Distribution shaping and scenario bundling for stochastic programs with endogenous uncertainty*

Marco Laumanns [†] Steven Prestwich [‡] Ban Kawas [§]

October 31, 2014

Abstract

Stochastic programs are usually formulated with probability distributions that are exogenously given. Modeling and solving problems with endogenous uncertainty, where decisions can influence the probabilities, has remained a largely unresolved challenge. In this paper we develop a new approach to handle decision-dependent probabilities based on the idea of distribution shaping. It uses a sequence of distributions, successively conditioned on the influencing decision variables, and characterizes these by linear inequalities. We demonstrate the approach on a pre-disaster planning problem of finding optimal investments to strengthen links in a transportation network, given that the links are subject to stochastic failure. Our new approach solves a recently considered instance of the Istanbul highway network to optimality within seconds, for which only approximate solutions had been known so far.

1 Introduction

Stochastic programming [4, 32] is an effective modeling approach for decision-making under uncertainty, particularly when probability distributions governing the uncertainty are readily available. In the classical paradigm of stochastic programming, it is assumed that decisions do not influence the probability space of the underlying stochastic processes—i.e. uncertainty is *exogenous* [20, 35, 37, 5]. For example, the standard formulation of a two-stage stochastic program with recourse is generally given as the problem

$$\begin{array}{ll} \text{Minimize}_{\,x} & g(x) + \mathrm{E}^{P^{\xi}}(Q(\xi, x)) \\ \text{subject to} & x \in \mathcal{X} \end{array} \tag{1}$$

[*] The contents of this paper has been presented at the EURO-INFORMS Joint International Meeting 2013 in Rome and the International Conference on Operations Research (OR 2014) in Aachen.

[†] IBM Research, Rüschlikon, Switzerland, mlm@zurich.ibm.com

[‡] University College Cork, Cork, Ireland, s.prestwich@cs.ucc.ie

[§] IBM Research, Yorktown Heights, NY, USA, bkawas@us.ibm.com

where ξ is a realization of the uncertain outcome of the stochastic process that is revealed in the second stage, P^{ξ} is a probability measure governing ξ and is *independent* of first stage decisions x and the set $\mathcal{X}$ defines the constraints on x. $Q(\xi, x)$ is the optimal value function of the second-stage recourse problem given by

$$\begin{array}{lll} \text{Minimize }_y & f(\xi, y) & (2) \\ \text{subject to} & x,\, y \in \mathcal{K} & \\ & \xi \in \Omega & \end{array}$$

where $\mathcal{K}$ is a set of constraints defined over *first and second* stage decisions x and y, respectively, and Ω is a finite sample space that is also *independent* of both x and y.

The exogenous uncertainty assumption allows for the design of efficient solution algorithms for real-sized problems and—from a practical point of view—can be satisfied by a wide range of applications. However, there is an important class of decision-making problems—for which stochastic programming is a suitable modeling approach—that violates this assumption [14, 21, 1] and can consequently be very challenging to solve. In this class of stochastic programs, decisions *can influence* the stochastic process and, therefore, uncertainty is *endogenous.*

This paper is concerned with stochastic programs under endogenous uncertainty in which optimization decisions can influence stochastic processes by altering the corresponding probability space. Decisions can *alter* probability spaces in at least two ways. They can alter the probability measures by making one random outcome more likely than another [29, 1]. Alternatively, they can also partially resolve the uncertainty influencing the reveal time and *alter* the set of possible future random outcomes [38, 14, 15]. In this paper we will consider the former case—which we refer to as the case of *decision-dependent probabilities*—when decisions can alter the probability measures governing the random outcome of stochastic processes. For example, for a 2-stage stochastic program, we consider formulations such as [10, 33]

$$\begin{array}{lll} \text{Minimize }_x & g(x) + \mathrm{E}^{P_x^{\xi}}(Q(\xi, x)) & (3) \\ \text{subject to} & x \in \mathcal{X} & \end{array}$$

where the probability measure P_x^{ξ} governing $\xi \in \Omega$ is *a function* of decisions x, while the sample space Ω is independent of x. In this case, if the function expressing the influence of decisions x on the probabilities of random outcomes is nontrivial, then the corresponding stochastic program may become significantly more challenging to analyze and solve.

Most solution approaches to handle endogenous uncertainty are based on approximations and heuristic algorithms [17, 40, 11, 39] and they heavily rely on problem structure [10]. General exact solution methods are not yet attainable. A recent review of solution methods for multistage stochastic programs with endogenous uncertainty is given in [16].

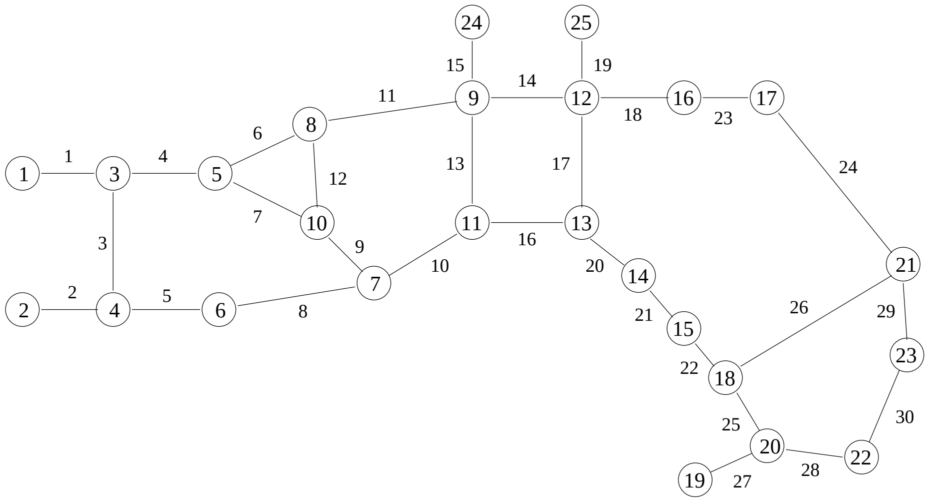

Figure 1: Istanbul road network

In this paper, we develop a novel approach to handle endogenous uncertainty in stochastic programs for the case of decision-dependent probabilities. We focus on the class of stochastic programs with *binary* decisions [37]. Unlike previous approaches—which are based on linear or convex approximations of polynomials that are induced by decision-dependent probabilities—our approach provides an *exact* formulation and solution method.

One component of the new approach, which we call *distribution shaping*, enables an efficient characterization of decision-dependent probability measures based on a key observation; “neighboring” probability measures—in the sense of influencing binary decisions—are *linearly dependent*, and this linear relationship can be expressed via Bayes’ Rule. Accordingly, we derive a successive polyhedral characterization of probability measures as a function of decisions and formulate the corresponding stochastic program as an exact mixed-integer program (MIP), if the recourse only depends on random variables, and as a mixed-integer bilinear program, if the recourse also depends on decision variables.

Another component of our approach enables us to solve exactly the derived MIP various applications, in this case for a pre-disaster planning problem [29], for which only approximate solutions were known previously. We utilize techniques from *constraint programming*, such as dynamic interchangeability, and the structure of the recourse function to cluster scenarios of uncertain random variables into scenario *bundles.* We consider scenarios with both binary and general m-ary structure. We generalize our distribution shaping method to work with scenario bundles and derive the corresponding MIPs.

The rest of the paper is organized as follows: In Section 2, we describe a real-world use-case of 2-stage stochastic programs under endogenous uncertainty—a

link	t_e	c_e	p_e	link	t_e	c_e	p_e
1	2.46	80	0.80	16	4.26	940	0.55
2	2.20	80	0.80	17	3.64	300	0.70
3	8.00	320	0.80	18	4.19	520	0.60
4	2.56	260	0.70	19	1.98	40	0.80
5	4.57	160	0.80	20	2.45	800	0.55
6	3.44	420	0.60	21	1.80	40	0.80
7	4.19	160	0.80	22	1.97	160	0.70
8	5.60	620	0.60	23	1.61	40	0.80
9	3.71	120	0.80	24	8.09	620	0.60
10	4.44	340	0.70	25	2.87	260	0.70
11	7.11	940	0.55	26	6.35	780	0.60
12	4.03	160	0.80	27	2.27	800	0.55
13	5.02	620	0.60	28	3.91	120	0.80
14	4.55	1180	0.50	29	4.11	220	0.70
15	1.36	40	0.80	30	2.27	500	0.60

Table 1: Problem parameters

pre-disaster planning problem. We discuss the challenges introduced by decision-dependent probabilities and motivate the need for efficient solution algorithms. We then describe our distribution shaping method in Section 3, for both binary scenarios and general m-ary scenarios. In Section 4, we detail the scenario bundling approach based on constraint programming techniques and also derive an extension of the distribution shaping method to work with scenario bundles. We demonstrate the effectiveness of the new approach in Section 5 by solving the real-world use-case from Section 2 exactly, which was previously only known to have approximate solutions, while exact solutions had been considered out of reach. We also demonstrate the approach on randomly generated and even larger instances of the same use-case. In Section 6, we discuss the applicability of both, the distribution shaping and scenario bundling, methods to other use-cases, such as stochastic PERT. We give some concluding remarks and future directions in Section 7.

2 Pre-disaster planning under endogenous uncertainty

The problem we exemplarily address and solve with the new method developed in this paper was first described by Peeta *et al.* [29]. They cite evidence that the probability of a major earthquake occurring in the next few decades with its epicenter in Istanbul has been estimated as 62.6±15%; that this is likely to cause tens of billions of dollars worth of damage; that the Turkish government plans to invest \$400 million to strengthen infrastructure for earthquake resistance;

and that a key element of this plan is to retrofit selected highways to maximize accessibility after an earthquake. We now recapitulate the problem setting and given input data and then discuss its stochastic programming formulation and its technical challenges.

2.1 Problem setting and input data

The Istanbul highway network is represented in Figure 1 by an undirected graph $G = (V, E)$ with 25 nodes V and 30 edges or links E. Each link represents a section of a highway may that may be disrupted with some given probability in the event of an earthquake, while each node represents a junction. The failure probability of a link can be reduced by investing money in it, but there is a budget limiting the total investment. To maximize post-quake accessibility, a relevant objective is to minimize the expected shortest path between given origin and destination nodes in the network, by investing in carefully-chosen links. In fact the actual objective is to minimize a weighted sum of shortest path lengths between several origin-destination (O-D) pairs, the choice of which is based on likely earthquake scenarios in the Japan International Cooperation Agency Report of 2002.

We now sketch the stochastic model. For each link $e \in E$ define a binary decision variable x_e which is 1 if we invest in that link and 0 otherwise. Define a binary random variable r_e which is 1 if link e survives and 0 if it fails. Denote the survival (non-failure) probability of link e by p_e without investment and q_e with, the investment required for link e by c_e, the length of link e by t_e (the units used in [29] are not specified but are proportional to the actual distances), and the budget by B. If an O-D pair is not connected then the path length is taken to be a fixed number M representing (for example) the cost of using a helicopter. Actually, if they are only connected by long paths then they are considered to be unconnected, as in practice rescuers would resort to alternatives such as rescue by helicopter or sea. So Peeta *et al.* only consider a few (4–6) shortest paths for each O-D pair, and we shall refer to these as the *allowed paths*. In each case M is chosen to be the smallest integer that is greater than the longest allowed path length. They also consider a larger value of $M = 120$ that places a greater importance on connectivity, though using the same paths as with the smaller M values. To distinguish between these two usages we replace M by M_a (the length below which a path is allowed) and M_p (the penalty imposed when no allowed path exists). We fix M_a to the smaller values (not 120) for each O-D pair, and generate two sets of instances using $M_p = M_a$ and $M_p = 120$. All q_e values are set to 1 based on feedback from structural engineers. Three budget levels $B_1 = 1164$, $B_2 = 2328$ and $B_3 = 3492$ are considered, corresponding to 10%, 20% and 30% of the total cost 11640 of investing in all links. The other problem parameters are specified in Table 1, and are taken from Peeta *et al.* who used data from the 2003 Master Earthquake Plan of the Istanbul municipality.

2.2 Stochastic programming approach and difficulty

The earthquake problem can be modeled as a 2-stage stochastic program. In the first stage we decide which links to invest in by assigning values to the x_e, then link failures occur randomly with probabilities depending of the x_e, causing values to be assigned to the r_e. In the second stage we choose a shortest path between the O-D pair, given the surviving links. If they are no longer connected by an allowed path then the value M_p is used instead of a path length. For a given O-D pair the expected length is computed over all scenarios, and minimizing this value over a set of given O-D pairs is the objective.

This is a large problem to solve by stochastic programming methods because each of the 30 links is independently affected by an earthquake, giving 2^{30} scenarios. Though optimization time is not critical in pre-disaster planning, handling a billion scenarios is intractable. In fact, even determining the expected shortest path lengths for a *given* solution is hard, it is a classical problem in network reliability and known to be #P-complete. Instead Peeta *et al.* sample a million scenarios, and approximate the objective function by a monotonic multilinear function. They show that their method gives optimal or near-optimal results on smaller instances, and present results on the full-scale problem.

Another source of difficulty is that the problem has *endogenous uncertainty*: the decisions (which links to invest in) affects the probabilities of the random events (the link failures). Relatively little work has been done on such problems but they are usually much harder to solve by Stochastic Programming methods. For a survey on problems with endogenous uncertainty see [14], which mentions applications including network design and interdiction, server selection, facility location, and gas reservoir development. Other examples include clinical trial planning [9] and portfolio optimization [38].

These two features make the earthquake problem very challenging to solve to optimality. To do so, we introduce two new techniques: *distribution shaping* described in Section 3 and *scenario bundling* described in Section 4. While each of these techniques is original and useful by itself, they are very well combinable and turn out to be even more powerful when applied in conjunction.

3 Distribution shaping

As discussed above, our goal is to solve stochastic programs where decisions, typically first-stage decisions, influence the probability distribution on the scenarios. In this section we develop a new modeling approach that avoids the non-linearities that previous formulations introduce when multiplying probabilities.

We consider a finite sample space $\Omega = \{0,1\}^n$ and denote scenarios as $\xi = (\xi_1, \xi_2, \ldots, \xi_n) \in \Omega$. Let $\mathcal{F}$ be the power set of Ω. In order to capture the influence of the decisions on the probabilities, we define a family of probability measures $P_x : \mathcal{F} \to [0,1]$, where x is the decision variable vector that influences the probabilities. Hence, each choice of $x \in \mathcal{X}$, where $\mathcal{X}$ is the feasible set of

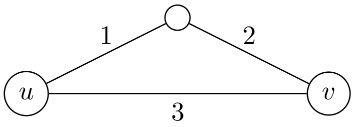

Figure 2: Graph with three nodes and three edges labeled 1, 2, 3.

those decisions, induces a probability space $(\Omega, \mathcal{F}, P_x)$ on the same underlying sample space and the same set of events, only the probability measures are different.

We consider a binary decision vector $x \in \{0,1\}^n$ and model its influence on the probability measures P_x as follows. With each sample component ξ_i two parameters p_i and q_i are associated, and the resulting probabilities as a function of the decision vector x are set to

$$P_x\left(\{\xi_i = \tilde{\xi}_i\}\right) = \begin{cases} (1-x_i)p_i + x_i q_i & \text{for } \tilde{\xi}_i = 1 \\ (1-x_i)(1-p_i) + x_i(1-q_i) & \text{for } \tilde{\xi}_i = 0. \end{cases}$$

Assuming independence of these events, the decision-dependent scenario probabilities are given by

$$P_x\left(\{\xi = \tilde{\xi}\}\right) = \prod_{i=1}^{n} P_x\left(\{\xi_i = \tilde{\xi}_i\}\right).$$

Using such products directly in an optimization model seems prohibitive as they result in polynomials of degree n in x. Therefore previous approaches relied either on linearizations [29] or convex approximations [11].

Instead of working with these polynomials directly and trying to approximate them, we proceed differently. Our key observation, which will enable an efficient characterization of the resulting probability measure, is that "neighboring" measures (in the sense of the x-values) have a very simple, linear relationship to each other, which can be interpreted as an application of Bayes' rule.

Consider two decision vectors x^0 and x^1 that only differ in one component i so that $x^1 - x^0 = \mathbf{e}_i$, the i-th unit vector. Then the two corresponding probability measures are related as

$$P_{x^1}\left(\{\xi = \tilde{\xi}\}\right) = \begin{cases} P_{x^0}\left(\{\xi = \tilde{\xi}\}\right) \cdot \frac{q_i}{p_i} & \text{if } \tilde{\xi}_i = 1, \\ P_{x^0}\left(\{\xi = \tilde{\xi}\}\right) \cdot \frac{1-q_i}{1-p_i} & \text{if } \tilde{\xi}_i = 0. \end{cases}$$

In other words, when changing x_i from 0 to 1 then the probabilities of all scenarios with $\xi_i = 1$ are uniformly scaled up so that all others must be scaled down accordingly to keep the total measure constant.

Let $n = 3$ and consider the graph shown in Figure 3. Edge i survives with probability $p_i = 0.8$ if $x_i = 0$ and with probability $q_i = 0.9$ if $x_i = 1$. Then the probabilities for the 2^3 joint edge survival or failure scenarios for the two different decision vectors $x^0 = (0,0,0)$ and $x^1 = (0,1,0)$ are given in Table 3.

edges			$P_x\{\xi = \tilde{\xi}\}$		
1	2	3	$x^0 = (0,0,0)$		$x^1 = (0,1,0)$
0	0	0	0.008	· 0.1 / 0.2	0.004
0	0	1	0.032	· 0.1 / 0.2	0.016
0	1	0	0.032	· 0.9 / 0.8	0.036
0	1	1	0.128	· 0.9 / 0.8	0.144
1	0	0	0.032	· 0.1 / 0.2	0.016
1	0	1	0.128	· 0.1 / 0.2	0.064
1	1	0	0.128	· 0.9 / 0.8	0.144
1	1	1	0.512	· 0.9 / 0.8	0.576

Table 2: Probability measure induced by two neighboring decision vectors

For any fixed decision vector x, the resulting probability measure P_x can be easily computed by applying the above scaling successively for all components x_i set to 1. However, to use this in an optimization model, we should be able to express this scaling not only for fixed x but as a function of x. We can exploit the fact that x is binary and derive a (successive) polyhedral characterization of the resulting distribution as follows. For a given x and all $k \in \{1, \dots n\}$, let $\check{x}^k$ be defined such that

$$\check{x}^k_i = \begin{cases} x_i & \text{if } i \leq k \\ 0 & \text{else} \end{cases}$$

and define $\pi^\xi_k = P_{\check{x}^k}(\{\xi\})$ for all k and $\xi \in \Omega$, so π_k is an auxiliary variable carrying the probability measure induced by $\check{x}^k$. Then for any $k < n$ the successive probability measures π_{k-1} and π_k must fulfill the set of linear inequalities

$$\begin{aligned}
\pi^\xi_k &\leq \frac{q_k}{p_k} \cdot \pi^\xi_{k-1} + 1 - x_k && \forall \xi \in \Omega : \xi_k = 1 && \text{(enforce scaling up)} \\
\pi^\xi_k &\leq \frac{1 - q_k}{1 - p_k} \cdot \pi^\xi_{k-1} + 1 - x_k && \forall \xi \in \Omega : \xi_k = 0 && \text{(enforce scaling down)} \\
\pi^\xi_k &\leq \pi^\xi_{k-1} + x_k && \forall \xi \in \Omega && \text{(no scaling if } x_k = 0\text{)} \\
\sum_\xi \pi^\xi_k &= 1.
\end{aligned}$$

where we naturally define $\check{x}^0 = (0, 0, \dots, 0)$, whose induced distribution is obviously given as a function of the parameters p_i only as

$$\pi^\xi_0 = \prod_{i:\xi_i=0} (1 - p_i) \cdot \prod_{i:\xi_i=1} p_i \tag{4}$$

and is thus given input. For a more compact notation in what follows, let $\mathcal{P}_k(\pi_{k-1}, x_k)$ denote the $(2^n \cdot n)$-dimensional polyhedron characterizing the feasible set for π_k as a function of π_{k-1} and x_k via the above inequalities.

We can now formulate a generic stochastic program using the decision variables x and auxiliary variables π_k introduced above. Let $f : \Omega \to \mathbb{R}$ be the random variable representing the cost in the different scenarios and E^{P_x} denote the expectation under the probability measure P_x. Then the problem

$$\begin{aligned}
&\text{Minimize} \qquad && \mathrm{E}^{P_x}(f) = \sum_{\xi\in\Omega} \pi_n^\xi \cdot f(\xi) && && (5)\\
&\text{subject to} && \pi_k \in \mathcal{P}_k(\pi_{k-1}, x_k) && \forall k \in \{1,\dots,n\} && (6)\\
& && x \in \mathcal{X} && && (7)
\end{aligned}$$

is a mixed-integer linear program for minimizing the expected cost, given that the cost in each scenario is a constant for each scenario and does not depend on other decision variables. This is the case for the type of problems considered in this paper. However, if $f(\xi)$ for instance was a recourse function of a stochastic linear program, then we would obtain a mixed-integer bi-linear program instead. Also note that it is straightforward to apply other convex risk measures instead of the expected value in the same manner, for example the conditional value-at-risk.

The above MIP has only n binary variables, but $2^n \cdot n$ auxiliary continuous variables and can thus become too large to be explicitly represented and solved in this form already for moderately sized problems. Therefore, in Section 4, we will present a way to effectively cluster scenarios that result in similar—in our case identical—cost into scenario bundles, by exploiting the structure of the recourse function f. This will allow us to work on a considerably smaller but equivalent partition of the sample space than by considering each scenario individually. We will then show how a distribution-shaping MIP similar to the one developed above can be applied to those coarse-grained partitions.

3.1 Beyond binary scenarios

The scaling technique does not only work for binary scenarios as presented above, but extends to any discrete distribution. For example, let $\Omega = \{1,\dots,m\}^n$ and assume given parameter vectors $p_i = (p_i^1, p_i^2, \dots, p_i^m), q_i = (q_i^1, q_i^2, \dots, q_i^m) \in [0,1]^m$ with $\sum_{j=1}^m p_i^j = 1$ and $\sum_{j=1}^m q_i^j = 1$ and for all $i \in \{1,\dots,n\}$. We can consider the decision-dependent probability measure to be

$$P_x\left(\{\xi_i = j\}\right) = (1-x_i)p_i^j + x_i q_i^j \qquad \text{for all } j \in \{1,\dots,m\}.$$

The resulting polyhedral characterization of successive measures is then

$$\begin{aligned}
\pi_k^\xi &\le \frac{q_k^j}{p_k^j} \cdot \pi_{k-1}^\xi + 1 - x_k \qquad && \forall j \in \{1,\dots,m\} \text{ and } \forall \xi \in \Omega : \xi_k = j,\\
\pi_k^\xi &\le \pi_{k-1}^\xi + x_k && \forall \xi \in \Omega,\\
\sum_\xi \pi_k^\xi &= 1.
\end{aligned}$$

We will discuss the case of $m = 3$ values per scenario component in the context of stochastic project planning problems in Section 6.

4 Scenario bundling

This section describes a new method for collecting scenarios into bundles. First we provide background on the constraint programming symmetry breaking ideas on which it is based.

4.1 Solution bundling in constraint programming

An early form of symmetry that has received considerable attention is *(value) interchangeability* [12].

Definition 1. *A value a for variable v is* fully interchangeable *with value b if and only if every solution in which $v = a$ remains a solution when b is substituted for a and vice-versa.*

If two values are interchangeable then one of them can be removed from the domain, reducing the size of the problem; alternatively they can be replaced by a single meta-value, and thus collected together in a Cartesian product representation of the search space. Both approaches avoid revisiting equivalent solutions. Several variants of interchangeability were defined in [12] and subsequent work in this area is surveyed in [22]. The relevant variant here is called *dynamic interchangeability*.

Definition 2. *A value a for variable v is* dynamically interchangeable *for b with respect to a set A of variable assignments if and only if they are fully interchangeable in the subproblem induced by A.*

Values may become interchangeable during backtrack search after some variables have been assigned values, so even a problem with no interchangeable values may exhibit dynamic interchangeability under some search strategy. This is an example of the more general concept of *conditional symmetry* [13] in which symmetry occurs at certain nodes in a search tree.

As an example consider the vertex coloring example in Figure 3 adapted from [12]. Neither vertices labeled with variables X and Y, nor Y and Z, may be assigned the same color because they are adjacent in the graph. Under the assignment Z=orange the values green and yellow in the domain of variable Y are fully interchangeable, because green and yellow are not in the domain of X so any solution in which Y=green can be transformed to a solution which is identical except that Y=yellow. However, under the assignment Z=yellow this no longer holds: for example assigning X=blue, Y=green and Z=yellow is a solution, but reassigning Y=yellow leads to a non-solution because Y and Z take the same color. Thus symmetry occurs in part of the search tree but not all of it. Note our assumption that Y is assigned first: dynamic interchangeability may occur in some search trees but not others, depending on the branching heuristic.

Interchangeable values can be exploited to group similar solutions together in *bundles*, a term used in [6, 18, 23] and other work. Bundles are Cartesian products of sets of values, which have been used in constraint programming

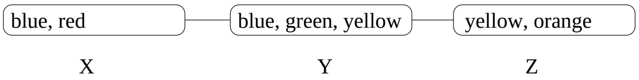

Figure 3: Dynamic interchangeability in vertex coloring

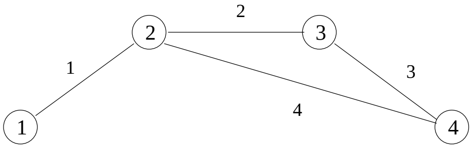

Figure 4: A small network example

to represent related solutions compactly in *solution bundles* [18], *cross product representations* [19], *maximal consistent decisions* [24], *solution clusters* [28] and the *maximal encoding* for Boolean satisfiability problems [30]. A drawback with interchangeability is that it does not seem to occur in many real applications [7, 27, 41] so it has received less attention than (for example) variable and value symmetries. Properties related to dynamic interchangeability were also investigated in [3, 30] but otherwise little or no work has been done on it. One of the contributions of this paper is to demonstrate the usefulness of dynamic interchangeability for scenario reduction in a stochastic problem.

4.2 Combining scenarios into bundles

We shall detect and exploit dynamic interchangeability in the random variables of the earthquake problem. As an illustration consider the simple example in Figure 4 with links $e \in \{1, \ldots, 4\}$. We set $t_e \equiv 1$, $p_e \equiv 0.8$, $q_e \equiv 1$, $c_e \equiv 1$, $B = 1$ and $M_a = M_p = 3.5$ so that both possible paths between nodes 1 and 4 are allowed. We must choose one link to invest in, to minimize the expected shortest path length between nodes 1 and 4. There are 16 scenarios and the optimal decision is to invest in link 1, giving an expected shortest path length of 2.236.

Some scenarios can be considered together instead of separately. For example consider the four scenarios $(1,0,0,1)$, $(1,0,1,1)$, $(1,1,0,1)$ and $(1,1,1,1)$, where the numbers indicate the survival (1) or failure (0) of links 1 to 4. Survival has probability 0.8 and failure 0.2 so these scenarios have probabilities 0.0256, 0.1024, 0.1024 and 0.4096 respectively. As links 1 and 4 survive in all four scenarios, it is irrelevant whether or not links 2 and 3 survive because they cannot be part of a shortest path: the path containing links 1 and 4 is shorter. We can therefore merge these four scenarios into a single expression $(1, *, *, 1)$, where the meta-value $*$ denotes interchangeability: the values 0 and 1 for links 2 and 3 are interchangeable. The expression represents the Cartesian product $\{1\} \times \{0, 1\} \times \{0, 1\} \times \{1\}$ of scenarios. The probability associated with this product of scenarios is $0.8 \times (0.8 + 0.2) \times (0.8 + 0.2) \times 0.8 = 0.64$, which is equal to the sum of the 4 scenario probabilities.

We shall call a product such as $(1, *, *, 1)$ a *scenario bundle* by analogy with solution bundles in constraint programming. Note that this usage is of course distinct from *bundle methods* in Stochastic Programming [34].

Another way of viewing scenario bundling is as an application of *stochastic dominance* [25]: the objective function associated with one choice (0 or 1) is at least as good as with another choice (1 or 0). In our case this holds in every scenario so it is the simplest form of stochastic dominance: *statewise* (or *zeroth order*) *dominance*. However, this is usually defined as a strict dominance by adding an extra condition: that one choice is strictly better than the other in at least one state (or scenario). In our case neither value is better so this is a weak dominance. In fact we have two values that each weakly dominate the other, a relationship that can be viewed as a *symmetry*: the tree is exactly the same whichever value we use for a link. There does not seem to be an accepted term such as "stochastic symmetry" for this phenomenon so we propose its use.

We note that the network reliability literature describes methods for evaluating and approximating the reliability networks. These include ways of pruning irrelevant parts of a network that have connections to our approach, though we have not found a direct parallel. For a discussion of these ideas see [8].

4.3 Finding small bundle sets

Bundling scenarios together may lead to faster solution of some stochastic problems. However, for the earthquake problem it is impractical to enumerate a billion scenarios then look for ways of bundling some of them together, as we did in the above example. Instead we dynamically enumerate scenarios by tree search on the random variables (the *scenario tree*) and apply symmetry breaking as we search.

Consider a node in the scenario tree at which links $1 \ldots i - 1$ have been realized, so that components $s_1 \ldots s_{i-1}$ have been assigned values, and we are about to assign a value to s_i corresponding to link i. Denote by C_i the shortest O-D path length including i, under the assumption that all unrealized links survive; and denote by F_i the shortest O-D path length not including i, under the assumption that all unrealized links fail (using M_p when no path exists). So C_i is the minimum shortest path length including i in all scenarios below this scenario tree node, while F_i is the maximum shortest path length not including i in the same scenarios. They can be computed by temporarily assigning $s_i \ldots s_{|E|}$ to 1 or 0 respectively, and applying a shortest path algorithm. Now if $C_i = F_i$ then the value assigned to s_i is irrelevant: the shortest path length in each scenario under this tree node is independent of the value of s_i, so the values are interchangeable. In this case there is no need to branch and we can simply assign value $*$ to the variable. This observation is the core of our method. Note that the interchangeability of an unrealized link implies that all other unrealized links are also interchangeable. This fact can be used to speed up interchangeability detection by avoiding unnecessary tests.

links				
3	2	4	1	p
0	*	0	*	0.0400
0	*	1	0	0.0320
0	*	1	1	0.1280
1	0	0	*	0.0320
1	0	1	0	0.0256
1	0	1	1	0.1024
1	1	0	0	0.0256
1	1	0	1	0.1024
1	1	1	0	0.1024
1	1	1	1	0.4096

links				
1	4	2	3	p
0	*	*	*	0.2000
1	0	0	*	0.0320
1	0	1	0	0.0256
1	0	1	1	0.1024
1	1	*	*	0.6400

Table 3: Two scenario bundle sets for the small example

The order in which we assign the s variables affects the cardinality of the bundle set. Two bundle sets for the example are shown in Table 3 along with their link permutations, where p is the bundle probability. The set of size 10 is the largest possible and the set of size 5 is the smallest. Note that once we have obtained a bundle set we can discard the permutation used to derive it. We can also replace the symbol * by any domain value (we choose 0) and use it as a representative scenario for the bundle. For example the bundle $(1, 1, *, *)$ under link permutation $(1, 4, 2, 3)$ can be represented by the scenario $(1, 0, 0, 1)$ under link permutation $(1, 2, 3, 4)$, with the same associated probability.

Assuming a static variable ordering, the problem of finding the smallest cardinality scenario bundle set corresponds exactly to the problem of finding a variable permutation that minimizes the number of paths in a binary decision tree [31]. This is known to be NP-complete [42]. If we allow a dynamic variable ordering, in which the choice of random variable to assign next depends on which path we took to the current node, the problem becomes more complex. To obtain a good solution we introduce a dynamic branching heuristic as follows.

At each node of the scenario tree we have the current graph, and we must choose a random variable, representing an unrealized link, which may be set to 0, 1 or $*$. If it is set to 1 then the link survives and the graph is unchanged, otherwise the link is deleted from the graph (and restored on backtracking past the node). Treating $*$ as 0 instead of 1 means that the graph has fewer links, which significantly speeds up interchangeability detection. To choose a variable we first create an ordered list $L = \langle r_1, \ldots, r_k \rangle$ of all allowed paths between the O-D pair, in increasing order of path length. Using this list, for each unrealized link ℓ derive a vector $\langle v_1, \ldots v_k \rangle$ where $v_i = 0$ if ℓ is in r_i, and $v_i = 1$ otherwise. Then we choose the variable whose link has the lexicographically smallest vector. For example in the small network in Figure 4, $L = \langle \langle 1, 2, 3 \rangle, \langle 1, 4 \rangle \rangle$ with associated lengths $\langle 3, 2 \rangle$ and the four links have the following vectors: (1) $\langle 0, 0 \rangle$, (2) $\langle 1, 0 \rangle$, (3) $\langle 1, 0 \rangle$ and (4) $\langle 0, 1 \rangle$. So the first link to be chosen is 1, while the second depends on whether link 1 survives or fails. The motivation behind this heuristic is to choose unrealized links that appear in the shortest paths in the greatest number of scenarios, to maximize interchangeability.

4.4 Distribution shaping using bundles

The distribution shaping concept developed in Section 3 for individual scenarios can be generalized to work with scenario bundles. By using bundles instead of individual scenarios, the size of the resulting MIPs can often be reduced drastically without loss of information, as will be shown later in the computational experiments in Section 5.

As defined above, a scenario bundle is vector $s \in \{0, 1, *\}^n$ and represents a subset of scenarios

$$B_s = \{\xi \in \Omega \mid \xi_i = s_i \vee s_i = *, 1 \leq i \leq n\}. \tag{8}$$

The algorithm described in Section 4.3 returns a set of bundles S such that $\{B_s \mid s \in S\} \subset \mathcal{F}$ is a partition of Ω. Thus, by construction,

$$P(B_s) = \sum_{\xi \in B_s} P(\{\xi\}) \quad \text{and} \quad \sum_{s \in S} P(B_s) = 1.$$

Using distributivity, we can characterize the decision-dependent probability measures P_x for the scenario bundles directly by applying the same scaling as for the individual scenarios. Again, consider two decision vectors x^0 and x^1 with $x^1 - x^0 = \mathbf{e}_i$. The two corresponding probability measures are related as follows:

$$P_{x^1}(B_s) \;=\; \begin{cases} P_{x^0}(B_s) \cdot \frac{q_i}{p_i} & \text{if } s = 1, \\ P_{x^0}(B_s) \cdot \frac{1-q_i}{1-p_i} & \text{if } s = 0, \\ P_{x^0}(B_s) & \text{if } s = *. \end{cases}$$

Similar to Section 3, we can use the partial decision vector $\check{x}^k$ and define the auxiliary variables $\phi_k^s = P_{\check{x}^k}(B_s)$ for all k and $s \in S$. For any $k < n$ the successive probability measures ϕ_{k-1} and ϕ_k must fulfill the set of linear inequalities

$$\begin{aligned}
\phi_k^s &\leq \frac{q_k}{p_k} \cdot \phi_{k-1}^s + 1 - x_k && \forall s \in S : s_k = 1 && \text{(bound for scaling up)} \\
\phi_k^s &\leq \frac{1-q_k}{1-p_k} \cdot \phi_{k-1}^s + 1 - x_k && \forall s \in S : s_k = 0 && \text{(force scaling down)} \\
\phi_k^s &\leq \phi_{k-1}^s && \forall s \in S : s_k = * && \text{(no scaling)} \\
\phi_k^s &\leq \phi_{k-1}^s + x_k && \forall s \in S && \text{(allow scaling)} \\
\sum_s \phi_k^s &= 1
\end{aligned}$$

with initialization

$$\phi_0^s = \prod_{i:s_i=0} (1 - p_i) \cdot \prod_{i:s_i=1} p_i. \tag{9}$$

Let $\mathcal{P}_k(\phi_{k-1}, x_k)$ denote the $(|S| \cdot n)$-dimensional polyhedron characterizing the feasible set for ϕ_k as a function of ϕ_{k-1} and x_k via the above inequalities. The resulting stochastic program using the decision variables x and auxiliary variables ϕ_k is

$$\begin{aligned} &\text{Minimize} && \mathrm{E}^{P_x}(f) = \sum_{s \in S} \phi_n^s \cdot f(s) && && (10)\\ &\text{subject to} && \phi_k \in \mathcal{P}_k(\phi_{k-1}, x_k) && \forall k \in \{1, \ldots, n\} && (11)\\ & && x \in \mathcal{X} && && (12) \end{aligned}$$

The bundling approach can also be generalized to work with general m-ary scenarios that take values in $\Upsilon = \{1, 2, \ldots, m\}$. In this case, we define a bundle σ as a vector of subsets of Υ, that is, $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_n)$, where each $\sigma_i \subseteq \Upsilon$. The corresponding subsets of scenarios are given by

$$B_\sigma = \{\xi \in \Omega \mid \xi_i \in \sigma_i, 1 \leq i \leq n\}. \tag{13}$$

We assume to be given a set of bundles Σ such that $\{B_\sigma \mid \sigma \in \Sigma\} \subset \mathcal{F}$ is a partition of Ω.

The probability measures corresponding to two decision vectors x^0 and x^1 with $x^1 - x^0 = \mathbf{e}_i$ are related as

$$P_{x^1}(B_\sigma) \;=\; P_{x^0}(B_\sigma) \cdot \frac{\sum_{j \in \sigma_i} q_i^j}{\sum_{j \in \sigma_i} p_i^j}$$

therefore the polyhedral characterization of successive measures is

$$\begin{aligned} \psi_k^\sigma &\leq \frac{\sum_{j \in \sigma_i} q_k^j}{\sum_{j \in \sigma_i} p_k^j} \cdot \psi_{k-1}^\sigma + 1 - x_k && \forall \sigma \in \Sigma\\ \psi_k^\sigma &\leq \psi_{k-1}^\sigma + x_k && \forall \sigma \in \Sigma\\ \sum_\sigma \psi_k^\sigma &= 1 \end{aligned}$$

with initialization

$$\psi_0^\sigma = \prod_{i=1}^n \sum_{j \in \sigma_i} p_i^j. \tag{14}$$

Let $\mathcal{P}_k(\psi_{k-1}, x_k)$ denote the $(|\Sigma| \cdot n)$-dimensional polyhedron characterizing the feasible set for ψ_k as a function of ψ_{k-1} and x_k via the above inequalities. The resulting stochastic program using the decision variables x and auxiliary variables ψ_k is

$$\begin{aligned} &\text{Minimize} && \mathrm{E}^{P_x}(f) = \sum_{\sigma \in \Sigma} \psi_n^\sigma \cdot f(\sigma) && && (15)\\ &\text{subject to} && \psi_k \in \mathcal{P}_k(\psi_{k-1}, x_k) && \forall k \in \{1, \ldots, n\} && (16)\\ & && x \in \mathcal{X}. && && (17) \end{aligned}$$

instance	O-D pair	M_a	bundles	M_a	bundles
1	14–20	31	39	∞	195
2	14–7	31	29	∞	374
3	12–18	28	56	∞	175
4	9–7	19	26	∞	202
5	4–8	35	73	∞	195

Table 4: Bundle set sizes for the earthquake problem

4.5 Handling multiple O-D pairs

The above method applies to a single O-D pair, but Peeta *et al.* minimize the expected weighted sum $\mathrm{E}^{P_x}\left(\sum_{i=1}^{5} w_i f_i\right)$ of shortest path lengths f_i between several O-D pairs i for weights w_i (which in their experiments were all set to 1 [36]). Unfortunately, there is likely to be little interchangeability in this problem, especially if (as we would expect) the O-D pairs are chosen to cover most of the network: for a given link to be irrelevant to the lengths of several paths is much less likely than for one path.

We can avoid this drawback by exploiting linearity of expectation and rewriting the objective as $\sum_{i=1}^{5} w_i \mathrm{E}^{P_x}(f_i)$ so that each expected path length can be computed separately using its own bundle set. Consequently, the resulting stochastic MIP will work with different coarse-grained distributions depending on the O-D pair and thus involve a different set of auxiliary variables ϕ_i for each bundle set S_i.

5 Computational case studies

5.1 Pre-disaster planning

We now apply our distribution shaping method using scenario bundles to the earthquake problem. The scenario bundling method is implemented in the Eclipse [2] constraint logic programming system (which provides a library of graph algorithms) and executed on a 2.8 GHz Pentium 4 with 512 MB RAM. The results are given in Table 4 for each O-D pair considered separately, and took approximately 0.05 seconds each to compute. The table shows the instances numbered 1–5, the O-D pairs, the constant M_a, and the size of the corresponding bundle set. For each O-D pair i the bundle sets S_i are remarkably small, representing scenario reduction of several orders of magnitude. We also show bundle set sizes with all paths allowed ($M_a = \infty$) and discuss these below.

We have replaced 2^{30} scenarios by a total of 223 bundles over 5 bundle sets S_1 to S_5, which allows us to find exact solutions to the problem using distribution shaping. Using the notation introduced in the previous section, the stochastic MIP to minimize the expected total shortest paths lengths f_i over the 5 O-D

B	link investment plan	objective
	approximate solutions (low M_p)	
B_1	20 21 22 23	86.7168
B_2	10 17 20 21 22 23 25	70.0352
B_3	10 13 16 17 20 21 22 25	59.5317
	exact solutions (low M_p)	
B_1	10 17 21 22 23 25	83.0801
B_2	4 10 12 17 20 21 22 25	66.1877
B_3	3 4 10 16 17 20 21 22 25	57.6802
	approximate solutions (high M_p)	
B_1	9 10 12 15 21 22 23 25	215.67
B_2	4 9 10 17 20 21 22 23 25	121.818
B_3	4 5 7 9 10 12 13 15 17 20 21 22 23 25	87.9268
	exact solutions (high M_p)	
B_1	10 17 21 22 23 25	212.413
B_2	4 10 12 17 20 21 22 25	120.083
B_3	3 4 10 16 17 20 21 22 25	78.4017

Table 5: Approximate and exact solutions for the earthquake problem

pairs i is

$$\text{Minimize} \quad \mathrm{E}^{P_x}\left(\sum_{i=1}^{5} f_i\right) = \sum_{i=1}^{5}\sum_{s\in S_i} \phi_{i,n}^{s} \cdot f_i(s) \tag{18}$$

$$\text{subject to} \quad \phi_{i,k} \in \mathcal{P}_k(\phi_{i,k-1}, x_k) \quad \forall i \in \{1,\ldots,5\}, k \in \{1,\ldots,n\} \tag{19}$$

$$x \in \mathcal{X}. \tag{20}$$

Solution times for the stochastic MIPs for the different budgets are under 5 seconds each on a 2.4GHz Intel Core i5-520M with 4GB RAM using the MIP solver of IBM ILOG CPLEX Optimization Studio Version 12.6[1] with default parameter settings. Thus the total time for our method to find an optimal investment plan is of the order of 5 seconds, compared to the several minutes taken by the approximate method of Peeta *et al.*

Table 5 show the approximate results of Peeta *et al.* and our exact results, including our exact evaluation of the objective function values of their approximate solutions. The results largely validate the method of Peeta *et al.* as their solutions are of good quality. However, the exact solutions are up to 10% better than the approximate solutions so the improvement is significant.

Peeta *et al.* remark that links 10, 20, 21, 22, 23 and 25 are invested in under most of their plans, and the same is true of ours. However, in some cases our plans look quite different to theirs. For example with B_1 and low M_p we invest

[1] IBM, ILOG, and CPLEX are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies.

B	link investment plan	objective	MIP solution time [s]
B_1	10 17 21 22 23 25	204.08	297.30
B_2	4 10 12 17 20 21 22 25	118.834	946.38
B_3	3 4 10 16 17 20 21 22 25	76.5491	512.13

Table 6: Exact solutions for the earthquake problem considering all paths

in more links than they do, while with B_3 and high M_p the reverse is true. It is not obvious in either case why one solution is better than another, illustrating the impracticality of finding good solutions manually.

Table 4 also shows bundle sizes for $M_a = \infty$, that is considering all paths between each O-D pair without neglecting path above the threshold M_a. This case was considered unrealistic by Peeta *et al.* as rescuers would use boats or helicopters instead of taking a long road route, but we include it to test our method further. The bundle sets are larger and took 0.23 seconds each to generate, and it took much longer to solve the resulting much larger stochastic MIP. The results and computation times are shown in Table 6. It can be noticed that the optimal values are slightly lower compared to the same instances with reduced number of allowed paths. This is due to the fact that the actual value of the disallowed path, which in the full model is between M_a and M_p was rounded up in the reduced model to M_p. Nevertheless, the experiments show that our new method is able to solve the entire problem to optimality within a few minutes, even without neglecting any paths.

5.2 Experiments with random road networks

To further evaluate our method we generate random road networks, which for the sake of realism should be planar graphs. Several methods exist for doing this but there is no general agreement on which is best, so we adapt one of the simplest: a grid method of [26]. They start with a square grid representing a road network in an idealized city, and add random dead-end links. Dead-ends introduce interchangeability and this could be viewed as artificially creating instances to favor our method, so we do not explicitly generate them. Instead we delete random edges to obtain variation in the network topology.

The method we use is as follows. We start from a grid of n squares, which has $2n(n+1)$ links and $(n+1)^2$ intersections. We then randomly delete links until the ratio of links to intersections is approximately 1.2: this and subsequent design choices were made to obtain similar characteristics to the Istanbul network. If the graph is not connected then we reject it and generate another. To each link e we assign a random length t_e in the interval $[1, 5]$, and a survival probability before investment p_e in $[0.5, 0.8]$; all values are uniformly distributed. All post-investment survival probabilities q_e are set to 1. To control the number of allowed paths indirectly we introduce a parameter $\alpha > 1$: for an O-D pair with shortest path distance d between them, we allow all paths with length up to $d\alpha$ by setting $M_a = d\alpha$. Bundle set sizes using the dynamic greedy heuristic are

			bundles				
n	links	scenarios	min	1st	med	3rd	max
$\alpha = 1.1$							
1	5	32	2	2	2	3	7
2	11	2048	2	2	3	4	21
3	19	524288	2	3	4	5	32
4	30	1.1×10^9	3	4	8	17	68
5	43	8.8×10^{12}	2	4	7	22	99
6	59	5.8×10^{17}	2	4	7	19	210
7	77	1.5×10^{23}	2	5	9	25	1900
$\alpha = 1.5$							
1	5	32	2	2	2	3	7
2	11	2048	2	2	3	7	49
3	19	524288	2	3	4	17	102
4	30	1.1×10^9	3	8	37	111	936
5	43	8.8×10^{12}	2	4	11	166	7592
6	59	5.8×10^{17}	2	4	42	652	55350
7	77	1.5×10^{23}	2	12	36	187	$>10^5$
$\alpha = \infty$							
1	5	32	5	5	5	7	7
2	11	2048	2	19	34	44	65
3	19	524288	2	29	46	87	304
4	30	1.1×10^9	4	387	535	863	4339
5	43	8.8×10^{12}	2	119	1735	8971	43173
6	59	5.8×10^{17}	19	2309	26201	$>10^5$	$>10^5$
7	77	1.5×10^{23}	29	14591	$>10^5$	$>10^5$	$>10^5$

Table 7: Bundle set sizes for random road networks

shown in Table 7, for different network sizes n and values of α. In each case we report the minimum, first quartile, median, third quartile and maximum bundle set sizes for 32 random instances. The results show that as the number of links increases, the bundle set size grows far more slowly than the number of scenarios, so cases that are larger than the Istanbul network will also be exactly solvable. The problem size is now reduced by many orders of magnitude, but the bundle set size can depend strongly on the number of allowed paths as controlled by α, and with large α it can become unmanageable. However, with a limited number of allowed paths we typically obtain small bundle sets even for larger networks.

As evidence that our artificial networks are fairly realistic, consider the results for the Istanbul network in Table 4 with $M_a = \infty$. There are 30 links and all are paths allowed, and the bundle set size per O-D pair ranges between 175–374. This is not dissimilar to the median bundle set size of 535 for the case $n = 4$ and $\alpha = \infty$ which also has 30 links and all paths allowed.

6 Further applications

We now outline some possible future applications of our ideas. Firstly, distribution shaping and scenario bundling are separate ideas that can be applied independently. Distribution shaping can be applied to stochastic optimization problems with endogenous uncertainty and one or more stages, independently of the presence of interchangeable values. Conversely, if interchangeability can be detected then scenario bundling can be applied to stochastic problems without endogenous uncertainty, again with one or more stages. These applications will be investigated in future work. Scenario bundling can also be applied to robust optimization problems, and we have already experimented with an adjustable robust optimization version of the earthquake problem. The MIP model in this case is much simpler than distribution shaping and scales up to hundreds of thousands of bundles.

Other two-stage stochastic problems have a similar structure to the earthquake problem and can potentially benefit from our methods. As an example we now outline a problem in project management: a stochastic version of the PERT problem [10]. Suppose we have a set of tasks to be scheduled, some or all with uncertain duration, and precedence constraints that determine a partial order on tasks which can be represented as a graph. The random task durations might take continuous distributions, but we can approximate these by discrete distributions using the *three-point estimation* technique from management and information systems applications. Three figures are produced for each task based on educated guesses: the best-case estimate, the most likely estimate and the worst-case estimate. This reduces the problem to a discrete one. Suppose also that we have enough budget to invest money in some tasks, and that investing in a task reduces the probability for its worst-case duration to occur while increasing the probabilities of the average and the best case. As in the earthquake problem there is endogenous uncertainty: the first-stage decisions affect the probability distributions of the first-stage random variables. It would be an instance of the generalized distribution shaping approach applied to m-ary scenarios as described in Section 3.1. The makespan in any scenario can be computed by finding the *longest* path length from the root to the terminal node, and we can choose an investment that minimizes the expected makespan. The makespan depends on the critical path which is a longest path instead of a shortest path, but interchangeable values will still occur, and the same bundling techniques as described in Section 4 can be used to arrive at a stochastic MIP that is small enough to be solvable for realistic instances.

7 Conclusion and Outlook

We have studied a challenging class of stochastic programs in which stochastic processes are influenced by optimization decisions—stochastic programs under endogenous uncertainty. In particular, we considered stochastic programs with binary decision variables that can alter the probability measures governing the random variables. Most solution methods proposed in the literature for this class of problems are based on approximations and heavily rely on problem structure. We developed a novel approach for handling decision-depending probabilities that seems to be the first to provide *exact* formulations and solution methods. Our *distribution shaping* technique is based on successive polyhedral characterization of probability measures and enables the formulation of corresponding stochastic programs as exact MIPs or mixed-integer bi-linear programs. Our *scenario bundling* technique—which is inspired by symmetry and dynamic interchangeability in constraint programming—allows the exact solution of large-scale MIPs that are formulated using the distribution shaping approach extended to scenario bundles. We demonstrate the effectiveness of the proposed approach by solving exactly a real-world case study—a pre-disaster planning problem detailed in [29]—that was previously only known to have approximate solutions and where exact solutions were considered out of reach. We also validate the approach on randomly generated and even larger instances of the same use-case. As an ongoing and future direction, we are experimenting with an adjustable robust optimization version of the pre-disaster problem and have extended scenario bundling to include robust considerations. We have observed that corresponding MIPs scale up even more and can be much simpler to handle. We intend to extend our approach to address other classes of stochastic programs under endogenous uncertainty (e.g. by relaxing the binary restriction on decisions) and apply it to other applications, such as Stochastic PERT. Another further step would be to extend the techniques to multi-stage problems.

Acknowledgements

We would like to thank the audience and session chairs at EURO-INFORMS joint International Meeting 2013 in Rome and the International Conference on Operations Research (OR 2014) in Aachen, where we have presented this work, for the interesting discussion and insightful comments that helped to further improve the paper and widen the scope and applicability of the approach. We particularly thank our colleague Bruno Flach for sharing details about his approach [11] and numerous discussions on this topic.

References

[1] S. Ahmed. Strategic planning under uncertainty: Stochastic integer programming approaches. Ph.D. thesis, University of Illinois at Urbana-Champaign, 2000.

[2] K. R. Apt, M. Wallace. Constraint Logic Programming using Eclipse. Cambridge University Press, 2007.

[3] A. M. Beckwith, B. Y. Choueiry. On the Dynamic Detection of Interchangeability in Finite Constraint Satisfaction Problems. *7th International Conference on Principles and Practice of Constraint Programming, Lecture Notes in Computer Science* vol. 2239, 2001, p. 760.

[4] J. R. Birge, and F. V. Louveaux. Introduction to stochastic programming. Springer, 2011.

[5] J.R. Birge. Stochastic programming computation and applications. *INFORMS J. Comput.* 9, 1997, pp. 111133.

[6] B. Y. Choueiry, A. M. Davis. Dynamic Bundling: Less Effort for More Solutions. *5th International Symposium on Abstraction, Reformulation and Approximation, Lecture Notes in Computer Science* vol. 2371, Springer-Verlag, 2002, pp. 64–82.

[7] B. Y. Choueiry, G. Noubir. On the Computation of Local Interchangeability in Discrete Constraint Satisfaction Problems. *AAAI/IAAI*, 1998, pp. 326–333.

[8] C. J. Colbourn. Concepts of Network Reliability. Wiley Encyclopedia of Operations Research and Management Science, John Wiley & Sons, Inc., 2010.

[9] M. Colvin, C. T. Maravelias. A Stochastic Programming Approach for Clinical Trial Planning in New Drug Development. *Computers and Chemical Engineering* **32**(11):2626–2642, 2008.

[10] J. Dupačová. Optimization under exogenous and endogenous uncertainty. University of West Bohemia in Pilsen, 2006.

[11] B. Flach and M. Poggi: On a class of stochastic programs with endogenous uncertainty: theory, algorithm and application. Monografias em Ciencia da Computacão No 05/10, PUC, Rio, 2010. (presented at ISMP 2012)

[12] E. C. Freuder. Eliminating Interchangeable Values in Constraint Satisfaction Problems. *National Conference on Artificial Intelligence*, 1991, pp. 227–233.

[13] I. P. Gent, T. Kelsey, S. A. Linton, J. Pearson, C. M. Roney-Dougal. Groupoids and Conditional Symmetry. *13th International Conference on Principles and Practice of Constraint Programming, Lecture Notes in Computer Science* vol. 4741, 2007, pp. 823–830.

[14] V. Goel, I. E. Grossmann. A Class of Stochastic Programs with Decision Dependent Uncertainty. *Mathematical Programming* **108**(2):355–394, 2006.

[15] V. Goel, I. E. Grossmann. A stochastic programming approach to planning of offshore gas field developments under uncertainty in reserves. *Computers & Chemical Engineering* 28 (8), 2004, pp. 14091429.

[16] V. Gupta, I. E. Grossmann. Solution strategies for multistage stochastic programming with endogenous uncertainties. *Computers & Chemical Engineering* 35.11, 2011, pp. 2235-2247.

[17] V. Gupta, I. E. Grossmann. A new decomposition algorithm for multistage stochastic programs with endogenous uncertainties. *Computers & Chemical Engineering* 62, 2014, pp. 62-79.

[18] A. Haselböck. Exploiting Interchangeabilities in Constraint Satisfaction Problems. *13th International Joint Conference on Artificial Intelligence*, 1993, pp. 282–287.

[19] P. D. Hubbe, E. C. Freuder. An Efficient Cross Product Representation of the Constraint Satisfaction Problem Search Space. *10th National Conference on Artificial Intelligence*, San Jose, California, USA, 1992, pp. 421–427.

[20] T. W. Jonsbraten. Optimization models for petroleum field exploitation. Department of Business Administration. Norway, Stavanger College, 1998.

[21] T. W. Jonsbraten, R. J. B. Wets, D. L. Woodruff. A class of stochastic programs with decision dependent random elements. *Annals of Operations Research* 82, 1998, pp. 83106.

[22] S. Karakashian, R. Woodward, B. Y. Choueiry, S. D. Prestwich, E. C. Freuder. A Partial Taxonomy of Substitutability and Interchangeability. *10th International Workshop on Symmetry in Constraint Satisfaction Problems*, 2010. (Journal paper in preparation.)

[23] A. Lal, B. Y. Choueiry, E. C. Freuder. Neighborhood Interchangeability and Dynamic Bundling for Non-Binary Finite CSPs. *10th National Conference on Artificial Intelligence and 17th Innovative Applications of Artificial Intelligence Conference*, 2005, pp. 397-404.

[24] D. Lesaint. Maximal Sets of Solutions for Constraint Satisfaction Problems. *11th European Conference on Artificial Intelligence*, 1994, pp. 110–114.

[25] H. Levy. Stochastic Dominance and Expected Utility: Survey and Analysis. *Management Science* 38:555–593, 1992.

[26] A. P. Masucci, D. Smith, A. Crooks, M. Batty. Random Planar Graphs and the London Street Network. *European Physical Journal B* 71(2):259–271, 2009.

[27] N. Neagu. Studying Interchangeability in Constraint Satisfaction Problems. *8th International Conference on Principles and Practice of Constraint Programming, Lecture Notes in Computer Science* vol. 2470, 2002, pp. 787–788.

[28] A. J. Parkes. Exploiting Solution Clusters for Coarse-Grained Distributed Search. *IJCAI Workshop on Distributed Constraint Reasoning*, 2001.

[29] S. Peeta, F. Siebel Salman, D. Gunnec, K. Viswanath: Pre-disaster investment decisions for strengthening a highway network. Computers & Operations Research 37, 2010.

[30] S. D. Prestwich. Full Dynamic Interchangeability with Forward Checking and Arc Consistency. *Workshop on Modeling and Solving Problems With Constraints*, Valencia, 2004.

[31] S. D. Prestwich, M. Laumanns, B. Kawas. Value Interchangeability in Scenario Generation. *19th International Conference on Principles and Practice of Constraint Programming, Lecture Notes in Computer Science* vol. 8124, Springer 2013, pp. 587–595.

[32] A. Prékopa. Stochastic Programming. Springer, 1995.

[33] R. Y. Rubinstein, A. Shapiro. Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization by the Score Function Method. J. Wiley & Sons, Chichester, 1993.

[34] A. Ruszczyński. Decomposition Methods in Stochastic Programming. *Mathematical Programming* 79:333–353, 1997.

[35] N. V. Sahinidis. Optimization under uncertainty: State-of-the-art and opportunities. *Comput. Chem. Eng.* 28 (6-7), 2004, pp. 971983.

[36] S. Salman, personal communication.

[37] R. Schultz. Stochastic programming with integer variables. *Mathematical Programming* 97 (1-2), 2003, pp. 285309.

[38] S. Solak, J.-P. B. Clarke, E. L. Johnson, E. R. Barnes. Optimization of R&D Project Portfolios Under Endogenous Uncertainty. *European Journal of Operational Research* **207**:420–433, 2010.

[39] B. Tarhan, I. E. Grossmann, V. Goel. Stochastic programming approach for the planning of offshore oil or gas field infrastructure under decision-dependent uncertainty. *Industrial and Engineering Chemistry Research* 48 (6), 2009, pp. 3078-3097.

[40] P. Vayanos, D. Kuhn, and B. Rustem. Decision rules for information discovery in multi-stage stochastic programming. *Decision and Control and European Control Conference (CDC-ECC), 50th IEEE Conference on. IEEE*, 2011.

[41] R. Weigel, B. V. Faltings, B. Y. Choueiry. Context in Discrete Constraint Satisfaction Problems. *12th European Conference on Artificial Intelligence*, 1996, pp. 205–209.

[42] H. Zantema, H. L. Bodlaender. Sizes of Ordered Decision Trees. *International Journal of Foundations of Computer Science* **13**(3):445–458, 2002.