A DYNAMIC PROGRAMMING APPROACH FOR A CLASS OF ROBUST OPTIMIZATION PROBLEMS*

MARCIO COSTA SANTOS†, AGOSTINHO AGRA‡, MICHAEL POSS§, AND DRITAN NACE¶

Abstract. Common approaches to solve a robust optimization problem decompose the problem into a master problem (MP) and adversarial separation problems (APs). MP contains the original robust constraints, however written only for finite numbers of scenarios. Additional scenarios are generated on the fly by solving the APs. We consider in this work the budgeted uncertainty polytope from Bertsimas and Sim, widely used in the literature, and propose new dynamic programming algorithms to solve the APs that are based on the maximum number of deviations allowed and on the size of the deviations. Our algorithms can be applied to robust constraints that occur in various applications such as lot-sizing, TSP with time-windows, scheduling problems, and inventory routing problems, among many others. We show how the simple version of the algorithms leads to a FPTAS when the deterministic problem is convex. We assess numerically our approach on a lot-sizing problem, showing a comparison with the classical MIP reformulation of the AP.

Key words. Robust optimization, Budgeted uncertainty, Dynamic programming, Row-and-column generation, FPTAS.

AMS subject classifications.

1. Introduction. Robust optimization (RO) is a popular framework to handle the uncertainty that arises in optimization problems. The essence of RO lies in ensuring that feasible solutions satisfy the robust constraints for all parameter realizations in a given uncertainty set Ξ. Since the seminal work of [11], the framework has been used in numerous applications, see [9, 12, 21] and the references therein. The success of robust optimization can be explained mainly by the following reasons. First, it is simple to use and understand since it only requires the knowledge of uncertainty sets. Second, RO very often yields optimization problems that are not more difficult to solve than the original problems.

The classical approach to RO trades the robust constraints for convex reformulations. The initial works were based on conic duality (e.g. [9]) but recent works have extended the scope of convex reformulations by using other tools, such as Fenchel duality [8] or the result "primal worst equals dual best" [22]. In this work, we consider an alternative approach, based on decomposition algorithms. Our work falls into the recent trend that solves complex RO problems without reformulating the robust constraints as convex ones. Instead, we relax the problem into a so-called Master Problem, where each robust constraint is written only for a finite subset $\Xi^0 \subset \Xi$. Given a feasible solution to the master problem, we check whether the solution is

*This work was supported by the French ministries MAE and MENESR. Agostinho Agra was partially funded by FCT (Fundação para a Ciência e a Tecnologia) through CIDMA, UID/MAT/04106/2013, and through program COMPETE: FCOMP-01-0124-FEDER-041898 within project EXPL/MAT-NAN/1761/2013.

†UMR CNRS 7253 Heudiasyc, Université de Technologie de Compiègne, Centre de Recherches de Royallieu, 60200 Compiègne, France. (marcio.costa-santos@hds.utc.fr).

‡CIDMA, Department of Mathematics, University of Aveiro, 3810-193 Aveiro, Portugal. (aagra@ua.pt).

§UMR CNRS 5506 LIRMM, Université de Montpellier 2, 161 rue Ada, 34392 Montpellier Cedex 5, France. (michael.poss@lirmm.fr).

¶UMR CNRS 7253 Heudiasyc, Université de Technologie de Compiègne, Centre de Recherches de Royallieu, 60200 Compiègne, France. (dritan.nace@hds.utc.fr).

feasible for each robust constraint by solving adversarial separation problems (APs). In case one or more robust constraints are infeasible, we expand Ξ^0 by one or more vectors and solve the augmented Master Problem. The main advantage of decomposition algorithms over convex reformulations is that they can be applied to virtually any type of robust constraint as long as one can handle the corresponding AP. For instance, they can be applied to robust constraints defined by non-concave functions of the uncertain parameters [30], to uncertainty sets that are defined through integrality requirements [20], or to solve exactly two-stage RO problems [35].

We are proposing in this paper a dynamic programming algorithm (DPA) and a unified decomposition algorithm to solve the following optimization problem. Let d be a vector of parameters, z be a vector of optimization variables, g_i and h_i be affine functions for each $i = 1, \ldots, n$, and consider the robust constraint

$$f(\xi^\Sigma, z) \leq d^T z, \quad \forall \xi \in \Xi, \tag{1.1}$$

with

$$f(\xi^\Sigma, z) = \sum_{i=1}^{n} \max\{g_i(\xi_i^\Sigma, z), h_i(\xi_i^\Sigma, z)\}. \tag{1.2}$$

and where $\xi_i^\Sigma = \sum_{j=1}^{i} \xi_j$ denotes the sum of the first i components of uncertain vector ξ for $i = 1, \ldots, n$. Consider a cost vector c and a feasibility set $\mathcal{Z}$. We are interested in solving exactly the following type of robust optimization problems:

$$\begin{aligned} \min \quad & c^T z \\ \text{s.t.} \quad & z \in \mathcal{Z}, \qquad (1.3) \\ & f(\xi^\Sigma, z) \leq d^T z, \quad \forall \xi \in \Xi, \qquad (1.4) \end{aligned}$$

where (1.3) represents all constraints not affected by the uncertainty, including the possible integrality restrictions.

Definition (1.2) has its roots in lot sizing problems where, for each period in a given time horizon, one has to pay either storage or backlog cost (cost incurred by delaying the delivery of the product). It is common to assume demand uncertainty in these problems so that the total demand that must be handled at period i is equal to $\sum_{j=1}^{i} \xi_j$. For the sake of clarity, we delay to a later section of the paper an important extension of our approach where the elements of the summations $\sum_{j=1}^{i} \xi_j$ involved in (1.2) depend on the optimization variables z.

Our algorithm further requires that the uncertainty set be the budgeted uncertainty polytope Ξ^Γ introduced by [13, 14] and defined as

$$\Xi^\Gamma \equiv \left\{ \xi : \xi_i = \bar{\xi}_i + \widehat{\xi}_i \eta_i, i = 1, \ldots, n, \|\eta\|_\infty \leq 1, \|\eta\|_1 \leq \Gamma \right\}, \tag{1.5}$$

for positive integers $\widehat{\xi}_i$, $i = 1, \ldots, n$, arbitrary reals $\bar{\xi}_i$, $i = 1, \ldots, n$, and $\Gamma > 0$. The set is extremely popular in integer programming and network optimization and has been used in a wide range of applications. The main purpose of the paper is to study how to efficiently solve the separation problem for constraint (1.4) and uncertainty set Ξ^Γ, namely

$$(\mathbf{AP}) \qquad \max_{\xi \in \Xi^\Gamma} f(\xi^\Sigma, z).$$

1.1. Contributions and literature review. Traditionally, (**AP**) is either solved through a DPA based on the value of Γ and on the stock levels [15], a MILP with big-M coefficients [15], or approximated via decision rules (e.g. [10]). More recently, [23] proposed to solve an adversarial separation problem different from (**AP**), which is related to the expansion of the maxima in the definition of f. Differently from these works, we propose here to address (**AP**) via a DPA based on Γ and on $\widehat{\xi}$. Hence, our approach can be applied to a wider range of problem than the seminal work of [15] that focused on the lot-sizing problem. The worst-case running time of our approach depends on the value of $\widehat{\xi}$ and Γ, yielding a pseudo-polynomial time algorithm. When the deviations are small, our numerical experiments show that the DPA can be orders of magnitude faster than the classical MILP reformulation. Moreover, we show that our DPA gives rise to a FPTAS for (**AP**) and the original robust problem whenever $\mathcal{Z}$ is a convex set and an additional technical assumption is satisfied. We also extend our DPA to combinatorial problems with lower time windows and inventory distribution problems. Notice also that, unlike [23] that considers bi-affine functions g_i and h_i, we consider these functions affine herein.

We mention that dynamic programming has already been successfully applied to other RO problems with uncertainty set Ξ^{Γ}. One of the first works in that aspect is [28] which presents a DPA to solve the robust knapsack problem. The approach is compared numerically to other solution algorithms in [29] where the authors also study the space complexity of the algorithm. The seminal idea of [28] is extended by [32] to any robust combinatorial optimization problems with cost uncertainty whose deterministic counterpart can be solved by a DPA. Differently from [28, 29, 32] which solve the full RO problem by a DPA, the authors of [3] use a DPA to solve the adversarial problem that arises in robust vehicle routing problems with time windows. The complexity of their algorithm is linear in Γ. A common characteristic of the aforementioned works is that the deterministic version of the problem studied therein (or the AP in case of [3]) can be solved by a DPA. Thus, these works show how to extend the deterministic DPA to handle the robust versions of the problems. Fortunately, the extension only multiplies by Γ the number of states used in the deterministic DPA, so that the robust problems pertain to the same complexity class as their deterministic counterparts. In contrast, the authors of [31] prove that the robust shortest path problem with time windows is $\mathcal{NP}$-hard in the strong sense while its deterministic version is $\mathcal{NP}$-hard in the weak sense. They propose a DPA for the shortest path problem with time windows which multiplies the deterministic number of states by an exponential function of Γ, yielding an algorithm with exponential running-time.

The algorithm presented herein is different from the above papers on mainly two aspects. First, like [15], it does not extend an existing deterministic algorithm because solving (**AP**) is trivial in the deterministic context. Second, the number of states of our algorithm depends on Γ and $\widehat{\xi}$, unlike previous works which only involve Γ. Hence, when each component of $\widehat{\xi}$ is bounded by a polynomial function of the input data, our DPA runs in polynomial time. Otherwise, its running-time is pseudo-polynomial.

1.2. Structure of the paper. In the next section, we present the general algorithmic procedure used to decompose the original problem into a master problem and an adversarial separation problem. Then, we discuss the computational complexity of the adversarial separation problem, showing that it is a special case of different $\mathcal{NP}$-hard optimization problems. In Section 3.1, we present a simple DPA for (**AP**) defined previously, which we extend in Section 3.2 to a FPTAS. We explain in Section 4 how to generalize the framework to encompass more complex problems such

as the robust traveling salesman with deadlines. We propose in Section 5 subtle extensions for our simple DPA, each one described for a particular problem. Namely, we present in Section 5.1 an inventory distribution problem where the adversarial problem can be decomposed into smaller problems linked by a simple constraint, and illustrate in Section 5.2 on the traveling-salesman problem how to handle full time windows rather than deadlines constraints. Our numerical experiments, realized on the lot-sizing problem, are presented in Section 6. Finally, the appendix contains the $\mathcal{NP}$-hardness proofs.

2. The framework. We describe in this section the general decomposition algorithm used in this paper. Then, we discuss the complexity of the adversarial problem.

2.1. Decomposition. We are interested in solving exactly the following type of robust optimization problems:

$$\begin{aligned} & & \min \quad & c^T z & \\ (\mathbf{P}) \qquad & & \text{s.t.} \quad & z \in \mathcal{Z}, & (2.1)\\ & & & f(\xi^\Sigma, z) \leq d^T z, \quad \forall \xi \in \Xi^\Gamma. & (2.2) \end{aligned}$$

Constraint (2.1) contains all restrictions not affected by uncertainty, including the possible integrality restrictions on z. Constraint (2.2) is a robust constraint characterized by a function that satisfies (1.2). For the sake of simplicity, we consider a unique robust constraint in (**P**), one readily extends the approach described below to problems with K constraints of type (2.2).

The problem (**P**) contains infinite numbers of variables and constraints, making it intractable as such. Here, we tackle the problem by generating a finite subset of variables and constraints on the fly in the course of the decomposition algorithm presented below. Let $\Xi^0 \subset \Xi^\Gamma$ be a finite set. Since Ξ^0 is finite, we can reformulate

$$f(\xi^\Sigma, z) \leq d^T z, \quad \forall \xi \in \Xi^0, \tag{2.3}$$

as the following finite set of linear inequalities, written for each $\xi \in \Xi^0$:

$$\sum_{i=1}^{n} \varphi_i^\xi \leq d^T z, \tag{2.4}$$

$$\varphi_i^\xi \geq g_i(\xi_i^\Sigma, z), \quad \forall i = 1, \ldots, n, \tag{2.5}$$

$$\varphi_i^\xi \geq h_i(\xi_i^\Sigma, z), \quad \forall i = 1, \ldots, n, \tag{2.6}$$

where φ^ξ is an additional vector of optimization variables. Our approach is based on the above linear reformulation of (2.3). Specifically, we relax constraints (2.2) for all elements in Ξ^Γ but Ξ^0 and replace robust constraint (2.2) by its linear reformulation, obtaining the Master Problem

$$\begin{aligned} & \min \quad & c^T z & \\ (\mathbf{MP}) \qquad & \text{s.t.} \quad & z \in \mathcal{Z}, & \\ & & \sum_{i=1}^{n} \varphi_i^\xi \leq d^T z, \quad \forall \xi \in \Xi^0, & \\ & & \varphi_i^\xi \geq g_i(\xi_i^\Sigma, z), \quad \forall \xi \in \Xi^0, i = 1, \ldots, n, & \\ & & \varphi_i^\xi \geq h_i(\xi_i^\Sigma, z), \quad \forall \xi \in \Xi^0, i = 1, \ldots, n. & \end{aligned}$$

Given a feasible solution z^* to (**MP**), one checks the feasibility of z^* for (**P**) by solving an adversarial problem. Let ξ^* be the optimal solution for the adversarial problem. If $f(\xi^{*\Sigma}, z^*) > d^T z^*$, then $\Xi^0 \leftarrow \Xi^0 \cup \{\xi^*\}$, and the corresponding optimization vector φ^{ξ^*} and constraints (2.4)–(2.6) are added to (**MP**). Therefore, the overall algorithm is a row-and-column generation algorithm in the line of those proposed in [3, 35].

2.2. Illustration: Robust Lot-Sizing Problem. Consider the Robust Lot-Sizing Problem (RLSP) defined for a finite planning horizon $\{1, \ldots, n\}$. For each time period $i = 1, \ldots, n$, we are given a capacity C_i, holding cost p_i, a shortage cost s_i, and a production cost c_i. We assume that the vector of demands ξ belongs to set Ξ^Γ, where $\overline{\xi}_i$ represents the nominal demand in period i, and $\widehat{\xi}_i$ represents the maximum allowed demand deviation in period i. The amount that needs to be produced in each time period must be decided before the actual demand value is revealed. In contrast, stock levels and backlogged demands are adjusted to each individual demand realization.

The problem can be modeled as follows. Variable x_i represents the amount produced at period i and variable θ represents the total storage and backlog costs. For each $\xi \in \Xi$, ξ_i^Σ represents the total demand up to time period i for demand vector ξ. The formulation for the RLSP follows.

$$\begin{aligned} & \min \quad c^T x + \theta \\ (2.7) \quad & \text{s.t} \quad 0 \le x_i \le C_i, \quad \forall i = 1, \ldots, n, \\ (2.8) \quad & \theta \ge \sum_{i=1}^{n} \max \left\{ s_i \left(\xi_i^\Sigma - \sum_{j=1}^{i} x_j \right), -p_i \left(\xi_i^\Sigma - \sum_{j=1}^{i} x_j \right) \right\}, \quad \forall \xi \in \Xi^\Gamma. \end{aligned}$$

We see readily that the problem is a special case of (**P**) for $z = (x, \theta)$. Namely, $\mathcal{Z}$ is the set defined by (2.7), the components of d corresponding to x and θ are equal to 0 and 1, respectively, and functions f, g and h are defined by $f(\xi^\Sigma, x, \theta) = \sum_{i=1}^{n} \max \left\{ s_i \left(\xi_i^\Sigma - \sum_{j=1}^{i} x_j \right), -p_i \left(\xi_i^\Sigma - \sum_{j=1}^{i} x_j \right) \right\}$, $g_i(\xi^\Sigma, x, \theta) = s_i(\xi_i^\Sigma - \sum_{j=1}^{i} x_j)$ and $h_i(\xi^\Sigma, x, \theta) = -p_i(\xi_i^\Sigma - \sum_{j=1}^{i} x_j)$.

Observe that the adversarial problem depends only on the quantities produced. Hence the approach holds if more complex models are considered for the decision problem, such as set-up costs, start-up costs, etc.

2.3. Complexity of the adversarial problem. Omitting the dependency on z, the adversarial problem $\max_{\xi \in \Xi^\Gamma} f(\xi^\Sigma)$ considered in this paper optimizes a function of the form $f(\xi^\Sigma) = \sum_{i=1}^{n} f_i(\xi_i^\Sigma)$ where f_i is a convex function for each $i = 1, \ldots, n$. Hence, f is convex so that its maximum is attained at least at one of the extreme points of polytope Ξ^Γ, which we denote $\text{ext}(\Xi^\Gamma)$:

$$\text{ext}(\Xi^\Gamma) \equiv \left\{ \xi : \xi_i = \overline{\xi}_i + \widehat{\xi}_i \eta_i, i = 1, \ldots, n, \eta \in \{-1, 0, 1\}^n, \|\eta\|_1 \le \Gamma \right\}.$$

There exist a few cases in which problem (**AP**) is easy. For instance, when Γ is constant (i.e. it is not part of the input), $\text{ext}(\Xi^\Gamma)$ contains a polynomial number of elements so that (**AP**) can be solved in polynomial time by inspection. More interestingly, the problem can also be solved in polynomial time whenever $\Gamma = n$, which corresponds to Ξ^Γ being a box uncertainty set, see [26]. More recently, the authors of [6] have shown that a closely related problem where each function f_i involves only component ξ_i of the uncertain vector is also easy.

While the $\mathcal{NP}$-hardness of (**AP**) is still unknown, simple generalizations of the problem are difficult. For instance, optimizing a general convex function over $\text{ext}(\Xi^\Gamma)$ is $\mathcal{APX}$-hard, see Proposition A.1 in the Appendix. We study next problems more closely related to (**AP**). We show that if we generalize either the uncertainty set Ξ^Γ or the set of admissible functions f_i, the resulting problem is $\mathcal{NP}$-hard. Namely, consider the following generalization of problem (**AP**):

$$(\widetilde{\mathbf{AP}}) \qquad \max_{\xi \in \widetilde{\Xi}} \sum_{i=1}^{n} \tilde{f}_i(\xi_i^\Sigma).$$

- If $\widetilde{\Xi}$ is a polytope having a compact description and functions $\tilde{f}_i$ are convex functions of the form $\tilde{f}_i(\xi_i^\Sigma) = \max\{g_i(\xi_i^\Sigma), h_i(\xi_i^\Sigma)\}$ where g_i and h_i are affine functions, then $(\widetilde{\mathbf{AP}})$ is $\mathcal{NP}$-hard. This result is stated in Proposition A.2 given in the Appendix.
- If $\widetilde{\Xi}$ is the set $\text{ext}(\Xi^\Gamma)$ and $\tilde{f}_i$ are general non-negative functions such that $\tilde{f}(\xi^\Sigma) = \sum_{i=1}^{n} \tilde{f}_i(\xi_i^\Sigma)$ is positive, then $(\widetilde{\mathbf{AP}})$ is $\mathcal{NP}$-hard. This result is stated in Proposition A.3 given in the Appendix.

In view of the above results and of the numerical difficulty of solving problem (**AP**), our conjecture is that the problem is $\mathcal{NP}$-hard.

On the other hand, if the deviations are constant or if their size can be bounded by a polynomial function of the input data then (**AP**) can be solved in polynomial time. This is an immediate consequence of the DPA presented in the next section.

3. A dynamic programming algorithm. We present in Section 3.1 a simple DPA to solve the adversarial problem for a simplification of Ξ^Γ. We show then in Section 3.2 how to approximate (**AP**) and (**P**) through a FPTAS when some additional assumptions are satisfied. We show how to extend the DPA to the general set Ξ^Γ and to larger classes of functions in Section 3.3 and in Section 3.4, respectively.

3.1. Exact algorithm. For the sake of simplicity, we consider in what follows that Γ is integer and that Ξ does not include downward deviations ($\eta \in \{0,1\}^n$). We discuss in Section 3.3 how these simplifications can be relaxed. We further assume that z is fixed throughout the section so that, for each $i = 1, \ldots, n$, we can simplify the writing of affine functions $g_i(\xi_i^\Sigma, z)$ and $h_i(\xi_i^\Sigma, z)$ by removing the explicit dependency on z. We obtain affine functions

$$g_i(\xi_i^\Sigma) = g_i^0 + g_i^1 \xi_i^\Sigma \quad \text{and} \quad h_i(\xi_i^\Sigma) = h_i^0 + h_i^1 \xi_i^\Sigma,$$

where the dependency on z is hidden in the independent terms g_i^0 and h_i^0. Then, we define $f_i(\xi_i^\Sigma) = \max\{g_i(\xi_i^\Sigma), h_i(\xi_i^\Sigma)\}$, for each $i = 1, \ldots, n$ and $f(\xi^\Sigma) = \sum_{i=1}^{n} f_i(\xi_i^\Sigma)$.

We are interested here in solving the optimization problem $\max_{\xi \in \Xi^\Gamma} f(\xi^\Sigma)$. Notice that because f is convex, its maximum is always reached at an extreme point of Ξ^Γ, yielding discrete optimization problem

$$\max_{\xi \in \text{ext}(\Xi^\Gamma)} f(\xi^\Sigma). \tag{3.1}$$

Problem (3.1) may not be easy to solve since $\text{ext}(\Xi^\Gamma)$ contains an exponential number of elements and function f is non-linear. However, recall that, because of the definition of Ξ^Γ, we do not need to know the entire vector $\xi \in \text{ext}(\Xi^\Gamma)$ to compute $f(\xi^\Sigma)$.

In fact, it is enough to know the cumulative uncertainties $\xi_i^\Sigma = \sum_{t=1}^i \xi_t$ for each $i = 1, \ldots, n$, which are equivalent to the cumulative deviations $\sum_{t=1}^i \xi_t - \sum_{t=1}^i \overline{\xi}_t$ for each $i = 1, \ldots, n$ because $\xi \in [\overline{\xi}, \overline{\xi} + \widehat{\xi}]$. With this in mind, we introduce

$$f_i'(\phi_i) = \max\{g_i(\overline{\xi}_i^\Sigma) + g_i^1\phi_i, h_i(\overline{\xi}_i^\Sigma) + h_i^1\phi_i\},$$

obtained from f_i by treating separately the cumulative mean $\overline{\xi}_i^\Sigma = \sum_{t=1}^i \overline{\xi}_t$ and the cumulative deviation $\phi_i = \xi_i^\Sigma - \overline{\xi}_i^\Sigma$. Namely, let $\eta \in \{0,1\}^n$ be a binary vector that satisfies $\|\eta\|_1 \leq \Gamma$ and let $\xi \in \text{ext}(\Xi^\Gamma)$ be the associated vector of uncertain parameters, defined as $\xi_i = \overline{\xi}_i + \widehat{\xi}_i\eta_i$ for each $i = 1, \ldots, n$. One readily checks that $f_i(\xi_i^\Sigma) = f_i'(\phi_i)$ if and only if $\phi_i = \sum_{t=1}^i \widehat{\xi}_t\eta_t$. Therefore, adversarial problem (3.1) can be rewritten as

$$\begin{array}{lll} & \max & \displaystyle\sum_{i=1}^n f_i'(\phi_i) \\ (\mathbf{AP}) & \text{s.t.} & \displaystyle\phi_i = \sum_{t=1}^i \widehat{\xi}_t\eta_t, \quad \forall i = 1, \ldots, n, \\ & & \displaystyle\sum_{i=1}^n \eta_i \leq \Gamma, \\ & & \eta_i \in \{0,1\}, \quad \forall i = 1, \ldots, n. \end{array}$$

Up to now we have shown that the optimal solution cost of (**AP**) only depends on the cumulative deviations ϕ_i for each $i = 1, \ldots, n$. To obtain a DPA, we still need a way to enumerate only the most promising cumulative deviations. Let $\overline{\phi}$ be the maximum allowed cumulative deviation, that is, $\overline{\phi} = \max_{S \subseteq \{1,\ldots,n\}:|S|=\Gamma} \sum_{i \in S} \widehat{\xi}_i$. We define $\alpha(j, \gamma, \phi)$, for each triple of integers $1 \leq j \leq n, 0 \leq \gamma \leq \Gamma$ and $0 \leq \phi \leq \overline{\phi}$, as the optimal value of the restricted problem for set $\{1, \ldots, j\}$ with at most γ deviations and a cumulative deviation of ϕ:

$$\begin{array}{lll} \alpha(j,\gamma,\phi) = \max & \displaystyle\sum_{i=1}^j f_i'(\phi_i) & \\ \text{s.t.} & \phi_j = \phi, & (3.2) \\ & \displaystyle\phi_i = \sum_{t=1}^i \widehat{\xi}_t\eta_t, \quad \forall i = 1, \ldots, j, & (3.3) \\ & \displaystyle\sum_{i=1}^j \eta_i \leq \gamma, & (3.4) \\ & \eta_i \in \{0,1\}, \quad \forall i = 1, \ldots, j. & (3.5) \end{array}$$

Let $\alpha(j, \gamma, \phi) = -\infty$ if the feasible set defined by (3.2) − (3.5) is empty because the value of ϕ cannot be reached by a sum of deviations. Hence, we have that $\alpha(1, 0, 0) = f_1'(0)$, $\alpha(1, \gamma, \widehat{\xi}_1) = f_1'(\widehat{\xi}_1)$ for each $1 \leq \gamma \leq \Gamma$, and $\alpha(1, \gamma, \phi) = -\infty$ for the remaining cases.

We see immediately that the optimal solution cost of the adversarial problem (**AP**), denoted by opt(**AP**), can be computed as $\text{opt}(\mathbf{AP}) = \max_{\phi=0,\ldots,\overline{\phi}} \alpha(n, \Gamma, \phi)$. Moreover, we see easily by contradiction that $\alpha(n, \gamma, \phi)$ satisfies the functional equation stated below.

LEMMA 3.1. *For $j > 1$, each $\alpha(j, \gamma, \phi)$ can be obtained using the following recursion:*

$$\alpha(j, \gamma, \phi) = f'_j(\phi) + \max\{\alpha(j-1, \gamma, \phi), \alpha(j-1, \gamma-1, \phi - \widehat{\xi}_j)\}, \tag{3.6}$$

for each $j = 2, \ldots, n, \gamma = 0, \ldots, \Gamma$, and $\phi = 0, \ldots, \overline{\phi}$.

The computation of $f'_j(\phi)$ can be done in constant time for each $j = 2, \ldots, n$ and $\phi = 0, \ldots, \overline{\phi}$, yielding a pseudo-polynomial worst-case complexity for our DPA.

LEMMA 3.2. *Problem (**AP**) can be solved by a DPA in $\mathcal{O}(n\Gamma\overline{\phi})$ operations.*

Using the equivalence between separation and optimization (e.g. [24]), it follows that (**P**) can be solved in pseudo-polynomial time whenever $\mathcal{Z}$ is an *easy* convex set, which is the case for the RLSP.

COROLLARY 3.3. *Consider problem (**P**) and let $\mathcal{Z}$ be a convex set that has a polynomial time separation oracle. Then, problem (**P**) can be solved in pseudo-polynomial time.*

Proof. We present next a simple cutting-plane algorithm for solving (**P**). Let J be a non-negative integer and $\mathbf{f}_i^j(z)$ be an affine function of z for each $1 \leq i \leq n, 1 \leq j \leq J$. We solve (**P**) by a cutting-plane algorithm based on the following relaxation:

$$(\mathbf{R}) \quad \begin{array}{ll} \min & c^T z \\ \text{s.t.} & z \in \mathcal{Z}, \\ & \displaystyle\sum_{i=1}^{n} \mathbf{f}_i^j(z) \leq d^T z, \quad \forall j = 1, \ldots, J, \end{array}$$

initialized with $J = 0$. Given a feasible solution z^* to (**R**), we solve the adversarial problem. If $\max_{\xi \in \Xi^\Gamma} f(\xi^\Sigma, z^*) > d^T z^*$, we let ξ^* be an optimal solution of the maximization problem, set $J \leftarrow J+1$, and add a new constraint to (**R**) where $\mathbf{f}_i^j(z) = g_i(\xi^{*\Sigma}, z)$ if $g_i(\xi^{*\Sigma}, z^*) = \max\{g_i(\xi^{*\Sigma}, z^*), h_i(\xi^{*\Sigma}, z^*)\}$ and $\mathbf{f}_i^j(z) = h_i(\xi^{*\Sigma}, z)$, otherwise. □

Lemma 3.2 states that solving the adversarial problem using the proposed dynamic approach can be considered an interesting option when the sum of deviations $\overline{\phi}$ is not too large. The cases when the deviations are large correspond to the situations where the uncertain parameters can assume a wide range of values and therefore the decision maker is very conservative or very little is known about the uncertain parameters. We also see that the algorithm is polynomial when Γ is constant since ϕ can take at most n^Γ different values.

3.2. Fully polynomial time approximation scheme. We show next how to modify our DPA to obtain a FPTAS for problems (**P**) that satisfy additional assumptions. Our approach works in two steps. First, we adapt to (**AP**) the FPTAS proposed for the knapsack problem by [27]. Their main idea is to reduce the precision on the parameters by dividing them with a well-chosen number, identical for all parameters. Our approach holds whenever functions f_i and g_i satisfy the the technical assumption stated below. Then, we show that whenever $\mathcal{Z}$ is convex, a FPTAS for (**AP**) can be turned into a FPTAS for (**P**).

ASSUMPTION 1. *Consider problem (**AP**) described by (g^0, g^1, h^0, h^1, z) and let $\boldsymbol{\xi} = \max_{i=1,\ldots,n} \widehat{\xi}_i$. There exists a positive function χ of (g^0, g^1, h^0, h^1) such that $LB =$*

$\boldsymbol{\xi}\chi(g^0, g^1, h^0, h^1)$ *is a lower bound for* $\text{opt}(\mathbf{AP})$ *and*

$$\frac{\left|\max_{i=1,\ldots,n}\{g_i^0, g_i^1, h_i^0, h_i^1\}\right|}{\chi(g^0, g^1, h^0, h^1)} \leq \mathcal{P}(n), \tag{3.7}$$

where $\mathcal{P}(n)$ *is a polynomial in* n.

Let us illustrate Assumption 1 on the RLSP, assuming that $\min\{s_n, p_n\} > 0$. For that problem, we see that a lower bound for the optimal solution of the problem is $LB = \frac{\boldsymbol{\xi}\min\{s_n,p_n\}}{2}$, so that (3.7) becomes $\frac{2\max_{i=1,\ldots,n}\{s_i,p_i\}}{\min\{s_n,p_n\}} \leq \mathcal{P}(n)$ for some polynomial $\mathcal{P}(n)$. For instance, requiring that $\mathcal{P}(n)$ be equal to some constant $\lambda > 0$ yields the set of instances for which $\frac{\max_{i=1,\ldots,n}\{s_i,p_i\}}{\min\{s_n,p_n\}} \leq \frac{\lambda}{2}$. Considering polynomials of higher degrees yields a larger set of admissible instances while increasing the computational complexity of the resulting FPTAS.

LEMMA 3.4. *Consider problem* $(\mathbf{AP})$ *such that Assumption 1 holds. There exists a FPTAS for* $(\mathbf{AP})$.

Proof. For any $\epsilon > 0$, we let $K = \frac{\epsilon\boldsymbol{\xi}\chi(g^0,g^1,h^0,h^1)}{2n(\Gamma+1)\left|\max_{i=1,\ldots,n}\{g_i^0,g_i^1,h_i^0,h_i^1\}\right|}$ and define the new mean value $\overline{\xi}_i' = \frac{\overline{\xi}_i}{K}$ and deviation $\widehat{\xi}_i' = \lfloor\frac{\widehat{\xi}_i}{K}\rfloor$ for each $i = 1, \ldots, n$, and $\overline{\phi}' = \max_{S\subseteq\{1,\ldots,n\}:|S|=\Gamma}\sum_{i\in S}\widehat{\xi}_i'$. Then, execute the DPA presented in the previous section to $(\mathbf{AP})$ using the vector of deviations $\widehat{\xi}'$. Using notation $\boldsymbol{\xi}' = \lfloor\frac{\boldsymbol{\xi}}{K}\rfloor$, we see that the running time of the algorithm is polynomial in $(n, \Gamma, 1/\epsilon)$ since

$$\mathcal{O}(n\Gamma\overline{\phi}') = \mathcal{O}(n\Gamma^2\boldsymbol{\xi}') = \mathcal{O}\left(n\Gamma^2\left\lfloor\frac{\boldsymbol{\xi}}{K}\right\rfloor\right) = \mathcal{O}\left(n\Gamma^2\left\lfloor\frac{n\Gamma\mathcal{P}(n)}{\epsilon}\right\rfloor\right).$$

We are left to show that the optimal solution to the problem with $\widehat{\xi}'$ is an $(1-\epsilon)$-approximate solution for the original problem.

Let $\eta', \eta^* \in \{\eta|\eta \in \{0,1\}^n, \|\eta\|_1 \leq \Gamma\}$ be the solution returned by the above algorithm and the optimal solution, respectively, and let $\text{profit}(\cdot)$ and $\text{profit}'(\cdot)$ denote the profit of any element of $\{0,1\}^n$ using deviations $\widehat{\xi}$ and $\widehat{\xi}'$, respectively. Clearly, $\text{profit}(\eta') \leq \text{opt}(\mathbf{AP})$. Then, recall from the definition that $K\,\text{profit}'(\eta) =$

$$\sum_{i=1}^{n}\max\left\{g_i^0K + g_i^1\sum_{t=1}^{i}\left(\overline{\xi}_t + \eta_tK\left\lfloor\frac{\widehat{\xi}_t}{K}\right\rfloor\right), h_i^0K + h_i^1\sum_{t=1}^{i}\left(\overline{\xi}_t + \eta_tK\left\lfloor\frac{\widehat{\xi}_t}{K}\right\rfloor\right)\right\},$$

for any $\eta \in \{\eta|\eta \in \{0,1\}^n, \|\eta\|_1 \leq \Gamma\}$ and observe that $\left|\widehat{\xi}_t - K\left\lfloor\frac{\widehat{\xi}_t}{K}\right\rfloor\right| \leq K$. Hence, for any $\eta \in \{\eta|\eta \in \{0,1\}^n, \|\eta\|_1 \leq \Gamma\}$ we have that

$$|\,\text{profit}(\eta) - K\,\text{profit}'(\eta)| \leq n(\Gamma+1)K\left|\max_{i=1,\ldots,n}\{g_i^0, g_i^1, h_i^0, h_i^1\}\right|. \tag{3.8}$$

Therefore,

$$\begin{aligned}
\text{profit}(\eta') &\geq K\,\text{profit}'(\eta') - n(\Gamma+1)K\left|\max_{i=1,\ldots,n}\{g_i^0, g_i^1, h_i^0, h_i^1\}\right| \\
&\geq K\,\text{profit}'(\eta^*) - n(\Gamma+1)K\left|\max_{i=1,\ldots,n}\{g_i^0, g_i^1, h_i^0, h_i^1\}\right| \\
&\geq \text{profit}(\eta^*) - 2n(\Gamma+1)K\left|\max_{i=1,\ldots,n}\{g_i^0, g_i^1, h_i^0, h_i^1\}\right| \\
&= \text{opt}(\mathbf{AP}) - \epsilon LB \geq (1-\epsilon)\text{opt}(\mathbf{AP}),
\end{aligned}$$

proving the result. □

The lemma below shows that the existence of a FPTAS for (**AP**) can be translated into a FPTAS for special cases of problem (**P**).

LEMMA 3.5. *Consider the following special case of problem* (**P**)

$$(\mathbf{P}') \qquad \begin{array}{ll} \min & c^T z + \theta \\ s.t. & z \in \mathcal{Z} \\ & f(\xi^\Sigma, z) \leq \theta, \quad \forall \xi \in \Xi^\Gamma. \end{array}$$

Assume that $\mathcal{Z}$ is a convex set that has a polynomial time separation oracle and that there exists a FPTAS for $\max_{\xi\in\Xi^\Gamma} f$. There exists a FPTAS for ($\mathbf{P}'$).

Proof. We must show that for each $\epsilon > 0$, we can provide in polynomial time an $(1+\epsilon)$-approximate solution to ($\mathbf{P}'$). Our approach relies on the cutting-plane algorithm from Corollary 3.3 with the difference that each (**AP**) is now solved with the FPTAS to provide an $\frac{1}{1+\epsilon}$-approximate solution. Let (z', θ') be the solution returned by the approximate cutting plane algorithm. We claim that $(z', (1+\epsilon)\theta')$ is the desired approximate solution. Clearly, $(z', (1+\epsilon)\theta')$ is computed in polynomial time. Then, we must verify that

$$\text{opt}(\mathbf{P}') \leq c^T z' + (1+\epsilon)\theta' \leq (1+\epsilon)\text{opt}(\mathbf{P}').$$

To prove the first inequality, we rewrite ($\mathbf{P}'$) as

$$\min_{z\in\mathcal{Z}} c^T z + F(z),$$

where $F(z) = \max_{\xi\in\Xi^\Gamma} f(\xi^\Sigma, z)$. Since θ' is an $\frac{1}{1+\epsilon}$-approximate solution of the corresponding (**AP**), we have that $\theta' \geq \frac{1}{1+\epsilon}F(z')$. Hence,

$$c^T z' + (1+\epsilon)\theta' \geq c^T z' + F(z') \geq \text{opt}(\mathbf{P}').$$

We prove the second inequality by contradiction. Assuming the inequality does not hold, we obtain:

$$\text{opt}(\mathbf{P}') < \frac{c^T z'}{1+\epsilon} + \frac{1+\epsilon}{1+\epsilon}\theta' \leq c^T z' + \theta'. \tag{3.9}$$

Moreover, because θ' is an approximate solution of the corresponding (**AP**), we have

$$c^T z' + \theta' \leq c^T z' + F(z') \leq \text{opt}(\mathbf{P}'),$$

which is in contradiction with (3.9). □

3.3. General uncertainty set. We discuss below how to extend the DPA to handle the general uncertainty set Ξ^Γ.

Downward deviations. Downward deviations of ξ can be handled by replacing constraints (3.4) and (3.5) in the definition of $\alpha(j,\gamma,\phi)$, by $\sum_{i=1}^{j}|\eta_i| \leq \gamma$ and $\eta_i \in \{-1,0,1\}$, respectively. The recursion formula (3.6) is then adapted to:

$$\alpha(j,\gamma,\phi) = f'_j(\phi) + \max\{\alpha(j-1,\gamma,\phi), \alpha(j-1,\gamma-1,\phi-\widehat{\xi}_j), \alpha(j-1,\gamma-1,\phi+\widehat{\xi}_j)\},$$

for each $j = 2,\ldots,n, \gamma = 0,\ldots,\Gamma$, and $\phi = 0,\ldots,\overline{\phi}$. The FPTAS for (**AP**) still holds in this case.

Fractional Γ. If Γ is fractional one can take advantage from the fact that the extreme points of Ξ^Γ can have at most one fractional η_i. Let $\Gamma^I = \lfloor\Gamma\rfloor$ and $\Gamma^F = \Gamma - \Gamma^I$. Hence (**AP**) can be solved by applying the DPA $n+1$ times, with Γ replaced by Γ^I. In the first iteration, we suppose that η has no fractional component (no other change in data is required). In each of the remaining n iterations, we assume that $\eta_j = \Gamma^F$. Then, we redefine $\overline{\xi}_j$ as $\overline{\xi}_j + \Gamma^F\widehat{\xi}_j$ and $\widehat{\xi}_j$ as 0 for that iteration. This approach works because the DPA does not require that $\overline{\xi}_i$ be integer. The FPTAS for (**AP**) also holds in this case.

3.4. Other objective functions. We discuss next whether the DPA and the related FPTAS can be extended to more general functions. One the one hand, the DPA holds when $f_i : \mathbb{R} \to \mathbb{R}$ is any quasi-convex function (quasi-convexity is required for replacing Ξ^Γ by $\text{ext}(\Xi^\Gamma)$). Clearly, the computational complexity of the resulting DPA increases according to the computational cost of evaluating each function f_i. For instance, for the type of functions f_i considered in this paper, this complexity is $O(1)$. A simple example of non-linear function arises in scheduling problems that minimize the squared deviation, where the cost of finishing a job i having a deadline d_i at time ξ_i^Σ is defined as $f_i(\xi^\Sigma) = (\xi_i^\Sigma - d_i)^2$. On the other hand, the FPTAS extends easily only to functions f_i defined as the maxima of K affine functions. This extension could be used to address lot-sizing problems with piecewise linear convex holding costs for instance [34]. The extension to K affine functions (instead of 2 as in (1.2)) carries over immediately to problems (**P**) and (**P**′). However, the row-and-column generation algorithm described in Section 2.1 and its cutting-plane version used in Corollary 3.3 are not designed to handle the more general quasi-convex functions mentioned for the DPA.

The separation problem is not affected by the dependency of f on z, so that the DPA and FPTAS extend directly to more complex functions of z. However, the row-and-column generation algorithm would then involve solving non-linear (**MP**). For instance, allowing g_i and h_i to be bi-affine functions, as seen in the recent literature on robust optimization [23, 6], would lead to bilinear (**MP**) when considering the more general models described in the next section.

4. Variable order.

4.1. Introducing permutations. We discuss in this section how to extend the algorithmic framework discussed in Section 2 to handle problems where the order used to define $\xi_i^\Sigma = \sum_{j=1}^{i}\xi_j$ must depend on the values taken by the optimization variables z. Consider for instance the one-machine scheduling problem that minimizes tardiness. Namely, each job i has a given deadline d_i and there is a cost for finishing the job after the deadline that is proportional to the delay. One readily sees that the finishing times of the jobs in a schedule, denoted by z, can be defined by cumulative

sums similar to ξ^{Σ} but depending on the order used in the schedule z. More precisely, the order of the jobs in z can be described by a permutation of the jobs $\{1,\dots,n\}$. Let $\tau_z(i)$ denote the order of job i in the schedule z. The finishing time of job i is given by

$$y_i(\xi,z)=\sum_{j=1}^{\tau_z(i)}\xi_{\tau_z^{-1}(i)}.$$

Given a penalty weight w_i for each job i, the total penalty cost of the schedule z is

$$f(\xi,z)=\sum_{i=1}^{n}\max\{w_i(y_i(\xi,z)-d_i),0\}.$$

We detail in the next subsection a similar construction for the vehicle routing problem with deadlines. Notice that for more complex problems it is useful to introduce two distinct functions. For instance, in assignment type project scheduling problems [16], the processing time of a job depends on the amount of resource that is assigned to the job. Hence a solution z would not only describe the order of the jobs, but also their processing times. We would then let τ_z represent the order in which the jobs are realized, while π_z would be a function from $\{1,\dots,n\}$ to $\{1,\dots,m\}$ that would characterize the processing times of the jobs among the m available processing times. Notice that our framework does not handle repetitions of the components of ξ in the partial summations so that π_z must be an injective function.

With these applications in mind, we generalize problem (**P**) as follows. We consider two positive integers n and m, such that $m\geq n$ and $\Xi^{\Gamma}\subset\mathbb{R}^m$ and consider the robust constraint

$$f(y(\xi,z),z)\leq d^Tz,\quad \xi\in\Xi^{\Gamma}, \tag{4.1}$$

with f defined as follows.

ASSUMPTION 2. *For every $z\in\mathcal{Z}$, we can define a permutation τ_z of $\{1,\dots,n\}$ and an injective function $\pi_z:\{1,\dots,n\}\to\{1,\dots,m\}$ with $y_i(\xi,z)=\sum_{j=1}^{\tau_z(i)}\xi_{\pi_z(j)}$ such that*

$$f(y(\xi,z),z)=\sum_{i=1}^{n}\max\{g_i(y_i(\xi,z),z),h_i(y_i(\xi,z),z)\}. \tag{4.2}$$

When z is fixed, we use the shorter notation $f(y(\xi),z)$.

The following extends the algorithm proposed in Section 2.1 by considering a vector of optimization variables y^{ξ} for each $\xi\in\Xi$, which models the function $y(\xi,z)$ in the new master problem. Namely, let us denote by $\mathcal{Y}$ the linear restrictions that link optimization vectors z and y^{ξ} such that for any $z\in\mathcal{Z}$ the projection of $\mathcal{Y}$ on y^{ξ} is reduced to singleton $\{(\sum_{t=1}^{\tau_z(1)}\xi_{\pi_z(t)},\dots,\sum_{t=1}^{\tau_z(n)}\xi_{\pi_z(t)})\}$. We illustrate the set in the next subsection. The counterpart of (**P**) under Assumption 2 is

$$\begin{aligned}
&\min && c^Tz\\
(\mathbf{Pvar})\qquad &\text{s.t.} && z\in\mathcal{Z},\\
&&& (z,y^{\xi})\in\mathcal{Y},\quad \forall\xi\in\Xi^{\Gamma},\\
&&& f(y^{\xi},z)\leq d^Tz,\quad \forall\xi\in\Xi^{\Gamma}.
\end{aligned} \tag{4.3}$$

The difference between (**P**) and (**Pvar**) lies in constraints (4.3), which ensure that the optimization vector y^ξ takes the right value for any $z \in \mathcal{Z}$ and $\xi \in \Xi^\Gamma$. The row-and-column-generation algorithm depicted in Section 2.1 extends naturally to cope with the new variables and constraints. More precisely, given a feasible solution z^* to (**MP**), one checks the feasibility of z^* for (**Pvar**) by solving an adversarial problem, which is identical to (**AP**) because z^* is fixed. Let ξ^* be the optimal solution for the adversarial problem. If $f(y^{\xi^*}, z^*) > d^T z^*$, then $\Xi^0 \leftarrow \Xi^0 \cup \{\xi^*\}$, and the corresponding optimization vectors y^{ξ^*} and φ^{ξ^*} and constraints $(z, y^{\xi^*}) \in \mathcal{Y}$ and

$$\begin{aligned}
&\sum_{i=1}^{n} \varphi_i^{\xi^*} \leq d^T z \\
&\varphi_i^{\xi^*} \geq g_i(y^{\xi^*}, z), \quad \forall i = 1, \ldots, n \\
&\varphi_i^{\xi^*} \geq h_i(y^{\xi^*}, z), \quad \forall i = 1, \ldots, n
\end{aligned}$$

are added to (**MP**).

We show next that the problems modeled by (**Pvar**) must involve non-connected feasibility sets $\mathcal{Z}$. Roughly speaking, the next result formalizes the intuitive idea that the permutations are non-trivial only if $\mathcal{Z}$ is non-connected, which happens in mixed-integer linear programs for instance.

LEMMA 4.1. *If $\mathcal{Z}$ is a connected set, then for all $z, z' \in \mathcal{Z}$ it holds that*

$$\sum_{t=1}^{\tau_z(i)} \xi_{\pi_z(t)} = \sum_{t=1}^{\tau_{z'}(i)} \xi_{\pi_{z'}(t)}, \text{ for each } i = 1, \ldots, n. \tag{4.4}$$

Proof. Let $\xi \in \text{ext}(\Xi^\Gamma)$ be fixed and let the projection of $\mathcal{Y}$ on its component y be represented by function $\mathbf{Y}(z) = \mathcal{Y} \cap \{z\}$, for all $z \in \mathcal{Z}$. Function $\mathbf{Y}$ is continuous in z, because its image is characterized by constraints that are affine functions of z. Suppose now that $\mathcal{Z}$ is connected and that there exists $z, z' \in \mathcal{Z}$ such that (4.4) does not hold and let $\delta > 0$ be a positive real. Denote by $\overrightarrow{zz'}$ a path in $\mathcal{Z}$ from z to z' and consider $a, b \in \overrightarrow{zz'}$ with $\|a - b\| \leq \delta$ such that (4.4) does not hold. By assumption and recalling that $\widehat{\xi}$ is integer, there exists an index i such that

$$|\mathbf{Y}_i(a) - \mathbf{Y}_i(b)| = \left| \sum_{t=1}^{\tau_a(i)} \xi_{\pi_a(t)} - \sum_{t=1}^{\tau_b(i)} \xi_{\pi_b(t)} \right| \geq 1,$$

proving the discontinuity of $\mathbf{Y}$ at a. □

The lemma implies that when $\mathcal{Z}$ is connected, constraints (4.3) can be removed from (**Pvar**) getting back to the simpler version (**P**).

4.2. Illustration: Robust Traveling Salesman Problem with deadlines. Here we consider a variant of the Traveling Salesman Problem where a deadline is considered for the visit to each client. The resulting problem is called the TSPD. The problem occurs in many practical situations and has been studied previously in the stochastic programming literature, see for instance [17]. The Robust TSPD, denoted RTSPD, is defined as follows. We are given a complete digraph $G = (V, A)$ with $V = \{0, 1, \ldots, n\}$ where node 0 is the depot, costs c_{ij} for crossing arc $(i, j) \in A$, and a deadline b_i associated with each node but the depot. If the vehicle arrives after time b_i, a penalty cost, denoted by s_i, is incurred per time unit of the violated time. We

assume that the traveling times ξ_{ij} are uncertain and belong to the aforementioned budgeted polytope where $\overline{\xi}_{ij}$ is the regular traveling time and $\widehat{\xi}_{ij}$ is the maximum possible delay.

In our formulation below, binary variable x_{ij} indicates whether arc (i,j) is in the solution or not, and continuous variable y_i^ξ represents the time of visit to vertex i when the vector of traveling times ξ is considered. We assume node 0 is the depot, from where the vehicle departs. The RTSPD with time windows can be modeled as follows.

$$\min \quad \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} + \theta \tag{4.5}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{ij} = 1, \quad \forall j \in V, \tag{4.6}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad \forall i \in V, \tag{4.7}$$

$$y_0^\xi = 0, \quad \forall \xi \in \Xi^\Gamma, \tag{4.8}$$

$$y_j^\xi \geq y_i^\xi + \xi_{ij} - M(1 - x_{ij}), \quad \forall \xi \in \Xi^\Gamma, (i,j) \in A, \tag{4.9}$$

$$\theta \geq \sum_{i=1}^{n} \max\{s_i(y_i^\xi - b_i), 0\}, \quad \forall \xi \in \Xi^\Gamma, \tag{4.10}$$

$$x_{ij} \in \{0,1\}, \quad \forall (i,j) \in A, \tag{4.11}$$

where M is a big-M constant. Constraints (4.6) and (4.7) ensure that each node is visited once. Constraints (4.8) and (4.9) define the time variables for the visit to each node, which also forbids cycles. Constraints (4.10) models the penalty incurred for not meeting the deadline at each node, and adds the penalty to θ.

We have yet to show that the above formulation is a special case of (**Pvar**). Here, set $\mathcal{Z}$ contains all $z = (x, \theta)$ where θ is an unrestricted real and x is a Hamiltonian cycle, that is, a binary vector that satisfies constraints (4.6) and (4.7) and whose associated subgraph does not contain cycles. Strictly speaking, the above formulation is not a special case of (**Pvar**) because cycles are forbidden through constraints (4.9), which are constraints that characterize $\mathcal{Y}$ and should therefore not contribute to the definition of $\mathcal{Z}$ as introduced in Section 4.1. Hence, to be truly coherent with (**Pvar**), one should add to the model classical constraints to forbid cycles (e.g. subtour inequalities, cut inequalities, MTZ inequalities, ...), which we omit to keep our model as simple as possible. Set $\mathcal{Y}$ is then defined by constraints (4.8) and (4.9). Namely, consider a fixed $z = (x, \theta) \in \mathcal{Z}$ where x is a fixed Hamiltomian cycle, and let $\tau_z(i)$ denote the position of node i in the cycle, starting from the depot ($\tau_z(0) = 0$), and let $\pi_z(i)$ denote the arc that comes in position i in the cycle. One readily verifies that (4.8) and (4.9) yield the following restriction for y_i^ξ for each $i = 1, \ldots, n$:

$$y_i^\xi \geq \sum_{t=1}^{\tau_z(i)} \xi_{\pi_z(i)}. \tag{4.12}$$

Again, to be faithful to (**Pvar**), constraint (4.12) should hold to equality. Although this could be enforced by complementing the formulation with additional linear constraints with big-M constants, these are unnecessary because the equality holds in any

optimal solution of the RTSPD. Finally, function $f(y(\xi,z),z)$ can be defined from the rhs of (4.10), $\sum_{i=1}^{n} \max\{s_i(y_i(\xi,z)-b_i),0\}$, which satisfies Assumption 2. Therefore, the decomposition of the problem can be done similarly to Section 2. Given a solution $z \in \mathcal{Z}$ to the master problem, the adversarial problem looks for the scenario ξ that maximizes $f(y(\xi),z)$, namely,

$$\text{opt}(\mathbf{AP}) = \max_{\xi \in \Xi^{\Gamma}} \left\{ \sum_{i=1}^{n} f_i(\xi) \right\} \text{ where } f_i(\xi) = \max\{s_i(y_i(\xi)-b_i),0\}.$$

Differently from the RLSP presented in the previous subsection, we see that the meaning of $y_i(\xi,z)$ depends here on vector $z=(x,\theta)$, since the order in which the nodes and arcs are visited depends on the specific Hamiltonian cycle x.

5. Extensions. In this section we discuss two extensions of the adversarial problem and the DPA. For the sake of simplicity, these extensions are presented for specific optimization problems.

5.1. A simple inventory distribution problem. In this section we consider an example of the case where the adversarial problem can be separated into k subproblems with a single linking constraint which is the constraint imposing a maximum number of Γ deviations. Each subproblem coincides with (**AP**). The approach consists in solving each subproblem (**AP**) with the DPA for each possible number of deviations and then, in a second stage, combine the k subproblems.

We exemplify this situation below with a robust variant of a simplistic inventory distribution problem. We are given a set k of retailers and a set of n time periods. The company controls the inventory at the retailers and needs to decide when and how much to deliver in each time period at each retailer. We define the following parameters: C is the distribution capacity per period for each retailer and $\xi_{ij}, s_{ij}, p_{ij}, f_{ij}$ represent for retailer j in period i, the demand, backlogging cost, holding cost and fixed transportation cost, respectively. As before, demand vector ξ is uncertain and belong to uncertainty set Ξ^{Γ}_{IR} defined as $\Xi^{\Gamma}_{IR} = \left\{\xi : \xi_{ij} = \overline{\xi}_{ij} + \widehat{\xi}_{ij}\eta_{ij}, i=1,\ldots,n, j=1,\ldots,k, \|\eta\|_{\infty} \leq 1, \|\eta\|_1 \leq \Gamma\right\}$. Hence, Γ is the total number of deviations allowed.

In our simplistic variant presented below, the only distribution variables are the continuous variables x_{ij} that describe the quantity delivered to retailer j in period i. Hence, x_{ij} plays the role of the former production variable x_i used in Section 2.2. A formulation for the Robust Inventory Routing Problem $RIRP$ follows.

$$\begin{aligned}
\min \quad & f^T x + \theta \\
\text{s.t} \quad & 0 \leq x_{ij} \leq C, \quad \forall i=1,\ldots,n, j=1,\ldots,k, \\
& \theta \geq \sum_{i=1}^{n}\sum_{j=1}^{k} \max\left\{ s_{ij}\sum_{t=1}^{i}(\xi_{tj}-x_{tj}), -p_{ij}\sum_{t=1}^{i}(\xi_{tj}-x_{tj}) \right\}, \quad \forall \xi \in \Xi^{\Gamma}_{IR}. \qquad (5.1)
\end{aligned}$$

Unlike the simple lot-sizing from Section 2.2, it is not possible here to sum up total demands up to period i in variable y_i^{ξ} since the latter depends also on the particular retailer. One way to avoid this difficulty is to ignore the correlation of demands for

different retailers, replacing the above model with

$$\begin{aligned}
\min \quad & f^T x + \sum_{j=1}^{k} \theta_j \\
\text{s.t} \quad & 0 \le x_{ij} \le C, \quad \forall i = 1, \dots, n, j = 1, \dots, k, \\
& \theta_j \ge \sum_{i=1}^{n} \max \left\{ s_{ij} \left(\xi_i^{\Sigma} - \sum_{t=1}^{i} x_{tj} \right), -p_{ij} \left(\xi_i^{\Sigma} - \sum_{t=1}^{i} x_{tj} \right) \right\}, \\
& \qquad \forall j = 1, \dots, k, \xi \in \Xi_{uncor}^{\Gamma^j},
\end{aligned}$$

with $\Xi_{uncor}^{\Gamma^j} = \left\{ \xi : \xi_i = \overline{\xi}_i + \widehat{\xi}_i \eta_i, i = 1, \dots, n, \|\eta\|_\infty \le 1, \|\eta\|_1 \le \Gamma^j \right\}$, and $\xi_i^{\Sigma} = \sum_{t=1}^{i} \xi_t$.

The uncorrelated model yields k adversarial problems, each of them identical to the adversarial problem of the RLSP. Still, it may not be satisfactory in some applications to ignore the correlations of demands among different retailers. Hence, we explain below how to solve the adversarial problem induced by constraint (5.1). As mentioned previously, for each retailer $j \in \{1, \dots, k\}$, we have an adversarial problem which coincides with the adversarial problem of the RLSP. The k problems are linked through the maximum number of deviations Γ. For each $j \in \{1, \dots, m\}$, $\gamma \in \{0, \dots, \Gamma\}$ let $\delta_{j\gamma}$ denote the value of the adversarial problem for the RLSP for retailer j with γ deviations, computed by the DPA from Section 3. The value of the adversarial problem for the RIRP is given by the best combination of these k subproblems:

$$\begin{aligned}
\text{opt}(\mathbf{AP'}) = \max \Bigg\{ & \sum_{j=1}^{k} \sum_{\gamma=0}^{\Gamma} \delta_{j\gamma} u_{j\gamma} : \sum_{j=1}^{k} \sum_{\gamma=0}^{\Gamma} \gamma u_{j\gamma} \le \Gamma, u_{j\gamma} \in \{0,1\}, \\
& \qquad j = 1, \dots, k, \gamma = 0, \dots, \Gamma \Bigg\}.
\end{aligned}$$

where $u_{j\gamma}$ is a binary variable that indicates whether γ deviations are considered for retailer j. This linking problem, known as the Linking Set Problem can be solved by dynamic programming in $\mathcal{O}(k\Gamma)$ operations, see [4]. Recall that $\overline{\phi} = \max_{S \subseteq \{1,\dots,n\}: |S| = \Gamma} \sum_{i \in S} \widehat{\xi}_i$. We obtain the following result.

LEMMA 5.1. *Problem* $(\mathbf{AP'})$ *can be solved by a DPA in* $\mathcal{O}(n\Gamma\overline{\phi} + k\Gamma)$ *operations.*

Notice that, for the sake of clarity, we presented a simplistic inventory routing problem. However, our approach extends directly to inventory routing problems where the routing and/or distribution is modeled by introducing binary variables (e.g. [33]), as well as problems with different capacities, since these refinements do not affect $(\mathbf{AP'})$. We also point out that similar adversarial problems occur in other problems such as the vehicle routing problems with deadlines with uncertain traveling times where the number of delays is bounded by Γ. In the latter problem, it can be important to bound the total number of delays for all vehicle by a single value of Γ.

5.2. The TSP with time-windows. The RTSPD considered in the previous section is a simplification of the widely studied TSP where a full time window $[a_i, b_i]$ is associated to each node different from the depot, thus complementing the deadline b_i with a lower limit a_i. The difficulty of handling a full time window, instead of

a deadline, depends on the type of constraints considered: soft or hard, where a soft constraint can be violated at some cost, and a hard constraint must always be satisfied. For instance, the deadline introduced for the RTSPD in the previous section is a soft constraint. A hard left constraint imposes that if the salesman arrives at node before a_i he must wait until a_i. We examine next the four different combinations for left and right time limits. If left and right time limits are soft, with unitary penalty costs denoted by p and s, respectively, the resulting problem can be formulated as the RTSPD studied previously replacing constraint (4.10) with

$$\theta \geq \sum_{i=1}^{n} \max\{s_i(y_i^\xi - b_i), -p_i(y_i^\xi - a_i)\}, \quad \forall \xi \in \Xi^\Gamma.$$

One readily sees that the above robust constraint is defined by a function that satisfies Assumption 2. If left time limit is soft, with unitary penalty cost p, and right time limit is hard, the problem can be formulated as the RTSPD studied previously replacing constraint (4.10) with robust constraints

$$\theta \geq \sum_{i=1}^{n} \max\{0, -p_i(y_i^\xi - a_i)\}, \quad \forall \xi \in \Xi^\Gamma, \tag{5.2}$$

$$y_i^\xi \leq b_i, \qquad \forall i = 1, \ldots, n, \xi \in \Xi^\Gamma. \tag{5.3}$$

One readily sees that both robust constraints above are defined by functions that satisfy Assumption 2. Actually, solving the (**AP**) associated to constraint (5.3) can be done in polynomial time since it comes down to computing the Γ highest components of $\widehat{\xi}$ among $\{\widehat{\xi}_1, \ldots, \widehat{\xi}_i\}$ for each $i = 1, \ldots, n$. If both time limits are hard, the problem does not enter the framework studied in this paper. The authors of [3] propose different approaches for that problem and show that the corresponding (**AP**) can be solved in polynomial time. The last case, denoted RTSPTW in what follows (standing for Traveling Salesman Problem with Time Windows), considers hard left time limit and soft right time limit with unitary penalty cost p (see [1] for an application). This case is more complex than the previous ones. Still, we show in the balance of the section that the associated adversarial problem can also be solved by a DPA in pseudo-polynomial time.

The mathematical model for the RTSPTW is close to the one given for the TSPD. In addition to (4.5) – (4.11), we must consider a new constraint imposing a lower bound on the start time of each visit:

$$y_i^\xi \geq a_i \quad \forall \xi \in \Xi^\Gamma, i \in V. \tag{5.4}$$

Unfortunately, the addition of constraints (5.4) to the formulation of the RTSPD prevents the RTSPTW from being a special case of problem (**Pvar**). Namely, let $z = (\theta, x) \in \mathcal{Z}$ where x is the incidence vector of a Hamiltonian cycle and θ the worst-case cost due to the delay, and let τ_z and π_z be the functions defined in Section 4.2. To avoid carrying the functions throughout, we assume without loss of generality that $\tau_z(i) = i$ and $\pi_z(i) = i$ for each $i = 1, \ldots, n$. Constraints (5.4) break the structure witnessed for the RTSPD since the arrival time $y_i(\xi)$ at node i can no longer be defined as an affine function of ξ, such as (4.12) used for the RTSPD. Namely, the wait of the salesman in case he arrives at node i before a_i yields arrival times that can be formulated recursively as

$$y_i(\xi) = \max(a_i, a_{i-1} + \xi_{(i-1)i}), \tag{5.5}$$

for each $i = 1, \ldots, n$, where we recall that $(i-1, i)$ denotes the arc that enters i and leaves its predecessor $i-1$ in the Hamiltonian cycle. Therefore, penalty function $f(y(\xi), z) = \sum_{i=1}^{n} \max\{s_i(y_i(\xi) - b_i), 0\}$, with $y_i(\xi)$ defined in (5.5), does not satisfy Assumption 2 because y is not affine. Fortunately, it is possible to adapt our DPA to handle this difficulty. For the sake of simplicity, we explain the generalization of the DPA directly for the RTSPTW, rather than using the general framework depicted in Section 4.

Expanding recursively the maxima in (5.5), $y_i(\xi)$ can be expressed as (see [2, 18])

$$y_i(\xi) = \max_{\ell=1,\ldots,i} \left\{ a_\ell + \sum_{t=\ell}^{i-1} \xi_{t(t+1)} \right\}. \tag{5.6}$$

As for the RTSPD, our objective penalizes the violation of the right time window b_i with unitary penalty cost s_i. Hence the adversarial problem is in this case $\text{opt}(\mathbf{AP}^*) = \max_{\xi \in \Xi^\Gamma} \left\{ \sum_{i=1}^{n} f_i(\xi) \right\}$ where

$$f_i(\xi) = s_i \left[\max_{\ell=1,\ldots,i} \left\{ a_\ell + \sum_{t=\ell}^{i-1} \xi_{t(t+1)} \right\} - b_i \right]^+ = \max_{\ell=1,\ldots,i} \left\{ s_i \left[a_\ell + \sum_{t=\ell}^{i-1} \xi_{t(t+1)} - b_i \right]^+ \right\},$$

where we used the simplified notation $[x]^+$ for $\max\{0, x\}$. Let $\xi_{[\ell i]}$ denote the subvector $\{\xi_{t(t+1)} : t = \ell, \ldots, i-1\}$ and define $\overline{f}_{\ell i}(\xi_{[\ell i]}) = s_i[a_\ell - b_i + \sum_{t=\ell}^{i-1} \xi_{t(t+1)}]^+$. Hence, $f_i(\xi) = \max_{\ell=1,\ldots,i} \overline{f}_{\ell i}(\xi_{[\ell i]})$. Let $\beta(m, \gamma)$ be the value of the optimal solution of the restricted problem defined for the subpath $1, \ldots, m$ with at most γ deviations:

$$\beta(m, \gamma) = \max_{\xi \in \Xi^\gamma_{[1m]}} \left\{ \sum_{i=1}^{m} \max_{\ell=1,\ldots,i} \overline{f}_{\ell i}(\xi_{[\ell i]}) \right\},$$

where $\Xi^\gamma_{[\ell i]} \equiv \left\{ \xi : \xi_t = \overline{\xi}_t + \widehat{\xi}_t \eta_t, t = \ell, \ldots, i, \eta \in \{0,1\}^{i-\ell+1}, \|\eta\|_1 \leq \gamma \right\}$. Clearly, $\text{opt}(\mathbf{AP}^*) = \beta(n, \Gamma)$.

The rest of the section is devoted to the construction of a DPA to compute $\beta(n, \Gamma)$. Notice that for any t, $\sum_{i=t}^{m} \overline{f}_{ti}$ satisfies (1.2), so that the sum can be optimized over the set $\Xi^\gamma_{[tm]}$ in pseudo-polynomial time by appling the algorithm presented in Section 3.1. Let us denote $\overline{f}^\beta(\xi_{[1m]}) = \sum_{i=1}^{m} \max_{\ell=1,\ldots,i} \overline{f}_{\ell i}(\xi_{[\ell i]})$ so that $\beta(m, \gamma)$ can be rewritten as

$$\beta(m, \gamma) = \max_{\xi \in \Xi^\gamma_{[1m]}} \overline{f}^\beta(\xi_{[1m]}).$$

The algorithm from Section 3.1 cannot be used directly to optimize $\overline{f}^\beta(\xi_{[1m]})$ because of the maximization involved in the definition of $\overline{f}^\beta(\xi_{[1m]})$. Hence, we used an alternative recursion based on the key lemma below. The lemma expresses $\overline{f}^\beta(\xi_{[1m]})$ from the set of functions $\left\{ \overline{f}^\beta(\xi_{[1t]}) : 1 \leq t \leq m-1 \right\}$ and the sums $\left\{ \sum_{i=t}^{m} \overline{f}_{ti}(\xi_{[ti]}) : 1 \leq t \leq m \right\}$. We show in the balance of the section how this leads to a DPA.

LEMMA 5.2. *Let $\xi \in \Xi^\gamma$ be fixed and $m \in \{2\dots,n\}$. It holds that*

$$\overline{f}^\beta(\xi_{[1m]}) = \max_{t=1,\dots,m}\left\{\overline{f}^\beta(\xi_{[1(t-1)]}) + \sum_{i=t}^{m}\overline{f}_{ti}(\xi_{[ti]})\right\}.$$

Proof. Since ξ is fixed, we simplify notations and denote $\overline{f}_{\ell i}(\xi_{[\ell i]})$ as $\overline{f}_{\ell i}$ in the rest of the proof. Notice that from the definition of $\overline{f}_{\ell m}$, the following holds for each $i \in \{\ell,\dots,m\}$:

$$\arg\max_{\ell=1,\dots,i}\overline{f}_{\ell m} = \arg\max_{\ell=1,\dots,i}\overline{f}_{\ell i}.$$

Therefore, if $\arg\max_{\ell=1,\dots,m}\overline{f}_{\ell m} = t$ and $t \le i$, then $\arg\max_{\ell=1,\dots,i}\overline{f}_{\ell i} = t$. This can be equivalently written as

$$\overline{f}_{tm} = \max_{\ell=1,\dots,m}\overline{f}_{\ell m} \quad\Rightarrow\quad \overline{f}_{ti} = \max_{\ell=1,\dots,i}\overline{f}_{\ell i} \text{ for all } i = t,\dots,m. \tag{5.7}$$

The counterpart of (5.7) for the whole sum $\sum_{i=1}^{m}\max_{\ell=1,\dots,i}\overline{f}_{\ell i}$ is

$$\overline{f}_{tm} = \max_{\ell=1,\dots,m}\overline{f}_{\ell m} \quad\Rightarrow\quad \sum_{i=1}^{m}\max_{\ell=1,\dots,i}\overline{f}_{\ell i} = \sum_{i=1}^{t-1}\max_{\ell=1,\dots,i}\overline{f}_{\ell i} + \sum_{i=t}^{m}\overline{f}_{ti},$$

and the result follows by taking the maximum over all $t = 1,\dots,m$ because we do not know in advance which corresponds to $\arg\max_{\ell=1,\dots,m}\overline{f}_{\ell m}$. □

Using Lemma 5.2 we have, for each $2 \le m \le n$ and $0 \le \gamma \le \Gamma$:

$$\begin{aligned}
\beta(m,\gamma) &= \max_{\xi\in\Xi^\gamma_{[1m]}}\left\{\overline{f}^\beta(\xi_{[1m]})\right\}\\
&= \max_{\xi\in\Xi^\gamma_{[1m]}}\max_{t=1,\dots,m}\left\{\overline{f}^\beta(\xi_{[1(t-1)]}) + \sum_{i=t}^{m}\overline{f}_{ti}(\xi_{[ti]})\right\}\\
&= \max_{t=1,\dots,m}\max_{\delta=0,\dots,\gamma}\left\{\max_{\xi\in\Xi^\delta_{[1(t-1)]}}\overline{f}^\beta(\xi_{[1(t-1)]}) + \max_{\xi\in\Xi^{\gamma-\delta}_{[tm]}}\sum_{i=t}^{m}\overline{f}_{ti}(\xi_{[ti]})\right\}\\
&= \max_{t=1,\dots,m}\max_{\delta=0,\dots,\gamma}\left\{\beta(t-1,\delta) + \max_{\xi\in\Xi^{\gamma-\delta}_{[tm]}}\sum_{i=t}^{m}\overline{f}_{ti}(\xi_{[ti]})\right\}\\
&= \max_{t=1,\dots,m}\max_{\delta=0,\dots,\gamma}\left\{\beta(t-1,\delta) + \overline{F}(t,m,\gamma-\delta)\right\}, \qquad (5.8)
\end{aligned}$$

where $\overline{F}(t,m,\gamma-\delta) = \max_{\xi\in\Xi^{\gamma-\delta}_{[tm]}}\sum_{i=t}^{m}\overline{f}_{ti}(\xi_{[ti]})$. Furthermore, for $m = 1$ and each $\gamma \in \{0,\dots,\Gamma\}$, we have

$$\beta(1,\gamma) = \max_{\xi\in\Xi^\gamma_{[11]}}\overline{f}_{11}(\xi_{[11]}) = s_1[a_1 - b_1]^+ = 0. \tag{5.9}$$

Combining (5.8) and (5.9), we obtain a DPA to solve (**AP***).

We conclude the section by showing that (5.8) yields a pseudo-polynomial DPA. Notice that function

$$\bar{f}_{tm}(\xi_{[tm]}) = \sum_{i=t}^{m}\overline{f}_{ti}(\xi_{[ti]})$$

n	$\Gamma^{0.10}$	$\Gamma^{0.05}$	$\Gamma^{0.01}$
50	11	13	18
100	14	18	24
200	20	25	34

TABLE 1
Values of Γ^ϵ obtained by rounding up the values prescribed by the probabilistic bound from [14].

satisfies Assumption 2 for each $1 \leq t \leq m \leq n$. Hence, we can apply the DPA from Section 3 to compute $\overline{F}(t,m,\gamma) = \max_{\xi \in \Xi^\gamma_{[tm]}} \bar{f}_{tm}(\xi_{[tm]})$ for each $1 \leq t \leq m \leq n$ and $0 \leq \gamma \leq \Gamma$. Namely, let α_t be the table used to compute $\overline{F}(t,n,\Gamma)$ through the DPA from Section 3. We readily see that

$$\overline{F}(t,m,\gamma) = \max_{\phi=0,\ldots,\overline{\phi}} \alpha_t(m-t,\gamma,\phi),$$

for each $1 \leq t \leq m \leq n$ and $0 \leq \gamma \leq \Gamma$. Therefore, we can obtain all values in $\{\overline{F}(t,m,\gamma) : 1 \leq t \leq m \leq n, 0 \leq \gamma \leq \Gamma\}$ by applying the DPA from Section 3 to $\overline{F}(t,n,\Gamma)$ for each $1 \leq t \leq n$. This yields a computational time of $O(n^2\Gamma\overline{\phi})$ which is done in a pre-processing phase. Once all values of $\overline{F}$ have been computed, (5.8) can be solved in $O(n^2\Gamma^2)$. Since $\Gamma \leq \overline{\phi}$, we obtain the following worst-case complexity.

LEMMA 5.3. *Problem* (**AP***) *can be solved by a DPA in* $\mathcal{O}(n^2\Gamma\overline{\phi})$ *operations.*

We finish the section by noticing that constraints similar to time windows appear also in scheduling problem with release times and deadlines [7]. In the deterministic version of the problem, we consider a set $N = \{1,\ldots,n\}$ of tasks. For each task $i \in N$ we are given a release time a_i, which is the time at which task i can start being processed, a deadline b_i and a duration ξ_i. The objective of the problem is to find a feasible schedule of the tasks in a single machine such that the weighted sum of the deadline violations are minimized. In the robust approach, the durations of the tasks are uncertain and belongs to uncertainty set Ξ^Γ. Similarly to the discussion made for the traveling salesman problem, one can consider different versions of the problem depending on weather the release times and deadlines are soft or hard constraints. Then one readily sees that for each of these variants, the adversarial problem is similar to the aforementioned adversarial problems.

6. Computational experiments. We compare in this section our DPA and the classical MIP formulation for solving the lot-sizing problem with row-and-column generation algorithms.

6.1. Instances and details. We consider three numbers of periods: 50, 100, and 200. For each one of them we create four sets of instances: $S1$, $S2$, $S3$ and $S4$. For all sets we generate the storage cost for each period randomly and uniformly from an interval $[5,10]$. The difference between the sets lies in the backlog cost. For each $i \in \{1,2,3,4\}$, instances in Si are defined by backlog cost equal to i times the storage cost in each period. For all instances the nominal demand is generated randomly from interval $[50,100]$. Then, we consider five levels of deviations, ranging from 10% to 50% of the nominal demand. We round up the obtained deviations to ensure that they are integer. Finally, we also consider three different values for the budget of uncertainty Γ, motivated by the probabilistic bounds given in [14] and provided in Table 1. Namely, the value of Γ^ϵ is such that there is a probability of $1-\epsilon$ that the

n	10%		20%		30%		40%		50%	
	DPA	MIP	DPA	MIP	DPA	MIP	DPA	MIP	DPA	MIP
50	0.051	18.6	0.078	15.2	0.091	10.2	0.094	7.67	0.115	7.56
100	0.358	70.3	0.519	52.1	0.537	29	0.634	23.5	0.674	21.5
200	2.77	1,600	3.72	940	3.95	417	4.69	326	5.23	183

TABLE 2
Arithmetic means of the solution times.

real cost will be no higher than the optimal solution cost whenever ξ is composed of independent and identically distributed random variables.

Our experiments compare the DPA with the well-known MIP reformulation of (**AP**) (e.g. [15]) recalled below:

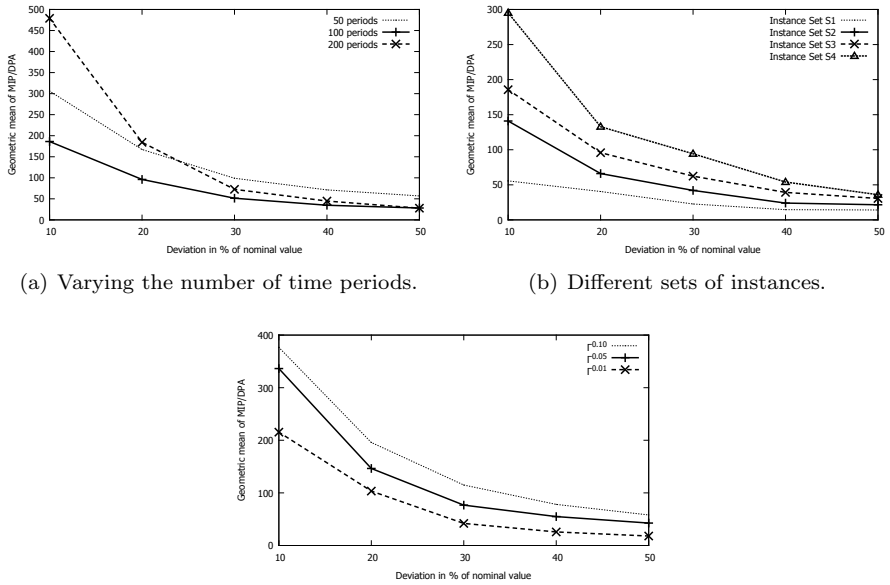
$$
\begin{aligned}
\max \quad & \sum_{i=1}^{n} \varphi_i, \\
\text{s.t.} \quad & \varphi_i \le g_i^0 + g_i^1 y_i + M_i u_i, \qquad \forall i = 1, \dots, n, \\
& \varphi_i \le h_i^0 + h_i^1 y_i + M_i (1 - u_i), \qquad \forall i = 1, \dots, n, \\
& y_i = \sum_{i=1}^{i} \overline{\xi}_i + \widehat{\xi}_i \eta_i, \qquad \forall i = 1, \dots, n, \\
& \sum_{i=1}^{n} \eta_i \le \Gamma, \qquad \forall i = 1, \dots, n, \\
& \eta \in \{0,1\}^n,\ u \in \{0,1\}^n,\ y \ge 0.
\end{aligned}
$$

For each $i = 1, \dots, n$, variables φ_i and y_i represent the value of functions $f_i(y_i(\xi))$ and $y_i(\xi)$, respectively. Then, for each $i = 1, \dots, n$ variable u_i is equal to 0 iff $g_i(y_i(\xi))$ is larger than $h_i(y_i(\xi))$ and M_i is a large predefined value that cannot be smaller than $|f_i(y_i(\xi))|$ for every $\xi \in \Xi^\Gamma$.

The DPA was coded in C++ and compiled in a GNU G++ 4.5 compiler. The MIP formulation was implemented in C++ using Cplex Concert Technology 12.5 [19]. The numerical experiments were carried out in an Intel(R) Core(TM) i7 CPU M60, 2.6Hz 4GB Ram machine.

6.2. Results. We provide in Figure 1 geometric means of the solution time of MIP divided by the solution time of DPA. The standard deviations of the solution times are illustrated on Figure 2 for both approaches. We present on Table 2 the arithmetic means of the solution times for the different number of time periods and levels of deviations.

Figure 1 shows that DPA clearly outperforms MIP, with means ranging up to 475 when $n = 200$ and the deviation is 10%. The different parameters strongly impact the respective solution times. First, we see on the charts from Figure 1 that, as expected from its theoretical complexity, higher levels of deviation slow down DPA. The number of time periods strongly affect both approaches, see Table 2. When the deviation is small, Figure 1(a) shows that the ratio between MIP and DPA increases with the value of n. In contrast, high levels of deviations tend to reduce the ratio between MIP and DPA. Figure 1(b) depicts the sensitivity of the ratio between the storage and backlog costs. Our (unreported) results show that MIP is highly sensitive to the ratio, while DPA is not affected at all, explaining the results of Figure 1(b).

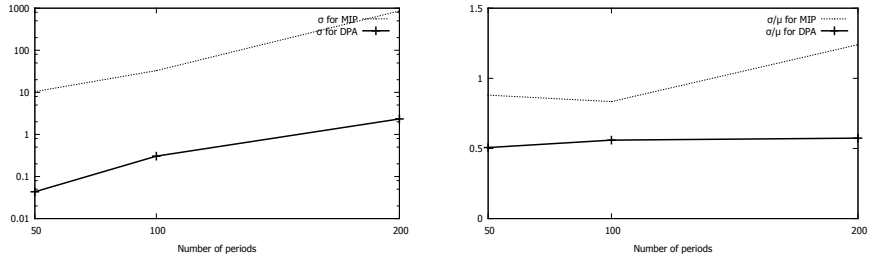


(a) Varying the number of time periods.

(b) Different sets of instances.

(c) Different values of Γ.

FIG. 1. *Geometric mean of (time MIP)/(time DPA).*



(a) Standard deviation represented in logarithmic scale.

(b) Standard deviation divided by the arithmetic mean (μ).

FIG. 2. *Standard deviation of the solution times σ when varying the number of time periods.*

Finally, Figure 1(c) shows that higher values of Γ yields smaller ratios on average. Here, both approaches are strongly affected by the value of Γ, and the figure shows than DPA is more affected than MIP since the ratio decreases significantly when Γ rises.

The variations of the solution times are represented on Figure 2 for DPA and MIP, through the standard deviation (σ). Figure 2(a) presents these standard deviations in a logarithmic scale, which shows that the solution times of MIP vary between 2 and 3 order of magnitude more than the solution times of DPA. Figure 2(b) shows these standard deviations in a relative scale, dividing them by the associated arithmetic means. The figure shows that in a relative scale, DPA varies roughly twice as much as MIP.

To conclude, our simple experiments show that DPA can be orders of magnitude

faster than MIP, especially when the deviation level is low. Moreover, the absolute variability of MIP is much higher than the one of DPA, some instances being particularly hard to solve for the former. Notice also that we compared a simplistic implementation of DPA to the well-engineered MIP solver from CPLEX. It is likely that rules to eliminate dominated states would further improve our results, but this is out of the scope of the current paper.

7. Acknowledgements. We thank Artur Alves Pessoa for fruitful discussions about approximation algorithms and FPTAS.We are also grateful to the referees for their constructive comments that have helped improving the presentation of the paper and for suggesting Proposition A.1.

Appendix. $\mathcal{NP}$-hardness results.

PROPOSITION A.1. *Let f be a convex function. Optimization problem $\max_{\xi\in\Xi^\Gamma} f(\xi)$ is $\mathcal{APX}$-hard.*

Proof. Consider the independent set problem on an undirected graph $G=(V,E)$ with n nodes where the goal is to decide whether there is an independent set of size at least $k<n$. Then, the Ξ^Γ be the uncertainty set characterized by $\overline{\xi}=0$, $\widehat{\xi}_i=1$ for each $i=1,\ldots,n$ and $\Gamma=k$, and f be the piecewise linear convex function defined by $f(\xi)=\max_z\left\{\sum_{i=1}^n \xi_i z_i : z_i+z_j\le 1, \{i,j\}\in E, z\ge 0\right\}$. For any $\xi\in\Xi^\Gamma$, $f(\xi)\le k$. Moreover, G has an independent set of size at least k if and only if $\max_{\xi\in\Xi^\Gamma} f(\xi)=k$. Therefore, the problem of maximizing f over Ξ^Γ is $\mathcal{NP}$-hard.

To show that the problem is $\mathcal{APX}$-hard, consider the independent set problem on 3-regular (cubic) graphs, which is known to be $\mathcal{APX}$-hard [5]. Hence, there is a constant $\alpha>0$ such that it is $\mathcal{NP}$-hard to distinguish whether a cubic graph $G=(V,E)$ has an independent set of size at least k or at most $(1-\alpha)k$. If this is the case that G has an independent set of size at least k, then $\max_{\xi\in\Xi^\Gamma} f(\xi)=k$. Otherwise, if the independent set of G has size at most $(1-\alpha)k$, then one can show that $\max_{\xi\in\Xi^\Gamma} f(\xi)\le (1-\alpha)k+\frac{3\alpha k}{4}=\left(1-\frac{\alpha}{4}\right)h$. This implies that maximizing f over Ξ^Γ is $\mathcal{APX}$-hard. □

PROPOSITION A.2. *Let $\widetilde{\Xi}$ be an uncertainty polytope having a compact description. Optimization problem $\max_{\xi\in\widetilde{\Xi}} f(\xi^\Sigma)$ is $\mathcal{NP}$-hard.*

Proof. We consider in this proof function $\hat{f}(\xi)=\sum_{i=1}^n \left|\sum_{j=1}^i \xi_j\right|$, obtained from (1.2) by choosing $g_i(\xi_i^\Sigma)=\xi_i^\Sigma$ and $h_i(\xi_i^\Sigma)=-\xi_i^\Sigma$ for each $i=1,\ldots,n$ where the dependency on z is omitted for the sake of simplicity. We prove first that problem

$$\max_{\xi\in\widetilde{\Xi}} \hat{f}(\xi) \tag{A.1}$$

has the same optimal solution cost as problem

$$\max_{\xi\in\widetilde{\Xi}'} f^*(\xi), \tag{A.2}$$

with $f^*(\xi)=\sum_{i=1}^n |\xi_i|$ and $\widetilde{\Xi}'$ a linear transformation of $\widetilde{\Xi}$. To see this, consider invertible linear transformation $\alpha(\xi)=(\xi_1,\xi_1+\xi_2,\xi_1+\xi_2+\xi_3,\ldots,\xi_1+\xi_2+\ldots+\xi_n)$.

Then, $\max_{\xi\in\widetilde{\Xi}} \hat{f}(\xi) = \max_{\xi\in\widetilde{\Xi}} \sum_{i=1}^{n} \left|\sum_{j=1}^{i} \xi_j\right| = \max_{\xi\in\widetilde{\Xi}} \sum_{i=1}^{n} |\alpha_i(\xi)| = \max_{\xi\in\widetilde{\Xi}} f^*(\alpha(\xi)) = \max_{\xi\in\alpha(\widetilde{\Xi})} f^*(\xi)$, and the equality holds for $\widetilde{\Xi}' = \alpha(\widetilde{\Xi})$. Notice that $\widetilde{\Xi}'$ can be described by the same number of inequalities as $\widetilde{\Xi}$ and these can be computed in polynomial time. Therefore, the $\mathcal{NP}$-hardness of (A.1) follows from the $\mathcal{NP}$-hardness of (A.2), which was proven in [25, Lemma 3.2]. □

PROPOSITION A.3. *Let $\tilde{f}_i$ be a non-negative function for each $i = 1, \ldots, n$ and $\tilde{f}(\xi^\Sigma) = \sum_{i=1}^{n} \tilde{f}_i(\xi_i^\Sigma)$, with $\tilde{f}$ being a positive function. Optimization problem $\max_{\xi\in\mathrm{ext}(\Xi^\Gamma)} \tilde{f}(\xi^\Sigma)$ is $\mathcal{NP}$-hard.*

Proof. The decision problem associated with the optimization problem takes the following form: given $\overline{\xi}, \widehat{\xi}$, and $\Gamma > 0$, non-negative functions $\tilde{f}_i(\xi_i^\Sigma)$, $i = 1, \ldots, n$, and $A \in \mathbb{R}$, is there a $\xi \in \Xi^\Gamma$ such that $\sum_{i=1}^{n} \tilde{f}_i(\xi_i^\Sigma) \geq A$? We show below that the partition problem can be reduced to the above decision problem. Recall that in the partition problem we are given m positive integers $a_i, i \in M = \{1, \ldots, m\}$ and wish to determine whether there exists a partition $(S, M \setminus S)$ of M such that $\sum_{i\in S} a_i = \sum_{i\in M\setminus S} a_i = \sum_{i\in M} a_i/2$.

For the reduction consider $n = m + 2$, $\overline{\xi}_i = 0, i \in N, \widehat{\xi}_i = a_i, i \in M, \widehat{\xi}_{m+1} = \widehat{\xi}_{m+2} = 0$. Let $A = \sum_{i=1}^{n} a_i$ and define $\tilde{f}_i(\xi_i^\Sigma) = 0, i \in M$, and

$$\tilde{f}_{m+1}(\xi_{m+1}^\Sigma) = \min\left\{\sum_{i=1}^{n} \xi_i, \frac{A}{2}\right\}, \ \tilde{f}_{m+2}(\xi_{m+2}^\Sigma) = \min\left\{A - \sum_{i=1}^{n} \xi_i, \frac{A}{2}\right\}.$$

Hence, from the definition of $\tilde{f}_i$ for $i = 1, \ldots, m+2$, it follows directly that $\sum_{i=1}^{n} f_i(\xi_i^\Sigma) \leq A$. Moreover, the equality holds if and only if $\tilde{f}_{m+1}(\xi_{m+1}^\Sigma) = \tilde{f}_{m+2}(\xi_{m+2}^\Sigma) = A/2$ which is equivalent to find a partition of N with $\sum_{i\in S} a_i = \sum_{i\in M\setminus S} a_i = \frac{A}{2}$ where $S = \{j \in \{1, \ldots, n\} : \xi_j = \widehat{\xi}_j\}$. □

REFERENCES

[1] A. AGRA, M. CHRISTIANSEN, AND A. DELGADO, *Mixed integer formulations for a short sea fuel oil distribution problem*, Transportation Science, 47 (2013), pp. 108–124.

[2] A. AGRA, M. CHRISTIANSEN, R. M. V. FIGUEIREDO, L. M. HVATTUM, M. POSS, AND CRISTINA REQUEJO, *Layered formulation for the robust vehicle routing problem with time windows*, in Combinatorial Optimization - Second International Symposium, ISCO 2012, Athens, Greece, April 19-21, 2012, Revised Selected Papers, 2012, pp. 249–260.

[3] A. AGRA, M. CHRISTIANSEN, R. M. V. FIGUEIREDO, L. M. HVATTUM, M. POSS, AND C. REQUEJO, *The robust vehicle routing problem with time windows*, Computers & OR, 40 (2013), pp. 856–866.

[4] A. AGRA AND C. REQUEJO, *The linking set problem: a polynomial special case of the multiple-choice knapsack problem*, Journal of Mathematical Sciences, 161 (2009), pp. 919–929.

[5] P. ALIMONTI AND V. KANN, *Some apx-completeness results for cubic graphs*, Theoretical Computer Science, 237 (2000), pp. 123 – 134.

[6] A. ARDESTANI-JAAFARI AND E. DELAGE, *Robust optimization of sums of piecewise linear functions with application to inventory problems*. Working paper.

[7] P. BAPTISTE, C. LE PAPE, AND W. NUIJTEN, *Constraint-based scheduling: applying constraint programming to scheduling problems*, vol. 39, Springer, 2001.

[8] A. Ben-Tal, D. den Hertog, and J.-P. Vial, *Deriving robust counterparts of nonlinear uncertain inequalities*, Mathematical Programming, (2014), pp. 1–35.

[9] A. Ben-Tal, L. El Ghaoui, and A.S. Nemirovski, *Robust optimization*, Princeton University Press, 2009.

[10] A. Ben-Tal, B. Golany, A. Nemirovski, and J.-P. Vial, *Retailer-supplier flexible commitments contracts: A robust optimization approach*, Manufacturing & Service Operations Management, 7 (2005), pp. 248–271.

[11] A. Ben-Tal and A. Nemirovski, *Robust convex optimization*, Mathematics of Operations Research, 23 (1998), pp. 769–805.

[12] D. Bertsimas, D.B. Brown, and C. Caramanis, *Theory and applications of robust optimization*, SIAM Review, 53 (2011), pp. 464–501.

[13] D. Bertsimas and M. Sim, *Robust discrete optimization and network flows*, Math. Program., 98 (2003), pp. 49–71.

[14] D. Bertsimas and M. Sim, *The price of robustness*, Operations Research, 52 (2004), pp. 35–53.

[15] D. Bienstock and N. Özbay, *Computing robust basestock levels*, Discrete Optimization, 5 (2008), pp. 389–414.

[16] P. Brucker, A. Drexl, R. Möhring, K. Neumann, and E. Pesch, *Resource-constrained project scheduling: Notation, classification, models, and methods*, European journal of operational research, 112 (1999), pp. 3–41.

[17] A. M. Campbell and B. W. Thomas, *Probabilistic traveling salesman problem with deadlines*, Transportation Science, 42 (2008), pp. 1–21.

[18] M. Chardy and O. Klopfenstein, *Handling uncertainties in vehicle routing problems through data preprocessing*, in proceedings of the Third International Conference on Information Systems, Logistics and Supply Chain, 2010.

[19] CPLEX, *IBM ILOG CPLEX 12.5 Reference Manual*, 2013.

[20] M. Fischetti and M. Monaci, *Cutting plane versus compact formulations for uncertain (integer) linear programs*, Mathematical Programming Computation, 4 (2012), pp. 239–273.

[21] V. Gabrel, C. Murat, and A. Thiele, *Recent advances in robust optimization: An overview*, European Journal of Operational Research, 235 (2014), pp. 471–483.

[22] B. L. Gorissen, A. Ben-Tal, H. Blanc, and D. Den Hertog, *A new method for deriving robust and globalized robust solutions of uncertain linear conic optimization problems having general convex uncertainty sets*, Operations Research, (2014). Online First.

[23] B. L. Gorissen and D. den Hertog, *Robust counterparts of inequalities containing sums of maxima of linear functions*, European Journal of Operational Research, 227 (2013), pp. 30 – 43.

[24] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, Springer-Verlag, 1993.

[25] E. Guslitser, *Uncertainty immunized solutions in linear programming*, master's thesis, TECHNION-ISRAEL INSTITUTE OF TECHNOLOGY, 2002.

[26] D. A. Iancu, M. Sharma, and M. Sviridenko, *Supermodularity and affine policies in dynamic robust optimization*, Operations Research, 61 (2013), pp. 941–956.

[27] O. H. Ibarra and C. E. Kim, *Fast approximation algorithms for the knapsack and sum of subset problems*, Journal of the ACM (JACM), 22 (1975), pp. 463–468.

[28] O. Klopfenstein and D. Nace, *A robust approach to the chance-constrained knapsack problem*, Oper. Res. Lett., 36 (2008), pp. 628–632.

[29] M. Monaci, U. Pferschy, and P. Serafini, *Exact solution of the robust knapsack problem*, Computers & OR, 40 (2013), pp. 2625–2631.

[30] A. A. Pessoa and M. Poss, *Robust network design with uncertain outsourcing cost*, INFORMS Journal on Computing, 27 (2015), pp. 507–524.

[31] A. A. Pessoa, L. Di Puglia Pugliese, F. Guerriero, and M. Poss, *Robust constrained shortest path problems under budgeted uncertainty*, Networks, 66 (2015), pp. 98–111.

[32] M. Poss, *Robust combinatorial optimization with variable cost uncertainty*, European Journal of Operational Research, 237 (2014), pp. 836–845.

[33] O. Solyali, J.-F. Cordeau, and G. Laporte, *Robust inventory routing under demand uncertainty*, Transportation Science, 46 (2012), pp. 327–340.

[34] Z. M. Teksan and J. Geunes, *A polynomial time algorithm for convex cost lot-sizing problems*, Operations Research Letters, 43 (2015), pp. 359–364.

[35] B. Zeng and L. Zhao, *Solving two-stage robust optimization problems by a constraint-and-column generation method*, Operations Research Letters, 41 (2013), pp. 457–461.