ECAI 2014
T. Schaub et al. (Eds.)

doi:10.3233/978-1-61499-419-0-741

Symmetry Breaking for Exact Solutions in Adjustable Robust Optimisation

S. D. Prestwich[1] and **M. Laumanns**[2] and **B. Kawas**[3]

Abstract.
One of the key unresolved challenges in Adjustable Robust Optimisation is how to deal with large discrete uncertainty sets. In this paper we present a technique for handling such sets based on symmetry breaking ideas from Constraint Programming. In earlier work we applied the technique to a pre-disaster planning problem modelled as a two-stage Stochastic Program, and we were able to solve exactly instances that were previously considered intractable and only had approximate solutions. In this paper we show that the technique can also be applied to an adjustable robust formulation that scales up to larger instances than the stochastic formulation. We also describe a new fast symmetry breaking heuristic that gives improved results.

1 Introduction

Robust Optimisation (RO) [6] is a methodology that was developed for uncertain decision environments in which: (i) input data and probability distributions governing random processes are not known exactly or are not readily available; (ii) small data perturbations can heavily affect the feasibility or optimality of decision problems; and (iii) the underlying decision problems are large-scale (data and decision vectors have large dimensions).

RO aims to construct solutions that are immunized against parameter uncertainty, and unlike in Stochastic Programming the uncertainty model is deterministic and set-based. RO also aims to derive tractable formulations of the decision model and address the curse of dimensionality that Stochastic Programming suffers from. Robust convex optimisation dates back to a 1973 paper [21], which was the first to consider a linear optimisation model whose solutions are feasible for all data belonging to a convex set. However, the approach suffered from over-conservatism and did not gain traction until the late 1990s and early 2000s [3, 4, 10, 13, 14] when less conservative models were proposed by using ellipsoidal and polyhedral uncertainty and introducing the notion of a *budget of uncertainty*. The latter provides decision-makers with a way of controlling conservatism, and the ability to choose trade-offs between robustness and performance that match their risk aversion. An alternative to approximation methods is the use of metaheuristics, for example [8].

Adjustable RO (ARO) [5], also called Robust Adaptable Optimisation or Multi-Stage RO, is an extension to the methodology. It is also a worst-case approach but it provides more flexibility and better solutions that are still immunized against data uncertainty. In ARO a distinction is made between *here and now* and *wait and see* (or *recourse*) decisions. The latter, unlike the former, can be adjusted to the realisation of the uncertain data. ARO methodologies have gained attention in recent years, especially for solving practical problems with high infeasibility costs such as those in network and transportation systems [1]. As detailed in [5], the flexibility given by ARO comes at the cost of intractability, even when the two stages are simple Linear Programs (LPs).

To address the intractability of ARO a number of approximation methods have been proposed. For example [5] restrict the second-stage recourse to be an affine function of the realized data, which enables the reformulation of the two-stage ARO to a single-stage problem that can be solved efficiently. More recently, two variants of a master-subproblem framework have been developed: (i) Benders-like decomposition methods [7, 22] in which cutting planes are supplied to the master problem using revealed uncertainty and dual information of the second-stage recourse problem; and (ii) a column-and-constraint generation method [24] in which recourse variables are generated only for significant scenarios of values taken by the uncertain variables. When the recourse problem is an LP, both these methods converge to an optimal solution in a finite number of iterations. When the recourse is a Mixed Integer Program (MIP), however, the Benders-like method is inapplicable as strong duality does not hold and exact dual information is not attainable. The column-and-constraint generation method is still applicable for MIP recourse, but identifying significant scenarios can prove challenging. In [25] this issue is addressed and an exact nested column-and-constraint generation algorithm is proposed.

In this paper we propose an exact method for solving some hard RO and ARO problems, inspired by ideas from Constraint Programming (CP). First, consider a general RO formulation:

$$\text{minimize}_{\mathbf{x}}\ a \text{ subject to } f(\mathbf{x},\mathbf{u}) \le a,\ g(\mathbf{x},\mathbf{u}),\ \forall \mathbf{u} \in \mathcal{U}$$

where $\mathbf{x}$ is a vector of decision variables, f is a real-valued objective function, g represents the constraints, and the uncertainty parameters $\mathbf{u}$ take values in the *uncertainty set* $\mathcal{U}$. We refer to each vector $\mathbf{u}$ as a scenario. If $\mathcal{U}$ is small then this formulation can be solved directly by an appropriate (depending on the form of $f, g, \mathbf{x}, \mathbf{u}$) solver, simply by enumerating the constraints for all possible values $\mathbf{u}$. However, even in the case of finite $\mathcal{U}$ it is often far too large to handle directly. Instead it is transformed to an equivalent or approximately equivalent form that can be solved efficiently. Results are good for many classes of problem, but RO methods cannot be applied to all problems. Similar formulations can be given for ARO problems with two or more stages, by introducing a copy of each recourse variable for each scenario.

Our proposal is to use the above formulation (or its ARO equiv-

[1] Insight Centre for Data Analytics, University College Cork, Ireland, email: s.prestwich@cs.ucc.ie
[2] IBM Research — Zurich, 8803 Rueschlikon, Switzerland, email: mlm@zurich.ibm.com
[3] IBM Research — Watson, NY, USA, email: bkawas@us.ibm.com

alent) directly, if we can sufficiently reduce the size of $\mathcal{U}$ by detecting symmetries between its scenarios. That is, if we can show that $g(\mathbf{x}, \mathbf{u}_1) \leftrightarrow g(\mathbf{x}, \mathbf{u}_2)$ and $f(\mathbf{x}, \mathbf{u}_1) = f(\mathbf{x}, \mathbf{u}_2)$ $\forall \mathbf{u}_1, \mathbf{u}_2 \in \mathcal{S}$ for some $\mathcal{S} \subseteq \mathcal{U}$ then we need only consider one arbitrary member $\mathbf{u} \in \mathcal{S}$, so $\mathcal{S}$ can be replaced by $\{\mathbf{u}\}$ in $\mathcal{U}$. In this way we can reduce equivalence classes of scenarios to single scenarios, which may drastically reduce the size of $\mathcal{U}$ without resorting to approximations. The success of this approach depends on the presence of symmetry in the uncertainty set, but it does not (at least in principle) depend on the form of the problem: if all constraints are linear then we have a MIP model, but we could equally handle more general constraints and use a CP model. Where standard RO methods can be applied our approach is unnecessary because those methods are very effective. But when they cannot be applied, for example in CP or ARO, our approach may be a useful alternative to approximations and metaheuristics.

We shall test our proposal using a case study: an ARO problem related to robust shortest paths but with additional complicating features. It is adapted from a real-world pre-disaster planning problem that has previously been modelled as a 2-stage stochastic program [17, 18]. While our stochastic approach in [18] was able to quickly solve to optimality instances previously considered intractable, our new ARO approach greatly improves scalability. We also present a symmetry breaking heuristic that is much faster than our previous method and breaks more symmetry. The problem is described in Section 2, a symmetry breaking approach for this type of problem is described in Section 3, it is applied to the problem in Section 4, and conclusions are drawn in Section 5.

2 A case study

First we describe the problem that inspired our research, which was described by Peeta *et al.* [17].

2.1 Problem description

The Turkish government must choose which links in the Istanbul road network to invest in, to strengthen them against a possible future earthquake. The objective is to facilitate rescue operations under a large number of possible scenarios. This is a real-world pre-disaster planning problem with data taken from national and international reports.

The Istanbul road network is represented by an undirected graph $G = (V, E)$ with 25 nodes V and 30 edges or links E. Each link represents a highway with a given length t_e, and may fail with some given probability p_e, while each node represents a junction. The failure probability of a link can be reduced to q_e by investing money in it with cost c_e, but a budget B limits the total investment. To maximise post-quake accessibility, the objective is to minimise the sum of the expected shortest path lengths between a given set of origin and destination (O-D) nodes in the network, by investing in carefully-chosen links. Five O-D pairs were chosen to represent rescue operations between hospitals and areas of high population.

If an O-D pair is unconnected 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 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 allowing 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 for each O-D pair, and choose either a low M_p value (M_a) or the high value. The total investment budget is B and three budget levels B_1, B_2, B_3 are considered, corresponding to 10%, 20% and 30% of the cost of investing in all links. See [17] for all figures, the road network topology and the O-D pairs.

2.2 A robust formulation

We ignore the probabilities p_e, q_e for our robust approach, and instead of minimising expected path lengths we take a robust objective. 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. Also define a binary uncertain variable r_e which is 1 if link e survives and 0 if it fails. In the first stage of the ARO problem we decide which links to invest in by assigning values to the x_e, then link failures occur and cause values to be assigned to the r_e. In the second stage we choose a shortest path between the O-D pairs, based on the surviving links. If they are no longer connected by an allowed path, or if the path is longer than M_a, then the value M_p is used instead of a path length.

Several possible RO approaches have been described for shortest paths [12] and many other problems. The uncertainty set may be a discrete set of scenarios or based on continuous intervals of possible values. For the earthquake problem either a link fails or it survives, so intervals cannot be used. It does not make sense to consider fractional values because we are not aiming to maximise flow through the network: if an ambulance can traverse a link then this is sufficient, even if it must do so slowly because of partial damage.

The simplest robust objective is to minimise the worst case path length, but this is generally considered far too conservative as it considers only one unlikely scenario (the worst case) and ignores what happens in all others. An interesting objective is the *bw*-robustness criterion [19], in which we maximise the number of scenarios under which the path length (in this problem) is no greater than *b*, while guaranteeing that it is no greater than *w* under any scenario. We could apply this objective to our problem with the proviso that path length is modified to include the penalty M_p when no path exists. Unfortunately there is no reasonable value for *w* because of the scenario where all links fail. We could simply set *w* to 0 and maximise the number of scenarios in which the path length (or penalty) is no greater than some value *b*. But a drawback with this objective is that it is very sensitive to the value of M_p, making it non-trivial to choose *b*. The *rw*-criterion is better suited to shortest path problems in which link lengths are uncertain, rather than link survival.

A better objective for our problem was described in [9, 10]. For each uncertain variable v we assume a *nominal* value $\tilde{v}$ and a *worst case* value $\tilde{v} + \hat{v}$. In our case $\tilde{v} = 1$ and $\hat{v} = -1$: the nominal case is the survival of a link while the worst case is its failure. We then assume that the very worst scenarios (in which almost all links fail) are highly unlikely and can reasonably be ignored. We introduce a parameter Γ called the *budget of uncertainty* to represent the maximum number of worst case values to be considered. For our problem this means that we must choose the number Γ of links that might reasonably fail in an earthquake. Though it is not obvious which value to choose, Γ is independent of other parameters such as M_p. Using this type of objective with binary uncertain variables is called *cardinality constrained uncertainty*, and it allows us to optimise over a large number of bad cases while ignoring highly unlikely worst cases. By varying Γ we can adjust the level of risk aversion. The uncertainty set is the set of all binary vectors with exactly Γ zeroes.

However, there is a complication: as in the stochastic approaches of [17, 18] we assume that a link always survives if we invest in it. When decisions affect uncertainty this is sometimes called *endogenous uncertainty*, and this non-standard feature can make problems much harder to model and solve. In our problem, given a sufficient budget we can invest in more than $n-\Gamma$ links for any given Γ (where n is the number of links), thus preventing the assumed worst cases of Γ failures. Moreover, because different links may have different investment costs, this contradiction may occur in some scenarios and not in others. To handle this complication we modify $\mathcal{U}$ so that *at most* Γ links are allowed to fail. Now our model may contain redundant constraints, because a scenario with Γ zeroes subsumes a scenario with a subset of those zeroes, but this does not affect the correctness of the model.

To choose an investment plan we solve the following MIP:

$$\begin{array}{ll} \text{Minimize } \sum_{k\in\mathcal{P}} z_k & \\ \text{subject to } \; z_k \geq f^s\left(1-\sum_{e\in C^s} x_e\right) & \forall s\in\mathcal{U}_k \text{ and } \forall k\in\mathcal{P} \\ \sum_{e\in E} c_e x_e \leq B & \\ x_e \in \{0,1\} & \forall e\in E \end{array}$$

where $\mathcal{P}$ is a set of O-D pairs, z_k is a real auxiliary variable for O-D pair k, for each pair $k \in \mathcal{P}$ we have a set $\mathcal{U}_k$ of scenarios (in the problem as described these are all the same set but below we shall use different sets), in scenario $s \in \mathcal{U}_k$ the path length is f^s, and $C^s \subseteq E$ is the set of failing links in scenario s. Notice that for this problem we can precompute all recourse decisions (the shortest paths) to find the recourse function values f^s, so recourse variables do not appear in the MIP. To handle the endogenous uncertainty, if we invest in a link e then any scenario in which it fails has its z_k-constraint disabled ($x_e = 1$ so the right hand side becomes non-positive). The drawback with this model is that the sets $\mathcal{U}_k$ may be very large: even though we do not enumerate all 2^n possible scenarios there are still $\sum_{i=0}^{\Gamma}\binom{n}{i}$. To address this we exploit symmetries between scenarios, as described in the next Section.

3 Scenario bundling

In this section we show how to represent the set of scenarios by a much smaller but equivalent set of scenarios, which makes the problem tractable. This exploits a particular form of symmetry that, as we shall show, occurs in shortest path problems.

3.1 Value interchangeability

An early form of symmetry that has received considerable attention in CP is *(value) interchangeability* [11]:

Definition. 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 [11] and subsequent work in this area is surveyed in [15]. The relevant variant here is called *dynamic interchangeability*.

Definition. 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.

3.2 Combining scenarios into bundles

Dynamic interchangeability occurs in shortest path problems such as the earthquake problem. As an illustration consider the simple network in Figure 1 with links $e \in \{1, 2, 3, 4\}$. We set all lengths $t_e \equiv 1$, investment costs $c_e \equiv 1$, budget $B = 2$, $\Gamma = 3$ and $M_a = M_p = 4$. We must choose two links to invest in, to minimize the shortest path length between nodes 1 and 4 over a number of worst-case scenarios. If we restricted scenarios to exactly Γ zeroes, as is usual, then there would be 4 scenarios in $\mathcal{U}$. But we only forbid more than Γ zeroes so there are 15: only scenario $(0, 0, 0, 0)$ is excluded. The optimal plan is of course to invest in links 1 and 4, giving a robust shortest path length of 2. That is, if we invest in links 1 and 4 then we have a guaranteed path of length 2 over all scenarios in which those two links survive (scenarios in which either fails are forbidden). There are four such scenarios: $(1, 0, 0, 1)$, $(1, 1, 0, 1)$, $(1, 0, 1, 1)$ and $(1, 1, 1, 1)$. But for our solution method we must consider $\mathcal{U}$ in the absence of investments so we still have 15 scenarios to consider.

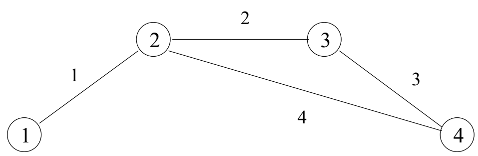

Figure 1. A small network example

Some scenarios can be considered together instead of separately. For example links 1 and 4 survive in the above four scenarios, and 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. We shall call a product such as $(1, *, *, 1)$ a *scenario bundle* by analogy with solution bundles in CP. (This usage is distinct from *bundle methods* in Stochastic Programming [20].) Though interchangeability is defined here on uncertain variables instead of decision variables, the principle is the same.

3.3 Finding small bundle sets

It is impractical to enumerate a large number of scenarios then look for ways of bundling some of them together, as we did in the small example. Instead we enumerate scenarios by tree search on the uncertain variables r_e (the *scenario tree*) and apply symmetry breaking during search. To limit the search to scenarios with at most Γ failures we post a constraint $\sum_e r_e \geq n - \Gamma$.

Consider a node in the scenario tree at which links $1 \dots i-1$ have been realized, so that variables $x_1 \dots x_{i-1}$ have been assigned values, and we are about to assign a value to x_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 $x_i \dots x_n$ to 1 or 0 respectively, and applying a shortest path algorithm. Now if $C_i = F_i$ then the value assigned to x_i is irrelevant: the shortest path length in each scenario under this tree node is independent of the value of x_i, so the values are interchangeable. In this case there is no need to branch and we can simply assign value * to the variable. Moreover, 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.

The order in which we assign the x variables affects the cardinality of the bundle set. Two alternative bundle sets $\mathcal{U}_1, \mathcal{U}_2$ for the example are shown in Table 1 along with their link permutations, and the path length f^s for each bundle. Once we have obtained a bundle set we can discard the permutation used to derive it. We can replace the symbol * by any domain value and use the result as a representative scenario for the bundle: we choose 1. The end result is a reduced uncertainty set of scenarios such as $\mathcal{U}_2'$.

links				
3	2	4	1	f^s
0	*	0	*	4
0	*	1	0	4
0	*	1	1	2
1	0	0	*	4
1	0	1	0	4
1	0	1	1	2
1	1	0	0	4
1	1	0	1	3
1	1	1	0	4

$\mathcal{U}_1$ bundles

links				
1	4	2	3	f^s
0	*	*	*	4
1	0	0	*	4
1	0	1	0	4
1	0	1	1	3
1	1	*	*	2

$\mathcal{U}_2$ bundles

links			
1	2	3	4
0	1	1	1
1	0	1	0
1	1	0	0
1	1	1	0
1	1	1	1

$\mathcal{U}_2'$ scenarios

Table 1. Bundle and scenario sets for the small example

Notice that both $\mathcal{U}_1$ and $\mathcal{U}_2$ implicitly include scenario $(0,0,0,0)$ which is forbidden by $\Gamma = 3$. This will often occur but is not a problem: the path length in all scenarios in a bundle is the same, so any forbidden scenario is equivalent to a permitted scenario in the same bundle. (Replacing * by 1 eliminates these cases anyway.) From now on we shall refer to reduced uncertainty sets $\mathcal{U}'$ of scenarios instead of bundle sets.

Using a static variable ordering, the problem of finding the smallest cardinality $\mathcal{U}'$ corresponds exactly to the problem of finding a variable permutation that minimizes the number of paths in a binary decision tree [18]. This is known to be NP-complete [23]. In [18] we used hill-climbing to find a good permutation; but because each local move required a complete tree search to evaluate, this dominated the total execution time. If we allow a dynamic variable ordering, in which the choice of uncertain variable to assign next depends on which path we took to the current node, the problem becomes more complex but smaller $\mathcal{U}'$ can potentially be found. To obtain a small $\mathcal{U}'$ quickly we introduce a dynamic branching heuristic as follows.

At each node of the scenario tree we must choose an uncertain variable r_e representing an unrealized link e, and assign it to 0, 1 or *. To choose a variable we first create an ordered list $L = \langle p_1, \dots, p_k \rangle$ of all allowed paths between the O-D pair, in increasing order of path length under the assumption that all unrealised links survive. Using this list, for each unrealized link e construct a vector $\langle v_1, \dots v_k \rangle$ where $v_i = 0$ if e is in p_i and $v_i = 1$ otherwise. Then choose the variable whose link has the lexicographically smallest vector. For example in the small network in Figure 1, $L = \langle \langle 1,2,3 \rangle, \langle 1,4 \rangle \rangle$ 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 Application to the case study

We now apply reduced scenario sets to the case study problem by reducing the uncertainty sets $\mathcal{U}_k$ in the MIP model of Section 2.2. We can use a different reduced set for each O-D pair, which is fortunate because there are likely to be few links that are interchangeable with respect to all five pairs simultaneously. Treating the pairs separately greatly increases the symmetry in the problem.

4.1 Original instances

First we tackle the original instances of [17] using our robust method. What value should we choose for Γ? In general this is highly problem-dependent, but it turns out that choosing any $\Gamma \geq 5$ produces the same $\mathcal{U}'$ so we only have a few values to try, and we tried all of them. Table 2 shows robust and optimal investment plans and their actual objective values (computed using our stochastic model from [18][4]) for each of the six instances (three budget levels and two M_p values) and $\Gamma = 1 \dots 5$. It shows that $\Gamma = 2$ consistently gave best results: for low M_p the investment plans have objectives 1.7%, 3.7% and 0% above optimal for B_1, B_2, B_3 respectively, and for high M_p 8.9%, 14.2% and 0%. These results are quite good considering that they ignore the link survival probabilities. The probabilities are anyway only estimates, and a robust objective is arguably more appropriate; see the RO literature for arguments on this point.

The uncertainty set reduction 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. For $\Gamma = 2$ the reduced scenario set sizes are 22, 12, 20, 9 and 16 for the five O-D pairs, and took approximately 0.03 seconds each to compute. We have replaced 2^{30} scenarios by a total of 79 scenarios, representing a scenario reduction of over 7 orders of magnitude, and this makes the problem tractable. Solution times for the robust MIPs are approximately 0.03 seconds on a 2.4GHz Intel Core i5-520M with 4GB RAM using the MIP solver of IBM ILOG CPLEX Optimization Studio Version 12.6[5] with default parameter

[4] [18] contained a typographical error: link 3 in the B_3 plan for high M_p instead appeared in the B_2 plan.

[5] 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.

settings. Thus the total time for our method to find an investment plan is less than 0.2 seconds, compared to the several minutes taken by the approximate method of Peeta *et al.* and approximately 5 seconds by our stochastic approach.

We also computed reduced scenario set sizes with all paths allowed ($M_a = \infty$) and $\Gamma = 2$: 22, 21, 20, 9 and 16, almost the same as before. 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 how reliant our method is on restricting the number of allowed paths: using $\Gamma = 2$ it is hardly affected at all. These sets took approximately 0.05 seconds each to generate and the ARO model is again solved in approximately 0.03 seconds. Our stochastic MIP model takes several minutes to solve these instances, which have several hundred scenarios, again showing that the robust approach is more scalable than the stochastic one.

B	M_p	case	link investment plan	objective
B_1	low	$\Gamma=1$	21 22 25	89.511
B_2	low	$\Gamma=1$	10 17 20 21 22 25	70.035
B_3	low	$\Gamma=1$	10 13 16 17 20 21 22 25	59.532
B_1	low	$\Gamma=2$	3 4 21 22 25	84.524
B_2	low	$\Gamma=2$	3 4 12 17 20 21 22 25	68.621
B_3	low	$\Gamma=2$	3 4 10 16 17 20 21 22 25	57.680
B_1	low	$\Gamma=3$	3 4 21 22 25	84.524
B_2	low	$\Gamma=3$	3 4 7 10 12 13 21 22 25	75.800
B_3	low	$\Gamma=3$	10 13 16 17 20 21 22 25	59.532
B_1	low	$\Gamma=4$	3 4 21 22 25	84.524
B_2	low	$\Gamma=4$	3 4 7 10 12 13 21 22 25	75.800
B_3	low	$\Gamma=4$	10 13 16 17 20 21 22 25	59.532
B_1	low	$\Gamma=5$	3 4 21 22 25	84.524
B_2	low	$\Gamma=5$	3 4 7 10 12 13 21 22 25	75.800
B_3	low	$\Gamma=5$	10 13 16 17 20 21 22 25	59.532
B_1	low	optimal	10 17 21 22 23 25	83.080
B_2	low	optimal	4 10 12 17 20 21 22 25	66.188
B_3	low	optimal	3 4 10 16 17 20 21 22 25	57.680
B_1	high	$\Gamma=1$	21 22 25	255.473
B_2	high	$\Gamma=1$	10 17 20 21 22 25	140.058
B_3	high	$\Gamma=1$	10 13 16 17 20 21 22 25	85.786
B_1	high	$\Gamma=2$	3 4 21 22 25	231.311
B_2	high	$\Gamma=2$	3 4 12 17 20 21 22 25	137.126
B_3	high	$\Gamma=2$	3 4 10 16 17 20 21 22 25	78.402
B_1	high	$\Gamma=3$	3 4 21 22 25	231.311
B_2	high	$\Gamma=3$	3 4 7 10 12 13 21 22 25	171.489
B_3	high	$\Gamma=3$	10 13 16 17 20 21 22 25	85.786
B_1	high	$\Gamma=4$	3 4 21 22 25	231.311
B_2	high	$\Gamma=4$	3 4 7 10 12 13 21 22 25	171.489
B_3	high	$\Gamma=4$	10 13 16 17 20 21 22 25	85.786
B_1	high	$\Gamma=5$	3 4 21 22 25	231.311
B_2	high	$\Gamma=5$	3 4 7 10 12 13 21 22 25	171.489
B_3	high	$\Gamma=5$	10 13 16 17 20 21 22 25	85.786
B_1	high	optimal	10 17 21 22 23 25	212.413
B_2	high	optimal	4 10 12 17 20 21 22 25	120.083
B_3	high	optimal	3 4 10 16 17 20 21 22 25	78.402

Table 2. Robust and optimal solutions for the case study

The tractability of the shortest path algorithm certainly contributes to the efficiency of our method. However, if the recourse computation involved the solution of a manageable number of tractable constraint satisfaction problems, this would still be practicable.

4.2 Larger random instances

The original instances turn out to be very easy for our approach, so to test it further we generate larger 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 [16]. 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 s squares, which has $n = 2s(s+1)$ links and $(s+1)^2$ 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 uniformly distributed in the interval $[1, 5]$. 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 $M_a = d\alpha$.

Reduced scenario set sizes using the dynamic greedy heuristic are shown in Table 3, for different network sizes s and values of α. Following results for the original instances we set $\Gamma = \frac{n}{15}$ (rounded to the nearest integer). In each case we report the minimum, first quartile, median, third quartile and maximum reduced set sizes for 32 random instances. We also show the value of $\binom{n}{\Gamma}$ which is the number of scenarios we would theoretically need to consider in a pure ARO approach. The results show that as the number of links increases the reduced sets grow slowly, so cases that are larger than the Istanbul network are also solvable. Even when the number of scenarios is large, the robust MIP model is much more efficient than the stochastic MIP model, and can solve instances with hundreds of thousands of scenarios in minutes.

				scenarios				
s	n	scenarios	$\binom{n}{\Gamma}$	min	1st	med	3rd	max
$\alpha = 1.1$								
2	11	2048	11	2	2	3	4	6
3	19	524288	19	2	3	4	4	7
4	30	1.1×10^9	435	3	4	8	15	48
5	43	8.8×10^{12}	12341	2	4	7	22	91
6	59	5.8×10^{17}	455126	2	4	7	19	210
7	77	1.5×10^{23}	2.0×10^7	2	5	9	25	1900
$\alpha = 1.5$								
2	11	2048	11	2	2	3	4	6
3	19	524288	19	2	3	4	4	7
4	30	1.1×10^9	435	3	8	19	30	56
5	43	8.8×10^{12}	12341	2	4	11	82	360
6	59	5.8×10^{17}	455126	2	4	42	336	1799
7	77	1.5×10^{23}	2.0×10^7	2	12	36	187	46415
$\alpha = \infty$								
2	11	2048	11	2	2	3	4	6
3	19	524288	19	2	3	4	4	7
4	30	1.1×10^9	435	4	13	21	32	56
5	43	8.8×10^{12}	12341	2	21	65	116	360
6	59	5.8×10^{17}	455126	19	146	420	747	2641
7	77	1.5×10^{23}	2.0×10^7	29	553	1883	5895	47700

Table 3. Reduced scenario set sizes for random road networks

5 Conclusion

We have shown that an ARO approach to a pre-disaster planning problem is made possible by exploiting symmetry between scenarios, and finds high-quality solutions despite ignoring the probabilities in the problem. An earlier stochastic approach found optimal solutions but the ARO approach has two advantages: it is much more scalable, and it does not require the user to estimate link survival probabilities.

Reducing uncertainty sets in ARO problems is a new application area for CP symmetry breaking methods, and we expect that other ARO problems will exhibit different forms of symmetry. Our method is also potentially useful for robust versions of CP problems, for which RO methods are inapplicable. In future work we shall explore the exploitation of symmetries in other ARO applications.

Our approach is related to that of [25] in that we have a two-stage ARO problem with a discrete uncertainty set and uncertain parameters that are a function of the first-stage decision variables. However, with our approach we are able to provide to the first-stage problem the recourse functions and associated constraints up-front, and not in an iterative fashion as in [25]. Moreover, we do not identify significant scenarios or exclude redundant ones: instead we identify sets of equivalent scenarios and replace them by a single representative, thus providing only one representative corresponding recourse function value and constraint. Another distinction to note here is that — similar to our work in [18] — the revealed uncertainty information and the recourse is generated during bundling as opposed to within the optimisation model.

Acknowledgments

The authors gratefully acknowledge the financial support provided by the Irish Industrial Development Agency under reference 199954. This publication has emanated from research supported in part by a research grant from Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289.

REFERENCES

[1] A. Atamtürk, M. Zhang. Two-Stage Robust Network Flow and Design Under Demand Uncertainty. *Operations Research* **55**(4):662–673, 2007.

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

[3] A. Ben-Tal, A. Nemirovski. Robust Solutions of Uncertain Linear Programs. *Operations Research Letters* **25**:1–13, 1999.

[4] A. Ben-Tal, A. Nemirovski. Robust Solutions of Linear Programming Problems Contaminated With Uncertain Data. *Mathematical Programming* **88**:411–424, 2000.

[5] A. Ben-Tal, A. Goryashko, E. Guslitzer, A. Nemirovski. Adjustable Robust Solutions Of Uncertain Linear Programs. *Mathematical Programming* **99**:351–376, 2004.

[6] A. Ben-Tal, L. El Ghaoui, A. Nemirovski. Robust Optimization. Princeton Series in Applied Mathematics. Princeton University Press, Princeton, NJ, 2009.

[7] D. Bertsimas, E. Litvinov, X. A. Sun, J. Zhao, T. Zheng. Adaptive Robust Optimization for the Security Constrained Unit Commitment Problem. *IEEE Transactions on Power Delivery* **28**(1):52–63, 2013.

[8] D. Bertsimas, O. Nohadani, K. M. Teo. Nonconvex Robust Optimization for Problems with Constraints. *INFORMS Journal on Computing* **22**(1):44–58, 2010.

[9] D. Bertsimas, M. Sim. Robust Discrete Optimization and Network Flows. *Mathematical Programming* **98**:49–71, 2003.

[10] D. Bertsimas, M. Sim. The Price of Robustness. *Operations Research* **52**:35–53, 2004.

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

[12] V. Gabrel, C. Murat, L. Wu. New Models for the Robust Shortest Path Problem: Complexity, Resolution and Generalization. *Annals of Operations Research* **207**:97–120, 2013.

[13] L. El Ghaoui, H. Lebert. Robust Solutions to Least-Square Problems With Uncertain Data. *SIAM Journal on Matrix Analysis and Applications* **18**(4):1035–1064, 1997.

[14] L. El Ghaoui, F. Oustry, H. Lebert. Robust Solutions to Uncertain Semidefinite Programs. *SIAM Journal on Optimization* **9**:33–52, 1998.

[15] 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.)

[16] 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.

[17] S. Peeta, F. S. Salman, D. Gunnec, K. Viswanath. Pre-Disaster Investment Decisions for Strengthening a Highway Network. *Computers & Operations Research* **37**:1708–1719, 2010.

[18] 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. (An extended journal paper is under review.)

[19] B. Roy. Robustness in Operational Research and Decision Aiding: a Multi-Faceted Issue. *Journal of Operational Research* **200**(3):629–638, 2010.

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

[21] A. Soyster. Convex Programming With Set-Inclusive Constraints and Applications to Inexact Linear Programming. *Operations Research* **21**:1151–1154, 1973.

[22] A. Thiele, T. Terry, M. Epelman. Robust Linear Optimization with Recourse. Technical Report TR09-01, Industrial and Operations Engineering, University of Michigan, Ann Arbor, MI, 2009.

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

[24] B. Zeng, L. Zhao. Solving Two-Stage Robust Optimization Problems Using a Column-and-Constraint Generation Method. *Operations Research Letters* **41**(5):457–461, 2013.

[25] L. Zhao, B. Zeng. An Exact Algorithm for Two-Stage Robust Optimization With Mixed Integer Recourse Problems. University of South Florida, Working Paper, 2011.