European Journal of Operational Research 251 (2016) 739–750

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Decomposition approaches for recoverable robust optimization problems

J. M. van den Akker[a,*], P. C. Bouman[b,1], J. A. Hoogeveen[a], D. D. Tönissen[c,1]

[a] Department of Information and Computing Sciences Utrecht University, Princetonplein 5, 3584 CC Utrecht, The Netherlands
[b] Rotterdam School of Management Erasmus University, Burgemeester Oudlaan 50, 3062 PA Rotterdam, The Netherlands
[c] School of Industrial Engineering Eindhoven University of Technology, Den Dolech 2, 5612 AZ Eindhoven, The Netherlands

ARTICLE INFO

Article history:
Received 16 October 2014
Accepted 1 December 2015
Available online 4 January 2016

Keywords:
Recoverable robustness
Column generation
Branch-and-price
Knapsack
Shortest path

ABSTRACT

Real-life planning problems are often complicated by the occurrence of disturbances, which imply that the original plan cannot be followed anymore and some recovery action must be taken to cope with the disturbance. In such a situation it is worthwhile to arm yourself against possible disturbances by including recourse actions in your planning strategy. Well-known approaches to create plans that take possible, common disturbances into account are robust optimization and stochastic programming. More recently, another approach has been developed that combines the best of these two: recoverable robustness. In this paper, we solve recoverable robust optimization problems by the technique of branch-and-price. We consider two types of decomposition approaches: separate recovery and combined recovery. We will show that with respect to the value of the LP-relaxation combined recovery dominates separate recovery. We investigate our approach for two example problems: the size robust knapsack problem, in which the knapsack size may get reduced, and the demand robust shortest path problem, in which the sink is uncertain and the cost of edges may increase. For each problem, we present elaborate computational experiments. We think that our approach is very promising and can be generalized to many other problems.

1. Introduction

Most optimization algorithms rely on the assumption that all input data are deterministic and known in advance. However, in many practical optimization problems, such as planning in public transportation or health care, data may be subject to changes. To deal with this uncertainty, different approaches have been developed. In case of *robust optimization* (see Ben-Tal, Ghaoui, and Nemirovski (2009); Bertsimas and Sim (2004)) we choose the solution with minimum cost that remains feasible for a given set of disturbances in the parameters. In case of *stochastic programming* (Birge & Louveaux, 1997), we take *first stage decisions* on basis of the current information and, after the true value of the uncertain data has been revealed, we take the *second stage* or *recourse decisions*. The objective here is to minimize the cost of the first stage decisions plus the expected cost of the recourse decisions. The recourse decision variables may be restricted to a polyhedron through the so-called technology matrix (Birge & Louveaux, 1997). Summarized, robust optimization wants the initial solution to be completely immune for a predefined set of disturbances, while stochastic programming includes a lot of options to postpone decisions to a later stage or change decisions in a later stage.

Recently, the notion of *recoverable robustness* (Liebchen, Lübbecke, Möhring, & Stiller, 2009) has been developed, which combines robust optimization and second-stage recovery options. Recoverable robust optimization computes solutions, which for a given set of scenarios can be recovered to a feasible solution according to a set of pre-described, fast, and simple recovery algorithms. The main difference between recoverable robustness and stochastic programming is the way in which recourse actions are limited. The property of recoverable robustness that recourse actions must be achieved by applying a simple algorithm instead of being bounded by a polyhedron makes this approach very suitable for combinatorial problems. As an example, consider the planning of buses and drivers in a large city. We may expect that during rush hours buses may be delayed, and hence may be too late to perform the next trip in their schedule. In case of robust optimization, we can counter this only by making the time between two consecutive trips larger than the maximum delay that we want to

* Correspondig author. Tel.: +31 302533989.
E-mail addresses: J.M.vandenAkker@uu.nl (J.M. van den Akker), PBouman@rsm.nl (P.C. Bouman), J.A.Hoogeveen@uu.nl (J.A. Hoogeveen), D.D.Tonissen@tue.nl (D.D. Tönissen).

[1] The research was performed while this author was at Utrecht University.

http://dx.doi.org/10.1016/j.ejor.2015.12.008
0377-2217/

take into account. This may lead to a very conservative schedule. In case of recoverable robustness, we are allowed to change, if necessary, the bus schedule, but this is limited by the choice of the recovery algorithm. For example, we may schedule a given number of stand-by drivers and buses, which can take over the trip of a delayed driver/bus combination. Especially in the area of railway optimization recoverable robust optimization methods have gained a lot of attention (see e.g. Caprara, Galli, Kroon, Maróti, and Toth (2010); Cicerone et al. (2009)).

In this paper we present a *Branch-and-Price* approach for solving recoverable robust optimization problems. We present two new types of solution approaches: *Separate Recovery Decomposition* (SRD) and *Combined Recovery Decomposition* (CRD). These approaches can be used to model many problems; we will test them on the size robust knapsack problem and on the demand robust shortest path problem.

This paper extends our conference paper (Bouman, van den Akker, Hoogeveen, Demetrescu, & Halldorsson, 2011) by presenting a general definition of our decomposition approaches, a proof that the LP-relaxation of the CRD model dominates the LP-relaxation of the SRD model, and a further study of the solution algorithm for the demand robust shortest path problem. This study includes different column generation strategies and elaborate computational experiments.

To the best of our knowledge, Bouman et al. (2011) and this paper are the first ones applying column generation to recoverable robust optimization. Another decomposition approach, namely Benders decomposition, is used by Cacchiani, Caprara, Galli, Kroon, and Maróti (2008) to assess the Price of Recoverability for recoverable robust rolling stock planning in railways.

The remainder of the paper is organized as follows. In Section 2, we define the concept of recoverable robustness. In Section 3, we present our two different decomposition approaches, and we show the general result that the LP-relaxation of the CRD Model is stronger than the LP-relaxation of the SRD model. In Section 4, we consider the size robust knapsack problem. We investigate the two decomposition approaches in a branch-and-price framework and we present computational experiments in which we compare different solution algorithms. Besides algorithms based on Separate and Combined Recovery Decomposition, we test hill-climbing, dynamic programming, and branch-and-bound. The experiments indicate that Separate Recovery Decomposition performs best. Section 5 is devoted to the demand robust shortest path problem. Since Separate Recovery Decomposition does not seem to be appropriate for this problem, we focus on Combined Recovery Decomposition and consider the settings of the branch-and-price algorithm in more detail. In our experiments we show that the column generation strategy has a significant influence on the computation time. Finally, Section 6 concludes the paper.

2. Recoverable robustness

In this section we formally define the concept of recoverable robustness. We are given an optimization problem

$$P = \min\{f(x)|x \in F\},$$

where $x \in \mathbb{R}^n$ are the decision variables, f is the objective function, and F is the set of feasible solutions.

Disturbances are modeled by a set of discrete scenarios S. We use F_s to denote the set of feasible solutions for scenario $s \in S$, and we denote the decision variables for scenarios s by y^s. The set of algorithms that can be used for recovery are denoted by $\mathcal{A}$, where $A(x, s) \in \mathcal{A}$ determines a feasible solution y^s from a given initial solution x in case of scenario s. In case of planning buses and drivers a scenario corresponds to a set of bus trips that are delayed, and the algorithms in $\mathcal{A}$ decide about the use of standby drivers.

The *recovery robust* optimization problem is now defined as:

$$\mathcal{RRP}_{\mathcal{A}} = \min\{f(x) + g(\{c^s(y^s)|s \in S\})|x \in F,\ A \in \mathcal{A},\ \forall_{s\in S} y^s = A(x, s)\}.$$

Here, $c^s(y^s)$ denotes the cost associated with the recovery variables y^s, and g denotes the function to combine these cost into the objective function. There are many possible choices for g. A few examples are as follows:

1. $g(\{c^s(y^s) = 0$. This models the situation where our only concern is the feasibility of the recovered solutions.
2. $g(\{c^s(y^s) = max_{s\in S} c^s(y^s)$, that is, it models the maximal cost of the recovered solutions y^s. This corresponds to minimizing the worst-case cost. If $c^s(y^s)$ measures the deviation of the solution y^s from x, we minimize the maximum deviation from the initial solution. Note that this deviation may also be limited by the recovery algorithms.
3. $g(\{c^s(y^s) = \sum_{s\in S} p_s c^s(y^s)$, where p_s denotes the probability that scenarios s occurs. This corresponds to minimizing the expected value of the solution after recovery.

In the remainder of the paper we will consider applications that use either a function of sum or max type for $g(\{c^s(y^s)$.

Although earlier papers on recoverable robustness (e.g. Liebchen et al. (2009)) consider the latter type of definition of g as two-stage stochastic programming, we think that the requirement of a pre-described easy recovery algorithm makes this definition fit into the framework of recoverable robustness.

3. Decomposition approaches

We discuss two decomposition approaches for recovery robust optimization problems. In both cases we reformulate the problem such that we have to select one solution for the initial problem and one for each scenario. The difference consists of the way we deal with the scenarios.

3.1. Separate Recovery Decomposition

In *Separate Recovery Decomposition*, we select an initial solution and *separately* we select a solution for each scenario. This means that for each feasible initial solution $k \in F$ we have a decision variable x_k signaling if this solution is selected; similarly for each feasible solution for each scenario $q \in F_s$ we have a decision variable y^s_q. In the formulation we enforce that we select exactly one initial solution and one solution for each scenario. The recovery constraints enforces that for each scenario the initial solution can be transformed into a feasible solution by the given recovery algorithm. We assume that the recovery constraint and the objective function can be expressed *linearly*. We now obtain an Integer Linear Programming formulation which is formulated as follows (for maximization objective):

$$\max \sum_{k\in F} c_k x_k + \sum_{s\in S} \sum_{q\in F_s} c^s_q y^s_q$$

subject to

$$\sum_{k\in F} x_k = 1 \tag{1}$$

$$\sum_{q\in F_s} y^s_q = 1 \text{ for all } s \in S \tag{2}$$

$$A_1 x + A^s_2 y^s \leq b_s \quad \text{for all } s \in S \tag{3}$$

$$x_k \in \{0, 1\} \quad \text{for all } k \in F \tag{4}$$