Heuristics for two-stage stochastic K-adaptability

Marianna De Santis[0000−0002−1189−5917], Federica Donnini[0009−0004−0377−2046]

Department of Information Engineering, University of Florence,
Via di Santa Marta 3 50139 Florence, Italy,
marianna.desantis@unifi.it federica.donnini@unifi.it

Abstract. Two-stage stochastic programs are used to model problems with uncertain data, where a decision maker first decides the values of first-stage variables, then observes the values of the uncertain data, and finally decides the values of the second-stage decisions. In many applications, however, it is convenient to only have a small set of precomputed second-stage solutions from which to choose. The K-adaptability approach for two-stage problems under uncertainty calculates K second-stage solutions already in the first stage. In the case where uncertainty can be modeled by a finite number of scenarios, the K-adaptability problem can be formulated as a minimization problem over all the possible partitions of the scenario set [4,7]. We present new partition-based heuristic methods able to provide upper bounds for the K-adaptability problem for two-stage stochastic optimization. Experimental results on linear and quadratic knapsack and facility location problems are presented.

Keywords: Stochastic Programming · Heuristics and Metaheuristics.

1 Introduction

The *extensive formulation* of two-stage stochastic programs [2] where uncertainty is represented by a finite set of scenarios S, $|S| = \ell > 0$, is as follows:

$$\begin{aligned} \min\ & f(x) + \sum_{s\in S} p_s g_s(y_s) \\ & x \in X, \\ & y_s \in Y_s(x), \quad s \in S, \end{aligned} \tag{1}$$

where x are the first-stage variables, and the second-stage variables are explicitly described for each scenario $s \in S$ by the corresponding vector y_s. The sets $X \subseteq \mathbb{R}^{n_x}$, $Y_s(x) \subseteq Y \subseteq \mathbb{R}^{n_y}$ may include integrality constraints on the variables x and y_s for each $s \in S$. Functions $f : \mathbb{R}^{n_x} \to \mathbb{R}$ and $g_s : \mathbb{R}^{n_y} \to \mathbb{R}$ are the first and second-stage objective functions respectively. We consider the most general case in which uncertainty can affect both the objective function and the feasible set and we do not assume the functions to be linear.

In this paper, we study the K-adaptable version of the two-stage stochastic optimization problem (1). The objective is that of finding, in the first stage, not only the optimal value of variable x, but also an optimal set of K second-stage solutions. In the second stage, once the uncertain parameters are known, we proceed to choose the best policy for the current scenario, among the set of precomputed solutions. The K second-stage solutions are determined so that they are feasible for the chosen first-stage solution and the objective value of the best of these solutions, calculated in each scenario, is optimal in expectation. Formally, the problem can be formulated as follows (see also [5]):

$$\begin{aligned} \min\ & f(x) + \sum_{s \in S} p_s \min\left\{ g_s(y) : y \in Y_s(x) \cap \left\{ y^1, \dots, y^K \right\} \right\} \\ & x \in X, \\ & y^1, \dots, y^K \in Y. \end{aligned} \tag{K-2SSP}$$

We call an instance of (K-2SSP) feasible if $x \in X$ and $y^1, \dots, y^K \in Y$ exist such that, for each $s \in S$, at least one $y^k,\ k \in [K]$, is feasible, i.e., $y^k \in Y_s(x)$. Note that we use the notation $[K] := \{1, \dots, K\}$.

The K-adaptability approach was first formulated in [1] for robust optimization problems and it has been extensively studied in its applications to robust optimization [6,10]. In [4] K-adaptability was studied for the first time for stochastic optimization under (linear) objective uncertainty. Exact solution methods are proposed for solving the K-adaptability problem, however, the authors point out that heuristics represent an effective alternative, as they consistently deliver near-optimal solutions within a short computation time. In [7] the K-adaptability approach was extended to optimization problems with uncertainty in the constraints and in [8] heuristic approaches to deal with this setting are formulated and tested. In [5] K-adaptability for two-stage stochastic programs is formulated for the first time and a new exact solution method relying on partitions of the scenario set is presented.

In this paper, we generalize and extend the heuristics from [4,8] and apply them to solve the K-adaptability approach for two-stage stochastic (nonlinear) programs. To the best of our knowledge this is the first time that heuristics are formulated and tested for this setting. In prior works, the K-adaptability problem was either feasible by construction [3], or assumed to be feasible through the inclusion of dummy policies that are valid for all scenarios [8]. In contrast, we do not adopt this assumption and instead we introduce an additional step in the heuristic to attempt to recover feasibility when it is not guaranteed. Our numerical results on linear instances of the two-stage stochastic knapsack problem with setup and the capacitated facility location problem show that, even in the two-stage setting, heuristic approaches consistently yield near-optimal solutions within short computation times. Numerical results on quadratic instances are also provided.

2 Heuristic solution strategies for two-stage stochastic $\boldsymbol{K}$-adaptability

The K-adaptability problem (K-2SSP) can be reformulated as a minimization problem over all the possible partitions of the scenario set S. In particular, we say that $\mathcal{P} = (P_1, \dots, P_K)$ is a partition of S of order K, if $\cup_{k\in[K]} P_k = S$, $P_k \neq \emptyset,\ \forall k \in [K]$, and $P_{k_1} \cap P_{k_2} = \emptyset$ for each $k_1, k_2 \in [K],\ k_1 \neq k_2$. Then, ($K$-2SSP) is equivalent to

$$\min\{pip(\mathcal{P}) \mid \mathcal{P} \text{ is a partition of } S \text{ of order } K\}, \tag{2}$$

where, given a partition $\mathcal{P}$, we define the corresponding *partition induced problem* as

$$\begin{aligned} \min\ & f(x) + \sum_{k\in[K]} \sum_{s\in P_k} p_s g_s(y^k) \\ & x \in X, \\ & y^k \in Y_{P_k}(x), \quad k \in [K], \end{aligned} \tag{$pip(\mathcal{P})$}$$

where $Y_{P_k}(x) := \cap_{s\in P_k} Y_s(x)$, for each $k \in [K]$. Problem ($pip(\mathcal{P})$) is feasible only if $x \in X$ exists such that $Y_{P_k}(x) \neq \emptyset,\ \forall\ k \in [K]$. If ($pip(\mathcal{P})$) is feasible, then its optimal solution, which we indicate with $(x_P, y_P^1, \dots, y_P^K)$, is the *solution induced by partition* $\mathcal{P}$ and provides a valid upper bound for (K-2SSP) . We say that partition $\mathcal{P}$ is *feasible* if the corresponding problem ($pip(\mathcal{P})$) is feasible.

Note that every feasible solution $(x, y^1, \dots, y^K)$ of (K-2SSP) induces a partition of the scenario set (or even more than one). Indeed, for each $s \in S$, we define

$$sip(s; x, y^1, \dots, y^K) = \{k \in [K] \mid y^k \in Y_s(x)\} \tag{3}$$

as the set of indices corresponding to the second-stage solutions that are feasible for that scenario. For ease of notation we will refer to the previous sets as $sip(s)$. Since $(x, y^1, \dots, y^K)$ is feasible, each set $sip(s)$ is non-empty and we say that $\mathcal{P}$ is a partition induced by $(x, y^1, \dots, y^K)$ if $\mathcal{P}$ is defined by setting

$$P_k := \{s \in S \mid k \in \arg\min\{p_s g_s(y^j) \mid j \in sip(s)\}\}, \quad k \in [K], \tag{4}$$

where, if for a certain $s \in S$ more than one index $j \in sip(s)$ exists realizing $\min\{p_s g_s(y^j)\}$, we add s to the element P_k such that k is the lowest numbered index realizing the minimum. Finally, for each $k \in [K]$ such that $P_k = \emptyset$, let $k' \in [K]$ such that $|P_{k'}| > 1$. We select $s \in P_{k'}$, and we set $P_k = \{s\}$ and $P_{k'} = P_{k'} \setminus \{s\}$. Following this procedure, $\mathcal{P}$ is a partition of S by construction.

Finally, we say that $\mathcal{P} = (P_1, \dots, P_K)$ is a *subpartition of* S if $\cup_{k\in[K]} P_k \subset S$, $P_k \neq \emptyset,\ \forall k \in [K]$, and $P_{k_1} \cap P_{k_2} = \emptyset$ for each $k_1, k_2 \in [K],\ k_1 \neq k_2$. That is, the elements of a subpartition do not cover the whole scenario set. Note that, if $\mathcal{P}$ is a subpartition of S the corresponding problem ($pip(\mathcal{P})$) is still well defined, however its solution is not necessarily a feasible solution of (K-2SSP).

2.1 A basic heuristic for two-stage stochastic K-adaptability

Given an initial partition $\mathcal{P}$ of S, we generalize the basic heuristic from [4,8] to the K-adaptability problem for two-stage stochastic programs. The resulting heuristic method is described in Algorithm 1.

Algorithm 1 Basic Heuristic

```
Input: $\mathcal{P}$ partition of $S$
Output: $(x, y^1, \dots, y^K)$
1: Set $\mathcal{P}' = (\emptyset, \dots, \emptyset)$
2: while $\mathcal{P}' \neq \mathcal{P}$ do
3:     Set $\mathcal{P}' \leftarrow \mathcal{P}$
4:     Compute the solution $(x, y^1, \dots, y^K)$ induced by $\mathcal{P}'$
5:     Compute the partition $\mathcal{P}$ induced by $(x, y^1, \dots, y^K)$
6: end while
```

In line 4, a solution $(x, y^1, \dots, y^K)$ induced by $\mathcal{P}'$ can be found by solving $(pip(\mathcal{P}))$ with $\mathcal{P} = \mathcal{P}'$. Then, in line 5, a partition induced by a solution can be computed as shown in (4), after having defined the sets $sip(s; x, y^1, \dots, y^K)$ for each $s \in S$. Observe that, if we apply the basic heuristic in Algorithm 1 with a feasible partition $\mathcal{P}$ as input, we obtain a sequence of partitions whose feasibility is always guaranteed, and the algorithm terminates once the current partition cannot be further improved. However, if the first partition $\mathcal{P}$ is not feasible, i.e. $(pip(\mathcal{P}))$ is infeasible, then Algorithm 1 cannot be applied since $\mathcal{P}$ does not induce a solution. Additionally, note that in [4,8] the absence of first stage variables allowed for the decomposition of the partition induced problem into smaller independent problems, one for each element of the partition. Since here we have first stage variables and we are considering objective functions that are not necessarily linear, solving the problem $(pip(\mathcal{P}))$ to check the feasibility of a partition $\mathcal{P}$, and to compute an induced solution, might be time consuming. Motivated by these considerations, we propose a novel heuristic procedure that evaluates the feasibility of a partition without solving the full partition induced problem. If the feasibility of the current partition cannot be proven, the algorithm iteratively modifies it until a feasible partition is found.

2.2 Recovering feasibility

Finding a feasible solution for (K-2SSP) is equivalent to finding a partition $\mathcal{P}$ of S such that $(pip(\mathcal{P}))$ is feasible. For the K-adaptability problem studied in [4], since the uncertainty only affected the objective function, every partition induced problem was feasible. In [8], constraint uncertainty is considered, however, it is also assumed that a dummy policy with infinitely large cost exists, so that the feasibility of the partition induced problem is always guaranteed. In our setting, we do not make this assumption, therefore, given a partition $\mathcal{P}$ the corresponding partition induced problem might be infeasible.

Algorithm 2 provides a heuristic method to recover (or prove) feasibility when a given partition $\mathcal{P}$ is such that its corresponding ($pip(\mathcal{P})$) is infeasible (or when no feasible solution is found before a given time limit) by iteratively modifying the partition until a feasible one is obtained.

The algorithm starts from a partition $\mathcal{P}$ of S and from a feasible first-stage variable $x \in X$, for example we can set $x = x^*$ where $(x^*, \{y_s^*\}_{s \in S})$ is the optimal solution of (1). At each iteration, the first-stage variable is fixed to a value $\hat{x}$, so that we can decompose the corresponding problem ($pip(\mathcal{P})$) into independent components, one for each $k \in [K]$, i.e. we can define the problems

$$\begin{aligned} \min \sum_{s \in P_k} & p_s g_s(y^k) \\ & y^k \in Y_{P_k}(\hat{x}). \end{aligned} \qquad (decp(\hat{x}, k))$$

These new problems allow us to localize infeasibility within specific elements P_k. If all such problems are feasible, then the current partition $\mathcal{P}$ is feasible and the algorithm terminates. Otherwise, the algorithm identifies all indices $k \in [K]$ such that ($decp(\hat{x}, k)$) is infeasible and removes a scenario from each corresponding element P_k, provided that $|P_k| > 1$. This step aims at isolating the source of infeasibility by progressively reducing the size of the infeasible subsets. At this point $\mathcal{P}$ is a subpartition of S, since it does not cover the whole scenario set. As we have observed in the previous section, the problem ($pip(\mathcal{P})$) remains well-defined and it contains fewer constraints since we have removed some scenarios. If ($pip(\mathcal{P})$) is infeasible (or its feasibility cannot be proven within the time limit), the algorithm continues to remove scenarios from $\mathcal{P}$ until feasibility is achieved. Once a feasible solution $(\tilde{x}, \tilde{y}^1, \ldots, \tilde{y}^K)$ is found, a new partition can be defined. If $(\tilde{x}, \tilde{y}^1, \ldots, \tilde{y}^K)$ induces a partition $\tilde{\mathcal{P}}$, then we conclude that $\tilde{\mathcal{P}}$ is feasible and the algorithm terminates. Otherwise, we define the partition $\tilde{\mathcal{P}}$ that is obtained from the current subpartition $\mathcal{P}$, by adding all the scenarios that had previously been removed from $\mathcal{P}$ to a set different from their original one. At this point, the first stage solution is updated to $\tilde{x}$ and the procedure restarts with a new first stage variable $x = \tilde{x}$ and a new (still not necessarily feasible) partition $\mathcal{P} = \tilde{\mathcal{P}}$.

The procedure is guaranteed to make progress: if necessary, repeated removals eventually lead to a subpartition where each element P_k contains a single scenario, in which case ($pip(\mathcal{P})$) is feasible, since the original (1) is feasible. At this point, either a feasible partition is induced, or the algorithm restarts from a new first-stage decision and partition configuration. While the method is heuristic in nature, it provides an effective and systematic way to explore different partitions of the scenario set to recover or prove feasibility in settings where it cannot be easily guaranteed a priori.

Note that if $\mathcal{I}(\hat{x})$ is empty (see line 3), we have a feasible partition $\mathcal{P}$ and a feasible solution of (K-2SSP) given by $\hat{x}$ and the $\hat{y}^j$ found as the optimal solutions of ($decp(\hat{x}, k)$). Similarly, if Algorithm 2 terminates in line 15, then $(\tilde{x}, \tilde{y}^1, \ldots, \tilde{y}^K)$ is a feasible solution of (K-2SSP). Note that Algorithm 2 never requires the solution of a partition induced problem. Indeed, in line 10 problem ($pip(\mathcal{P})$) is always solved for a subpartition of S.

The behavior of the algorithm is illustrated in Example 1, which shows how, starting from an infeasible partition, the removal and reinsertion mechanism leads to the unique feasible partition of the problem.

Example 1. Consider the problem with no first-stage variables, where $S = \{1, \dots, 6\}$, $K = 2$, and without objective uncertainty, i.e. $g = g_s$ for each $s \in S$. Consider the feasible regions defined as

$$Y_s = \left\{ y \in \{0,1\} \middle| y \le 0.5 - \frac{1}{2s} \right\}, \quad s \in \{2,4,6\},$$

and

$$Y_s = \left\{ y \in \{0,1\} \middle| y \ge 0.5 + \frac{1}{2s} \right\}, \quad s \in \{1,3,5\}.$$

Then, $Y_{s_1} \cap Y_{s_2} = \emptyset$, for each $s_1 \in \{1,3,5\}$ and $s_2 \in \{2,4,6\}$, while $\cap_{s \in \{2,4,6\}} Y_s = \{y = 0\}$ and $\cap_{s \in \{1,3,5\}} Y_s = \{y = 1\}$. Therefore, the only partition such that the corresponding $(pip(\mathcal{P}))$ is feasible is (a permutation of) $\mathcal{P}^* = (\{1,3,5\},\{2,4,6\})$. If we start Algorithm 2 with input $\mathcal{P} = (\{1,3,6\},\{2,4,5\})$ and if, in line 6, we remove $s = 6$ from P_1, and $s = 5$ from P_2, then $\mathcal{P}$ reduces to the subpartition $\mathcal{P} = (\{1,3\},\{2,4\})$. The corresponding $(pip(\mathcal{P}))$ is feasible, and its optimal solution is $\tilde{y}^1 = 1$ and $\tilde{y}^2 = 0$, which induces the feasible partition $(\{1,3,5\},\{2,4,6\})$.

3 Numerical results

In this section, we present the numerical tests we performed to evaluate the effectiveness and efficiency of the heuristic strategies described in the previous sections. In particular, we define two heuristic methods, which we will refer to as `PR-H` and `TS-KA-H`. Method `PRH` starts from an initial partition $\mathcal{P}$ and solves $(pip(\mathcal{P}))$ to compute an upper bound. In case $(pip(\mathcal{P}))$ (and then $\mathcal{P}$) is infeasible, or if its feasibility cannot be proven by a solver within a time limit, we apply Algorithm 2. As soon as a feasible partition $\mathcal{P}$ is found `PR-H` terminates by computing the optimal solution of $(pip(\mathcal{P}))$ to obtain an upper bound. The second method, `TS-KA-H`, starts from the feasible partition found with `PR-H` and then uses Algorithm 1 to improve the incumbent upper bound.

To define the initial partition, we consider two different strategies. First, we consider a permutation of the scenario set S and we produce a balanced partition by splitting the scenario set into subsets of the same cardinality (when possible). We will refer to this strategy as *random*. As a second strategy, we apply the k-means algorithm to partition the scenario set S into K subsets, where scenarios that have similar uncertain parameters in the left-hand side of the constraints are grouped together. We mention [8] for other strategies to select the initial partition.

As a first set of experiments, we address linear two-stage stochastic instances obtained from the knapsack problem with setup and the capacitated facility location problem. As a second set of experiments, in order to deal with nonlinear

Algorithm 2 `Partition Repair Heuristic`

```
Input: $(\mathcal{P}, x)$
 1: Set $\hat{x} = x$ and $\mathcal{U} = \emptyset$
 2: Let $\mathcal{I}(\hat{x}) = \{k \in [K] | (decp(\hat{x}, k))$ is infeasible$\}$
 3: if $\mathcal{I}(\hat{x}) = \emptyset$ then return $\mathcal{P}$
 4: else
 5:     for $k \in \mathcal{I}(\hat{x})$ such that $|P_k| > 1$ do
 6:         Choose $s \in P_k$
 7:         $P_k \leftarrow P_k \setminus \{s\}, \mathcal{U} \leftarrow \mathcal{U} \cup \{s\}$
 8:         Choose $k(s) \in [K],\ k(s) \neq k$
 9:     end for
10:     if $(pip(\mathcal{P}))$ is feasible then
11:         Let $(\tilde{x}, \tilde{y}^1, \dots, \tilde{y}^K)$ be the optimal solution of $(pip(\mathcal{P}))$
12:         for $s \in S$ do
13:             Compute $sip(s) = sip(s; \tilde{x}, \tilde{y}^1, \dots, \tilde{y}^K)$ as in (3)
14:         end for
15:         if $(\tilde{x}, \tilde{y}^1, \dots, \tilde{y}^K)$ induces a partition $\mathcal{P}$ as in (4) then return $\mathcal{P}$
16:         else
17:             Let $\tilde{\mathcal{P}}$ be the subpartition induced by $(\tilde{x}, \tilde{y}^1, \dots, \tilde{y}^K)$
18:             for $s \in \mathcal{U}$ such that $sip(s) = \emptyset$ do
19:                 $\tilde{P}_{k(s)} \leftarrow \tilde{P}_{k(s)} \cup \{s\}$
20:             end for
21:             Set $x = \tilde{x}$ and $\mathcal{P} = \tilde{\mathcal{P}}$ and go to 1
22:         end if
23:     else
24:         Go to 5
25:     end if
26: end if
```

two-stage stochastic problems, we address instances that involve quadratic objectives. The extended formulation of the problems, as well as the parameters used to build the instances, can be found in [5].

For both problems we considered $n_x = 5$ first-stage variables and $n_y = 10$ second-stage variables. Concerning the number of scenarios, we used $\ell \in \{50, 100\}$, while $K \in \{3, 5\}$. For each combination of parameters K and ℓ we randomly generated 5 instances. To test the performance of the methods in a context where the feasibility of a partition is not guaranteed, we considered additional constraints requiring a minimum capacity to be satisfied. Specifically, the minimum capacity threshold was set to $p \in \{60\%, 70\%, 80\%\}$ for the knapsack instances and $p \in \{20\%, 40\%, 60\%\}$ for the capacitated facility location instances (both linear and quadratic).

Using the optimal solution of the two-stage stochastic problem, computed offline, or the best lower bound found within a time limit, as a valid lower bound lb for the K-adaptability problem, we consider the following performance measures: (i) the number of instances for which no upper bound was found within a time limit; (ii) the average optimality gap, computed as $\min\left\{\frac{ub-lb}{ub}, 1\right\}$, as well as the number of instances for which the gap is strictly less than 1 (reported within the parenthesis); and (iii) the average time t required to obtain the first upper bound ub. Averages are computed over the instances for which an upper bound is obtained within the time limit.

For the linear instances, we use a time limit of 120 seconds and we compare our heuristics with the performance of the MIQP solver of Gurobi [9] on a compact formulation of (1) (see [5]). For the MIQP solver of Gurobi, we show the results for both the first detected upper bound, as well as for the best incumbent solution found within the time limit.

The results for the linear instances are reported in Tables 1 and 2. In both cases, we observe that, for the smaller values of p, `PR-H` quickly finds a feasible partition and thus an upper bound. As the value of p increases, finding a feasible partition becomes increasingly harder, in particular for the case of $\ell = 100$ and $K = 3$, and the average time to find a first feasible solution is longer. Similarly, `TS-KA-H` is successful in improving the gap for smaller values of p. Whenever finding a first feasible partition is challenging, the basic heuristic cannot further improve the incumbent upper bound. Note that, for the facility location instances, starting from the initial partition found using k-means leads to generally better optimality gaps, however, for the knapsack instances there is no clear benefit in using k-means over a random initial partition.

Finally, both heuristics obtain upper bounds that are better than the first one found by the MIQP solver of Gurobi, and in some cases these bounds are even better than the best incumbent solution identified by the MIQP solver within the 120-second time limit.

For the quadratic instances, we considered a time limit of 30 minutes, both to compute the solution of the two-stage stochastic problem, to be used as a lower bound, and also for the heuristic methods. In the majority of the instances, the two-stage stochastic problem could not be solved within the time limit, thus the best lower bound found was used in place of the optimal solution as lb to compute the optimality gaps. The results for the quadratic knapsack instances are shown in Table 3, while we do not report the table for the quadratic facility location instances since, for all the settings we considered, no feasible solutions could be found within the time limit. Table 3 highlights how, for the quadratic instances, the average computation time needed to find a first feasible partition is much longer. While in most cases `PR-H` is able to find a feasible partition, the addition of the basic heuristic in `TS-KA-H` rarely succeeds in improving the best incumbent upper bound. As for the linear instances, different initial partitions lead to different results, but neither option consistently outperforms the other.

4 Conclusions

We study the K-adaptability approach for two-stage stochastic programs where uncertainty is represented by a finite set of scenarios. We generalize a basic heuristic approach from the literature to the two-stage setting. We highlight its

Table 1. Results for the two-stage stochastic knapsack problem with setup.

p	ℓ	K	PR-H						TS-KA-H						miqp					
			random			k-means			*random*			k-means			first			best		
			TL	gap	t	TL	gap	t	TL	gap	t	TL	gap	t	TL	gap	t	TL	gap	t
0.6	50	3	0	0.94 (1)	0.2	0	0.95 (3)	0.2	0	0.79 (2)	0.5	0	0.62 (4)	0.4	0	1.00 (0)	0.1	0	0.32 (5)	120.0
0.6	50	5	0	0.88 (3)	0.4	0	0.69 (5)	0.3	0	0.42 (5)	0.7	0	0.29 (5)	0.6	0	1.00 (0)	0.1	0	0.36 (5)	120.0
0.6	100	3	0	0.85 (2)	3.8	0	1.00 (0)	2.7	0	0.84 (2)	3.8	0	0.97 (1)	2.9	0	1.00 (0)	0.1	0	0.46 (5)	120.0
0.6	100	5	0	0.90 (1)	0.8	0	0.98 (1)	0.5	0	0.88 (1)	1.2	0	0.52 (5)	0.9	0	1.00 (0)	1.4	0	0.54 (5)	120.0
0.7	50	3	0	0.85 (2)	1.3	0	0.63 (5)	1.4	0	0.85 (2)	1.3	0	0.62 (5)	1.4	0	0.89 (1)	0.2	0	0.41 (5)	120.0
0.7	50	5	0	0.85 (2)	0.7	0	0.84 (2)	0.4	0	0.72 (3)	1.1	0	0.55 (4)	0.7	0	1.00 (0)	0.4	0	0.46 (5)	120.0
0.7	100	3	0	0.70 (4)	49.9	0	0.77 (3)	10.7	0	0.70 (4)	50.0	0	0.77 (3)	10.7	0	1.00 (0)	2.7	0	0.98 (1)	120.0
0.7	100	5	0	0.82 (4)	5.4	0	0.56 (5)	3.3	0	0.79 (4)	5.5	0	0.52 (5)	3.4	0	1.00 (0)	1.5	0	0.65 (5)	120.0
0.8	50	3	2	0.97 (1)	12.4	1	1.00 (0)	41.1	2	0.97 (1)	12.4	1	1.00 (0)	41.1	0	1.00 (0)	20.9	0	0.97 (1)	120.0
0.8	50	5	0	0.74 (5)	2.4	0	0.71 (4)	3.5	0	0.74 (5)	2.4	0	0.71 (4)	3.6	0	1.00 (0)	1.6	0	0.89 (5)	120.0
0.8	100	3	5	-	-	5	-	-	5	-	-	5	-	-	5	-	-	5	-	-
0.8	100	5	0	0.78 (5)	46.0	0	0.88 (5)	41.4	0	0.78 (5)	46.2	0	0.86 (5)	41.5	0	1.00 (0)	7.4	0	0.95 (2)	120.0

Table 2. Results for two-stage the stochastic capacitated facility location problem.

p	ℓ	K	PR-H						TS-KA-H						miqp					
			random			k-means			*random*			k-means			first			best		
			TL	gap	t	TL	gap	t	TL	gap	t	TL	gap	t_1	TL	gap	t	TL	gap	t
0.2	50	3	0	0.29 (5)	0.7	0	0.28 (5)	0.7	0	0.27 (5)	1.4	0	0.26 (5)	1.8	0	0.50 (5)	2.8	0	0.23 (5)	120.0
0.2	50	5	0	0.25 (5)	0.8	0	0.28 (5)	0.8	0	0.22 (5)	4.0	0	0.22 (5)	2.5	0	0.44 (5)	2.0	0	0.27 (5)	120.0
0.2	100	3	0	0.28 (5)	10.1	0	0.28 (5)	3.3	0	0.26 (5)	11.1	0	0.26 (5)	5.4	0	0.39 (5)	4.2	0	0.28 (5)	120.0
0.2	100	5	0	0.28 (5)	1.6	0	0.28 (5)	2.8	0	0.25 (5)	5.1	0	0.23 (5)	5.3	0	0.38 (5)	11.1	0	0.27 (5)	120.0
0.4	50	3	1	0.26 (4)	1.4	1	0.25 (4)	2.3	1	0.24 (4)	2.4	1	0.23 (4)	3.1	1	0.39 (4)	4.3	1	0.28 (4)	120.0
0.4	50	5	0	0.24 (5)	2.8	0	0.27 (5)	4.6	0	0.22 (5)	3.8	0	0.21 (5)	6.0	1	0.37 (4)	5.4	1	0.22 (4)	120.0
0.4	100	3	1	0.22 (4)	34.4	1	0.20 (4)	26.4	1	0.22 (4)	34.7	1	0.20 (4)	26.4	1	0.29 (4)	30.6	1	0.27 (4)	120.0
0.4	100	5	0	0.28 (5)	18.5	1	0.16 (4)	11.0	0	0.27 (5)	19.3	1	0.14 (4)	12.6	1	0.32 (4)	17.9	1	0.23 (4)	120.0
0.6	50	3	4	0.24 (1)	90.6	5	-	-	4	0.24 (1)	90.6	5	-	-	5	-	-	5	-	-
0.6	50	5	2	0.13 (3)	31.8	3	0.08 (2)	28.1	2	0.13 (3)	32.0	3	0.08 (2)	28.3	3	0.20 (2)	8.6	3	0.19 (2)	120.0
0.6	100	3	5	-	-	5	-	-	5	-	-	5	-	-	5	-	-	5	-	-
0.6	100	5	5	-	-	5	-	-	5	-	-	5	-	-	5	-	-	5	-	-

Table 3. Results for the quadratic two-stage stochastic knapsack problem with setup.

p	ℓ	K	PR-H						TS-KA-H					
			random			k-means			*random*			k-means		
			TL	gap	t	TL	gap	t	TL	gap	t	TL	gap	t
0.6	50	3	0	0.96 (1)	443.9	1	1.00 (0)	431.0	0	0.96 (1)	444.4	1	0.96 (1)	431.2
0.6	50	5	0	1.00 (0)	5.6	0	0.92 (2)	12.1	0	1.00 (0)	6.0	0	0.91 (2)	12.5
0.6	100	3	4	1.00 (0)	49.6	4	1.00 (0)	23.0	4	1.00 (0)	49.6	4	1.00 (0)	23.0
0.6	100	5	0	0.93 (1)	584.7	0	1.00 (0)	706.8	0	0.93 (1)	584.8	0	1.00 (0)	706.8
0.7	50	3	0	0.96 (1)	443.0	1	1.00 (0)	428.5	0	0.96 (1)	443.1	1	1.00 (0)	428.6
0.7	50	5	0	1.00 (0)	5.6	0	0.92 (2)	11.9	0	1.00 (0)	6.0	0	0.91 (3)	12.3
0.7	100	3	4	1.00 (0)	94.1	4	1.00 (0)	545.8	4	1.00 (0)	94.1	4	1.00 (0)	545.8
0.7	100	5	0	0.93 (1)	581.5	0	1.00 (0)	699.2	0	0.93 (1)	581.5	0	1.00 (0)	699.2
0.8	50	3	0	0.96 (2)	645.0	1	0.99 (1)	494.4	0	0.96 (2)	645.1	1	0.99 (1)	494.5
0.8	50	5	0	1.00 (0)	5.6	0	0.92 (2)	12.0	0	1.00 (0)	6.0	0	0.91 (3)	12.3
0.8	100	3	5	-	-	5	-	-	5	-	-	5	-	-
0.8	100	5	0	0.93 (1)	582.0	0	1.00 (0)	701.2	0	0.93 (1)	582.1	0	1.00 (0)	701.2

limitations in this context, when the feasibility of the K-adaptability problem cannot be guaranteed a priori. We develop a new heuristic method that explores different partitions of the scenario set, in order to find a feasible one, from which an upper bound of the K-adaptability problem can be computed. Once a feasible partition is found, the basic heuristic can be used to improve the current upper bound. We test these heuristics on linear and quadratic instances of two-stage stochastic knapsack and facility location problems. Experimental results on linear instances show that, in most cases, our method finds feasible solutions with small optimality gaps within short computation times, outperforming the MIQP solver of Gurobi. Moreover, our method is able to compute feasible solutions for two-stage stochastic quadratic knapsack instances. In contrast, no feasible solutions are obtained within a time limit of 30 minutes for two-stage stochastic quadratic facility location instances, highlighting the increased difficulty of the K-adaptability problem when a nonlinear objective function is considered.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Bertsimas, D., Caramanis, C.: Finite adaptability in multistage linear optimization. IEEE Transactions on Automatic Control **55**(12), 2751–2766 (2010)
2. Birge, J.R., Louveaux, F.: Introduction to stochastic programming. Springer (1997)
3. Buchheim, C., Kurtz, J.: Min–max–min robust combinatorial optimization. Mathematical Programming **163**(1), 1–23 (2017)
4. Buchheim, C., Pruente, J.: K-adaptability in stochastic combinatorial optimization under objective uncertainty. European Journal of Operational Research **277**(3), 953–963 (2019)
5. Donnini, F., De Santis, M., Kurtz, J.: K-adaptability for two-stage stochastic optimization. Optimization online (2025), https://optimization-online.org/?p=32681
6. Hanasusanto, G.A., Kuhn, D., Wiesemann, W.: K-adaptability in two-stage robust binary programming. Operations Research **63**(4), 877–891 (2015)
7. Malaguti, E., Monaci, M., Pruente, J.: K-adaptability in stochastic optimization. Mathematical Programming **196**(1), 567–595 (2022)
8. Malaguti, E., Monaci, M., Pruente, J.: Heuristic algorithms for k-adaptability. Available at SSRN 5266670 (2025)
9. Optimization, G.: Inc., gurobi optimizer. Gurobi optimizer
10. Subramanyam, A., Gounaris, C.E., Wiesemann, W.: K-adaptability in two-stage mixed-integer robust optimization. Mathematical Programming Computation **12**, 193–224 (2020)