Robust combinatorial optimization problems with knapsack constraints under interdiction uncertainty

Alejandro Crema

Escuela de Computación, Facultad de Ciencias, Universidad Central de Venezuela.

Contributing authors: alejandro.crema@ciens.ucv.ve;

Abstract

We present an algorithm for finding near-optimal solutions to robust combinatorial optimization problems with knapsack constraints under interdiction uncertainty. We incorporate a heuristic for generating feasible solutions in a standard row generation approach. Experimental results are presented for set covering, simple plant location, and min-knapsack problems under a discrete-budgeted interdiction uncertainty set introduced in this work.

Keywords: Combinatorial optimization, Robust optimization, Interdiction uncertainty, Bulk-robust optimization

1 Introduction

To simplify the exposition we prefer to present directly the notation in section 1.1. In section 1.2 we present the Robust combinatorial optimization problem under interdiction uncertainty [1] when the problem is defined with knapsack constraints and in section 1.3 we present the Bulk-robust combinatorial optimization problem ([2], [3]) which results for the mentioned special case an equivalent problem. In section 1.4 we present a summary of our contribution and in section 1.5 we present some remarks and the organization of the paper.

1.1 Notation

1. if U is an optimization problem then: (i) $F(U)$ is its feasible solution set and $v(U)$ is its optimal value with the usual convention $v(U) = +\infty(-\infty)$ for the minimization (maximization) case when an optimal solution does not exist and (ii) let ϕ the cost function for the minimization case and let $\epsilon \geq 0$ then $\mathbf{y}$ is an ϵ-optimal solution for U if $\mathbf{y} \in F(U)$ and $\frac{\phi(\mathbf{y})-v(U)}{\phi(\mathbf{y})} \leq \epsilon$

2. if $n \geq 1$ then $[n] = \{1, \cdots, n\}$

3. let $K \geq 1$, let Y a non-empty set and let $\mathbf{x}^k \in Y$ for all $k \in [K]$ then $\{\mathbf{x}^k\}_1^K = \{\mathbf{x}^1, \cdots, \mathbf{x}^K\} \subseteq Y$

4. let $n \geq 1$, if $\mathbf{x} \in \{0,1\}^n$ then $supp(\mathbf{x}) = \{j : \mathbf{x}_j = 1,\ j \in [n]\}$

5. $\mathbf{0}(\mathbf{1})$ denotes a vector with all coordinates equal to $0(1)$ and the dimensions will be clear each time in the context

6. let $r \geq 1$, if $\mathbf{y}, \mathbf{w} \in \{0,1\}^r$ then $\mathbf{y} \otimes \mathbf{w}$ is defined as follows: $\mathbf{y} \otimes \mathbf{w}_j = \mathbf{y}_j \mathbf{w}_j\ \forall j \in [r]$

7. let $m, n \geq 1$, let $A \in \mathbb{R}^{m \times n}$ and let $\mathbf{y}, \mathbf{w} \in \{0,1\}^n$ then $Ay \otimes w = Az$ with $z = y \otimes w$

1.2 Robust combinatorial optimization problems with knapsack constraints under interdiction uncertainty

Let $m, n \geq 1$, let $A \in \mathbb{Z}_+^{m \times n}$, let $\mathbf{b} \in \mathbb{Z}_+^m$ with $\mathbf{b} > \mathbf{0}$, and let $\phi : \{0,1\}^n \longrightarrow \mathbb{R}_+$. The *nominal problem* to be considered is a problem in $\mathbf{x}$ defined as follows:

$$\begin{aligned} \min\ & \phi(\mathbf{x}) && \mathcal{P} \\ s.t.\ & A\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \in \{0,1\}^n \end{aligned}$$

Several important combinatorial problems are defined as $\mathcal{P}$, for example: the set covering (SC) problem, the quadratic set covering problem, the min-knapsack (mK) and the multidimensional min-knapsack problem with integer weights, the quadratic min-knapsack and the quadratic multidimensional min-knapsack problem with integer weights, the selection problem and the simple plant location (SPL) problem.

Let $S \subseteq \{0,1\}^n$ be the implicitly defined *interdiction uncertainty set*. If $\mathbf{s} \in S$ let $\mathcal{X}(\mathbf{s}) = \{\mathbf{x} \in \{0,1\}^n : A\mathbf{x} \otimes (\mathbf{1} - \mathbf{s}) \geq \mathbf{b}\}$. Let $S_f = \{\mathbf{s} \in S : \mathcal{X}(\mathbf{s}) \neq \emptyset\}$. If $\mathbf{s} \in S$ then $\mathbf{s}$ is an *scenario*. If $\mathbf{s} \in S$ and $\mathcal{X}(\mathbf{s}) \neq \emptyset$ then $\mathbf{s}$ is a *feasible scenario*. If $\mathbf{x} \in \{0,1\}^n$ and $\mathbf{s} \in S_f$ we say that $\mathbf{x}$ *covers* $\mathbf{s}$ if and only if $\mathbf{x} \in \mathcal{X}(\mathbf{s})$.

The *Robust combinatorial optimization problem with knapsack constraints under interdiction uncertainty* is a problem in $\mathbf{x}$ defined as follows:

$$\begin{array}{lll} \min & \phi(\mathbf{x}) & \mathcal{I} \\ s.t. & A\,\mathbf{x} \otimes (1-\mathbf{s}) \geq \mathbf{b} \quad \forall \mathbf{s} \in S_f & \\ & \mathbf{x} \in \{0,1\}^n & \end{array}$$

Let us suppose that $F(\mathcal{P}) \neq \emptyset$. Let us suppose that $\mathbf{0} \in S$. Since $F(\mathcal{P}) \neq \emptyset$ then (i) $\mathcal{X}(\mathbf{0}) \neq \emptyset$, (ii) $\mathbf{0} \in S_f$ and (iii) $S_f \neq \emptyset$. Note that $\mathbf{1} \in F(\mathcal{I})$ and then there is an optimal solution. Note that $\mathbf{x} \in F(\mathcal{I})$ if and only if $\mathbf{x}$ covers $\mathbf{s}$ for all $\mathbf{s} \in S_f$.

Let $\mathbf{w} \in \mathbb{Z}_+^n$, $\mathbf{c} \in \mathbb{R}_+$ and $W > 0$. If S is defined by $S = \{\mathbf{s} \in \{0,1\}^n : \mathbf{w}^t\mathbf{s} \leq W\}$ and $\phi(\mathbf{x}) = \mathbf{c}^t\mathbf{x}$ then there is a compact formulation for $\mathcal{I}$ in the case where the constraints are of cardinality ([1]).

Let us suppose that a compact formulation for $\mathcal{I}$ either does not exist or we we do not known it.

A standard row generation approach to solve $\mathcal{I}$, mentioned in [4] as the scenarios-dynamic-approach, is to solve a sequence of relaxations defined with a subset of S_f until a solution $\mathbf{x}$ is generated with $\mathbf{x} \in F(\mathcal{I})$. If the solution $\mathbf{x}$ obtained by solving a relaxation does not belong to $F(\mathcal{I})$, the rows of some feasible scenario not previously considered and not covered by $\mathbf{x}$ are added to define a new relaxation.

If $\underline{S} = \{\mathbf{s}^k\}_1^K \subseteq S_f$ the corresponding relaxation $\mathcal{I}(\underline{S})$ is a problem in $\mathbf{x}$ defined as follows:

$$\begin{array}{lll} \min & \phi(\mathbf{x}) & \mathcal{I}(\underline{S}) \\ s.t. & A\,\mathbf{x} \otimes (1-\mathbf{s}^k) \geq b \qquad \forall \mathbf{k} \in [K] & \\ & \mathbf{x} \in \{0,1\}^n & \end{array}$$

The standard approach outlined has a crucial drawback: there is a lack of upper bounds that would allow to stop the algorithm execution with ϵ-optimal solutions.

1.3 Bulk-robust combinatorial optimization

Let $\Omega = \{\mathbf{x} \in \{0,1\}^n : \forall \mathbf{s} \in S_f \; \exists\, \mathbf{y}(\mathbf{s}) \in \{0,1\}^n$ such that $A\mathbf{y}(\mathbf{s}) \geq \mathbf{b}$, $\mathbf{y}(\mathbf{s}) \leq (\mathbf{1}-\mathbf{s})$ and $\mathbf{y}(\mathbf{s}) \leq \mathbf{x}\}$.

The Bulk-robust combinatorial optimization problem ([2], [3]) with knapsack constraints is a problem in $\mathbf{x}$ defined as follows :

$$\min \; \phi(\mathbf{x}) \;\; s.t. \;\; \mathbf{x} \in \Omega \qquad \mathcal{B}u$$

Remark 1. *$\mathcal{I}$ and $\mathcal{B}u$ are equivalent problems.*

Proof: Let $\mathbf{x} \in F(\mathcal{I})$ and let $\mathbf{y}(\mathbf{s}) = \mathbf{x} \otimes (\mathbf{1} - \mathbf{s})$ for all $\mathbf{s} \in S_f$ then $\mathbf{y}(\mathbf{s}) \in \{0,1\}^n$, $A\mathbf{y}(\mathbf{s}) \geq \mathbf{b}$, $\mathbf{y}(\mathbf{s}) \leq (\mathbf{1} - \mathbf{s})$ and $\mathbf{y}(\mathbf{s}) \leq \mathbf{x}$, therefore $\mathbf{x} \in \Omega$. Let $\mathbf{x} \in \Omega$ and let $\mathbf{s} \in S_f$ then there existe $\mathbf{y}(\mathbf{s}) \in \{0,1\}^n$ such that $A\mathbf{y}(\mathbf{s}) \geq \mathbf{b}$, $\mathbf{y}(\mathbf{s}) \leq (\mathbf{1} - \mathbf{s})$ and $\mathbf{y}(\mathbf{s}) \leq \mathbf{x}$, therefore $A\mathbf{x} \otimes (\mathbf{1} - \mathbf{s}) \geq A\mathbf{x} \otimes \mathbf{y}(\mathbf{s}) \geq \mathbf{b}$ and $\mathbf{x} \in F(\mathcal{I})$. Therefore, $\mathcal{I}$ and $\mathcal{B}u$ are equivalent problems □

1.4 Our contribution

We will show that we can overcome the drawback of the standard row generation approach. Some points to note are the following:

- we will present a greedy heuristic to generate: (i) a solution in $F(\mathcal{I})$ starting from $\mathbf{x}$ not in $F(\mathcal{I})$ with which obtaining of ϵ-optimal solutions becomes possible and (ii) a set of feasible scenarios not previously considered such that $\mathbf{x}$ does not cover them

- an ϵ-optimal algorithm to solve $\mathcal{I}$ with a row-and-bound generation approach will be presented

- we present a new non-uniform interdiction uncertainty sets resulting from applying the fundamental idea of an uniform budgeted interdiction uncertainty set to a set of explicitly defined basic binary scenarios

- we will present experimental results for problems whose nominal version corresponds to set covering, simple plant location and min-knapsack problems.

1.5 Remarks and paper organization

We will now present some remarks to clarify the context and the organization of the paper:

- in order to simplify the exposition, the auxiliary problems, the greedy heuristic, and the resulting ϵ-optimal algorithm will be presented assuming that the sequence of auxiliary problems and relaxations used are solved exactly. Subsequently, appropriate remarks will be made when considering tolerances if necessary

- the study of the computational complexity of $\mathcal{I}$ and $\mathcal{B}u$ is beyond the scope of the paper. Results on complexity for several kind of constraints can be found in [1],[2],[3],[5],[6],[7] and [8]

- the paper is organized as follows: section 2 presents the theoretical results, auxiliary problems and greedy heuristic that allow us to define an ϵ-optimal algorithm, which is presented in the same section. Section 3 presents the non-uniform interdiction uncertainty sets. Section 4 presents the experimental results and finally section 5 presents the conclusions and possible extensions of the work.

2 Theoretical results and algorithms

Remark 2. *Let* $\mathbf{s} \in S$ *then:* $\mathbf{s} \in S_f$ *if and only if* $A(\mathbf{1}-\mathbf{s}) \geq \mathbf{b}$

Proof: If $\mathbf{s} \in S_f$ then there exists $\mathbf{x}(\mathbf{s}) \in \{0,1\}^n$ such that $\mathbf{b} \leq A\mathbf{x}(\mathbf{s}) \leq A(\mathbf{1}-\mathbf{s})$. If $A(\mathbf{1}-\mathbf{s}) \geq \mathbf{b}$ we have that $\mathbf{1}-\mathbf{s} \in \mathcal{X}(\mathbf{s})$ and then $\mathbf{s} \in S_f$ $\square$

2.1 Adversarial problem

Next we present the *adversarial problem* to verify whether a solution is feasible and, if not, to generate a feasible scenario that is not covered.

Lemma 1. *Let* $\mathbf{x} \in \{0,1\}^n$*. Let* $Q(\mathbf{x})$ *be a problem in* $(\mathbf{z},\mathbf{s})$ *defined as follows:*

$$\max \sum_{i\in[m]} \mathbf{z}_i \qquad Q(\mathbf{x})$$

$$s.t. \sum_{j\in[n]} A_{ij}\mathbf{x}_j(1-\mathbf{s}_j) \leq \mathbf{z}_i(b_i-1) + (1-\mathbf{z}_i)\sum_{j\in[n]} A_{ij}\mathbf{x}_j \qquad \forall i \in [m] \qquad (*_1)$$

$$A(\mathbf{1}-\mathbf{s}) \geq \mathbf{b} \qquad (*_2)$$

$$\mathbf{z} \in \{0,1\}^m,\ \mathbf{s} \in S$$

then:

1. $F(Q(\mathbf{x})) \neq \emptyset$

2. *If* $(\mathbf{z},\mathbf{s}) \in F(Q(\mathbf{x}))$ *then* $\mathbf{s} \in S_f$

3. $v(Q(\mathbf{x})) \geq 1$ *if and only if* $\mathbf{x} \notin F(\mathcal{I})$

4. *Let* $\underline{S} = \{\mathbf{s}^k\}_1^K \subseteq S_f$*, let* $\mathbf{x} \in F(\mathcal{I}(\underline{S}))$ *and let* $(\mathbf{z},\mathbf{s})$ *be an optimal solution for* $Q(\mathbf{x})$*. If* $v(Q(\mathbf{x})) \geq 1$ *then* $\mathbf{s} \notin \{\mathbf{s}^k\}_1^K$ *and* $\mathbf{x}$ *does not cover* $\mathbf{s}$*.*

Proof:

1. If $\mathbf{z} = \mathbf{0}$ then constraints in $(*_1)$ are satisfied for all $\mathbf{s} \in S$ and if $\mathbf{s} = \mathbf{0}$ then constraints in $(*_2)$ are satisfied, therefore $(\mathbf{0},\mathbf{0}) \in F(Q(\mathbf{x}))$

2. If $(\mathbf{z},\mathbf{s}) \in F(Q(\mathbf{x}))$ then $\mathbf{s} \in S$ with $A(\mathbf{1}-\mathbf{s}) \geq \mathbf{b}$ and, from remark 2, $\mathbf{s} \in S_f$

3.(a) Let $(\mathbf{z},\mathbf{s})$ be an optimal solution with $v(Q(\mathbf{x})) \geq 1$, then we have that $\mathbf{s} \in S_f$ and there exists $i \in [m]$ such that $\mathbf{z}_i = 1$ and then $\sum_{j\in[n]} A_{ij}\mathbf{x}_j(1-\mathbf{s}_j) \leq b_i - 1$. Therefore $\mathbf{x}$ does not cover $\mathbf{s}$ and $\mathbf{x} \notin F(\mathcal{I})$

(b) Let us suppose that $\mathbf{x} \notin F(\mathcal{I})$ then there exists $\mathbf{s} \in S_f$ such that $\mathbf{x}$ does not cover it. Let $H(\mathbf{x},\mathbf{s}) = \{i \in [m] : \sum_{j\in[n]} A_{ij}\mathbf{x}_j(1-\mathbf{s}_j) \leq b_i - 1\}$. Let $\mathbf{z}_i = 1$ if

and only if $i \in H(\mathbf{x}, \mathbf{s})$. In that case we have that $(\mathbf{z}, \mathbf{s}) \in F(Q(\mathbf{x}))$, therefore $v(Q(\mathbf{x})) \geq 1$

4. Let $(\mathbf{z}, \mathbf{s})$ be an optimal solution with $v(Q(\mathbf{x})) \geq 1$, since $v(Q(\mathbf{x})) \geq 1$ then there exists $i \in [m]$ such that $\mathbf{z}_i = 1$ and in that case $\sum_{j \in [n]} A_{ij}\mathbf{x}_j(1 - \mathbf{s}_j) \leq b_i - 1$ and $\mathbf{x}$ does not cover $\mathbf{s}$. Since $\mathbf{x} \in F(\mathcal{I}(\underline{S}))$ we have that $\mathbf{s} \notin \{\mathbf{s}^k\}_1^K$ $\square$

2.2 Scenarios unfeasibility problem

Next we present a problem to verify if there are non-feasible scenarios.

Lemma 2. *Let R be a problem in $(\mathbf{z}, \mathbf{s})$ defined as follows:*

$$
\begin{aligned}
\max \quad & \sum_{i \in [m]} \mathbf{z}_i & & R \\
s.t. \quad & \sum_{j \in [n]} A_{ij}(1 - \mathbf{s}_j) \leq \mathbf{z}_i(b_i - 1) + (1 - \mathbf{z}_i) \sum_{j \in [n]} A_{ij} & \forall i \in [m] & \\
& \mathbf{z} \in \{0, 1\}^m, \ \mathbf{s} \in S &
\end{aligned}
$$

then:

1. $F(R) \neq \emptyset$

2. $v(R) \geq 1$ *if and only if* $S - S_f \neq \emptyset$

Proof:

1. $(\mathbf{0}, \mathbf{0}) \in F(R)$

2.(a) Let $(\mathbf{z}, \mathbf{s})$ be an optimal solution with $v(R) \geq 1$ then there exists $i \in [m]$ such that $\mathbf{z}_i = 1$, therefore $\sum_{j \in [n]} A_{ij}(1 - \mathbf{s}_j) \leq b_i - 1$ and from remark 2 we have that $\mathbf{s} \notin S_f$

(b) Let us suppose that $S - S_f \neq \emptyset$ and let $\mathbf{s} \in S - S_f$. Let $H(\mathbf{s}) = \{i \in [m] : \sum_{j \in [n]} A_{ij}(1 - \mathbf{s}_j) \leq b_i - 1\}$. Since $\mathbf{s} \notin S_f$ we have that $H(\mathbf{s}) \neq \emptyset$. Let $\mathbf{z}_i = 1$ if and only if $i \in H(\mathbf{s})$ then $(\mathbf{z}, \mathbf{s}) \in F(R)$, therefore $v(R) \geq 1$ $\square$

2.3 Greedy heuristic

Next we present a greedy heuristic to generate feasible solutions and non-covered unconsidered feasible scenarios.

Let $\underline{\mathbf{x}} \in \{0, 1\}^n$ with $\underline{\mathbf{x}} \notin F(\mathcal{I})$. What is done in the heuristic is the following: find $\mathbf{s} \in S_f$ such that $\mathbf{s}$ is not covered by $\underline{\mathbf{x}}$, add indexes to $supp(\underline{\mathbf{x}})$ to cover the new scenario at minimum cost, update $\underline{\mathbf{x}}$ and repeat the procedure until $\underline{\mathbf{x}} \in F(\mathcal{I})$.

Let $\underline{\mathbf{x}} \in \{0,1\}^n$ and let $\mathbf{s} \in S_f$ such that $\mathbf{s}$ is not covered by $\underline{\mathbf{x}}$. Let $I(\mathbf{s}, \underline{\mathbf{x}})$ be a problem in $\mathbf{x}$ define as follows

$$
\begin{array}{lll}
\min & \phi(\mathbf{x}) & I(\mathbf{s}, \underline{\mathbf{x}}) \\
s.t. & A\,\mathbf{x} \otimes (1-\mathbf{s}) \geq \mathbf{b} & \\
& \mathbf{x} \geq \underline{\mathbf{x}} & \\
& \mathbf{x} \in \{0,1\}^n &
\end{array}
$$

Note that $\mathbf{1} \in F(I(\mathbf{s}, \underline{\mathbf{x}}))$. The greedy heuristic is defined as follows:

Algorithm 1 Greedy heuristic: **H**

Require: Let $\underline{S} \subseteq S_f$, let $\mathbf{x}^1$ be an optimal solution for $\mathcal{I}(\underline{S})$ and let $(\mathbf{z}^*, \mathbf{s}^*)$ be an optimal solution for $Q(\mathbf{x}^1)$. Let us suppose that $v(Q(\mathbf{x}^1)) \geq 1$ and let $r = 1$

1: **while** $v(Q(\mathbf{x}^r)) \geq 1$ **do**
2: **Solve** $I(\mathbf{s}^*, \mathbf{x}^r)$ and let $\mathbf{x}^{r+1}$ be an optimal solution
3: **Solve** $Q(\mathbf{x}^{r+1})$ and let $(\mathbf{z}^*, \mathbf{s}^*)$ be an optimal solution
4: $r = r + 1$
5: **return** $\mathbf{x}^r$

Lemma 3. **H** *is finite and if* $\mathbf{x}^r$ *is the output then* $\mathbf{x}^r \in F(\mathcal{I})$.

Proof: From the definition of **H** we have for the first $\mathbf{s}^*$:$\sum_{j\in[n]} A_{ij}\mathbf{x}^1_j(1-\mathbf{s}^*_j) \leq b_i - 1$ for some $i \in [m]$. In step 2 we have that $\sum_{j\in[n]} A_{ij}\mathbf{x}^2_j(1-\mathbf{s}^*_j) \geq b_i$ for all $i \in [m]$ and then $\mathbf{x}^2 \neq \mathbf{x}^1$. Since $\mathbf{x}^2 \geq \mathbf{x}^1$ we have that $|supp(x^2)| \geq |supp(x^1)| + 1$. We can repeat the reasoning to proof that, as long as the algorithm does not stop, $|supp(\mathbf{x}^{r+1})| \geq |supp(\mathbf{x}^r)| + 1$ for all r in such a manner that in the worst case we will find $\mathbf{x}^r = \mathbf{1}$ for some r and the algorithm will stop because $\mathbf{1} \in F(\mathcal{I})$. When the algorithm stops because $v(Q(\mathbf{x}^r)) = 0$ we have that $\mathbf{x}^r \in F(\mathcal{I})$ $\square$

Note that the original $\mathbf{s}^*$ before step 1 and all the scenarios generated at step 3 while $v(Q(\mathbf{x}^r)) \geq 1$ are not covered with $\mathbf{x}^1$.

2.4 The non-vulnerable problem

Let NV the set of *non-vulnerables* variables defined as follows: $NV = \{j \in [n] : \mathbf{s}_j = 0\ \forall \mathbf{s} \in S\}$ and let $P(NV)$ the *non-vulnerable problem* defined as follows:

$$
\begin{array}{lll}
\min & \phi(\mathbf{x}) & P(NV) \\
s.t. & A\mathbf{x} \geq \mathbf{b} & \\
& \mathbf{x}_j = 0 \qquad \forall j \notin NV & \\
& \mathbf{x} \in \{0,1\}^n &
\end{array}
$$

If $F(P(NV)) \neq \emptyset$ and $\mathbf{x}_{nv}$ is an optimal solution then $\mathbf{x}_{nv} \in F(\mathcal{I})$ and then we have that $v(\mathcal{I}) \leq \min\{v(P(NV)), \phi(\mathbf{1})\}$.

2.5 An algorithm to find an ϵ-optimal solution for I

Let $\epsilon \geq 0$ be a predefined *relative global tolerance*. We are now in a position to define an algorithm to find $\mathbf{x}^+ \in F(\mathcal{I})$ such that $\mathbf{x}^+$ is an optimal solution or at least is an ϵ-optimal solution.

In order to simplify the exposition we present the algorithm with the relative global tolerance (ϵ) fixed. Some schemes are valid with ϵ updated as time progress and will be presented with the computational experience.

Algorithm 2 An algorithm to find an ϵ-optimal solution for $\mathcal{I}$: $\mathbf{A}(\epsilon)$

Require: Let $\epsilon \geq 0$, let $\underline{S} = \{\mathbf{0}\}$, let $LB = 0$ and let $UB = \min\{v(P(NV)), \phi(\mathbf{1})\}$

1: **while** $UB(1-\epsilon) > LB$ **do**
2: **Solve** $\mathcal{I}(\underline{S})$, let $\mathbf{x}^*$ be an optimal solution and let $LB = \phi(\mathbf{x}^*)$
3: **Solve** $Q(\mathbf{x}^*)$ and let $(\mathbf{z}, \mathbf{s})$ be an optimal solution
4: **if** $v(Q(\mathbf{x}^*)) = 0$ **then**
5: $UB = \phi(\mathbf{x}^*)$ and $\mathbf{x}^+ = \mathbf{x}^*$ ▷ in this case $UB = LB$
6: **if** $UB(1-\epsilon) > LB$ **then**
7: $\underline{S} = \underline{S} \cup \{\mathbf{s}\}$
8: **while** $v(Q(\mathbf{x}^*)) \geq 1$ **do**
9: $\mathbf{x} = \mathbf{x}^*$
10: **Solve** $I(\mathbf{s}, \mathbf{x})$ and let $\mathbf{x}^*$ be an optimal solution
11: **Solve** $Q(\mathbf{x}^*)$ and let $(\mathbf{z}, \mathbf{s})$ be an optimal solution
12: $\underline{S} = \underline{S} \cup \{\mathbf{s}\}$
13: **if** $\phi(\mathbf{x}^*) < UB$ **then**
14: $UB = \phi(\mathbf{x}^*)$ and $\mathbf{x}^+ = \mathbf{x}^*$
15: **return** $\mathbf{x}^+$, $gap = 100\frac{UB-LB}{UB}$

Lemma 4. $\mathbf{A}(\epsilon)$ *is finite and if* $\mathbf{x}^+$ *is the output then* $\mathbf{x}^+$ *is an optimal solution or at least is an* ϵ*-optimal solution for* $\mathcal{I}$

Proof: Note that: (i) Steps 8-12 correspond to executing **H** and will always run in a finite number of steps, (ii) all scenarios generated in steps 3 and 11 are feasible, (iii) if the algorithm stops with $LB = UB$ because of $v(Q(\mathbf{x}^*)) = 0$ at step 4 then $\mathbf{x}^+ = \mathbf{x}^*$ is an optimal solution for $\mathcal{I}$, (iv) if the algorithm stops because of $UB(1-\epsilon) \leq LB$ then $\mathbf{x}^+$ is an ϵ-optimal solution for $\mathcal{I}$.

In step 3 if $v(Q(\mathbf{x}^*)) \geq 1$ then $\mathbf{s} \notin \underline{S}$, therefore in the worst case $\mathcal{I}(S_f)$ is solved in step 2 by finding an optimal solution for $\mathcal{I}$ and the algorithm will stop □

3 Discrete-budgeted interdiction uncertainty set

Let $\Gamma \geq 1$, a uniform budgeted interdiction uncertainty set is defined as follows: $S = \{\mathbf{s} \in \{0,1\}^n : \sum_{j=1}^{n} \mathbf{s}_j \leq \Gamma\}$. Thus, the supports of the scenarios defining S are the subsets of $[n]$ with at most Γ elements. No correlation is assumed between the elements of $[n]$ that conditions their appearance in the scenarios. A step towards non-uniformity is to consider that there are vulnerable elements defining the set $V \subset [n]$ and non-vulnerable elements in the complement of V and in that case a particular case of a non-uniform budgeted interdiction uncertainty set is defined as follows: $S = \{\mathbf{s} : \sum_{j \in V} \mathbf{s}_j \leq \Gamma\}$. Again for elements in V no correlation is assumed.

On the other hand, it is usual to consider the discrete interdiction uncertainty set in the following form: let $K \geq 1$ and let $\{\mathbf{s}^k\}_1^K \subseteq \{0,1\}^n$ with $\mathbf{0} \in \{\mathbf{s}^k\}_1^K$ then $S = \{\mathbf{s}^k\}_1^K$. With this option, the scenario supports are subsets of $[n]$ that can be defined based on the existing correlations for the occurrence of the elements.

It is quite natural to define an uncertainty set based on the above two ideas. Let $K_B \geq 1$, let $\{\mathbf{s}_B^k\}_1^{K_B} \subseteq \{0,1\}^n$ an explicit *basic scenarios set* and let $\Gamma \geq 1$. What we do then in the *hard* version of the set we propose is to define the support of each scenario as the union of the supports of up to Γ basic scenarios and in the *soft* version we require that the support of each scenario be a subset of the union of the supports of at most Γ basic scenarios.

Formal definitions, the corresponding 0-1-Mixed Integer Linear Programming formulations and some basic properties are presented below.

3.1 Hard discrete-budgeted interdiction uncertainty set

The *hard discrete-budgeted interdiction uncertainty set* is defined as follows:

$$Sh(\{\mathbf{s}_B^k\}_1^{K_B}, \Gamma) = \{\mathbf{s} \in \{0,1\}^n : supp(\mathbf{s}) = \bigcup_{k:\mathbf{z}_k=1} supp(\mathbf{s}_B^k),\ \mathbf{e}^t\mathbf{z} \leq \Gamma,\ \mathbf{z} \in \{0,1\}^{K_B}\} =$$

$$\{\mathbf{s} \in \{0,1\}^n : \mathbf{z}_k \mathbf{s}_{Bj}^k \leq \mathbf{s}_j \leq \sum_{q \in [K_B]} \mathbf{z}_q \mathbf{s}_{Bj}^q\ \forall k \in [K_B]\ \forall j \in [n],\ \mathbf{1}^t\mathbf{z} \leq \Gamma,\ \mathbf{z} \in \{0,1\}^{K_B}\}$$

3.2 Soft discrete-budgeted interdiction uncertainty set

The *soft discrete-budgeted interdiction uncertainty set* is defined as follows:

$$Ss(\{\mathbf{s}_B^k\}_1^{K_B}, \Gamma) = \{\mathbf{s} \in \{0,1\}^n : supp(\mathbf{s}) \subseteq \bigcup_{k:\mathbf{z}_k=1} supp(\mathbf{s}_B^k),\ \mathbf{e}^t\mathbf{z} \leq \Gamma,\ \mathbf{z} \in \{0,1\}^{K_B}\} =$$

$$\{\mathbf{s} \in \{0,1\}^n : \mathbf{s}_j \leq \sum_{q \in [K_B]} \mathbf{z}_q \mathbf{s}_{Bj}^q\ \forall j \in [n],\ \mathbf{e}^t\mathbf{z} \leq \Gamma,\ \mathbf{z} \in \{0,1\}^{K_B}\}$$

3.3 Remarks

Note that:

1. $Sh(\{\mathbf{s}_B^k\}_1^{K_B}, \Gamma) \subseteq Ss(\{\mathbf{s}_B^k\}_1^{K_B}, \Gamma)$

2. if $supp(\mathbf{s}_B^{k_1}) \cap supp(\mathbf{s}_B^{k_2}) = \emptyset$ for all $k_1, k_2 \in [K_B]$ with $k_1 \neq k_2$ then:

   $$Sh(\{\mathbf{s}_B^k\}_1^{K_B}, \Gamma) = \{\mathbf{s} \in \{0,1\}^n : \mathbf{s}_j = \mathbf{z}_k \ \forall j \in supp(\mathbf{s}_B^k) \ \forall k \in [K_B], \mathbf{1}^t\mathbf{z} \leq \Gamma, \ \mathbf{z} \in \{0,1\}^{K_B}\}$$

3. $Sh(\{\mathbf{s}_B^k\}_1^{K_B}, 1) = \{\mathbf{0}\} \cup \{\mathbf{s}_B^k\}_1^{K_B}$ which is the discrete interdiction uncertainty set

4. if $\mathbf{s}_B^k = j_k \ \forall k \in [K_B]$ with $j_{k_1} \neq j_{k_2}$ for all $k_1, k_2 \in [K_B], k_1 \neq k_2$ and $V = \{j_1, \cdots, j_{K_B}\} \subseteq [n]$ then

   $$Sh(\{\mathbf{s}_B^k\}_1^{K_B}, \Gamma) = Ss(\{\mathbf{s}_B^k\}^{K_B}, \Gamma) = \{\mathbf{s} \in \{0,1\}^n : \sum_{j \in V} \mathbf{s}_j \leq \Gamma\}$$

   which is a uniform (non-uniform) budgeted interdiction uncertainty set when $K_B = n$ ($K_B < n$) .

4 Computational experience

The experiments have been performed on a personal computer as follows:Intel(R)Core(TM) i7-9750H CPU, @ 2.60 GHz Lenovo ThinkPad X1 Extreme Gen 2, 32.00 GB Ram and Windows 10 Pro Operating System. All the instances have been processed through ILOG-Cplex 12.10 from a DOcplex Python code. All the parameters of ILOG-Cplex 12.10 are in their default values unless otherwise indicated.

Computational experiences for the $\mathcal{I}$ and $\mathcal{B}u$ problems using mathematical programming algorithms to find optimal or ϵ-optimal solutions are not abundant in the literature. To the best of our knowledge, we can refer to two cases: (i) [1] presents experience for several types of problems with cardinality constraints (selection problem, job assignment problem and connected 2-edge subgraph problem) and the uncertainty set defined with a knapsack constraint with which the adversarial problem allows a compact formulation and (ii) in [9] a preliminary computational experience was presented (as far as we know not published later) using a branch and cut algorithm to solve $\mathcal{B}u$ with assignment constraints, $\phi(x) = \mathbf{c}^t\mathbf{x}$, $supp(\mathbf{s}) \in \{\{1\}, \cdots, \{n\}\}$ for all $\mathbf{s} \in S$ and $S = S_f$.

In this first computational experience with $\mathbf{A}(\epsilon)$ we will highlight the behavior of the algorithm, when the dimensions grow, in terms of the computation time, the final *gap* reached and the criterion with which the algorithm stopped. However, other statistics that may be useful for other types of analysis are presented and will be postponed for future occasions.

As usual when using mathematical programming it is very easy to define a configuration, to generate the synthesized data defining the problems, in such a way that the algorithms are very efficient. Such is the case of the configurations presented in the first rows of all tables presented. What was done starting from these easy configurations was to progressively increase the difficulty of the problems by manipulating some parameters and increasing the dimensions and uncertainty until it became evident that configurations were reached whose associated problems required a high computational work as will be seen in the tables. We hope that this first computational experience will demonstrate that the algorithm is able to find ϵ-optimal solutions for moderate dimensions even with a high degree of uncertainty.

In order to save time in experimentation, we implemented a flexible strategy for stopping the algorithm, which we call the tolerance-time-strategy (tts), based on increasing the *relative global tolerance* (ϵ) as time progresses.

The cases considered presented in section 4.1 have nominal versions corresponding to Set Covering, Simple plant location and min-Knapsack problems. SC and SPL problems are presented with specific motivations to help to understand a particular interpretation of the solution of $\mathcal{I}$. Also we include how the data, the basic scenarios and the uncertainty set used were generated. Details on the tolerance-time-strategy can be found in Section 4.2. The definition of the statistics can be found in section 4.3. The experimental results are presented in section 4.4.

Arbitrarily it was decided to use as uncertainty set S_s, for the SC problems, S_h for the SPL problems, which leads according to the basic scenarios defined to a classical budgeted uncertainty set over the servers, and S_h for the mK problems. The Γ parameter was taken at $\{3, 6\}$, which makes it practically impossible to find S_f explicitly for almost all cases considered.

4.1 The problems

4.1.1 Set covering

The authorities of a city want to organize the attention to the citizens (users) in public offices (servers) in such a way that each user is assigned to one or several servers located no farther than a certain pre-established distance. In the event that some of the servers temporarily collapse (e.g. due to traffic congestion or too many users being served) every user should have, except in extreme cases (unfeasible scenarios), some non-collapsed server available.

The city is divided into sectors and the users of each sector are assigned in block to one or several servers but then each user decides individually, and in each opportunity, which of the servers use among those to which was assigned. Each sector may or may not become a server. The authorities have an implicit description of the uncertainty set associated with the collapse of the servers.

Let S be the uncertainty set (implicitly known). If $\mathbf{s} \in S$ then $\mathbf{s}_j = 1$ indicates the temporary collapse of a server in sector j. The $\mathbf{0}$ scenario is in S and at every instant for a long time horizon some scenario is active. Let S_f be the set (not known a priori) of scenarios that can be successfully faced (feasible): those in which every user has an assigned non-collapsed server.

Let n be the number of sectors. Let $\mathbf{c}_j$ the cost per unit time to keep a server in sector j operational (unless a temporary collapse), let $A \in \{0,1\}^{n\times n}$ the matrix that indicates whether the users of sector i can be served in a server located in sector j ($A_{ij} = 1$) or not ($A_{ij} = 0$). It is desired to find the servers to be opened with minimum cost per unit time.

Let $\mathbf{x}_j = 1$ if it is decided use the sector j as a server, the nominal problem is defined as follows

$$\begin{array}{ll} \min & \mathbf{c}^t\mathbf{x} \\ s.t. & A\mathbf{x} \geq \mathbf{1} \\ & \mathbf{x} \in \{0,1\}^n \end{array} \qquad P$$

Note that if $\mathbf{x}$ is an optimal solution for $\mathcal{I}$ and $\mathbf{s} \in S_f$ is active then the available solution for $\mathbf{s}$ is $\mathbf{y}(\mathbf{s}) = \mathbf{x} \otimes (\mathbf{1} - \mathbf{s})$ and the population of sector i can be *well* served by any of the servers belonging to the set defined by $Serv(i,\mathbf{s}) = \{j : A_{ij}\mathbf{y}_j = 1,\ \mathbf{s}_j = 0,\ j \in [n]\} = \{j : A_{ij}\mathbf{x}_j(1-\mathbf{s}_j) = 1,\ j \in [n]\}$. If $\mathbf{s}_j = 1$ the interpretation is not exactly that $\mathbf{x}(\mathbf{s})_j = 1$ can not be implemented, the *new* interpretation in this case is that the users are not *well* served temporary from the server open in sector j and must use another server.

Note that if S_f is known explicitely then $\mathcal{I}$ is an ordinary set covering problem with $n \times |S_f|$ rows and n columns.

The data were generated at random as follows: (i) the coordinates of the sectors are chosen at random in the circle of ratio 1 in such a manner that the distance from the origin and the angle are uniform distribution on $[0,1]$ and $[0,2\pi]$ respectevily, (ii) the distance from sector i to sector j is the euclidian distance denoted $\mathbf{d}_{ij}$, (iii) let $\delta \in [0,1000]$ then $A_{ij} = 1$ if and only if $\mathbf{d}_{ij} < \frac{\delta}{1000} \max_{l,r\in[n]} \mathbf{d}_{lr}$, (iv) $\mathbf{c}_j$ is proportional (without loss of generality the proportionality constant is 1) to the total population assigned to a server in sector j, (v) let $\mathbf{P}_i$ the population of sector i chosen uniformly at random in $[0,100]$ then $\mathbf{c}_j = \sum_{i\in[n]} A_{ij}\mathbf{P}_i$.

Let $\beta \in [0,1000]$ and let $K_B \in [n]$. For $k \in [K_B]$ a neighborhood of sector k is found defined by those sectors no farther than a certain prespecified distance. Each neighborhood defines a basic scenario as follows:

.

$$supp(\mathbf{s}_B^k) = \{j \in [n] : \mathbf{d}_{jk} < \tfrac{\beta}{1000} \max_{l,r\in[n]} \mathbf{d}_{lr}\}\ \forall k \in [K_B]$$

The basic scenarios based on the defined neighborhoods are based on the assumption that, in general, there is a correlation in the collapse of neighboring sectors.

In this case we use $S = Ss(\{\mathbf{s}_B^k\}_1^{K_B}, \Gamma)$ and the configuration for a problem to be considered in the experimentation is identified in tables as $(n, K_B, \Gamma, \beta, \delta)$.

4.1.2 Simple plant location

Consider a set of r points in the plane. Each point is associated with a user and potentially a server. Let $\mathbf{d}_{ik}$ be the Euclidean distance between points i and k. Each user can be served from any server located no more than a prespecified distance away. Let $\mathcal{A} \in \{0,1\}^{r\times r}$ and let $\mathcal{A}_{ik} = 1$ if and only if the population of point i may be assigned to server in point k.

In order to write $\mathcal{I}$ with the same notation used in the rest of paper we can do:

(i) let $m = r$, $n = r \times r$ and let $A \in \{0,1\}^{m\times n}$ defined as follows:

$$A_{ij} = \begin{cases} \mathcal{A}_{ik} \ \forall (i,j): & i \in [m],\ j = (i-1)m + k \\ 0 \quad \forall (i,j): i \in [m],\ j \notin \{(i-1)m+1, \cdots, (i-1)m+m\} \end{cases}$$

(ii) let $\mathbf{x} \in \{0,1\}^n$ and $\mathbf{x}_j = 1$ if and only if the population of point i is assigned to a server in point k with $j = (i-1)m + k$

(iii) Let $\mathbf{F}_k$ a fixed cost of using a server at location k. Let $\mathbf{P}_i$ the population of point i. Let us suppose that the cost per unit time for a server open at j is defined with ϕ as follows: $\phi(\mathbf{x}) = \sum_{k\in[m]} \phi_k(\mathbf{x})$ with:

$$\phi_k(\mathbf{x}) = \begin{cases} \sum_{i\in[m]} \mathbf{P}_i \mathbf{x}_{(i-1)m+k} + \mathbf{F}_k \ \ if \sum_{i\in[m]} \mathbf{x}_{(i-1)m+k} \geq 1 \\ 0 \qquad\qquad\qquad\qquad if \sum_{i\in[m]} \mathbf{x}_{(i-1)m+k} = 0 \end{cases}$$

(iv) Let $\mathcal{S}$ be the uncertainty set (implicitly known). If $\mathbf{u} \in \mathcal{S}$ then $\mathbf{u}_k = 1$ indicates the temporary collapse of a server in sector k. The $\mathbf{0}$ scenario is in $\mathcal{S}$ and at every instant for a long time horizon some scenario is active. Let $S \subseteq \{0,1\}^n$ defined as follows: $\mathbf{s}_j = \mathbf{u}_k$ if $j = (i-1)m + k$ with $i, k \in [m]$.

The nominal problem is defined as follows:

$$\begin{array}{ll} \min & \phi(\mathbf{x}) \\ s.t. & A\mathbf{x} \geq \mathbf{1} \\ & \mathbf{x} \in \{0,1\}^n \end{array} \qquad P$$

Note that the nominal problem is independent of the $\mathbf{P}_i$ values because there is an optimal solution such that any point i is assigned to only one server k.

Note that if S_f is known explicitely then $\mathcal{I}$ is not an ordinary simple plant location problem.

The data were generated at random as follows: (i) the coordinates of locations are chosen at random in the circle of ratio 1 in such a manner that the distance from the origin and the angle are uniform distribution on $[0,1]$ and $[0,2\pi]$ respectevily, (ii) let $\delta \in (0,1000)$ then $\mathcal{A}_{ik} = 1$ if and only if $\mathbf{d}_{ik} < \frac{\delta}{1000} \max_{l,r\in[r]} \{\mathbf{d}_{lr}\}$, (iii) let $\mathbf{dist}_i$ the distance from the origen until the point i, then $\mathbf{P}_i$ is choosen from a uniform distribution in $[800(1-\mathbf{dist}_i), 1200(1-\mathbf{dist}_i]$ for all $i \in [r]$, (iv) let $F > 0$ then $\mathbf{F}_i$ is choosen from a uniform distribution in $[800F(1-\mathbf{dist}_i), 1200F(1-\mathbf{dist}_i]$ for all $i \in [r]$,

Let $K_B \in [r]$. For $k \in [K_B]$ a basic scenario is defined as follows:

$$supp(\mathbf{u}_B^k) = \{q : \mathcal{A}_{kq} = 1,\ q \in [r]\}$$

In this case we use $\mathcal{S} = \mathcal{S}_h(\{\mathbf{u}_B^k\}_1^{K_B}, \Gamma)$. This means that we use a discrete budgeted uncertainty set for the servers: up to Γ servers (with all its links) may colapse at the same time.

The configuration for a problem to be considered in the experimentation is defined as $(r, K_B, \Gamma, \delta, F)$

4.1.3 Min-Knapsack

Let $n \geq 1$ and let $m = 1$, let $\mathbf{c} \in \mathbb{Z}_+^n$, let $\phi(\mathbf{x}) = \mathbf{c}^t\mathbf{x}$, let $\mathbf{w} \in \mathbb{Z}_+$, let $A_{1j} = \mathbf{w}_j\ \forall j \in [n]$, let $W > 0$, let $\mathbf{b}_1 = W$ and let S be the uncertainty set (implicitly known).

The nominal problem is:

$$\begin{array}{ll} \min & \mathbf{c}^t\mathbf{x} \\ s.t. & \mathbf{w}^t\mathbf{x} \geq W \\ & \mathbf{x} \in \{0,1\}^n \end{array} \qquad P$$

Note that if S_f is known explicitely then $\mathcal{I}$ is an ordinary multidimesional knpasack problem with $n \times |S_f|$ rows and n columns.

We generate *Weakly correlated instances* ([10]) as follows: (i) $\mathbf{w}_j$ is chosen uniformly at random in [1000] for all $j \in [n]$, (ii) let $W = \lfloor \frac{1}{2} \sum_{j\in[n]} \mathbf{w}_j \rfloor$ and (iii) $\mathbf{c}_j$ is chosen uniformly at random in $[\max\{\mathbf{w}_j - 100, 1\}, \mathbf{w}_j + 100]$ for all $j \in [n]$.

Despite their name, weakly correlated instances have a very high correlation between the cost and weight of an item.

Let $\beta \in [0, 1000]$ and let $K_B \in [n]$. For $k \in [K_B]$ a basic scenario is (arbitrarily) generated as follows:

$$supp(\mathbf{s}_B^k) = \left\{ j \in [n] : |\tfrac{\mathbf{c}_k}{\mathbf{w}_k} - \tfrac{\mathbf{c_j}}{\mathbf{w}_j}| < \tfrac{\beta}{1000} \min_{i \in [n]} \{\tfrac{c_i}{w_i}\} \right\}$$

In this case we use $S = Sh(\{\mathbf{s}_B^k\}_1^{K_B}, \Gamma)$ and the configuration for a problem to be considered in the experimentation is defined as (n, K_B, Γ, β).

4.2 Tolerances, *gap*s and time management

Let ϵ be the *relative global tolerance* and let T be the *global time limit* (T is used also as limit time for solving $\mathcal{I}(\underline{S})$). Let t be the cumulative running time when leaving step 1.

1. Problems $P, P(NV), Q(\mathbf{x})$,R and $I(\mathbf{s}, \mathbf{x})$ are solved using the Cplex default *relative gap tolerance* ($mipgap = 0.0001$) and without time limit

2. Problem $\mathcal{I}(\underline{S})$ is solved with $mipgap(\mathcal{I}) = 0.01$

3. The relative global tolerance, ϵ, is adjusted in execution as follows:

   (a) a tolerance-time-strategy (tts) is defined as follows: let $L \geq 2$ and let $tts = ((T_1, \epsilon_1), \cdots, (T_L, \epsilon_L))$ such that: $0 = T_1 < \cdots < T_{L-1} < T_L = T$, $mipgap(\mathcal{I}) \leq \epsilon_1 \leq \cdots \leq \epsilon_{L-1} < \epsilon_L = +\infty$.

   (b) we run the algorithm as follows: if $T_i < t \leq T_{i+1}$ then we use $\epsilon = \epsilon_i$. Note that if $t > T_L = T$ then the algorithm will stop with a solution $\left(\frac{UB-LB}{UB}\right)$-optimal. In order to save time during experimentation the scheme relaxes the requirement for stopping as time progresses

   (c) usual tts has the form $\{(0, \epsilon), (T, \infty)\}$ which say that the algortithm stop if either an ϵ-optimal solution was found or $t > T$ with a $\left(\frac{UB-LB}{UB}\right)$-optimal solution. In order to save time in the experimentation we use more complex tts as indicated in the tables.

4. The lower bounds for $\mathcal{I}$ are rigorously updated using the lower bound available from Cplex (*bestbound*) at the end of the solution of $\mathcal{I}(\underline{S})$ and not the value of the obtained solution (if LB is the lower bound after solving $\mathcal{I}(\underline{S})$ then $LB = \max\{LB, bestbound\}$). Since $mipgap(\mathcal{I}) \leq \epsilon$ at all times the scheme guarantees that if a generated solution $\mathbf{x}^*$ is in $F(\mathcal{I})$ (step 4) it will meet the relative global tolerance of the algorithm by the time it is generated. Also, if the algorithm stops without $\mathbf{x}^* \in F(\mathcal{I})$ in step 4 then LB and UB are valid bounds.

4.3 Statistics to be presented in the tables

For each configuration considered, as indicated in the tables, a *set* of random instances is generated. For each instance, P, $P(NV)$ and $\mathcal{I}$, the nominal, nonvulnerable and robust problems, respectively, are solved. The $\mathcal{I}$ problem is solved according to the tts as indicated. Also we solve R problem.

Details of the results are presented in tables. The basic information comes in two types of tables, one presents statistics that have to do with the generation of the data and some important characteristics of the generated problems (tables 1,3 and 5 for SC,SPL and mK problems respectively) and the other presents statistics that have to do with the performance of the A(ϵ) algorithm with the tts used and some important characteristics of the solutions found (tables 2,4 and 6 for SC,SPL and mK problems respectively and table 7 for all problems).

The independent statistics of the ($\mathbf{A}(\epsilon)$ run are defined as follows:

1. $|set|$: the number of problems in the set
2. $\overline{\mathbf{sB}}$: the average of the cardinality of the basic scenarios support
3. $\overline{\widehat{\mathbf{sB}}}$: the average of the maximum cardinality of the basic scenarios support
4. $\overline{\mathbf{nv}}$: the average percentage of non-vulnerable elements
5. $\overline{\mathbf{dens}}$: the density of the A and $\mathcal{A}$ matrices defining the possible assignments in the SC and SPL problems, respectively
6. $\#_f^{nv}$: the number of problems in which the solution of the non-vulnerable problem is feasible in $\mathcal{I}$
7. $\#nf$: the number of problems in which there are infeasible scenarios

The dependent statistics of the $\mathbf{A}(\epsilon)$ run are defined as follows:

1. $\overline{\mathbf{s3}}$: the average number of resolved relaxations (scenarios generated) in step 2(3)
2. $\overline{\mathbf{sH}}$: the average of the total number of generated scenarios using the greedy heuristic
3. $\overline{\mathbf{tH}}$: the average execution time in seconds for the greedy heuristic
4. $\overline{\mathbf{t}}$: the average execution time in seconds
5. $\widehat{\mathbf{t}}$: the worst execution time in seconds
6. $\#_+^*$:the number of problems in which the algorithm ended up verifying that the solution generated in the relaxation ($\mathbf{x}^*$) is feasible in $\mathcal{I}$ and is the solution reported by the algorithm ($\mathbf{x}^+$)
7. $\%\#_+^*$: the percentage of problems in which the algorithm ended up verifying that the solution generated in the relaxation ($\mathbf{x}^*$) is feasible in $\mathcal{I}$ and is the solution reported by the algorithm ($\mathbf{x}^+$)
8. $\#_+^{nv}$: the number of problems in which the solution of the non-vulnerable problem ($\mathbf{x}^{nv}$) ended up being the solution reported by the algorithm ($\mathbf{x}^+$)
9. $\overline{\mathbf{gap}}$: the final average *gap* found
10. $\widehat{\mathbf{gap}}$): the worst final *gap* found
11. $\overline{\mathbf{ryx}}$: the average of the percentage increase of the active variables in the solution of $\mathcal{I}$ with respect to the nominal problem

12. $\widehat{\mathbf{rnv}\mathcal{I}}$: the average of the percentage increase in the value of the non-vulnerable problem with respect to the $\mathcal{I}$ problem (when the non-vulnerable problem is feasible)
13. $\overline{\mathbf{r}\mathcal{I}\mathbf{P}}$: the average of the percentage increase in the value of the $\mathcal{I}$ problem with respect to the nominal problem
14. $\#_{\alpha}^{>}$: the number of problems in which $gap > \alpha$ with the tts used with α as indicated

Table 7 summarizes the results in terms of time, *gap* and stopping condition as a function of dimensions and number of basic scenarios. The same table illustrates the influence of the matrix density for the SC and SPL cases.

In Table 8 we present experiments for the problems with the worst *gap* of some selected sets. In the left part of the table we can see the time (t) and the *gap* achieved with the original tts. On the right side of the table we can see the time and *gap* achieved without time limit ($tts = \{(0, \epsilon), (\infty, \infty)\}$ with ϵ as indicated). The percentage improvement in the upper bound value found is presented in the ΔUB column

Counters and time statistics will be presented as integer numbers and other statistics will be presented to two decimal places.

4.4 Experimental results

4.4.1 Set covering

For the SC problems we varied n by $\{300, 400, 600\}$, for each n we took $K_B = 0.5n$, Γ was set to 6, for each (n, K_B, Γ) β was varied by $\{5, 10\}$ and for each (n, K_B, Γ, β) δ was varied by $\{100, 150, 200\}$. For each of the sets denoted $SC1, \cdots, SC12$ 30 problems were considered for a subtotal of 360 cases. For each of the more demanding sets denoted $SC13, \cdots, SC18$ 10 problems were considered for a subtotal of 60 cases. In total, 420 SC problems were considered (see table 1).

In all the SC problems referenced in Tables 1 and 2 the following tts was used: $tts = ((0, 0.01), (600, 0.0125), (1200, 0.015), (1800, 0.0175), (2400, 0.02), (3000, 0.03), (3600.\infty))$ (in the worst case it was aimed to find solutions in no more than one hour with a *gap* at most 3%).

In 418 problems out of 420 total SC problems, the algorithm found solutions with *gap* at most 3% and the worst average was 1.59% for problems of the $SC13$ set (with $(n, K_B, \Gamma, \beta, \delta) = (600, 300, 6, 5, 100)$). The worst average time was 2026 seconds for the $SC18$ set (with $(n, K_B, \Gamma, \beta, \delta) = (600, 300, 6, 10, 200)$) (see Tables 1 and 2). The computation time and *gap* averages found clearly evolve as expected as a function of n, going from minutes for $n = 300$ to just over twenty minutes for $n = 600$, with the average *gap* going from less than 1% for $n = 300$ to 1.43% for $n = 600$) (see Table 7).

The two problems with *gap* greater than 3% in sets $SC13$ and $SC16$ (with *gap* 6.85% and 8.27% respectively) were solved with $tts = ((0, 0.03), (\infty, \infty))$ i.e., requiring to find a solution with *gap* less than 3% with no time limit. In both cases a

solution with *gap* less than 3% was found and in the worst case the run time was close to twelve hours (see Table 8). The percentage upper bound improvement for the SC cases is very low (3.0% in the best case), showing that the original solutions had a true *gap* close to 3%, lower than the *gap* verified at the time of stopping the algorithm.

In 87%, 77% and 33% of the problems with n fixed at 300,400,600 respectevily (see table 7), the algorithm stopped upon detecting a solution in $F(\mathcal{I})$ at steps 2-4 (in that cases the algorithm found a solution in $F(\mathcal{I})$ before upper bounds of quality were achieved using the greedy heuristic). In the rest of cases the algorithm stopped upon detecting a solution with a *gap* smaller than the one in effect at the time of the stop, this means that an upper bounds of quality were achieved using the greedy heuristic before sufficient scenarios were generated to find a solution at $F(\mathcal{I})$ in steps 2-4. The results seem to indicate that as the problem size grows it is more useful to have good bounds to stop the algorithm in a reasonable time before a solution in $F(\mathcal{I})$ is generated in a relaxation.

4.4.2 Simple plant location

For SPL problems (see table 3) we varied r by $\{200, 300, 400\}$, for each r we varied K_B by $\{0.25r, 0.50r\}$, for each (r, K_B) was made to vary Γ by $\{3, 6\}$, for each (r, K_B, Γ) was made to vary δ by $\{100, 200\}$ and for each (r, K_B, Γ, δ) was made to vary F by $\{20, 40\}$. For each of the configurations denoted $SP1, \cdots, SP20$ and $SP33, \cdots, SP36$ 30 problems were considered for a subtotal of 720 cases, for the demanding configurations denoted $S21, \cdots, SP24$ and $SP37, \cdots, SP40$ 10 cases were considered for a subtotal of 80 cases and finally for the demanding configurations denoted $SP25, \cdots, SP32$ and $SP41, \cdots, SP48$ 5 cases were taken for a subtotal of 80 cases for a total of 48 sets and 880 SPL problems.

In all problems that belong to the $SPL1, \cdots, SPL24$ and $SPL33, \cdots, SPL40$ sets contemplated in tables 3 and 4, the following *tts* was used: $tts = ((0, 0.01), (600, 0.0125), (1200, 0.015), (1800, 0.0175), (2400, 0.02), (3000, 0.03), (3600.\infty))$ (in the worst case it was aimed to find solutions in no more than one hour with a *gap* at most 3%).

For the more demanding $SPL25, \cdots, SPL32$ and $SPL41, \cdots, SPL48$ sets contemplated in tables 3 and 4, the following *tts* was used: $tts = ((0, 0.01), (600, 0.015), (1200, 0.02), (1800, 0.03), (2400, 0.04), (3000, 0.05), (7200.\infty))$ (in the worst case it was aimed to find solutions in no more than two hours with a *gap* at most 5%).

In 858 of the 860 cases of the first 40 sets, the algorithm found solutions with *gap* at most 3%. The worst average time was 2810 seconds for the $SPL26$ set (with $(n, K_B, \Gamma, \delta, F) = (300, 150, 3, 100, 40)$) and the worst average *gap* was 2.29% for the same set.

For the remaining 8 sets, all with $(r, K_B) = (400, 0.5r)$, the algorithm found solutions with *gap* at most 3% in 29 of the 40 problems solved. All 11 failures occurred at low density ($\delta = 100$). For sets with $\delta = 200$ the worst average time was 2406 seconds for the $SPL48$ set (with $(n, K_B, \Gamma, \delta, F) = (400, 200, 6, 200, 40)$) and the worst average *gap* was 2.31% for the same set.

The two problems with *gap* greater than 3% in $SPL26$ and $SC32$ sets and the worst case of $SPL41$,$SPL42$,$SPL45$ and $SPL46$ sets were solved with $tts = ((0, 0.03), (\infty, \infty))$ i.e. requiring to find a solution with *gap* less than 3% with no time limit. In the six cases a solution with *gap* less than 3% was found and in the worst case the execution time was more that 21 hours (see table 8). The percentage improvement for the upper bound for the SPL cases were very high in three cases.

Table 7 shows the evolution of the time, gap and percentage of times the algorithm stopped for finding a solution in $F(\mathcal{I})$ as the dimensions grow. From minutes and average gap less than 1% to hours and average gap greater than 3%. As in the case of SC the percentage decreases as the dimensions increase. Additionally, the influence of the density of $\mathcal{A}$ can be seen: the higher the density the lower the time, gap and percentage.

In 232 of 880 problems the algorithm stopped upon detecting a solution with a *gap* less than the one in effect at the time of the stop, this means that upper bounds on quality were reached using the greedy heuristic before sufficient scenarios were generated to find a solution in $F(\mathcal{I})$ when solving a relaxation.

4.4.3 Min-knapsack

For the mK problems (see table 5) we varied n in $\{500, 1000, 2000, 4000\}$, for each n we took K_B in $\{0, 1n, 0.2n\}$, fixed Γ at 6 and for each (n, K_B, Γ) we varied β in $\{100, 150, 200\}$. For each of the configurations denoted $K1, \cdots, K12$ 30 problems were considered for a subtotal of 360 problems. For each of the more demanding sets denoted $K13, \cdots, K24$ 5 cases were considered for a subtotal of 60 cases, thus 420 mK problems were considered.

In all the mK cases contemplated in tables 5 and 6, the following tts was used:

$$tts = ((0, 0.01), (300, 0.0125), (600, 0.015), (900, 0.0175), (1200, 0.02), (1800, \infty))$$

(in the worst cases it was aspired to find solutions with *gap* in at most one 2% in about half an hour).

For the 390 problems in the $K1, \cdots, K18$ sets, solutions with *gap*s of less than 2% were achieved in less than half an hour (the worst case was 1.89%). For the problems in the $K1, \cdots, K15$ configurations the average time did not exceed 10 min. For 27 of the 30 problems in the $K19, K24$ configurations the final *gap*s clearly exceeded 2%

with the worst case at 8.73%.

The worst cases in terms of *gap* was selected for each of the configurations $K19, \cdots, K24$. These 6 problems were solved with $tts = ((0, 0.02), (\infty, \infty))$ i.e. requiring to find a solution with *gap* less than 2% with no time limit. In all 6 cases a solution with *gap* less than 2% was found and in the worst case the execution time was a little more than two hours (see table 7).

It is noteworthy that in 389 of the 390 problems in the $K1, \cdots, K18$ configurations, the algorithm stopped upon detecting a solution with a *gap* smaller than the one in effect at the time of the stop (see column $\#_+^*$ with zeroes except for one instance). This means that in general upper bounds of quality were achieved using the greedy heuristic before sufficient scenarios were generated to find a solution at $F(\mathcal{I})$ in steps 2-4. In the 6 cases reported in Table 7 the algorithm stopped for the same reason.

Table 7 shows the evolution of the time and gap as the dimensions grow. From minutes and average gap less than 1% to more than half an hour and average gap greater than 3%.

5 Conclusions and further extensions

5.1 Conclusions

An algorithm was defined to find ϵ-optimal solutions for the Robust combinatorial optimization problems with knapsack constraints under interdiction uncertainty. It is not assumed that all scenarios are feasible. The key elements of the algorithm are: (i) solving an adversarial problem that allows to decide if a solution generated in one of the relaxations is a feasible solution and that finds some scenario feasible and not covered by the solution if it turns out not to be a feasible solution and (ii) applying a greedy heuristic that generates feasible solutions from a generated solution as well as feasible and uncovered scenarios, allowing to stop the algorithm when finding an ϵ-optimal solution.

It was experimented with three types of problems and from the experimentation it became evident that the proposed algorithm is able to generate in general, ϵ-optimal solutions (with $\epsilon \in [0, 0.03]$ in our experiments) for problems with moderate dimensions.

The experimentation was done on Discrete-budgeted interdiction uncertainty sets, introduced in this paper, that result from applying the fundamental idea of the discrete budgeted uncertainty to a set of explicitly defined binary scenarios.

5.2 Further extensions

There are at least three natural paths to try to define algorithms, based on mathematical programming, to deal with $\mathcal{I}$ with better performance than the proposed one, as follows:

1. use the same idea of the proposed algorithm to solve each relaxation, i.e., apply the proposed algorithm to the problem restricted to the set of scenarios already generated,

2. solve $\mathcal{I}$ using a Branch-and-cut algorithm in which the usual criteria (relaxation, branching, cutting, bounding), based on the set of scenarios already generated, are also valid for Sf,

3. replace the proposed greedy heuristic with a greedy heuristic specialized to the problem to be solved and solve the relaxations with appropriate algorithms to the case, e.g., one would use an algorithm for set-covering (multidimensional knapsack) problems when solving the relaxations corresponding to the robust set-covering (knapsack) problem.

References

[1] Goerigk, M., Khosravi, M.: Robust combinatorial optimization problems under budgeted interdiction uncertainty (2024) https://doi.org/10.1007/s00291-024-00772-0

[2] Adjiashvili, D.: Structural robustness in combinatorial optimization. In: DISS. ETH NR. 20327, (2012). https://doi.org/10.3929/ethz-a-007579109

[3] Adjiashvili, D., Stiller, S., Zenklusen, R.: Bulk-robust combinatorial optimization. Math. Program. **149**, 361–390 (2015) https://doi.org/10.1007/s10107-014-0760-6

[4] Pfetsch, M.E., Schmitt, A.: A generic optimization framework for resilient systems. Optimization Methods and Software **38**(2), 356–385 (2023) https://doi.org/10.1080/10556788.2022.2142581

[5] Hommelsheim, F.: Complexity of bulk-robust combinatorial optimization problems (2020) https://doi.org/10.17877/DE290R-21673

[6] Adjiashvili, D., Bindewald, V., Michaels, D.: Robust Assignments via Ear Decompositions and Randomized Rounding (2016). https://arxiv.org/abs/1607.02437

[7] Adjiashvili, D., Bindewald, V., Michaels, D.: Robust assignments with vulnerable nodes. CoRR **abs/1703.06074** (2017) 1703.06074

[8] Bindewald, V.: Bulk-robust assignment problems: hardness, approximability and

algorithms. PhD thesis, Dortmund University, Germany (2017). https://doi.org/10.17877/DE290R-19108 . http://hdl.handle.net/2003/37112

[9] Walter, M., Adjiashvili, D., Bindewald, V., Michaels, D.: Solving bulk-robust assingment problems to optimality. Aussois Combinatorial Optimization Workshop (2018)

[10] Pisinger, D.: Where are the hard knapsack problems? Computers and Operations Research **32**(9), 2271–2284 (2005) https://doi.org/10.1016/j.cor.2004.03.002

Table 1 *SC* problems

set	$\|set\|$	n	K_B	Γ	β	δ	$\overline{\mathbf{sB}}$	$\widehat{\overline{\mathbf{sB}}}$	$\overline{\mathbf{nv}}$	$\overline{\mathbf{dens}}$	$\#_f^{nv}$	$\#nf$
$SC1$	30	300	$0.5n$	6	5	100	1.08	3.30	48.31	7.19	2	28
$SC2$						150	1.10	3.70	48.11	13.48	24	5
$SC3$						200	1.10	3.77	48.35	21.23	30	0
$SC4$					10	100	1.34	7.17	44.45	6.84	1	29
$SC5$						150	1.37	6.83	44.14	13.38	29	1
$SC6$						200	1.34	6.57	43.77	21.69	29	1
$SC7$	30	400	$0.5n$	6	5	100	1.14	4.90	47.18	6.97	5	24
$SC8$						150	1.12	4.43	47.55	13.37	27	1
$SC9$						200	1.13	4.63	47.74	21.55	30	0
$SC10$					10	100	1.47	8.97	42.68	7.09	6	23
$SC11$						150	1.43	8.80	42.84	13.97	28	2
$SC12$						200	1.44	8.33	42.75	21.31	30	0
$SC13$	10	600	$0.5n$	6	5	100	1.16	5.50	46.80	6.83	6	3
$SC14$						150	1.19	7.00	46.63	13.23	10	0
$SC15$						200	1.16	5.10	47.06	20.99	10	0
$SC16$					10	100	1.68	12.50	40.63	6.79	7	2
$SC17$						150	1.70	13.30	39.91	13.51	10	0
$SC18$						200	1.67	13.80	40.80	21.22	10	0

Table 2 *SC* problems, $tts = ((0, 0.01), (600, 0.0125), (1200, 0.015), (1800, 0.0175), (2400, 0.02), (3000, 0.03), (3600, \infty))$

set	$\|set\|$	$\overline{\mathbf{s3}}$	$\overline{\mathbf{sH}}$	$\overline{\mathbf{tH}}$	$\bar{\mathbf{t}}$	$\widehat{\mathbf{t}}$	$\#_+^*$	$\#_\epsilon^{nv}$	$\overline{\mathbf{gap}}$	$\widehat{\mathbf{gap}}$	$\overline{\mathbf{ryx}}$	$\widehat{\mathbf{rnv\mathcal{I}}}$	$\overline{\mathbf{r\mathcal{I}P}}$	$\#_3^>$
$SC1$	30	8.40	51.07	56	65	201	30	1	0.71	1.00	24.93	0.00	21.88	0
$SC2$		8.60	40.00	48	65	216	25	22	0.87	1.00	14.29	0.00	11.70	0
$SC3$		6.97	27.37	34	52	114	25	27	0.93	1.00	7.33	0.00	5.43	0
$SC4$		18.40	94.13	138	189	3007	27	1	0.81	2.81	28.13	0.00	23.87	0
$SC5$		8.03	44.50	51	68	141	24	27	0.91	1.00	6.82	0.00	8.68	0
$SC6$		8.13	34.23	41	65	162	26	27	0.90	1.00	6.93	0.00	6.81	0
$SC7$	30	11.67	76.50	150	184	1115	27	4	0.89	1.24	23.06	0.00	18.50	0
$SC8$		9.73	50.70	97	155	389	20	26	0.93	1.00	7.52	0.00	8.67	0
$SC9$		7.70	30.17	61	134	630	23	30	0.94	1.00	4.40	0.00	4.65	0
$SC10$		11.23	93.83	185	224	682	26	5	0.90	1.09	24.86	0.00	21.12	0
$SC11$		9.13	58.47	120	191	586	23	26	0.95	1.00	7.85	0.00	7.66	0
$SC12$		10.37	60.70	141	336	802	21	29	0.96	1.22	11.78	0.00	6.85	0
$SC13$	10	13.10	113.30	463	966	3706	6	4	1.59	**6.85**	14.72	0.00	12.38	**1**
$SC14$		8.80	54.70	227	629	1306	6	10	1.08	1.47	5.28	0.00	4.68	0
$SC15$		9.10	45.60	215	1075	1927	3	10	1.12	1.50	5.67	0.00	3.89	0
$SC16$		12.60	150.10	653	1256	3699	3	7	1.82	**8.27**	13.62	0.00	13.49	**1**
$SC17$		11.20	91.20	416	1854	3292	1	10	1.44	1.75	6.21	0.00	5.98	0
$SC18$		9.40	62.30	309	2026	3073	1	10	1.52	2.88	8.06	0.00	4.74	0

Table 3 SPL problems

set	$\|set\|$	r	K_B	Γ	δ	F	$\overline{\mathbf{sB}}$	$\overline{\widehat{\mathbf{sB}}}$	$\overline{\mathbf{nv}}$	$\overline{\mathbf{dens}}$	$\#_f^{nv}$	$\#nf$
$SPL1$	30	200	$0.25r$	3	100	20	13.86	39.87	75.47	7.08	13	9
$SPL2$						40	13.84	40.33	75.25	6.99	12	11
$SPL3$					200	20	43.01	82.10	75.17	21.64	29	0
$SPL4$						40	41.53	80.63	75.35	21.06	30	0
$SPL5$				6	100	20	14.50	40.73	74.40	7.09	16	6
$SPL6$						40	13.48	40.23	76.17	7.06	15	11
$SPL7$					200	20	43.49	83.57	75.08	21.85	30	0
$SPL8$						40	43.76	81.37	74.68	21.56	30	0
$SPL9$	30	200	$0.5r$	3	100	20	14.05	40.63	49.11	6.88	1	22
$SPL10$						40	14.35	41.80	49.93	7.18	0	14
$SPL11$					200	20	43.40	82.90	49.78	21.61	30	0
$SPL12$						40	42.77	82.57	50.62	21.66	27	0
$SPL13$				6	100	20	15.10	42.80	48.83	7.38	1	17
$SPL14$						40	13.91	39.77	49.28	6.87	3	18
$SPL15$					200	20	43.27	82.03	49.65	21.46	27	0
$SPL16$						40	43.80	83.27	49.95	21.88	25	0
$SPL17$	30	300	$0.25r$	3	100	20	21.28	60.40	74.92	7.06	18	6
$SPL18$						40	20.81	59.07	74.62	6.83	21	6
$SPL19$					200	20	63.04	118.73	75.01	21.02	30	0
$SPL20$						40	63.38	121.43	75.31	21.40	30	0
$SPL21$	10			6	100	20	21.23	58.30	74.04	6.81	6	2
$SPL22$						40	21.14	61.60	75.13	7.07	7	1
$SPL23$					200	20	61.88	117.20	75.20	20.78	10	0
$SPL24$						40	63.71	120.10	74.96	21.19	10	0
$SPL25$	5	300	$0.5r$	3	100	20	21.39	63.00	49.28	7.02	0	1
$SPL26$						40	23.44	68.60	49.67	7.75	1	0
$SPL27$					200	20	65.55	124.80	50.02	21.83	5	0
$SPL28$						40	64.43	122.20	49.46	21.26	5	0
$SPL29$				6	100	20	21.84	62.40	49.58	7.20	0	2
$SPL30$						40	21.41	60.60	49.63	7.08	2	1
$SPL31$					200	20	64.52	122.40	49.84	21.43	5	0
$SPL32$						40	64.69	121.60	49.72	21.46	5	0
$SPL33$	30	400	$0.25r$	3	100	20	28.18	80.33	74.78	6.97	23	1
$SPL34$						40	27.20	78.50	74.83	6.76	28	1
$SPL35$					200	20	84.90	161.23	75.01	21.22	30	0
$SPL36$						40	85.06	162.47	75.25	21.49	30	0
$SPL37$	10			6	100	20	27.69	82.50	75.82	7.15	6	0
$SPL38$						40	28.27	85.70	75.50	7.24	9	1
$SPL39$					200	20	86.27	164.80	75.43	21.94	10	0
$SPL40$						40	85.06	159.50	74.89	21.20	10	0
$SPL41$	5	400	$0.5r$	3	100	20	28.95	83.00	49.49	7.17	2	1
$SPL42$						40	28.44	84.80	50.37	7.16	1	0
$SPL43$					200	20	88.12	165.40	50.00	22.02	5	0
$SPL44$						40	85.69	158.00	48.86	20.94	5	0
$SPL45$				6	100	20	25.93	77.60	49.97	6.48	1	0
$SPL46$						40	26.44	78.00	48.99	6.48	1	1
$SPL47$					200	20	85.04	161.40	50.58	21.51	5	0
$SPL48$						40	85.85	161.40	49.85	21.42	5	0

Table 4 SPL problems, $tts = ((0, 0.01), (600, 0.0125), (1200, 0.015), (1800, 0.0175), (2400, 0.02), (3000, 0.03), (3600, \infty)) * tts = ((0, 0.01), (600, 0.015), (1200, 0.02), (1800, 0.03), (2400, 0.04), (3000, 0.05), (7200, \infty))$

set	$\|set\|$	$\overline{\mathbf{s3}}$	$\overline{\mathbf{sH}}$	$\overline{\mathbf{tH}}$	$\overline{\mathbf{t}}$	$\widehat{\mathbf{t}}$	$\#^{*}_{+}$	$\#^{nv}_{\epsilon}$	$\overline{\mathbf{gap}}$	$\widehat{\mathbf{gap}}$	$\overline{\mathbf{ryx}}$	$\widehat{\mathbf{rnv\mathcal{I}}}$	$\overline{\mathbf{r\mathcal{I}P}}$	$\#^{>}_{\beta}$
$SPL1$	30	3.93	21.47	48	59	113	27	1	0.69	1.00	8.85	3.97	3.05	0
$SPL2$		3.60	20.23	45	55	139	28	0	0.63	1.00	11.87	5.25	2.56	0
$SPL3$		3.50	6.27	14	55	268	24	11	0.76	0.99	2.35	4.01	1.52	0
$SPL4$		2.97	4.83	11	39	120	26	9	0.65	0.99	3.18	2.92	1.55	0
$SPL5$		3.87	16.03	35	46	178	28	0	0.61	0.97	10.20	3.87	2.85	0
$SPL6$		3.40	13.47	30	41	161	26	0	0.65	0.99	14.40	7.25	2.49	0
$SPL7$		2.97	2.70	6	35	167	21	15	0.70	0.99	1.65	1.38	1.22	0
$SPL8$		3.00	2.63	6	42	300	23	10	0.67	1.00	2.92	4.10	1.43	0
$SPL9$	30	6.80	89.97	230	267	723	26	0	0.80	0.99	19.02	4.82	6.98	0
$SPL10$		6.47	93.30	237	280	913	27	-	0.75	1.20	24.58	-	5.96	0
$SPL11$		7.43	36.80	87	328	1139	26	4	0.74	0.98	4.63	4.44	3.08	0
$SPL12$		7.23	43.97	106	422	2056	29	0	0.78	1.57	7.90	7.07	4.39	0
$SPL13$		6.97	124.90	352	414	1461	29	0	0.78	1.00	20.55	1.39	6.73	0
$SPL14$		6.63	126.90	370	449	1543	28	0	0.82	1.00	27.68	11.38	6.19	0
$SPL15$		7.63	19.37	48	261	771	25	4	0.70	1.04	3.65	4.74	3.42	0
$SPL16$		6.63	16.20	38	299	1504	27	2	0.74	1.06	6.63	11.41	3.89	0
$SPL17$	30	4.97	43.07	220	346	904	22	3	0.84	1.21	5.72	2.64	2.77	0
$SPL18$		4.70	38.67	195	340	1266	22	2	0.73	0.97	8.47	4.51	2.76	0
$SPL19$		3.07	5.80	28	132	775	17	15	0.78	1.01	1.24	1.31	0.89	0
$SPL20$		3.70	8.77	42	254	555	22	11	0.87	1.00	2.02	2.01	1.35	0
$SPL21$	10	5.50	22.50	113	235	486	9	1	0.83	1.00	2.67	1.26	2.82	0
$SPL22$		4.60	46.40	246	392	901	9	0	0.82	1.00	9.83	4.20	2.82	0
$SPL23$		2.40	1.80	9	119	580	5	7	0.84	0.98	0.87	0.32	0.83	0
$SPL24$		3.80	4.20	21	167	462	9	3	0.81	0.99	1.60	2.02	1.29	0
$SPL25$	*5	8.40	210.80	1381	1814	3234	4	-	1.05	1.30	13.73	-	5.60	0
$SPL26$		9.00	209.40	1404	2810	7578	2	0	2.29	**7.44**	27.33	4.24	7.84	**1**
$SPL27$		6.40	42.80	219	965	1786	2	3	1.26	2.00	1.20	1.50	2.66	0
$SPL28$		7.00	62.60	328	1463	1999	3	1	1.29	2.72	7.67	4.34	3.58	0
$SPL29$		9.00	234.00	1620	2564	6090	5	0	0.81	0.99	13.60	0.00	6.10	0
$SPL30$		8.80	235.00	1702	2135	4343	3	0	1.03	1.95	18.67	6.87	5.62	0
$SPL31$		6.00	32.40	172	936	1488	1	4	1.31	1.98	0.80	0.51	2.82	0
$SPL32$		8.40	15.80	83	1569	2474	2	4	1.97	**3.59**	0.13	0.91	4.21	**1**
$SPL33$	30	5.07	48.63	453	1076	2423	19	7	1.02	1.69	4.81	2.35	2.17	0
$SPL34$		4.97	50.13	471	1246	3351	18	6	1.18	2.83	5.77	3.20	2.51	0
$SPL35$		1.97	2.33	20	163	794	4	27	0.81	1.07	0.25	0.48	0.60	0
$SPL36$		3.03	6.23	56	456	1514	18	22	0.85	1.25	0.73	0.94	0.96	0
$SPL37$	10	5.90	62.00	590	1438	2682	9	1	0.99	1.79	5.60	2.35	2.05	0
$SPL38$		4.90	36.30	339	867	1950	9	0	0.86	1.03	13.40	2.90	1.88	0
$SPL39$		3.00	5.20	45	336	799	3	7	0.91	1.14	0.48	0.34	0.72	0
$SPL40$		2.60	3.20	28	293	1466	7	6	0.93	1.40	1.57	0.68	1.10	0
$SPL41$	*5	9.00	253.20	3052	6041	7326	1	2	**4.87**	**9.82**	12.00	0.00	8.84	**2**
$SPL42$		6.40	253.20	3022	7388	9511	1	0	**9.46**	**23.44**	52.15	7.33	15.70	**2**
$SPL43$		4.40	42.60	383	1255	2089	0	5	1.59	2.16	0.00	0.00	2.23	0
$SPL44$		4.80	34.80	310	1844	2823	0	5	1.70	2.18	0.00	0.00	2.77	0
$SPL45$		5.20	348.00	4824	7741	8286	0	1	**26.53**	**54.18**	94.30	0.00	50.26	**5**
$SPL46$		6.00	229.60	2924	5948	9011	1	1	**5.75**	**16.40**	25.15	0.00	10.72	**2**
$SPL47$		4.60	10.20	92	1003	1496	1	4	1.21	1.51	0.20	0.48	1.35	0
$SPL48$		6.00	30.40	281	2406	3361	0	4	2.31	2.92	0.80	0.29	3.70	0

Table 5 mK problems

set	$\|set\|$	n	K_B	Γ	β	$\overline{\mathbf{sB}}$	$\overline{\widehat{\mathbf{sB}}}$	$\overline{\mathbf{nv}}$	$\#_f^{nv}$	$\#nf$
$mK1$	30	500	50	6	100	3.17	8.07	72.75	30	0
$mK2$					150	4.29	10.40	66.25	29	0
$mK3$					200	5.72	13.00	58.35	20	0
$mK4$			100	6	100	3.26	8.63	52.98	7	0
$mK5$					150	4.55	11.50	43.81	0	0
$mK6$					200	5.78	14.27	37.20	0	0
$mK7$	30	1000	100	6	100	5.03	12.80	62.21	28	0
$mK8$					150	7.14	16.87	52.47	3	0
$mK9$					200	9.05	20.80	44.92	0	0
$mK10$			200	6	100	4.01	13.07	40.68	0	0
$mK11$					150	7.00	17.50	30.97	0	0
$mK12$					200	9.11	21.60	25.25	0	0
$mK13$	5	2000	200	6	100	9.37	24.80	44.79	0	0
$mK14$					150	11.91	27.80	37.41	0	0
$mK15$					200	16.09	35.00	29.77	0	0
$mK16$			400	6	100	8.28	20.80	27.81	0	0
$mK17$					150	12.56	28.40	19.41	0	0
$mK18$					200	15.48	35.20	15.87	0	0
$mK19$	5	4000	400	6	100	15.72	36.80	30.03	0	0
$mK20$					150	22.30	48.80	22.13	0	0
$mK21$					200	29.91	62.60	19.11	0	0
$mK22$			800	6	100	14.78	36.00	16.38	0	0
$mK23$					150	22.40	50.00	12.47	0	0
$mK24$					200	29.86	63.80	9.73	0	0

Table 6 mK problems, $tts = ((0, 0.01), (300, 0.0125), (600, 0.015), (900, 0.0175), (1200, 0.02), (1800, \infty))$

set	$\|set\|$	$\overline{\mathbf{s3}}$	$\overline{\mathbf{sH}}$	$\overline{\mathbf{tH}}$	$\overline{\mathbf{t}}$	$\widehat{\mathbf{t}}$	$\#_+^*$	$\#_\epsilon^{nv}$	$\overline{\mathbf{gap}}$	$\widehat{\mathbf{gap}}$	$\overline{\mathbf{ryx}}$	$\widehat{\mathbf{rnv\mathcal{I}}}$	$\overline{\mathbf{r\mathcal{I}P}}$	$\#_2^>$
$mK1$	30	2.57	25.90	4	5	8	0	30	0.67	0.99	-3.80	0.00	4.17	0
$mK2$		5.63	59.63	12	19	306	0	29	0.69	1.06	-3.75	0.00	5.71	0
$mK3$		30.40	275.90	46	92	309	0	20	0.85	1.13	3.80	0.00	9.72	0
$mK4$		85.60	855.83	238	485	1213	0	3	1.25	1.85	5.98	1.37	9.95	0
$mK5$		56.57	821.87	223	423	922	0	-	1.23	1.55	11.08	-	13.87	0
$mK6$		87.4	924.50	273	442	906	1	-	1.26	1.55	16.83	-	17,37	0
$mK7$	30	14.63	209.03	95	115	308	0	9	0.95	1.07	-0.76	0.88	6.66	0
$mK8$		36.60	453.50	192	248	325	0	1	1.00	1.13	6.47	1.82	10.23	0
$mK9$		30.94	452.50	202	252	312	0	-	0.99	1.16	10.59	-	13.35	0
$mK10$		29.34	547.17	385	453	618	0	-	1.20	1.29	8.22	-	10.14	0
$mK11$		31.23	578.70	408	481	652	0	-	1.20	1.48	11.33	-	13.83	0
$mK12$		31.63	601.40	447	524	653	0	-	1.24	1.47	14.70	-	17.11	0
$mK13$	5	19.60	439.60	525	594	633	0	-	1.27	1.37	9.22	-	9.64	0
$mK14$		19.40	435.00	544	614	659	0	-	1.23	1.33	9.43	-	12.14	0
$mK15$		19.20	435.80	550	617	659	0	-	1.35	1.48	12.66	-	15.37	0
$mK16$		18.60	453.40	985	1081	1255	0	-	1.70	1.76	10.26	-	10.56	0
$mK17$		20.20	525.80	1216	1327	1528	0	-	1.89	1.99	13.22	-	14.52	0
$mK18$		18.00	531.80	1243	1343	1619	0	-	1.83	1.99	13.85	-	17.25	0
$mK19$	5	15.20	372.80	1647	1800	1929	0	-	**2.16**	**2.78**	8.16	-	10.75	**2**
$mK20$		13.80	392.20	1724	1867	1891	0	-	**3.18**	**3.66**	10.12	-	15.05	**5**
$mK21$		12.40	385.60	1761	1892	1956	0	-	**3.83**	**4.70**	12.31	-	18.28	**5**
$mK22$		6.40	219.40	1906	2047	2103	0	-	**4.53**	**5.31**	9.79	-	12.59	**5**
$mK23$		6.00	224.80	1996	2136	2212	0	-	**6.64**	**7.08**	13.33	-	18.36	**5**
$mK24$		6.20	209.20	1914	2053	2229	0	-	**7.83**	**8.73**	16.48	-	22.71	**5**

Table 7
SC problems,
$tts =$
$((0, 0.01), (600, 0.0125), (1200, 0.015), (1800, 0.0175), (2400, 0.02), (3000, 0.03), (3600, \infty))$
SPL problems, $tts =$
$((0, 0.01), (600, 0.0125), (1200, 0.015), (1800, 0.0175), (2400, 0.02), (3000, 0.03), (3600, \infty))$
$*\ tts =$
$((0, 0.01), (600, 0.015), (1200, 0.02), (1800, 0.03), (2400, 0.04), (3000, 0.05), (7200, \infty))$
mK problems,
$tts = ((0, 0.01), (300, 0.0125), (600, 0.015), (900, 0.0175), (1200, 0.02), (1800, \infty))$

P	n, r, n	K_B	δ	$\overline{\mathbf{t}}$	$\overline{\mathbf{gap}}$	$\widehat{\mathbf{gap}}$	$\#^{>}_{3,2,2}$	$\%\#^{*}_{+}$	$\lvert set\rvert$
SC	300	$0.5n$	-	84	0.86	2.81	0	87.22	180
	400	$0.5n$	-	204	0.93	1.24	0	77.77	
	600	$0.5n$	-	1301	1.43	**8.27**	**2**	33.33	60
SPL	200	$0.25r$	-	47	0.67	1.00	0	84.58	240
		$0.5r$	-	340	0.76	1.57	0	90.42	
	300	$0.25r$	100	336	0.79	1.21	0	77.50	80
			200	181	0.82	1.01	0	66.25	
		$*0.5r$	100	2331	1.29	**7.44**	**1**	70.00	20
			200	1234	1.45	**3.59**	**1**	20.00	
	400	$0.25r$	100	1159	1.06	2.83	0	68.75	80
			200	311	0.85	1.40	0	40.00	
		$*0.5r$	100	6679	**11.65**	**54.18**	**11**	20.00	20
			200	1627	1.70	2.92	0	0.00	
mK	500	$0.1n$	-	39	0.74	1.13	0	0.00	90
		$0.2n$	-	450	1.25	1.85	0	1.11	
	1000	$0.1n$	-	205	0.98	1.16	0	0.00	
		$0.2n$	-	486	1.22	1.48	0	0.00	
	2000	$0.1n$	-	609	1.29	1.48	0	0.00	15
		$0.2n$	-	1251	1.81	1.99	0	0.00	
	4000	$0.1n$	-	1853	**3.06**	**4.70**	**12**	0.00	
		$0.2n$	-	2078	**6.34**	**8.73**	**15**	0.00	

Table 8 Wort cases. Left side: original tts, right side: $tts = ((0, \epsilon), (\infty, \infty))$ (the algorithm only stops with $gap \leq \epsilon$). For SC and SPL we use $\epsilon = 0.03$, for mK we use $\epsilon = 0.02$

$\lvert set \rvert = 1$	t	gap	t	gap	ΔUB
$SC13$	3706	**6.85**	41334	2.97	0.21
$SC16$	3699	**8.27**	28238	2.96	3.30
$SPL26$	7578	**7.44**	15957	1.85	5.09
$SPL32$	2474	**3.59**	2799	1.31	2.31
$SPL41$	7326	**9.82**	7961	1.00	4.54
$SPL42$	9511	**23.44**	11214	2.03	19.21
$SPL45$	8286	**54.18**	76244	0.63	52.90
$SPL46$	8896	**16.40**	27157	2.82	12.67
$mK19$	1867	**2.78**	2047	1.98	0.78
$mK20$	1883	**3.66**	2517	1.75	1.80
$mK21$	1854	**4.70**	2821	1.75	2.70
$mK22$	2037	**5.31**	4783	1.97	2.19
$mK23$	2152	**7.08**	5587	1.93	4.01
$mK24$	1929	**8.73**	7681	1.79	4.84