BRNO UNIVERSITY OF TECHNOLOGY

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF MECHANICAL ENGINEERING

FAKULTA STROJNÍHO INŽENÝRSTVÍ

INSTITUTE OF MATHEMATICS

ÚSTAV MATEMATIKY

ADVANCED DECOMPOSITION METHODS IN STOCHASTIC CONVEX OPTIMIZATION

POKROČILÉ DEKOMPOZIČNÍ METODY VE STOCHASTICKÉ KONVEXNÍ OPTIMALIZACI

SHORT VERSION OF DOCTORAL THESIS

TEZE DIZERTAČNÍ PRÁCE

AUTHOR Ing. Jakub Kůdela

AUTOR PRÁCE

SUPERVISOR RNDr. Pavel Popela, Ph.D.

ŠKOLITEL

BRNO 2019

ABSTRACT

When working with stochastic programming problems, we frequently encounter optimization problems that are too large to be processed by routine methods of mathematical programming. However, in some cases the problem structure allows for a use of specialized decomposition methods that (when utilizing said structure) can be employed to efficiently solve very large optimization problems. This work focuses on two classes of stochastic programming problems that have an exploitable structure, namely two-stage stochastic programming problems and chance constrained problems, and the advanced decomposition methods that can be used to solve optimization problems in these two classes. We describe a novel warm-start cuts for the Generalized Benders Decomposition, which is used as a methods for the two-stage stochastic programming problems. For the class of chance constraint problems, we introduce an original decomposition method, that we named the Pool & Discard algorithm. The usefulness of the described decomposition methods is demonstrated on several examples and engineering applications.

KEYWORDS

stochastic optimization, stochastic programming, decomposition methods, two-stage stochastic programming problems, chance constrained problems

ABSTRAKT

Při práci s úlohami stochastického programování se často setkáváme s optimalizačními problémy, které jsou příliš rozsáhlé na to, aby byly zpracovány pomocí rutinních metod matematického programování. Nicméně, v někteých případech mají tyto problémy vhodnou strukturu, umožňující použití specializovaných dekompozičních metod, které lze použít při řešení rozsáhlých optimalizačních problémů. Tato práce se zabývá dvěma třídami úloh stochastického programování, které mají speciální strukturu, a to dvoustupňovými stochastickými úlohami a úlohami s pravděpodobnostním omezením, a pokročilými dekompozičními metodami, které lze použít k řešení problému v těchto dvou třídách. V práci popisujeme novou metodu pro tvorbu “warm-start” řezů pro metodu zvanou “Generalized Benders Decomposition”, která se používá při řešení dvoustupňových stochastických problémů. Pro třídu úloh s pravděpodobnostním omezením zde uvádíme originální dekompoziční metodu, kterou jsme nazvali “Pool & Discard algoritmus”. Užitečnost popsaných dekompozičních metod je ukázána na několika příkladech a inženýrských aplikacích.

KLÍČOVÁ SLOVA

stochastická optimalizace, stochastické programování, dekompoziční metody, úlohy dvoustupňového stochastického programování, úlohy s pravděpodobnostním omezením

KŮDELA, Jakub. *Advanced Decomposition Methods in Stochastic Convex Optimization*: short version of doctoral thesis. Brno: Brno University of Technology, Faculty of Mechanical Engineering, Institute of Mathematics, 2019. 27 p. Supervised by RNDr. Pavel Popela, Ph.D.

CONTENTS

1 INTRODUCTION

There are numerous introductory texts for stochastic programming. Among the ones we can recommend are the well-known books [17], [27], [4], and [28]. Since the purpose of this text is not to present a new insight into stochastic programming as such, the thesis uses for the introduction (Chapter 1) some parts and arguments from (in our opinion) an exceptional text [26].

The main body of the thesis comprises of three published articles [18, 19, 21] and a (for the time being) previously unpublished work comprising of description and numerical examination of a new algorithm for chance constrained optimization problems.

The article [18] (in Chapter 2) describes a decomposition algorithm suitable for two-stage convex stochastic problems called the Generalized Benders Decomposition and presents a new reformulation that incorporates a lower bound cut that serves as a warm-start, decreasing the overall computation time. Chapter 3 comprizes of the arcitle [21] – a real-world waste-management application of the two-stage stochastic programming problem, where the warm-start cut developed in Chapter 2 is utilized.

In Chapter 3 is described a new algorithm, called Pool & Discard algorithm (P&D algorithm), aimed at handling the chance constrained problems. It uses much of the theory of probabilistic robust design developed by Calafiore, Campi and Garatti in [6], [7], [8],[9], and [10] – this theory is summarized in the first, introductory, part of the chapter. In the remaining sections, the algorithm is described, examined and compared to other techniques for handling chance constrained problems (the ones presented in [1], [25], and [30]). Chapter 4 contains the article [19] – it describes a novel (convex) reformulation of otherwise rather involved engineering problem (optimal beam design). It can be also seen as the first application of the P&D algorithm with a trivial Pooling part (see the appropriate section in Chapter 3).

2 WARM-START CUTS FOR GENERALIZED BENDERS DECOMPOSITION

2.1 Main Ideas

In this section, we give a brief insight into the GBD, as it is not our intention to devote several pages to its thorough description. An interested reader can find an in-depth analysis of the method in the original paper [13] and in the works of Floudas in [11] and [12]. The problems GBD aims to solve are of the form:

$$\begin{array}{ll} \underset{x,y}{\text{minimize}} & f(x,y) \\ \text{subject to} & G(x,y) \leq 0, \quad x \in X, y \in Y, \end{array} \tag{2.1}$$

where $x \in X \subseteq \Re^{n_1}$, $y \in Y \subseteq \Re^{n_2}$, $f : \Re^{n_1} \times \Re^{n_2} \longrightarrow \Re$ is a real-valued objective function and $G : \Re^{n_1} \times \Re^{n_2} \longrightarrow \Re^m$ is an m-vector of constraint functions. The variable x is called a complicating variable in the sense that (2.1) is a much easier optimization problem in y when x is temporarily held fixed. The following conditions are required:

C1: Y is a nonempty, convex set and the functions f and G are convex for each fixed $x \in X$.

C2: The set

$$Z_x = \{z \in \Re^m : G(x,y) \leq z \text{ for some } y \in Y\}, \tag{2.2}$$

is closed for each fixed $x \in X$.

C3: For each fixed $x \in X \cap V$, where

$$V = \{x : G(x,y) \leq 0, \text{ for some } y \in Y\}, \tag{2.3}$$

one of the following conditions holds:

(i) the problem (2.1) has a finite solution and has an optimal multiplier vector for the inequalities.

(ii) the problem (2.1) is unbounded, that is, its objective function value goes to $-\infty$.

The particular situation we are interested in is when f and G are linearly separable in x and y, i.e.

$$\begin{array}{rcl} f(x,y) & = & f_1(x) + f_2(y), \\ G(x,y) & = & G_1(x) + G_2(y). \end{array} \tag{2.4}$$

The basic idea in GBD is the generation, at each iteration, of an upper bound and a lower bound on the optimal objective function value of (2.1).

2.2 GBD for Two-Stage Stochastic Programming Problems

In stochastic programming linear separability of the objective function and constraints is a very common property. Let us consider the following problem:

$$\begin{array}{ll} \underset{x,y_1,\ldots,y_K}{\text{minimize}} & f_1(x) + \sum\limits_{k=1}^{K} p(\xi_k) f_2(y_k, \xi_k) \\ \text{subject to} & G_{11}(x) \leq 0, \\ & G_{21}(\xi_k)x + G_{22}(y_k, \xi_k) \leq 0, \xi_k \in \Xi, \end{array} \tag{2.5}$$

where $f_1 : \Re^{n_1} \longrightarrow \Re$ is a convex function, all m_1 constraint functions $G_{11} : \Re^{n_1} \longrightarrow \Re^{m_1}$ are convex, and for all $\xi_k \in \Xi$ with $|\Xi| = K$ finite, $G_{21}(\xi_k)$ is a $m_2 \times n_1$ matrix, $f_2(\cdot, \xi_k) : \Re^{n_2} \longrightarrow \Re$ is convex, all m_2 constraint functions $G_{22}(\cdot, \xi_k) : \Re^{n_2} \longrightarrow \Re^{m_2}$ are convex, $P(\xi = \xi_k) \equiv p(\xi_k) > 0$, $\sum\limits_{k=1}^{K} p(\xi_k) = 1$.

The master problem corresponding to (2.5) has the following form:

$$\begin{array}{lll} \underset{x,\theta}{\text{minimize}} & f_1(x) + \theta & \\ \text{subject to} & G_{11}(x) \leq 0, & \\ & D_i x \leq d_i, & i = 1, \ldots, p, \\ & E_j x - \theta \leq e_j, & j = 1, \ldots, r, \end{array} \tag{2.6}$$

where $\theta \in \Re$ serves as the lower bound on the second stage objective value. Because of the structure of the two-stage stochastic programming problems, the subproblem separates into K independent subproblems (one for each scenario) in the form:

$$\begin{array}{ll} \underset{y_k}{\text{minimize}} & f_2(y_k, \xi_k) \\ \text{subject to} & G_{21}(\xi_k)x + G_{22}(y_k, \xi_k) \leq 0. \end{array} \tag{2.7}$$

2.3 Solution Procedure

Step 0. Set $c = 0, r = 0$, and $\epsilon > 0$.

Step 1. Solve (2.6) and obtain $(\bar{x}, \bar{\theta})$. The optimal objective value of (2.6) gives us a lower bound on optimal objective value of (2.5).

Step 2. For fixed $x = \bar{x}$ solve all K subproblems (2.7). One of two possibilities can happen.

Step 2A. For some k the subproblem (2.7) is infeasible. Solve the following problem:

$$\begin{array}{ll} \underset{y_k, v \geq 0}{\text{minimize}} & ||v||_1 \\ \text{subject to} & G_{21}(\xi_k)\bar{x} + G_{22}(y_k, \xi_k) \leq v, \end{array} \tag{2.8}$$

where $v \in \Re^{m_2}$ is a decision vector representing "slacks" in the constraints. Get $(\bar{y}_k, \bar{v})$ and from its dual obtain the optimal Lagrange multipliers λ. Set $c = c + 1$. Add a new row to the matrix D and vector d in (2.6):

$$D_c = \lambda^T G_{21}(\xi_k), d_c = \lambda^T(-G_{22}(\bar{y}_k, \xi_k)). \tag{2.9}$$

Return to Step 1.

Step 2B. All the subproblems have finite optimal values, we obtained $(\bar{y}_k, u_k)$, where u_k are optimal Lagrange multipliers. The evaluation of the objective of (2.5) at $(\bar{x}, \bar{y}_1, \ldots, \bar{y}_K)$ gives us an upper bound on its optimal value. Check for optimality: if

$$\bar{\theta} + \epsilon \geq \sum_{k=1}^{K} p(\xi_k) f_2(\bar{y}_k, \xi_k), \tag{2.10}$$

terminate, $(\bar{x}, \bar{y}_1, \ldots, \bar{y}_K)$ are ϵ-optimal [13]. Otherwise, set $r = r + 1$ and add a new row to the matrix E and vector e in (2.6):

$$\begin{aligned} E_r &= \sum_{k=1}^{K} p(\xi_k)(u_k^T G_{21}(\xi_k)), \\ e_r &= -\sum_{k=1}^{K} p(\xi_k)(f_2(\bar{y}_k, \xi_k) + u_k^T(G_{22}(\bar{y}_k, \xi_k)). \end{aligned} \tag{2.11}$$

Return to Step 1.

2.4 Reformulation with Bounding Cut

Let us define

$$\begin{aligned} \underset{x_k, y_k}{\text{minimize}} \quad & f_1(x_k) + f_2(y_k, \xi_k) \\ \text{subject to} \quad & G_{11}(x_k) \leq 0, \\ & G_{21} x_k + G_{22}(y_k, \xi_k) \leq 0, \end{aligned} \tag{2.12}$$

as the optimization problem for one particular realization $\xi_k \in \Xi$ and denote its optimal objective function value as $z(\xi_k)$. The wait-and-see solution is the solution without nonanticipativity constraints. We will denote the average of the optimal objective values of (2.12) as:

$$\text{WS} = \sum_{k=1}^{K} p(\xi_k) z(\xi_k). \tag{2.13}$$

Now we may compare this wait-and-see solution to the solution of (2.5). We will denote the optimal objective value of (2.5) as RP (the recourse problem [4]). The following inequality holds for any stochastic program:

$$\text{WS} \leq \text{RP}. \tag{2.14}$$

From this, we can see that WS creates a valid lower bound on the harder problem we are aiming to solve. The idea behind the reformulation is to include such a valid lower bound to the algorithmic procedure to "jumpstart" it and by doing so save on iterations, and, as a result, save on the overall computational effort and time. A natural temptation might be to solve a much simpler problem: the one obtained by replacing all random variables by their expected values. This is called the expected value problem, which is simply

$$\mathrm{EV} = z(\bar{\xi}), \tag{2.15}$$

where $\bar{\xi} = \sum_{k=1}^{K} p(\xi_k)\xi_k$.

Although WS is a valid bound, the computational effort for its enumeration is much higher compared to the effort to compute EV. However, EV does not necessarily have to play the role of a lower bound on RP; there are instances, where RP $\leq$ EV. For the purpose of deriving the reformulation, we will suppose that EV is a valid lower bound on RP. Suppose

$$\mathrm{EV} \leq \mathrm{RP}, \tag{2.16}$$

holds, then

$$f_1(x) + \sum_{k=1}^{K} p(\xi_k) f_2(y_k, \xi_k) \geq \mathrm{EV}, \tag{2.17}$$

holds for the optimum of (2.5). This inequality cannot be added directly to (2.5) since it would cease to be a convex program. The reformulation we propose is aimed at the master problem (2.6). A new variable z is introduced to bound the first-stage objective from above

$$f_1(x) \leq z, \tag{2.18}$$

which is a convexity preserving inequality. Furthermore, this new variable z added to the variable representing the second stage θ form a lower bound on the overall objective function. Finally, the bound

$$z + \theta \geq \mathrm{EV}, \tag{2.19}$$

since it is affine, can be added to (2.6), and the reformulation of the problem is

$$\begin{aligned}
\underset{z,x,\theta}{\text{minimize}} \quad & z + \theta \\
\text{subject to} \quad & f_1(x) \leq z, \\
& G_{11}(x) \leq 0, \\
& z + \theta \geq \mathrm{EV}, \\
& D_i x \leq d_i, \qquad i = 1, \ldots, p, \\
& E_j x - \theta \leq e_j, \quad j = 1, \ldots, r.
\end{aligned} \tag{2.20}$$

After this reformulation, the algorithm continues as usual, arriving at an ϵ-optimal solution in, preferably, a shorter time than its original counterpart.

Now, let us address what happens if (2.16) does not hold. One of two possibilities can occur, namely, that optimal objective function value (as determined by the algorithm) will be equal to EV, or that the problem will be infeasible. The price we pay for mistakenly using the cuts (2.16) is, in both cases, one iteration of the algorithm – i.e. after one iteration we can assess, if our algorithm will arrive at the desired solution, and, either restart it without (2.16) (possibly including WS instead), or continue.

However, certain situations can happen when we restart the algorithm without (2.16) and get the same result again. This occurs if the original problem is infeasible (in which case we have some serious model or data issues) or if EV = RP, in that case we would have to run the entire algorithm only to arrive at the same objective function value (which is a bit unfortunate, but unavoidable).

The solution procedure can be summarized in the following steps:

Step 0. Solve the expected value problem to get EV (2.15). Set $p = 0, r = 0$, and $\epsilon > 0$. Solve (2.20) and obtain $(\bar{z}, \bar{x}, \bar{\theta})$. If $\bar{z} + \bar{\theta} = \text{EV}$, terminate (and use the original method without the EV cut, or use WS instead). Otherwise, go to Step 2.

Step 1. Solve (2.20) and obtain $(\bar{z}, \bar{x}, \bar{\theta})$.

Step 2., Step 2A., Step 2B. The same as in section 2.3.

The rest of the chapter/article describes different implementation options (bunching and multicuts) as well as two numerical examples that demonstrate the usefulness of the warm-start formulation (as well as the usefulness of the bunching).

3 WASTE TRANSFER STATION PLANNING BY STOCHASTIC PROGRAMMING

3.1 Introduction

Since the situation in waste management is unknown due to the undecided support to the particular technology system and treatment from the government or the EU, the planning of future infrastructure is not secured from the investment point of view. This paper proposes a novel approach in the planning of transport infrastructure for efficient treatment of residual waste which is in line with all the possible cases of future development of waste management system. The future uncertainty (legislative development and support for different systems) in the treatment grid design is projected through the processing cost for different facilities at various locations. The computational approach was designed to handle real-life tasks in reasonable time. The case study is presented with the use of data from the Czech Republic.

3.2 Problem Description

The problem consists of deciding where to construct the transfer stations, what should be their respective capacities and from which producer of waste to which waste-processing plant should the cargo be send, provided that some of the data are uncertain. The road network is partly depicted in Fig 3.1 (and described by an incidence matrix A_1 in the mathematical model). This network had 24,770 arcs (roads) connecting the 6,258 nodes. The locations of the waste producers, the waste-processing plants and the possible locations for transfer stations are partially depicted in Fig 3.1. In the problem there were 6,258 places producing waste, 44 waste-processing plants (where 15 correspond to foreign facilities) and 116 possible places for the transfer stations.

To be able to differentiate between the transportation of waste that does or does not use the transfer stations, a separate road network was computed – for each possible transfer station was found the shortest path to each waste-processing plant. In this pre-processing step, 5,075 shortest path optimization problems were solved, resulting in the additional network with 5,075 arcs. The transfer of waste when using the transfer stations is assumed 3 times cheaper than the regular one. Each of the possible transfer stations can be constructed with 6 different capacities. Combining this with the 116 locations results in 696 binary first-stage decisions. The second-stage decisions are the flows on the arcs of the two networks and the amounts of waste processes at the plants, in total 29,889.

Type	Symbol	Description
Sets	$s \in S$	Set of scenarios
	$j \in J$	Set of nodes (cities)
	$i \in I \subset J$	Set of possible transfer stations
	$t \in T$	Set of possible options for transfer station capacities
Parameters	A_1	First incidence matrix (Fig. 3.1)
	A_2	Second incidence matrix (from pre-processing)
	c_1	Transfer costs, without the transfer stations (on A_1)
	c_2	Transfer costs, using transfer stations (on A_2)
	p_s	Probability of a scenario s
	$e_{i,t}$	Cost of a construction of a transfer station at location i, with capacity option t
	$k_{i,t}$	Capacity of a transfer station at location i with capacity option t
	$f_{j,s}$	Cost of processing waste at node j, scenario s
	r_j	Production of waste at node j
	q_j	Waste processing capacity of node j
Variables	$d_{i,t}$	Decision on building the transfer station at location i, with capacity option t; binary, first-stage
	$x_{1,s}$	Flows on A_1 in scenario s; continuous, second-stage
	$x_{2,s}$	Flows on A_2 in scenario s; continuous, second-stage
	$y_{j,s}$	Amount of processed waste in node j, scenario s; continuous, second-stage

Table 3.1: The notation.

The uncertain parameters that are considered in the model are the costs for processing the waste at the 44 different plants, which correspond with the legislation development and local conditions (such as the demand for heat, etc.). The number of scenarios for this model was set to 1,000 and so the model has almost 30M variables. The notation that is used to develop the mathematical model is described in Table 3.1.

To simplify the notation, some subscripts were hidden, meaning that the appropriate parameters/variables were stacked to form a vector of a fitting size (and the associated equalities/inequalities are meant for each element in the vector). The

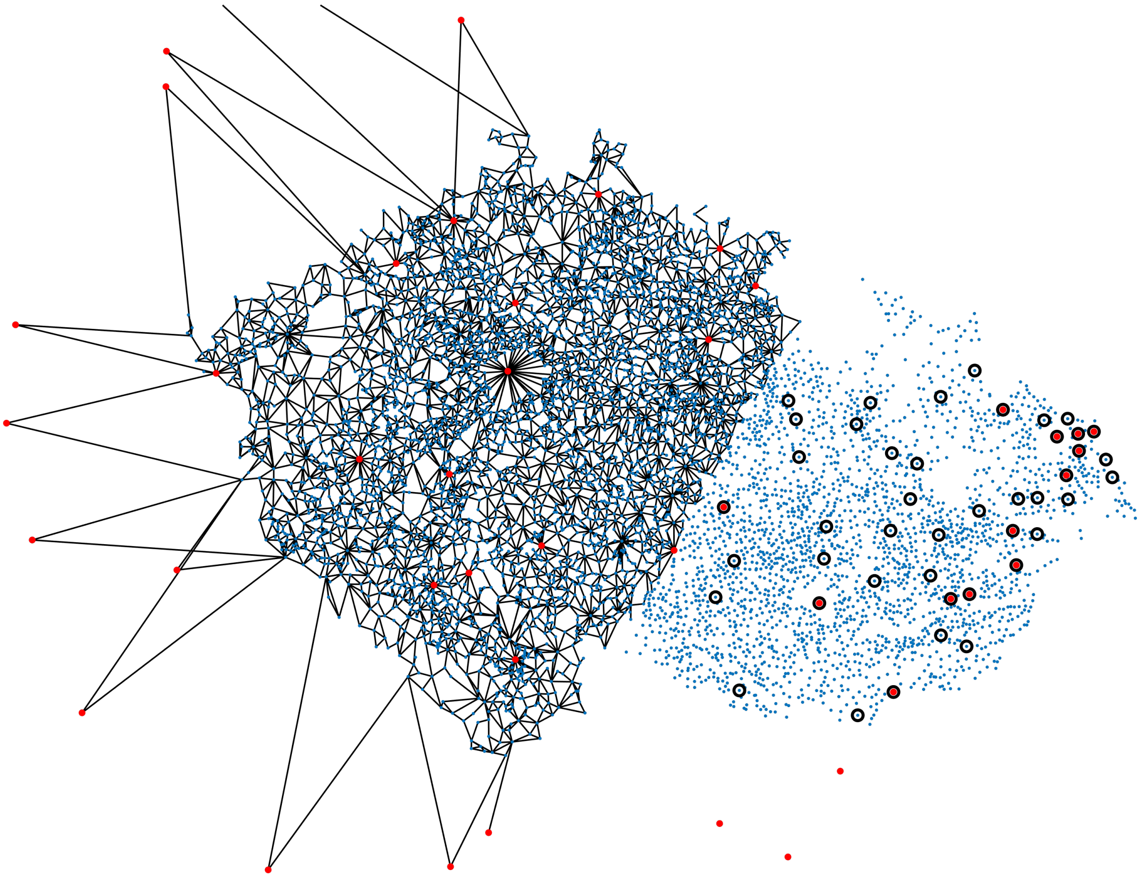


Fig. 3.1: A map showing the producers of waste (blue dots), the places processing waste (red dots). The road network (black lines) and the possible transfer stations (black rings) are shown on two separate parts.

mathematical model has the following form:

$$\begin{aligned}
\text{minimize} \quad & \sum_{i \in I, t \in T} e_{i,t} d_{i,t} + \sum_{s \in S} p_s (c_1^T x_{1,s} + c_2^T x_{2,s} + f_s^T y_s) && (3.1)\\
\text{subject to} \quad & A_1 x_{1,s} + A_2 x_{2,s} + y_s = r, \quad \forall s \in S, && (3.2)\\
& y_s \leq q, \quad \forall s \in S, && (3.3)\\
& \sum_{\text{flows from } i \in I} x_{2,s} \leq \sum_{t \in T} k_{i,t} d_{i,t}, \quad \forall s \in S, \forall i \in I, && (3.4)\\
& \sum_{t \in T} d_{i,t} \leq 1, \quad \forall i \in I, && (3.5)\\
& x_{1,s}, x_{2,s}, y_s \geq 0, \quad \forall s \in S, && (3.6)\\
& d_{i,t} \in \{0, 1\}, \quad \forall i \in I, \forall t \in T. && (3.7)
\end{aligned}$$

The objective function given by (3.1) is the expected waste transportation and processing costs and the building cost for building the transfer plants. The constraint (3.2) is the conservation of waste – at each node and for each scenario, the amount produced must be equal to the amount transported (by one of the two possibilities) plus the amount processed. The constraint (3.3) is an upper bound on the amount of waste that can be processed at a given node. The constraint (3.4) guarantees that the amount transferred using the transfer station i is less than the installed capacity of that transfer station. The constraint (3.5) ensures that at most one of the possible capacities is installed at location i. The last two constraints (3.6) and (3.7) are the nonnegativity and integrality constraint, respectively. The only constraints that do not depend on the scenarios are (3.5) and (3.7). The total number of constraints that depend on scenarios is 36,307, meaning that the model has over 36M constraints.

3.3 Implementation and Results

The model was solved using the Benders decomposition scheme described in [20] enhanced by the warm-start cuts developed in [18]. The results of the computation are best summarized in Fig 3.2 and Fig 3.3. Of the 116 possible locations, 71 were chosen as optimal places for the transfer stations. One scenario of optimal flows and the optimal places for the transfer stations is depicted in Fig 3.2 (the optimal places are the same for all scenarios, the flows are different).

The optimal expected cost was 260.14M EUR and the expected total distance traveled by all vehicles was 8.23M km, assuming that the regular flows are serviced by vehicles with capacity 10 t and the flows from transfer stations are serviced by vehicles with capacity 24 t (all fully loaded).

The histograms in Fig 3.3 represent the results for the 1,000 generated scenarios and show in detail the impact of building the transfer stations. The expected costs are 8% lower on average when building the transfer stations, the costs for transportation alone are 21% lower. The expected total distance traveled by all vehicles is reduced by 9% on average when building the transfer stations. However, this quantity has a much higher variance and, in some scenarios, is worse than the situation with no transfer stations. This inconvenience stems from the objective focusing only on costs – if some form of trade-off between costs and total distance was added to the objective function, the results would be more favourable towards lower total distance (at the price of increased costs). This might represent the situation when taking into account the environmental aspects is more important than the overall cost.

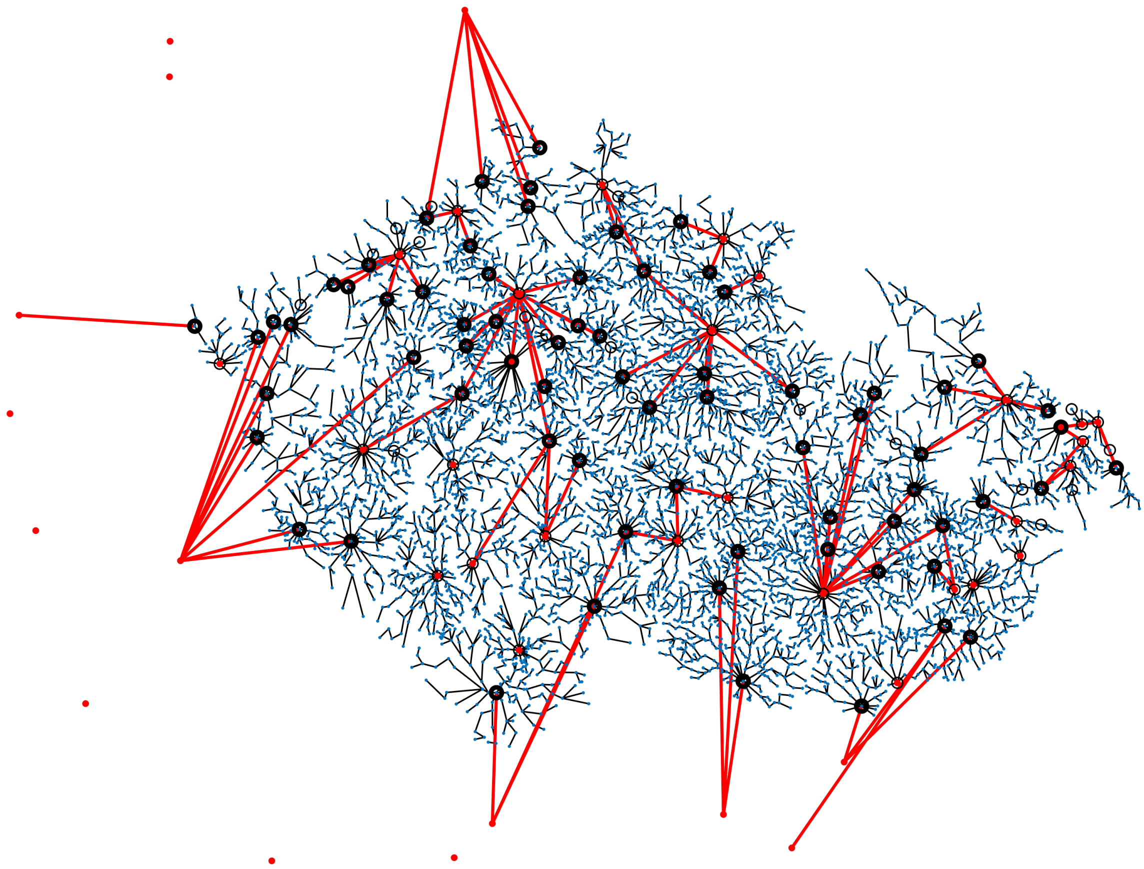


Fig. 3.2: A map showing the results for one of the scenarios – thick black rings correspond to the selected places for transfer stations, red lines are flows from these transfer stations, black lines are regular flows.

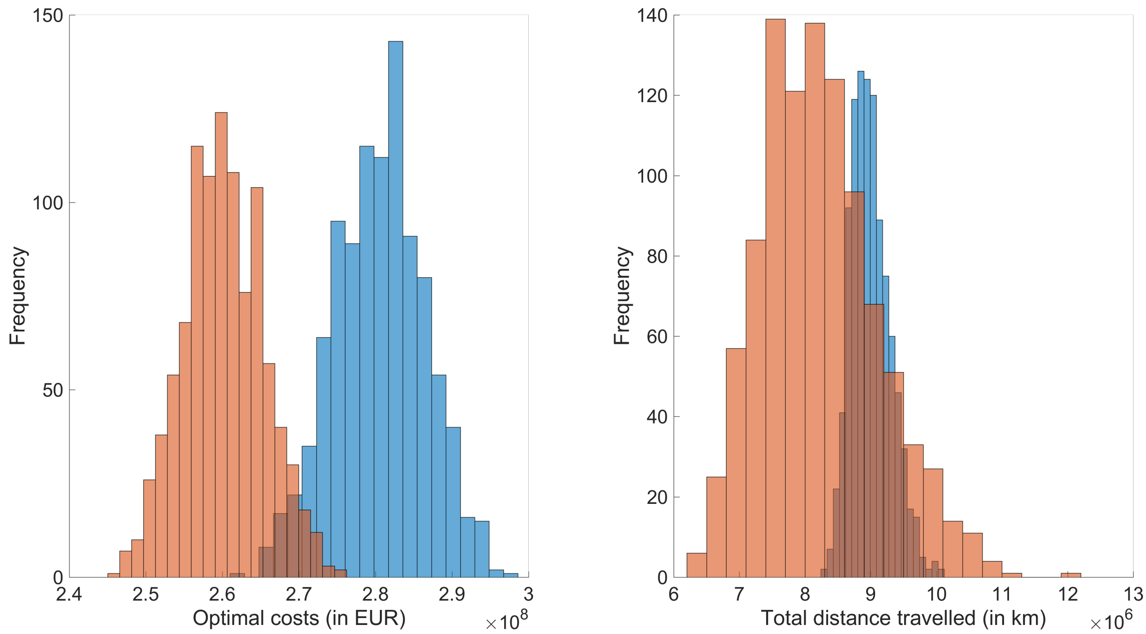


Fig. 3.3: Histograms of optimal cost and total distance travelled – the red one is using the transfer stations, the blue one is not.

4 CHANCE CONSTRAINED PROBLEMS

4.1 Introduction

The introduction into the topic is derived (more or less directly) from [9] – with most of the used notation adapted from [9] as well. Let $\mathcal{X} \subseteq \Re^{n_x}$ be a convex and closed domain of optimization and consider a family of constraints $x \in \mathcal{X}_\xi$ parameterized in $\xi \in \Xi$. The uncertain parameter ξ describes different instances of an uncertain optimization scenario. We adopt a probabilistic description of uncertainty and suppose that the support Ξ for ξ is endowed with a σ-algebra $\mathcal{D}$ and that a probability measure $\mathcal{P}$ is defined over $\mathcal{D}$. The probability measure $\mathcal{P}$ describes the probability with which the uncertain parameter ξ takes value in Ξ. Then, a chance constrained optimization program is written as:

$$\begin{aligned} \text{CCP}_\epsilon : \quad & \underset{x \in \mathcal{X}}{\text{minimize}} \quad c^T x \\ & \text{subject to} \quad \mathcal{P}\{\xi : x \in \mathcal{X}_\xi\} \geq 1 - \epsilon. \end{aligned} \tag{4.1}$$

Here, we assume that the σ-algebra $\mathcal{D}$ is large enough, so that $\{\xi : x \in \mathcal{X}_\xi\} \in \mathcal{D}$, i.e. $\{\xi : x \in \mathcal{X}_\xi\}$ is a measurable set. Also, linearity of the objective function can be assumed without loss of generality, since any objective of the form

$$\underset{x \in \mathcal{X}}{\text{minimize}} \quad c(x),$$

where $c(x) : \mathcal{X} \to \Re$ is a convex function, can be re-written as

$$\underset{x \in \mathcal{X}, y \geq c(x)}{\text{minimize}} \quad y,$$

where y is a scalar variable.

In the CCP_ϵ (4.1), constraint violation is tolerated, but the violated constraint set must be no larger than ϵ.

4.2 Sample Counterpart

We can view the variable $x \in \mathcal{X} \subseteq \Re^{n_x}$ as the "design variable". The family of possible instances is parameterized by an "uncertainty vector" $\xi \in \Xi \subseteq \Re^{n_\xi}$.

Assumption 4.2.1 (Convexity)
For each $\xi \in \Xi$ the sets $\mathcal{X}_\xi$ are convex and closed. ⋆

Assumption 4.2.1 requires convexity only with respect to the design variable x. We have the following definition:

Definition 4.2.2 (Probability of Violation)
Let $x \in \mathcal{X}$ be given. The probability of violation of x is defined as

$$\mathcal{V}(x) = \mathcal{P}\{\xi \in \Xi : g(x, \xi) > 0\}.$$

A solution x with small associated $\mathcal{V}(x)$ is feasible for most of the problem instances, i.e., it is approximately feasible for the worst-case problem. Any such solution is here named an "ϵ-level" solution:

Definition 4.2.3 (ϵ-Level Solution)
Let $\epsilon \in (0, 1)$. We say that $x \in \mathcal{X}$ is an ϵ-level robustly feasible (or, more simply, an ϵ-level) solution, if $\mathcal{V}(x) \leq \epsilon$.

Our ultimate goal is to devise an algorithm that returns a ϵ-level solution. To this purpose, we now introduce the "scenario" version of the worst-case design problem. In the "scenario design" we optimize the objective subject to a finite number of these randomly selected scenarios.

Definition 4.2.4 (Scenario Design Problem)
Assume that S independent identically distributed samples $\xi^1, \ldots, \xi^S$ are drawn according to probability $\mathcal{P}$. A scenario design problem is given by the convex program

$$\begin{array}{lll} \mathrm{SDP}_S: & \underset{x \in \mathcal{X}}{\text{minimize}} & c^T x \\ & \text{subject to} & g(x, \xi^i) \leq 0, \quad i = 1, \ldots, S. \end{array} \tag{4.2}$$

Assumption 4.2.5 (Feasibility)
For all possible extractions $\xi^1, \ldots, \xi^S$, the optimization problem (4.2) is either infeasible, or, if feasible, it attains a unique optimal solution.

The scenario problem SDP_S is a standard convex optimization problem with a finite number of constraints S. Since the constraints $g(x, \xi^i) \leq 0$ are randomly selected, the resulting optimal solution $\hat{x}_S$ is a random variable that depends on the multi-sample extraction $(\xi^1, \ldots, \xi^S)$. Therefore, $\hat{x}_S$ can be a ϵ-level solution for a given random extraction and not for another. In the theorem, the parameter β bounds the probability that is not a ϵ-level solution. In other words, we are looking for S big enough, such that the following inequality holds:

$$\sum_{i=0}^{n_x - 1} \begin{pmatrix} S \\ i \end{pmatrix} \epsilon^i (1 - \epsilon)^{S-i} \leq \beta. \tag{4.3}$$

The best explicit bound for this quantity to date can be found in [2]:

Theorem 4.2.6 (Feasibility I)
Let Assumption 4.2.5 be satisfied. Fix two real numbers $\epsilon \in (0,1)$ (level parameter) and $\beta \in (0,1)$ (confidence parameter). If

$$S \geq \left\lceil \frac{1}{\epsilon}\left(\ln\frac{1}{\beta} + n_x + \sqrt{2n_x \ln\frac{1}{\beta}}\right)\right\rceil \tag{4.4}$$

then, with probability no smaller than $1-\beta$, either the scenario problem SDP_S is infeasible and, hence, also the initial robust convex program is infeasible; or, SDP_S is feasible, and then its optimal solution $\hat{x}_S$ is ϵ-level robustly feasible.

Of the S generated scenarios, only some of these S will be "bounding" in the sense that they prevent the solution from "falling" to a lower objective value.

Definition 4.2.7 (Support Scenario)
Scenario $\xi^i, i \in \{1,\dots,S\}$, is a support scenario for the scenario problem SDP_S if its removal changes the optimal solution of SDP_S.

The following theorem, whose proof can be found in [8] or in a different form in [23], gives us the bound on the number of support scenarios:

Theorem 4.2.8 (Number of Support Scenarios)
The number of support scenarios for SDP_S is at most n_x, the size of x.

The number of support scenarios does not depend on the number of generated scenarios S. If all the S constraints are enforced, however, one cannot expect that good approximations of chance constrained solutions are obtained (cf. the numerical examinations in the thesis). Thus, we want to allow the solution to violate part of the sampled constraints to improve its objective value. A general removal procedure is formalized in the following definition:

Definition 4.2.9 (Constraint Removal Algorithm)
Let $k < S$. An algorithm $\mathcal{A}$ for constraints removal is any rule by which k constraints out of a set of S constraints are selected and removed. The output of $\mathcal{A}$ is the set $\mathcal{A}\{\xi^1,\dots,\xi^S\} = \{i_1,\dots,i_k\}$ of the indexes of the k removed constraints.

The sample-based optimization program where k constraints are removed as indicated by $\mathcal{A}$ is expressed as

$$\begin{aligned} \mathrm{SDP}^{\mathcal{A}}_{S,k}: \quad & \underset{x\in\mathcal{X}}{\text{minimize}} \quad c^T x \\ & \text{subject to} \quad g(x,\xi^i) \leq 0, \quad i \in \{1,\dots,S\} \setminus \mathcal{A}\{\xi^1,\dots,\xi^S\}, \end{aligned} \tag{4.5}$$

and its solution will be hereafter indicated as $x^*_{S,k}$. We introduce the following assumptions:

Assumption 4.2.10 (Constraint Violation)
Almost surely with respect to the multi-sample $(\xi^1, \ldots, \xi^S)$*, the solution* $x^*_{S,k}$ *of the sample-based optimization program* $\mathrm{SDP}^{\mathcal{A}}_{S,k}$ *violates all the* k *constraints that* $\mathcal{A}$ *has removed.*

The next Theorem (proved in [9]) provides theoretical guarantees that $\mathcal{V}(x^*_{S,k}) \leq \epsilon$, i.e. that the optimal solution $x^*_{S,k}$ of the optimization program $\mathrm{SDP}^{\mathcal{A}}_{S,k}$ is feasible for the CCP_ϵ.

Theorem 4.2.11 (Feasibility)
Let $\beta \in (0,1)$ *be any small confidence parameter value. If* S *and* k *are such that*

$$\begin{pmatrix} k + n_x - 1 \\ k \end{pmatrix} \sum_{i=0}^{k+n_x-1} \begin{pmatrix} S \\ i \end{pmatrix} \epsilon^i (1-\epsilon)^{S-i} \leq \beta, \tag{4.6}$$

then $\mathcal{P}^S\{\mathcal{V}(x^*_{S,k}) \leq \epsilon\} \geq 1 - \beta$.

4.3 Pool & Discard Algorithm

4.3.1 Pooling Part

The idea behind the Pooling part of the algorithm is the following: if one were to verbally describe the problem (4.2), the one word that came to our mind was "long", as there are much more constraints than decision variables. Moreover, the number of support constraints (or support scenarios), that the optimal solution of (4.2) depends upon is very small, when compared to the overall number of constraints (or scenarios).

The method consists of solving (4.2) by the following procedure. First, we start by completely neglecting the constraints in (4.2) that correspond to the different scenarios and solve this relaxed optimization problem. Then we find the most violated constraints, add them to the relaxed problem and find a new optimal solution – this step heavily exploits warm-starting. The Pooling part can be summarized as follows:
Step 0. Set $\delta > 0$, $\mathcal{I} = \emptyset$.
Step 1. Solve the following problem:

$$\begin{aligned} \underset{x \in \mathcal{X}}{\text{minimize}} \quad & c^T x \\ \text{subject to} \quad & g(x, \xi^i) \leq 0, \quad i \in \mathcal{I}, \end{aligned} \tag{4.7}$$

and obtain a solution $\hat{x}$.
Step 2. Check feasibility of the solution by computing the slacks s^i:

$$s^i = g(\hat{x}, \xi^i), \quad i \in \{1, \ldots, S\}. \tag{4.8}$$

Step 3. If $\max_{i\in\{1,\dots,S\}} s^i > \delta$, find the associated index of the maximum value $\hat{i} = \operatorname{argmax}_{i\in\{1,\dots,S\}} s^i$, add it to the set $\mathcal{I}$ and return to Step 1. Otherwise, set $x^* = \hat{x}, \mathcal{I}^* = \mathcal{I}$ and terminate.

The parameter δ, can (theoretically) be set to zero, but there are implementation issues that would lead to unfavourable results, cf. Section 4.3.3. By the end of this procedure, we not only get the optimal solution of (4.2), but also an index set $\mathcal{I}$ that contains the support scenarios.

4.3.2 Discarding Part

The Discarding part of the algorithm consists of utilizing the index set $\mathcal{I}$, finding the support scenarios among this set and finding the one scenario, whose removal decreases the optimal objective value the most – this is repeated k times, where k is either set a priori, or is terminated once an estimate of the probability of violation of obtained solution $\mathcal{V}(x)$ reaches certain threshold. The Discarding part can be summarized as follows:

Step 0. Solve the pooling part described above to obtain $\mathcal{I}^*$ and x^*. Set $\gamma > 0, k > 0, \mathcal{I}_p = \emptyset$.

Repeat k times, or terminate once an estimate of $\mathcal{V}(x^*)$ reaches a threshold:

Step 1. Find the set of support scenarios $\mathcal{I}_r \subset \mathcal{I}^*$ – either by examining the slacks $(s^i > -\gamma)$ or the associated dual variables $(\mu^i > \gamma)$.

Step 2. For each of the support scenarios $i_r \in \mathcal{I}_r$, solve the following problem:

$$\begin{aligned} \underset{x\in\mathcal{X}}{\text{minimize}} \quad & c^T x \\ \text{subject to} \quad & g(x,\xi^i) \le 0, \quad i \in \{1,\dots,S\} \setminus \{i_r \cup \mathcal{I}_p\}, \end{aligned} \tag{4.9}$$

using the Pooling part, warm-started by using $\mathcal{I} = \mathcal{I}^* \setminus i_r$ and $x = x^*$. Denote the solution to (4.9) as $x^*_{i_r}$, its optimal objective function value v^*_r and its final set of scenarios $\mathcal{I}^*_r$.

Step 3. Find the index with the best optimal objective value: $i^* = \operatorname{argmin}_{i_r} v^*_r$. Set $x^* = x^*_{i^*}, \mathcal{I}^* = \mathcal{I}^*_{i^*}$ and add the corresponding scenario to the set of permanently discarded ones $\mathcal{I}_p$.

The parameter γ can be, in theory, set to 0. What discourages us from doing so are the implementation issues discussed in Section 4.3.3.

4.3.3 Implementation

For our implementation we chose a relatively new programming language `Julia` [3], that is designed for high-performance numerical computing, and a modeling system

`JuMP`, which is domain-specific modeling language for mathematical optimization embedded in `Julia`. `JuMP` supports a wide variety of solvers, model modifications, warm-stars, and even different solver callbacks (lazy constraints, etc.) that, even they are not useful for the P&D algorithm, make it a very powerful modeling tool. The solver suitable fot the implementation of the P&D algorithm was `CPLEX 12.7` [16].

The parameter δ is the required feasibility of the solution. In theory, this could be set to zero, to guarantee that the solution of the Pooling part "really" solves the SDP_S formulation (4.2). The problem is that in some solvers the "optimal" solution they provide is not always strictly feasible. Among these solvers are `CPLEX` and `GUROBI` [14]. In `CPLEX`, the parameter that sets the tolerance for the feasibility of the optimal solution is `CPX_PARAM_EPRHS`, has a default value of 10^{-6} and can be set anywhere between 10^{-9} and 10^{-1}, but not to 0. In `GUROBI`, this parameter is called `FeasibilityTol`. This is the reason we need a nonzero δ, because when set to zero, the Pooling part can end up in an infinite loop, because it cannot produce a feasible point.

The second parameter, γ in the Discarding part, controls which scenarios will be treated as possible support scenarios. From complementary slackness [5] we know that any primal optimal x^* and dual optimal μ the following holds:

$$\mu^i g(x^*, \xi^i) = 0, i \in \{1, \ldots, S\}.$$

We can express this, equivalently as

$$\mu^i > 0 \Longrightarrow g(x^*, \xi^i) = 0,$$

or

$$g(x^*, \xi^i) < 0 \Longrightarrow \mu^i = 0.$$

Depending on whether or not we have access to the dual optimal μ, we inspect either the slacks or the dual variables μ, to find set of possible support scenarios. The issue of setting γ to 0 is one of numerical computing (and the feasibility tolerance mentioned earlier) – when reporting the optimal dual variables μ the solvers rarely return exactly 0, more often, we get values ranging from 10^{-8} to 10^{-16}. If we did set γ to 0 we would (likely) have to consider all the scenarios as possible support scenarios and the execution of the algorithm would be significantly prolonged.

5 CHANCE CONSTRAINED OPTIMAL BEAM DESIGN: CONVEX REFORMULATION AND PROBABILISTIC ROBUST DESIGN

5.1 Introduction

Optimal design problems in engineering frequently lead to optimization problems involving differential equations. One of the classes of these problems is shape optimization [15]. The particular shape optimization problem considered in this paper is the optimal design of a beam (be it a fixed beam, a cantilever beam, etc.) subjected to some kind of loading. This problem was previously also examined in [29] and [32], where the authors used the finite element method (FEM) and the finite difference method to approximate the ordinary differential equations (ODE) and solve the problem. Our paper shows that this beam design problem can be formulated as a geometric programming problem, which can be further transformed into a convex one, and thus can be efficiently solved. An important issue regarding the design is its reliability (see [22]). In the context of this paper the reliability of the design will mean that the constraints in the resulting optimization program should hold with high probability. In this paper we investigate the chance constrained beam design problem under more complicated random loads. We utilize the sampling approach (called Probabilistic Robust Design) developed in [7], [9] and [10] to obtain a manageable approximation of the chance constrained problem and use a scenario-deletion method to compute a trade-off between the reliability of the design and the objective value.

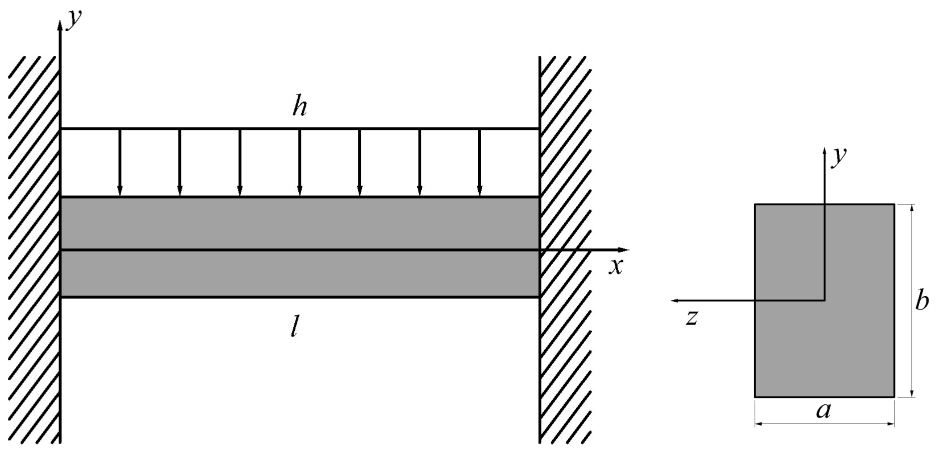
5.2 Problem Formulation

The problem is best described by Fig. 5.1. We consider a fixed beam of length l with rectangular cross-section that is subjected to a load $h(x)$ (with the opposite direction than the axis y), which is depicted in Fig. 5.1a. The task is to find the optimal design, in terms of the cross-section dimensions a and b (Fig. 5.1b), that minimizes the weight of the beam.

Naturally, given a load $h(x)$ the beam will deflect and will be subjected to a bending stress. The requirement for the design is that the maximum stress in the beam is less then a material-specific constant, that ensures that the design is safe. The problem can be formulated as the following ODE-constrained optimization program:

$$
\begin{aligned}
\underset{a,b,v(x)}{\text{minimize}} \quad & \rho a b l && && (5.1)\\
\text{subject to} \quad & E\frac{ab^3}{12}\frac{d^4v}{dx^4}(x) = h(x), && x \in [0,l], && (5.2)\\
& |E\frac{b}{2}\frac{d^2v}{dx^2}(x)| \leq \sigma_M, && x \in [0,l], && (5.3)\\
& v(0) = 0,\ \ \frac{dv}{dx}(0) = 0,\ \ v(l) = 0,\ \ \frac{dv}{dx}(l) = 0, && && (5.4)\\
& a_L \leq a \leq a_U, \quad b_L \leq b \leq b_U, && && (5.5)
\end{aligned}
$$

where ρ is the density of the material, $v(x)$ is the deflection of the beam (with the opposite direction than the axis y) in a point $x \in [0,l]$, E is the Young modulus, σ_M is the maximum stress allowed, and a_L, a_U, b_L, b_U are the bounds on the cross-section dimensions. The constraint (5.2) is the ODE that governs the deflection of the beam $v(x)$ given a specific load $h(x)$. The constraint (5.3) is the maximum allowed stress in the beam. The constraint (5.4) defines the boundary conditions for the ODE (i.e. that we have a fixed beam).



(a) The scheme of loaded beam. (b) Beam cross-section.

Fig. 5.1: The problem geometry.

5.2.1 FEM Problem Approximation and Solution

To tackle the problem (5.1)-(5.5) we use the FEM to approximate the ODE in (5.2) and (5.3). Following [31] (p. 25 – 27), the FEM approximation of the problem

(5.1)-(5.5) is then the following:

$$\begin{aligned} \underset{a,b,\mathcal{V}}{\text{minimize}} \quad & \rho abl & (5.6) \\ \text{subject to} \quad & E\frac{ab^3}{12}\mathbb{K}\mathcal{V} = h, & (5.7) \\ & |E\frac{b}{2}\mathbb{C}\mathcal{V}| \leq \sigma_M, & (5.8) \\ & a_L \leq a \leq a_U, \quad b_L \leq b \leq b_U. & (5.9) \end{aligned}$$

This problem has $2N$ variables ($2N+2$ in $\mathcal{V}$ of which 4 are fixed by boundary conditions, and 2 design variables), $2N+2$ constraints and a box constraints on a and b, and is non-convex. The crucial realization is that the stiffness matrix $\mathbb{K}$ is, by design, always invertible. Using this fact, we can rewrite (5.7) as:

$$\mathcal{V} = \frac{12}{Eab^3}\mathbb{K}^{-1}h, \tag{5.10}$$

and (5.8) becomes:

$$|\frac{6}{ab^2}\mathbb{C}\mathbb{K}^{-1}h| = \frac{1}{ab^2}|6\mathbb{C}\mathbb{K}^{-1}h| \leq \sigma_M. \tag{5.11}$$

Let us denote as v_M the maximum of $|6\mathbb{C}\mathbb{K}^{-1}h|$ over all the nodes of the FEM discretization. Since σ_M is the same for all $N+1$ nodes, the $N+1$ inequalities (5.8) are equivalent to a single inequality:

$$\frac{v_M}{ab^2} \leq \sigma_M. \tag{5.12}$$

Utilizing these results and neglecting the constants ρ and l in the objective (5.6), we can reformulate the problem (5.6)-(5.9) as the following equivalent problem:

$$\begin{aligned} \underset{a,b}{\text{minimize}} \quad & ab & (5.13) \\ \text{subject to} \quad & \frac{v_M}{ab^2} \leq \sigma_M, \quad a_L \leq a \leq a_U, \quad b_L \leq b \leq b_U, & (5.14) \end{aligned}$$

which is a geometric program, that can be transformed into a convex program. This problem has the following analytic solution (that is derived in the Appendix A in the thesis):

- if $\frac{v_M}{a_U b_U^2} > \sigma_M$, the problem is infeasible,
- if $\frac{v_M}{a_L b_L^2} \leq \sigma_M$, the solution is $a^* = a_L, b^* = b_L$,
- if $b = \sqrt{\frac{v_M}{a_L \sigma_M}}$ is within the bounds, $b^* = b, a^* = a_L$,
- else $a = \frac{v_M}{b_U^2 \sigma_M}$ and $a^* = a, b^* = b_U$.

5.2.2 Additional Variable, Constraints and Convex Reformulation

The structure of the problem allows us to consider the material constant E as a variable, without destroying the convexity of the upcoming reformulation. An additional restriction on the solution involves the maximum absolute deflection of the beam, which we denote as δ_M. In our FEM formulation, the vector $\mathcal{V}$ includes both the deflection of the beam and its first derivative in each node of the division.

$$|\mathcal{V}_i| \leq \delta_M, i = 1, 3, 5, \ldots, 2N+1, \tag{5.15}$$

which is equivalent to a single inequality

$$\max_{i=1,3,5,\ldots,2N+1} |\mathcal{V}_i| \leq \delta_M, \tag{5.16}$$

using (5.10) and denoting the maximum of the odd components of $|12\mathbb{K}^{-1}h|$ as w_M we get

$$\frac{w_M}{Eab^3} \leq \delta_M. \tag{5.17}$$

The final constraint restricts the ratio between b and a to be less then the maximum allowed r_M.

Adding these constraints to (5.13)-(5.14), treating E as a design variable and changing the objective yields the following geometric program:

$$\begin{aligned}
\underset{a,b,E}{\text{minimize}} \quad & E^p ab && (5.18)\\
\text{subject to} \quad & \frac{v_M}{\sigma_M} a^{-1} b^{-2} \leq 1, && (5.19)\\
& \frac{w_M}{\delta_M} E^{-1} a^{-1} b^{-3} \leq 1, && (5.20)\\
& \frac{1}{r_M} b a^{-1} \leq 1, && (5.21)\\
& a_L a^{-1} \leq 1, \frac{1}{a_U} a \leq 1, \;\; b_L b^{-1} \leq 1, \frac{1}{b_U} b \leq 1, \;\; E_L E^{-1} \leq 1, \frac{1}{E_U} E \leq 1, && (5.22)
\end{aligned}$$

where all the coefficients of the monomials in (5.18)-(5.22) are clearly positive, meaning we can use the following transformation to derive an equivalent linear program:

$$
\begin{aligned}
\underset{y_a, y_b, y_E}{\text{minimize}} \quad & y_a + y_b + p \cdot y_E && (5.23)\\
\text{subject to} \quad & -y_a - 2y_b + \log v_M - \log \sigma_M \leq 0, && (5.24)\\
& -y_a - 3y_b - y_E + \log w_M - \log \delta_M \leq 0, && (5.25)\\
& -y_a + y_b - \log r_M \leq 0, && (5.26)\\
& \log a_L \leq y_a \leq \log a_U,\ \ \log b_L \leq y_b \leq \log b_U,\ \ \log E_L \leq y_E \leq \log E_U. && (5.27)
\end{aligned}
$$

5.3 Random Loads and Robust Solution

In this paper, we assume that the randomness is in the load h. Instead of specifying the distribution of h by its cumulative distribution function or moment generating function (that would allow us to use the Bernstein approximation [24]), we devised a mechanism that produces random samples/scenarios. The sampling procedure is the following ($\mathcal{U}(a, b)$ denotes a uniform distribution):

0. Pick a random integer i between 1 and 4. Set $h(x) = 0$.
1. Repeat i times: Generate a Bernoulli trial.
   a) If 0, randomly pick 4 points $0 \leq x_a \leq x_b \leq x_c \leq x_d \leq l$ and add to $h(x)$ a trapezoidal load $h_a(x)$ between x_a and x_d. Height of the trapezoid is $h_M \sim \mathcal{U}(0, 1)$.
   b) If 1, sample $h_\mu \sim \mathcal{U}(0, l), h_\sigma \sim \mathcal{U}(0, l)$ and add to $h(x)$ the bell curve load:

$$h_b(x) = \frac{1}{h_\sigma \sqrt{2\pi}} e^{\frac{-(x-h_\mu)^2}{2h_\sigma^2}}.$$

2. Normalize the load $h(x)$: Pick $H \sim \mathcal{U}(8000\,\mathrm{N}, 15000\,\mathrm{N})$. Compute $h_i = \int_0^l h(x)dx$, and set $h(x) = \frac{H}{h_i} h(x)$.

5.4 Chance Constraints and Probabilistic Robust Design

The chance constrained formulation of the problem has the following form:

$$\begin{aligned}
&\underset{y_a,y_b,y_E}{\text{minimize}} && y_a + y_b + p \cdot y_E && (5.28)\\
&\text{subject to} && P\begin{pmatrix} -y_a - 2y_b + \log v_M(\xi) - \log \sigma_M \leq 0, \\ -y_a - 3y_b - y_E + \log w_M(\xi) - \log \delta_M \leq 0 \end{pmatrix} \geq 1 - \epsilon, && (5.29)\\
& && -y_a + y_b - \log r_M \leq 0, && (5.30)\\
& && \log a_L \leq y_a \leq \log a_U,\ \ \log b_L \leq y_b \leq \log b_U,\ \ \log E_L \leq y_E \leq \log E_U, && (5.31)
\end{aligned}$$

where $1 - \epsilon$ is the reliability level. The first part of the method is, again, to draw a large number S of scenarios and solve the following problem:

$$\begin{aligned}
&\underset{y_a,y_b,y_E}{\text{minimize}} && y_a + y_b + p \cdot y_E && && (5.32)\\
&\text{subject to} && -y_a - 2y_b + \log v_M(s) - \log \sigma_M \leq 0, && s = 1, \ldots, S, && (5.33)\\
& && -y_a - 3y_b - y_E + \log w_M(s) - \log \delta_M \leq 0, && s = 1, \ldots, S, && (5.34)\\
& && -y_a + y_b - \log r_M \leq 0, && && (5.35)\\
& && \log a_L \leq y_a \leq \log a_U,\ \ \log b_L \leq y_b \leq \log b_U,\ \ \log E_L \leq y_E \leq \log E_U, && && (5.36)
\end{aligned}$$

where the $2S$ constraints (5.33) and (5.34) can be reduced to the following 2 constraints:

$$-y_a - 2y_b + \max_s(\log v_M(s)) - \log \sigma_M \leq 0, \qquad (5.37)$$

$$-y_a - 3y_b - y_E + \max_s(\log w_M(s)) - \log \delta_M \leq 0. \qquad (5.38)$$

For a high enough choice of S, the optimal solution to (5.32)-(5.38) yields a feasible solution for the chance constrained problem with high probability (see [7]). As discussed in [9], we can remove the k scenarios at once or we can use a greedy approach that removes just one scenario at a time. In our case, the greedy approach makes perfect sense – there are only two scenarios (called support scenarios in [10]) whose removal can decrease the optimal objective value of (5.32)-(5.38):

$$s_1 = \underset{s}{\operatorname{argmax}}(\log v_M(s)) \quad \text{and} \quad s_2 = \underset{s}{\operatorname{argmax}}(\log w_M(s)).$$

To determine, which one of the two scenarios should be removed, we must solve two additional linear problems (with s_1 or s_2 temporarily removed) and compare their optimal objective values – this is repeated k times. There is one different approach we will discuss, and that is the approximation of the joint chance constraint (5.29) by individual chance constraints:

$$P(-y_a - 2y_b + \log v_M(\xi) - \log \sigma_M \le 0) \ge 1 - \epsilon_1, \tag{5.39}$$

$$P(-y_a - 3y_b - y_E + \log w_M(\xi) - \log \delta_M \le 0) \ge 1 - \epsilon_2, \tag{5.40}$$

which become

$$-y_a - 2y_b + \Phi_v^{-1}(1-\epsilon_1) - \log \sigma_M \le 0, \tag{5.41}$$

$$-y_a - 3y_b - y_E + \Phi_w^{-1}(1-\epsilon_2) - \log \delta_M \le 0, \tag{5.42}$$

where Φ_v^{-1} and Φ_w^{-1} are the (empirical) quantile functions of $\log v_M(\xi)$ and $\log w_M(\xi)$, and ϵ_1, $\epsilon_2 > 0$ are appropriately chosen. The problem then becomes:

$$\begin{aligned}
\underset{y_a, y_b, y_E}{\text{minimize}} \quad & y_a + y_b + p \cdot y_E & (5.43)\\
\text{subject to} \quad & -y_a - 2y_b + \Phi_v^{-1}(1-\epsilon_1) - \log \sigma_M \le 0, & (5.44)\\
& -y_a - 3y_b - y_E + \Phi_w^{-1}(1-\epsilon_2) - \log \delta_M \le 0, & (5.45)\\
& -y_a + y_b - \log r_M \le 0, & (5.46)\\
& \log a_L \le y_a \le \log a_U, \;\; \log b_L \le y_b \le \log b_U, \;\; \log E_L \le y_E \le \log E_U. & (5.47)
\end{aligned}$$

5.5 Numerical Results

Our goal is to obtain a trade-off curve between the optimal objective value and the reliability of the design. To achieve this we used our scenarios generation technique to draw two large sets of scenarios, where the first one contained S_1 and the second S_2 scenarios. The first one was used for the optimization part (i.e. solving (5.32)-(5.38)), the second one was used for the estimate of the reliability level ϵ. The method proceeded as follows:

0. Generate the two sets of scenarios.
   Repeat k times:
1. Solve (5.32)-(5.38) using the first set of scenarios. Obtain an optimal design.
2. Estimate the reliability of the design using the second set of scenarios: given a design in the form of a, b and E, the constraints (5.33)-(5.34) either both hold, or at least one of them does not hold. This outcome describes a binomial random variable – compute its point estimate (a fraction of scenarios for which at least one of the constraints did not hold) and its 99.9% confidence interval (using the Clopper-Pearson interval).

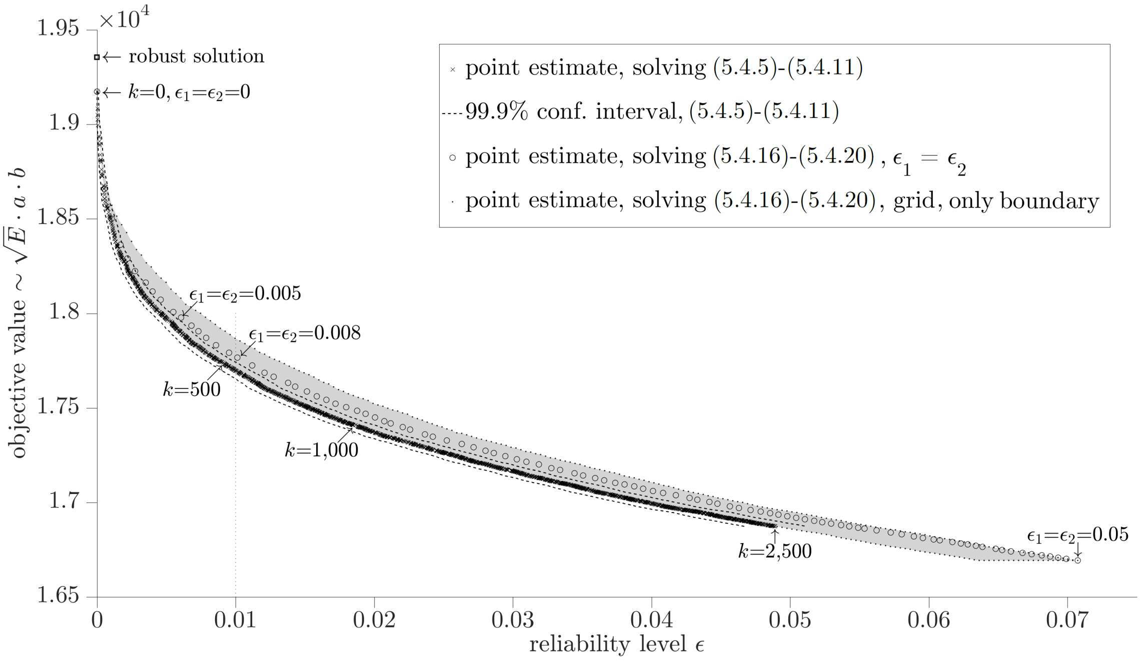


Fig. 5.2: The trade-off between reliability and optimal objective value.

3. Determine, which one of the two support scenarios to remove, and delete it from the first set of scenarios. Return to 1.

In Fig. 5.2 is depicted the trade-off between the reliability level ϵ and the optimal objective value using the two approaches (5.32)-(5.38) and (5.43)-(5.47). In the first approach we gradually remove the scenarios (upto $k = 2{,}500$) – the computational time for each iteration (two optimization problems, scenario removal) was around 0.4 s. In the second approach (5.43)-(5.47) we vary the values of $\epsilon_1 = \epsilon_2$ between 0 and 0.05 – the computational time for each value was around 0.2 s. Furthermore, used a grid of 1,001 steps for ϵ_1 and ϵ_2 between 0 and 0.05 and computed the results for all of these grid values (they fill the grey area in Fig. 5.2), this took 45 hours. The robust solution was computed using the results in the Appendix B (maximum point loads in $\frac{1}{2}l$ and $\frac{1}{3}l$).

The comparison between the two methods favours the scenario-removal one (5.32)-(5.38) over solving (5.43)-(5.47) with $\epsilon_1 = \epsilon_2$, as it produces designs with better objective value. For example, given the target (point estimate of) $\epsilon = 0.01$, the closest design produced by (5.43)-(5.47) is for $\epsilon_1 = \epsilon_2 = 0.008$, with the objective value $1.776{\cdot}10^4$, whereas the method using (5.32)-(5.38) with $k = 568$ deleted scenarios achieved the objective value $1.769{\cdot}10^4$. Moreover, the scenario-removal method (5.32)-(5.38) produced as good solutions as the best ones using the grid values for ϵ_1 and ϵ_2 and solving (5.43)-(5.47).

BIBLIOGRAPHY

[1] L. Adam and M. Branda. Nonlinear chance constrained problems: Optimality conditions, regularization and solvers. *Journal of Optimization Theory and Applications*, 170(2):419–436, 2016.

[2] T. Alamo, R. Tempo, A. Luque, and D. R. Ramirez. Randomized methods for design of uncertain systems: Sample complexity and sequential algorithms. *Automatica*, 52(1):160–172, 2015.

[3] J. Bezanson, A. Edelman, S. Karpinski, and Shah V. B. Julia: A fresh approach to numerical computing. *SIAM Review*, 59:65–98, 2017.

[4] J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer, 1st edition, 1997.

[5] S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 1st edition, 2004.

[6] G. C. Calafiore and M. C. Campi. Uncertain convex programs: randomized solutions and confidence levels. *Mathematical Programming, Ser. A*, 102(1):25–46, 2005.

[7] G. C. Calafiore and M. C. Campi. The scenario approach to robust control design. *IEEE Transactions on Automatic Control*, 51(5):742–753, 2006.

[8] M. C. Campi and S. Garatti. The exact feasibility of randomized solutions of uncertain convex programs. *SIAM Journal of Optimization*, 19(3):1211–1230, 2008.

[9] M. C. Campi and S. Garatti. A sampling-and-discarding approach to chance-constrained optimization: Feasibility and optimality. *Journal of Optimization Theory and Applications*, 148:257–280, 2011.

[10] A. Care, S. Garatti, and M. C. Campi. Scenario min-max optimization and the risk of empirical costs. *SIAM Journal on Optimization*, 25(4):2061–2080, 2015.

[11] C. A. Floudas. *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. Oxford University Press, 1st edition, 1995.

[12] C. A. Floudas and P. M. Pardalos. *Encyclopedia of Optimization*. Springer, 2nd edition, 2009.

[13] A. M. Geoffrion. Generalized Benders decomposition. *Journal of Optimization Theory and Applications*, 10(4):237–260, 1972.

[14] Gurobi Optimization. Gurobi optimizer reference manual, 2018.

[15] J. Haslinger and R. A. E. Mäkinen. *Introduction to Shape Optimization: Theory, Approximation, and Computation (Advances in Design and Control).* SIAM, 1st edition, 2003.

[16] IBM Corp. *IBM ILOG CPLEX Optimization Studio: CPLEX User's Manual. Version 12, Release 7.*, 2016.

[17] P. Kall and S. W. Wallace. *Stochastic Programming.* John Wiley & Sons, 1st edition, 1994.

[18] J. Kudela and P. Popela. Warm-start cuts for generalized Benders decomposition. *Kybernetika*, 53(6):1012–1025, 2017.

[19] J. Kudela and P. Popela. Chance constrained optimal beam design: Convex reformulation and probabilistic robust design. *Kybernetika*, 54(6):1201–1217, 2018.

[20] J. Kudela, P. Popela, R. Somplak, M. Malek, A. Rychtar, and D. Hrabec. The L-shaped method for large-scale mixed-integer waste management decision making problems. *Chemical Engineering Transactions*, 61:1087–1092, 2017.

[21] J. Kudela, R. Somplak, V. Nevrly, and T. Lipovsky. Robust waste transfer station planning by stochastic programming. *Chemical Engineering Transactions*, 70(1):889–894, 2018.

[22] I. Lanikova, P. Stepanek, and P. Simunek. Optimized design of concrete structures considering environmental aspects. *Advances in Structural Engineering*, 17(4):495–511, 2014.

[23] V. L. Levin. Application of E. Helly's theorem to convex programming, problems of best approximation and related questions. *Mathematics of the USSR-Sbornik*, 8:235–247, 1969.

[24] A. Nemirovski. On safe tractable approximations of chance constraints. *European Journal of Operational Research*, 219(3):707–718, 2012.

[25] A. Nemirovski and Shapiro A. Convex approximations of chance constrained programs. *SIAM Journal on Optimization*, 17(4):959–996, 2006.

[26] A. Nemirovski and A. Shapiro. *Continuous Optimization*, chapter On Complexity of Stochastic Programming Problems. Springer US, 2005.

[27] A. Prekopa. *Stochastic Programming.* Kluwer, 1st edition, 1995.

[28] A. Ruszczynski and A. Shapiro, editors. *Stochastic Programming*, volume 10 of *Handbooks in Operations Research and Management Science*. Elsevier, 1st edition, 2003.

[29] Z. Sabartova and P. Popela. Beam design optimization model with FEM based constraints. *Mendel*, 1:422–427, 2012.

[30] F. Shan, L. Zhang, and X. Xiao. A smoothing function approach to joint chance-constrained programs. *Journal of Optimization Theory and Applications*, 163(1):181–199, 2014.

[31] I. M. Smith and D. V. Griffiths. *Programming the finite element method*. John Wiley & Sons,, 4nd edition, 2004.

[32] E. Zampachova, P. Popela, and M. Mrazek. Optimum beam design via stochastic programming. *Kybernetika*, 46(3):571–582, 2010.