Klein Haneveld, W.K. 1986. On integrated chance constraints. *Duality in Stochastic Linear and Dynamic Programming*. Springer, 113–138.

Kouvelis, P., G. Yu. 1997. *Robust discerete optimization and its appliastions*. Kluwer Academic PublishKlu, Norwell, MA.

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

Lenstra, J. K., A. H. G. Rinnooy Kan. 1981. Complexity of vehicle routing and scheduling problems. *Networks* **11**(2) 221–227.

Mirchandani, P.B., R.L. Francis. 1990. *Discrete Location Theory*. Wiley-New York.

Mulvey, J.M., Vanderbei, S.A. R.J., Zenios. 1995. Robust optimization of large-scale systems. *Operations Research* **43** 264–281.

Natarajan, K., D. Pachamanova, M. Sim. 2008. Incorporating asymmetric distributional information in robust value-at-risk optimization. *Management Science* **54**(3) 573–585.

Natarajan, K., D. Pachamanova, M. Sim. 2009. Constructing risk measures from uncertainty sets. *Operations Research* **57**(5) 1129–1141.

Nemhauser, G.L., L.A. Wolsey. 1988. *Integer and combinatorial optimization*. Wiley-Interscience, New York, NY, USA.

Papadimitriou, C.H. 2003. Computational complexity. *Encyclopedia of Computer Science*. John Wiley and Sons Ltd.

Popescu, I. 2007. Robust mean-covariance solutions for stochastic optimization. *Operations Research* **55**(1) 98–112.

Ruszczynski, A., A. Shapiro, eds. 2003. *Stochastic Prog*. Handbooks in Operations Research and Management Science, Vol. 10. Elsevier, Amestredam.

Scarf, H. 1958. A min-max solution of an inventory problem. *Studies in The Mathematical Theory of Inventory and Production* 2001–2009.

Soyster, A. L. 1973. Convex programming with set-inclusive constraints and applications to inexact linear programming. *Operations Research* **21** 1154–1157.

Wolsey, L. A. 1998. *Integer programming*. Wiley-Interscience, New York, NY, USA.

Yue, J., B. Chen, M.-C. Wang. 2006. Expected value of distribution information for the newsvendor problem. *Operations Research* **54**(5) 1128–1136.