Light Robustness

Matteo Fischetti and Michele Monaci

DEI, Università di Padova, Italy
{matteo.fischetti, michele.monaci}@unipd.it

Preliminary draft: December 18, 2008

Abstract

We consider optimization problems where the exact value of the input data is not known in advance and can be affected by uncertainty. For these problems, one is typically required to determine a robust solution, i.e., a possibly suboptimal solution whose feasibility and cost is not affected heavily by the change of certain input coefficients. Two main classes of models have been proposed in the literature to handle uncertainty: stochastic programming models (offering great flexibility, but often too large in size to be handled efficiently), and robust optimization models (easier to solve but sometimes leading to very conservative solutions of little practical use). We invertigate a third way to model uncertainty, leading to a modelling framework that we call Light Robustness. Light Robustness couples robust optimization with a simplified two-stage stochastic programming approach, and has a number of important advantages in terms of flexibility and ease to use. In particular, experiments on both random and real word problems show that Light Robustness is often able to produce solutions whose quality is comparable with that obtained through stochastic programming or robust models, though it requires less effort in terms of model formulation and solution time.
Key words: Robust optimization, Stochastic Programming, Integer Linear Program, Multi-dimensional Knapsack, Train Timetabling.

1 Introduction

One of the basic assumption in mathematical programming is that the exact value of the input data is fixed and known in advance. This assumption can however be violated in many situations arising when real world problems are considered. This can be due to the fact that the parameters used in the model are just estimates of real parameters, or more generally to the effect of uncertainty affecting some parameters. When uncertainty is taken into account, an optimal solution with respect to the nominal values of the parameters can be suboptimal (or even infeasible) according to the actual parameters. Hence, small uncertainty in the input data can make the nominal optimal solution completely meaningless from a practical viewpoint.

Within the above setting, a main request when dealing with real world applications is to determine a *robust* solution, i.e., a solution that remains feasible even if some of the input coefficients change. In other words, one is required to determine a solution that is not necessarily optimal for the nominal objective function, but such that its feasibility and cost is not affected heavily by the change of some coefficients—at least for certain meaningful realizations of the input data.

In this paper we mainly focus on a Linear Program (LP) of the form

$$\min \sum_{j \in N} c_j \, x_j \tag{1}$$

$$\sum_{j \in N} a_{ij}\ x_j \leq b_i \qquad i \in M \tag{2}$$

$$x_j \geq 0 \qquad j \in N \tag{3}$$

where some coefficients of constraint matrix A can take a value, say $\widetilde{a}_{ij} \in [a_{ij}, a_{ij} + \hat{a}_{ij}]$, which is different from the nominal one (namely, a_{ij}). We further assume that the actual values of the coefficients are independent. Uncertainty on vectors b and c is not dealt with explicitly, as it can be handled in a straightforward way by just adding suitable artificial variables and/or constraints. We denote by $n = |N|$ and $m = |M|$ the number of variables and constraints in the LP model, respectively. Our approach extends easily to the Mixed-Integer Programming (MIP) case, where certain variables can only assume integer values.

Classical approaches for dealing with uncertainty can be classified as follows:

- Stochastic Programming (SP): find a solution that is optimal by considering possible recourse variables y^ω implementing corrective actions to be performed after a certain scenario $\omega \in \Omega$ has taken place (see, e.g., Birge and Louveaux [8], Ruszczynski and Shapiro [14], and Linderoth, Shapiro, and Wright [12]). This approach typically does not restrict the original solution space, but penalizes the feasible solutions by taking into account the cost of the corrective actions needed to face a certain scenario. The approach is quite powerful but requires the knowledge of the probability (and main features) of the various scenarios, and almost invariably leads to huge LPs that require very large computing times (though clever decomposition techniques have been proposed in the literature to speed-up their resolution).

- Robust Optimization (RO): uncertainty is associated with hard constraints restricting the solution space, i.e., one is required to find a solution that is still feasible for worst-case parameters chosen within a certain uncertainty domain (see, e.g., Ben-Tal and Nemirovski [2] and Bertsimas and Sim [6]). This is a simple way to model uncertainty, but it can lead to overconservative solutions that are quite bad in terms of cost (actually, a feasible solution may not exist at all).

In the present paper we analyze a third way to model uncertainty, leading to a modelling framework that we call *Light Robustness* (LR). Light Robustness can be viewed as a "flexible counterpart" of robust models, obtained through the following modelling steps. We first fix the maximum objective function deterioration that we are willing to accept in our model, by introducing a linear constraint of the type $c^T x \leq \overline{z}$. Then we define a "robustness goal" that we would like to achieve, and model it by using a classical robust optimization framework (e.g., through the Ben-Tal and Nemirovski [2] or Bertsimas and Sim [6] methods). In this way we obtain a robust model with no objective function, that however is likely to be infeasible. To cope with infeasibility, we introduce appropriate slack variables that allow for "local violations" of the robustness requirements, and define an auxiliary objective function aimed at minimizing the slacks. The LR slack variables play a role similar to second-stage recourse variables in SP models, as they penalize the corrective actions needed to restore feasibility. In this view, LR is a modelling framework combining the flexibility of SP (due to the presence of second-stage variables) and the modelling ease of RO. The underlying assumption is that the robust model already captures uncertainty in a sufficiently detailed way, so we do not need a cumbersome second-stage set of variables and constraints—simple slack variables are enough.

LR models are easy to formulate and to solve, and their applicability is potentially larger than robust models. However, it is not clear whether such a simple approach can deliver solutions that are comparable than those obtained through more involved stochastic programming or robust models. The computational experience reported in the present paper confirms the viability of the LR approach—at least in some practically relevant contexts.

The rest of the paper is organized as follows. In Section 2 we briefly review the Bertsimas and Sim approach [6]. Two LR variants are described in Sections 3 and 4, respectively, and are computationally tested on random instances in Section 5. A real-world application is addressed in Section 6. Finally, Section 7 draws some conclusions.

2 The Bertsimas and Sim approach

The first attempt to handle data uncertainty through mathematical models was performed by Soyster [16], who considered uncertain problems of the form

$$\min \{\sum_{j\in N} c_j\ x_j \mid \sum_{j\in N} A_j\ x_j \leq b,\ \forall A_j \in \mathcal{K}_j, j \in N\}$$

where $\mathcal{K}_j$ are convex sets associated with "column-wise" uncertainty. This approach tends to lead to overconservative models, thus to poor solutions in term of optimality. Ben-Tal and Nemirovski [2, 3, 4] defined less conservative models by considering ellipsoidal uncertainties. Moreover, [2] shows that the robust counterpart of an uncertain LP is equivalent to an explicit computationally tractable problem, provided that the uncertainty is itself "tractable". On the contrary, when the problem to be considered is an ILP, these nonlinear (convex) models become computationally hard problems.

Later on, Bertsimas and Sim (BS) [5, 6] considered a different concept of robustness. Their approach is based on the observation that, in real situations, it is unrealistic to assume that all coefficients take, at the same time, their worst-case value. So, it makes sense to define a robust model whose optimal solution remains feasible for every change of (at most) Γ_i coefficients in each row $i \in M$, where Γ_i is an input parameter associated to the expected robustness of the solution. (For sake of simplicity, we will implicitly assume that Γ_i is integer, although this is not required in the approach proposed in [6].) The robust counterpart of (1)–(3) is therefore defined by replacing each row $i \in M$ with the new constraint:

$$\sum_{j\in N} a_{ij}\ x_j + \beta(x, \Gamma_i) \leq b_i \tag{4}$$

where $\beta(x, \Gamma_i)$ is related to the level of protection with respect to uncertainty in the coefficients of row i, and is defined as

$$\beta(x, \Gamma_i) = \max_{S\subseteq N:|S|\leq\Gamma_i} \sum_{j\in S} \hat{a}_{ij}\ x_j \tag{5}$$

So, $\beta(x, \Gamma_i)$ is the maximum increase in the left-hand side of the i-th constraint evaluated for x^*, when at most Γ_i coefficients in row i take their worst-case value.

As already mentioned, parameter Γ_i allows the modeler to control the solution robustness: $\Gamma_i = 0$ means that robustness is not taken into account and the nominal constraint is considered, whereas $\Gamma_i = n$ means that each coefficient in row i can take its worst-case value, and corresponds to the conservative method by Soyster [16].

By using LP duality, the robust model can be formulated through the following LP:

$$\min \sum_{j \in N} c_j \; x_j \tag{6}$$

$$\sum_{j \in N} a_{ij} \; x_j + \Gamma_i \; z_i + \sum_{j \in N} p_{ij} \leq b_i \qquad i \in M \tag{7}$$

$$-\hat{a}_{ij} \; x_j + z_i + p_{ij} \geq 0 \qquad i \in M, j \in N \tag{8}$$

$$z_i \geq 0 \qquad i \in M \tag{9}$$

$$p_{ij} \geq 0 \qquad i \in M, j \in N \tag{10}$$

$$x_j \geq 0 \qquad j \in N \tag{11}$$

The robust formulation above, referred to as `BS` in the sequel, involves a number of variables and constraints that is polynomial in the input size. Note that the approach remains valid when MIPs are considered instead of just LPs, the only requirement being that term $\beta(x, \Gamma_i)$ can be formulated as an LP whose size is polynomial in the input size.

The BS approach provides solutions that are deterministically feasible if the coefficients change under the assumptions above, and are feasible with a high probability if more than Γ_i coefficients in row i are allowed to change.

3 The basic Light Robustness approach

Very often, the optimal robust solution found according to the BS definition can be considerably worse (with respect to the objective function value) than the optimal nominal solution, even if few coefficients are allowed to change in each row. This fact is dramatically emphasized for those problems where most of the coefficients are "structural" and the number of uncertain coefficients in each row is very small (as, e.g., in the train timetabling problem addressed in Section 6).

As already outlined in the introduction, our definition of Light Robustness is a compromise between the robustness of the solution with respect to uncertainty of the matrix coefficients, and the quality of the solution with respect to the objective function. Indeed, in our scheme we look for the most robust solution among those which are "not too far" from optimality for the nominal problem. To be more specific, given a robust optimization model such as the BS one (1)–(3), we define the LR counterpart as:

$$\min \sum_{i \in M} w_i \gamma_i \tag{12}$$

$$\sum_{j \in N} a_{ij} \; x_j + \beta(x, \Gamma_i) - \gamma_i \leq b_i \qquad i \in M \tag{13}$$

$$\sum_{j \in N} a_{ij} \; x_j \leq b_i \qquad i \in M \tag{14}$$

$$\sum_{j \in N} c_j \; x_j \leq (1 + \delta) \; z^* \tag{15}$$

$$x_j \geq 0 \qquad j \in N \tag{16}$$

$$\gamma_i \geq 0 \qquad i \in M \tag{17}$$

Slack variables γ_i act as second-stage recourse variables used to recover from a possible infeasibility, whose weighted sum is minimized by objective function (12). Each variable γ_i defines

the level of robustness of the solution with respect to uncertainty of parameters in row $i \in M$: in particular, γ_i takes a strictly positive value if the corresponding robust constraint i is violated. Constraint (15) imposes a maximum worsening of the objective function value with respect to z^*, defined as the value of the optimal solution of the nominal problem. The role of the input parameter δ in (15) is to balance the quality (optimality) and the feasibility (robustness) of the solution: $\delta = 0$ corresponds to the nominal problem (i.e., robustness is only taken into account to break ties among equivalent optimal solutions), while for $\delta = \infty$ the nominal objective function is not considered at all.

Note that the presence of constraints (13) combined with nominal constraints (14) and objective function (12), is equivalent to fix, for each variable γ_i, an upper bound equal to $\beta(x, \Gamma_i)$.

Weights w_i appearing in the objective function (12) are intended to compensate for possibly different scales for the constraints and can be set, e.g., to the Euclidean norm of each left-hand side coefficient vector. In the sequel we assume implicitly that all the constraints are stated in a comparable unit, hence we set $w_i = 1$ for all i. It is worth noting that the BS approach itself is intrinsically dependent on the specific formulation of LP model at hand, in the sense that it is not invariant with respect to transformations of the constraints that leave the feasible space of the nominal problem unchanged. In other words, the practical applicability of the BS approach (and hence of its LR counterpart) implicitly assumes that the original model is stated in a form that is "suited for robustness"–taking an LP-equivalent model can lead to meaningless results.

By using LP duality as in the BS approach, the LR counterpart of (1)–(3) becomes:

$$\min \sum_{i \in M} \gamma_i \tag{18}$$

$$\sum_{j \in N} a_{ij}\ x_j + \Gamma_i\ z_i + \sum_{j \in N} p_{ij} - \gamma_i \leq b_i \qquad i \in M \tag{19}$$

$$-\hat{a}_{ij}\ x_j + z_i + p_{ij} \geq 0 \qquad i \in M, j \in N \tag{20}$$

$$z_i \geq 0 \qquad i \in M \tag{21}$$

$$p_{ij} \geq 0 \qquad i \in M, j \in N \tag{22}$$

$$\sum_{j \in N} a_{ij}\ x_j \leq b_i \qquad i \in M \tag{23}$$

$$\sum_{j \in N} c_j\ x_j \leq (1 + \delta)\ z^* \tag{24}$$

$$\gamma_i \geq 0 \qquad i \in M \tag{25}$$

$$x_j \geq 0 \qquad j \in N \tag{26}$$

As stated, Light Robustness is a strongly dependent on the BS definition of robustness, hence it can be applied only in those cases in which uncertainty can be described by means of a linear formulation. However, different LR variants can be defined for specific problems. In fact, in our view LR is not a rigid technique, but a modelling framework where robustness is achieved by first enforcing a demanding robustness/optimality goal, and then by allowing for local violations of the constraints (absorbed by the slack variables) to deal with possible infeasibility issues. In the next section we analyze a different (and simpler) LR version whose definition does not rely on the BS model. A problem-specific LR definition will be addressed in Section 6.

4 A heuristic Light Robustness scheme

We next describe a modified LR scheme that is not based on the BS approach, but deals directly with the slack variables associated with the constraints of the nominal problem. The underlying assumption here is that the degree of robustness of a solution is somehow proportional to the slack left in the uncertain rows, to be used to absorb variations of the left-hand side coefficients. Determining the exact value of the slack in each row is of course a difficult task that depends on the whole solution x^* (and not just on the constraint slacks) and has to take into account interactions among the constraints, but it can be approached heuristically as follows.

Let x^* be an optimal solution of nominal problem (1)–(3), and let

$$L_i^* = \sum_{j \in N} (a_{ij} + \hat{a}_{ij})\ x_j^* - b_i$$

denote the maximum violation of constraint i with respect to solution x^*. We define by

$$U = \{i \in M : L_i^* > 0\}$$

the set of constraints that may be affected by uncertainty with respect to x^*. In other words, U contains the rows we want to take care of in terms of uncertainty, i.e., those rows for which enough slack should be given. We can assume without loss of generality $|U| \geq 1$, since otherwise the optimal solution x^* of the nominal problem would be feasible (and hence optimal) in any realization of the data.

We first solve the following LP

$$\max\ \sigma \tag{27}$$

$$\sum_{j \in N} a_{ij}\ x_j + s_i = b_i \qquad i \in M \tag{28}$$

$$\sigma \leq \frac{s_i}{L_i^*} \qquad I \in U \tag{29}$$

$$\sum_{j \in N} c_j\ x_j \leq (1 + \delta)\ z^* \tag{30}$$

$$x_j \geq 0 \qquad j \in N \tag{31}$$

$$s_i \geq 0 \qquad i \in M \tag{32}$$

which maximizes the minimum slack that can be assigned to any uncertain row. In order to take into account uncertainty on each row separately, the slack variable s_i in the i-th uncertain constraint (29) is heuristically normalized by dividing it by L_i^* ($i \in U$).

The LP above typically has several equivalent optimal solutions, due to its max-min nature. Indeed, objective function (27) only considers the row corresponding to the minimum normalized slack, hence there is no incentive in giving a large slack to the remaining rows—whereas this is very important for improving robustness. Thus, a second LP is solved in order to balance the slack among uncertain rows, while keeping the total amount of slack large enough. Given an optimal solution (x^*, s^*, σ^*) of model (27)–(32), we define the average and minimum value for the normalized slack as

$$s_{\text{avg}} = \frac{\sum_{i \in U} s_i^*/L_i^*}{|U|}$$

$$s_{\min} = \min\{s_i^*/L_i^* : i \in U\} \quad (= \sigma^*)$$

and solve the following LP

$$\min \sum_{i \in U} t_i \tag{33}$$

$$\sum_{j \in N} a_{ij}\ x_j + s_i = b_i \qquad i \in M \tag{34}$$

$$\sum_{j \in N} c_j\ x_j \leq (1+\delta)\ z^* \tag{35}$$

$$\frac{s_i}{L_i^*} + t_i \geq s_{\text{avg}} \qquad i \in U \tag{36}$$

$$x_j \geq 0 \qquad j \in N \tag{37}$$

$$s_i/L_i^* \geq s_{\min} \qquad i \in U \tag{38}$$

$$s_i \geq 0,\ t_i \geq 0 \qquad i \in U \tag{39}$$

In this model, for each uncertain constraint $i \in U$ we introduce an auxiliary variable t_i assuming a positive value if the associated normalized slack is smaller than the average. Objective function (33) penalizes the sum of these variables, so as balance the normalized slack among all constraints.

Although this method requires the solution of two LPs (actually, three if the nominal problem is also considered), our computational experiments reported in Section 5 show that the corresponding extra computing time is quite small in practice, due to the use of fast parametric reoptimization techniques.

5 Computational experiments on random data

In order to test the two LR approaches described in the previous sections, we performed computational experiments on knapsack and portfolio instances similar to those considered by Bertsimas and Sim in [6], and on variants of these instances.

Our computational measure of robustness for a given feasible solution $\tilde{x}$ of the nominal model (1)–(3) is provided by an external tool (called the external *validation tool* in the sequel) that generates 10,000 random *scenarios*, i.e., realizations of the input data according to a uniform distribution. The validation tools receives solution $\tilde{x}$ on input, and returns the probability of infeasibility of $\tilde{x}$ (of course, violation of a single constraint in model (1)–(3) is enough to declare $\tilde{x}$ infeasible for a certain scenario).

Since our LR schemes require an optimality threshold on input, namely $\overline{z} := (1+\delta)z^*$, a fair comparison with respect `BS` is not immediate. In our experiments we implemented the following scheme.

We first solved the BS model (6)–(11) so as to test `BS` approach alone. Since a main difficulty in using the BS approach is the definition of coefficients Γ_i to be used in (5), we heuristically fixed $\Gamma_i = \Gamma$ for all $i \in M$, and solved the corresponding BS model for increasing values of Γ, until a value was found, say $\Gamma^{\max}$, for which the corresponding solution is always feasible according to our external validation tool. We will refer to this solution as the *always-feasible* solution. The gap between the value of the optimal solution of the nominal problem and the value of the always-feasible solution is then used for defining threshold values $\overline{z}$. More specifically, we considered 9 threshold values obtained by allowing for a worsening

(with respect to the optimal nominal solution) of 1%, 5%, 10%, 25%, 50%, 60%, 70%, 80%, and 90% of such a gap.

Once the threshold value $\bar{z}$ is fixed, we ran all models and evaluated the robustness of the corresponding solution $\tilde{x}$ through our external validation tool. The basic LR model (12)–(17) was solved by setting all Γ_i's to a constant (quite large) value, so as to require a high level of protection against uncertainty.

The heuristic LR model (27)–(32) and (33)–(39) does not require any other parameter and was solved as described. As to the BS model (6)–(11), we embedded it into a binary search procedure that finds the maximum real value of $\Gamma \in [0, n]$ such that the optimal solution value for model (6)–(11) does not exceed $\bar{z}$. In order to limit computing time, binary search is halted as soon as the difference between the maximum and minimum Γ values is smaller than 0.1. The procedure is further speeded-up, at each binary-search iteration, by stopping the solution of model (6)–(11) as soon as a solution with value not greater than $\bar{z}$ is found. In a similar way, each iteration is halted whenever a proof is given that no such a solution exists. The value of Γ produced by the binary search procedure, say Γ^*, is therefore an approximation of the best possible value for model (6)–(11) when a solution having cost at most $\bar{z}$ is required. In the following we refer to this method as `BinBS`.

A fair comparison of `BinBS` and LR computing times is not immediate, since our experiment design is biased somehow in favor of the LR approach. Indeed, one could symmetrically fix the Γ value and apply binary search to LR to find the corresponding threshold value $\bar{z}$. According to our experience, in practical cases working with an optimality threshold is more natural than providing the Γ coefficient(s). In any case, the reported computing times for `BinBS` and LR have to compared with some caution.

The following tables report, separately for each problem, the results of each method for each threshold value $\bar{z}$, showing the probability that the solution found is infeasible along with the corresponding computing time. In addition, for method `BinBS` we report the value Γ^* found by the binary search procedure, and the average time required to perform a single binary-search iteration. All experiments have been performed on a AMD Athlon 64 Processor 3500+ using `ILOG-Cplex` 10.1 as LP/ILP solver.

5.1 Single Knapsack Problem

One of the most famous problems in Combinatorial Optimization is the *Knapsack Problem* (KP) in which one is given a set $N = \{1, \ldots, n\}$ of *items* and a knapsack of capacity W. Each item $j \in N$ has associated a positive *profit* p_j and a positive weight w_j, and the aim is to select a set of items in such a way that (i) the sum of the weights of the selected items does not exceed c, and (ii) the sum of the profits of the selected items is maximized. By introducing, for each item $j \in N$ a binary variable x_j taking value 1 iff item j is selected, the problem can be formulated as follows:

$$\max \sum_{j \in N} p_j \, x_j \tag{40}$$

$$\sum_{j \in N} w_j \, x_j \leq W \tag{41}$$

$$x_j \in \{0, 1\} \qquad j \in N \tag{42}$$

This problem is NP-hard, although pseudo-polynomial solution algorithms exist. For extensive studies on approaches to the knapsack problem, as well as to its variants or extensions,

the reader is referred to the books by Martello and Toth [13] and by Kellerer, Pferschy and Pisinger [11].

Following Berstimas and Sim [6], we tested our robust approach by generating a KP instance with $|N| = 200$, integer profits p_j randomly generated in $[16, 77]$, integer weights w_j randomly generated in $[20, 29]$, and $W = 4000$. Uncertainty was modelled by allowing each weight to differ by at most 10% with respect to its nominal value.

Table 1 reports the value of the optimal solution of model (6)–(11) using different values for Γ. In addition, the table gives the percentage worsening in the solution value, the required computing time, and the probability that the provided solution is infeasible. The results of Table 1 experimentally confirm the theoretical results provided in [6] for what concerns both the worsening of the solution value and the probability of infeasibility.

Table 1: Results on BS model (6)–(11) on a random knapsack problem

Γ	z	% wors.	% Infeas	Time
0	8801	0.0000	47.57	0.00
1	8800	0.0114	43.18	0.01
5	8786	0.1704	19.96	0.12
10	8773	0.3181	5.08	0.02
15	8754	0.5340	0.57	0.06
20	8740	0.6931	0.02	0.13
22	8732	0.7840	0.00	0.14

The optimal solution of the nominal (maximization) problem has value 8801, while the always feasible solution, provided by model (6)–(11) with $\Gamma = 22$, has value 8732. The derived threshold values and the corresponding results for each robust method are reported in Table 2. The first table row has the following meaning: fixing a lower bound of $\overline{z} = 8800$ on the solution profit, one can find a solution with infeasibility probability of 43.18% (43.10% for HLR); this solution if found in 0.06 CPU seconds by `BinBS` (each binary-search iteration taking 0.01 seconds on average), and corresponds to the choice $\Gamma^* = 0.98$, whereas LR and HLR require 0.01 and 0.02 seconds, respectively.

According to the table, for each threshold value $\tilde{z}$ the three methods deliver solutions with negligible differences in terms of robustness. This is not surprising, due the very simple structure of the KP problem. As expected, the LR approaches are faster than `BinBS` as no binary search is required.

5.2 Multi-dimensional Knapsack Problem

In order to validate our methods on a problem involving several constraints, we considered a Multi-dimensional Knapsack instance with $|M| = 10$ constraints. All coefficients and the associated deviations are generated as for the KP instance of Section 5.1, i.e., in the same way used in [6]. For this instance, the optimal solution of the nominal problem has value 8316, while the always feasible solution is provided by model (6)–(11) with $\Gamma = 24$ and has value 8238.

Computational results in Table 3 show that the solutions provided by `BinBS` and LR are quite similar in terms of robustness. On the other hand, computing times for LR are

Table 2: Results on a random knapsack problem

	BinBS				$LR_{(\Gamma = 20)}$		HLR	
$\bar{z}$	Γ^*	% Infeas	Time	Avg.Time	% Infeas	Time	% Infeas	Time
8800	0.98	43.18	0.06	0.01	43.18	0.01	43.10	0.02
8797	1.95	36.82	0.07	0.01	36.82	0.01	36.54	0.01
8794	2.93	30.61	0.06	0.01	30.61	0.01	30.61	0.02
8783	5.47	18.60	0.19	0.02	18.60	0.01	18.60	0.19
8766	11.33	3.13	0.16	0.02	3.25	0.02	3.20	0.02
8759	13.28	1.48	0.25	0.03	1.48	0.02	1.48	0.15
8752	16.60	0.31	0.13	0.01	0.31	0.02	0.31	0.01
8745	18.75	0.12	0.09	0.01	0.12	0.02	0.12	0.03
8738	20.90	0.01	0.20	0.02	0.02	0.18	0.01	0.02

considerably smaller (often by two orders of magnitude) than those required by `BinBS`. E.g., for threshold $\bar{z} = 8296$ `BinBS` required 27.37 seconds, whereas LR took just 0.24 seconds. At first glance, this is quite surprising since the average BS time for a single binary-search iteration is 2.74 seconds, i.e., 10 times larger than LR. A similar situation arises for $\bar{z} = 8277$ and 8269. The explanation is that, during binary search, the BS model has to deal with weird (noninteger) values for the Γ_i coefficients appearing in (7), which makes these constraints numerically nasty and the solution of the overall problem much harder. The LR models, instead, do not suffer from this problem, due the greater flexibility granted by the presence of slack variables.

As to HLR, it provides even (slightly) better results than `BinBS` and LR in terms of robustness, and requires much shorter computing times. E.g., for $\bar{z} = 8296$ it provides a solution with about 10% less probability of infeasibility than `BinBS`, and requires about 3 orders of magnitude less computing time.

Table 3: Results on the multi-dimensional knapsack instance

	BinBS				$LR_{(\Gamma = 20)}$		HLR	
$\bar{z}$	Γ^*	% Infeas	Time	Avg.Time	% Infeas	Time	% Infeas	Time
8315	0.59	86.97	1.88	0.19	88.42	0.21	86.97	0.10
8312	2.34	81.01	0.84	0.08	82.25	0.16	81.01	0.11
8308	3.32	78.20	0.88	0.09	71.53	0.11	71.58	0.08
8296	5.47	64.70	27.37	2.74	61.42	0.24	53.64	0.09
8277	12.70	5.62	4.45	0.45	6.74	0.14	5.62	0.08
8269	14.45	4.02	37.26	3.73	4.77	0.17	4.02	0.08
8261	18.75	0.30	2.38	0.24	0.30	0.30	0.30	0.11
8253	20.90	0.08	2.75	0.28	0.09	0.28	0.08	0.06
8245	22.66	0.05	12.08	1.21	0.15	2.05	0.05	0.09

In order to analyze the performance of various robust approaches on more demanding settings, we performed additional experiments on the multi-dimensional knapsack instance

described above.

We first considered the situation arising when coefficients of the first constraint have more uncertainty than those of the other constraints. In particular, the original instance is considered, but each coefficient in the first constraint is allowed to differ by at most 50% with respect to its nominal value, while uncertainty for coefficients in the remaining rows is at most 10%, as in the previous experiment. The corresponding computational results are given in Table 4 and confirm the previous findings: all three methods provided solutions with similar robustness (the only exception being $\bar{z} = 8313$ and 8286, where LR produced significantly more robust solutions), and HLR is faster than LR, which is in turn much faster than `BinBS`.

Table 4: Results on the multi-dimensional knapsack instance with larger uncertainty of the coefficients in the first row

	`BinBS`				$LR_{(\Gamma = 20)}$		HLR	
$\bar{z}$	Γ^*	% Infeas	Time	Avg.Time	% Infeas	Time	% Infeas	Time
8313	1.56	89.68	1.24	0.12	86.44	0.27	89.68	0.08
8301	4.30	73.34	6.32	0.63	73.35	0.10	74.51	0.08
8286	7.03	41.21	0.72	0.07	28.20	0.29	40.85	0.05
8241	10.16	8.99	0.96	0.10	7.30	1.35	8.99	0.04
8167	14.84	0.77	0.69	0.07	0.84	0.08	0.84	0.04
8137	16.60	0.29	1.23	0.12	0.29	0.06	0.29	0.04
8108	18.36	0.14	3.05	0.31	0.12	0.26	0.13	0.06
8078	20.31	0.09	0.70	0.07	0.08	0.74	0.08	0.07
8048	22.07	0.02	0.55	0.06	0.04	0.06	0.01	0.06

Finally, we considered two cases where the number of uncertain coefficients in each row is not a constant. In particular, let $J_i \subseteq N$ denote the index set of the uncertain coefficients in each row i ($i = 1, \ldots, 10$). We considered case $|J_i| = 10*(11-i)$, i.e., 100 uncertain coefficients arise in the first row, 90 in the second, and 10 in the last row. The set of uncertain coefficients in each row is generated according to a uniform distribution, and each uncertain coefficient can differ by at most 10% with respect to the nominal value.

Note that, in the new setting, defining a same value for all Γ_i's does not make sense for `BS`. Thus, according to [6], we considered a value θ representing the normalized number of uncertain coefficients, and defined $\Gamma_i = \theta\ |J_i|$ for each row i. Accordingly, binary search was executed with an accuracy equal to 10^{-3} on the value of θ, while LR was executed with $\theta = 1$.

In the instance addressed in Table 5, there is no correlation among uncertain coefficients in different rows. On the contrary, in the instance of Table 6 uncertainty was generated so that a coefficient can be uncertain in row i only if the coefficient in the same column is uncertain in row $i - 1$, thus inducing a certain degree of correlation among uncertain coefficients in different rows.

Results in Tables 5 and 6 confirm once again that the LR approaches, in spite of their simplicity, are able to produce solutions that turn out to be equally (or even more) robust than those produced by `BinBS`, in much shorter computing times. In particular, HLR qualifies as the method of choice for producing robust solutions for multi-dimensional knapsack problems.

Table 5: Results on the multi-dimensional knapsack instance when the number of uncertain coefficients in each row is not a constant

	BinBS				$LR_{(\theta = 1.0)}$		HLR	
$\bar{z}$	θ^*	% Infeas	Time	Avg.Time	% Infeas	Time	% Infeas	Time
8315	0.015	77.89	0.58	0.06	77.80	0.09	77.89	0.08
8313	0.024	65.70	0.57	0.06	62.27	0.03	65.70	0.10
8311	0.043	40.60	0.39	0.04	60.39	0.05	40.60	0.04
8304	0.073	22.40	0.31	0.03	32.08	0.05	23.39	0.06
8292	0.108	20.64	6.82	0.68	13.28	1.24	27.12	0.11
8287	0.148	0.39	0.52	0.05	6.58	4.13	0.39	0.04
8283	0.167	0.09	0.58	0.06	2.47	1.00	0.20	0.05
8278	0.186	0.04	0.53	0.05	0.31	0.34	0.04	0.08
8273	0.204	0.00	0.75	0.08	0.14	0.02	0.02	0.06

Table 6: Results on the multi-dimensional knapsack instance when the number of uncertain coefficients in each row is not a constant, and high correlation among uncertainty in different rows exists

	BinBS				$LR_{(\theta = 1.0)}$		HLR	
$\bar{z}$	θ^*	% Infeas	Time	Avg.Time	% Infeas	Time	% Infeas	Time
8315	0.015	78.27	0.52	0.05	78.27	0.08	78.27	0.09
8313	0.024	64.52	0.58	0.06	59.38	0.02	64.52	0.10
8311	0.043	40.95	0.41	0.04	77.45	0.05	40.95	0.04
8304	0.073	20.60	0.33	0.03	59.56	0.05	22.86	0.06
8292	0.109	19.93	5.74	0.57	12.93	1.70	22.00	0.14
8287	0.149	0.45	0.76	0.08	2.16	1.31	0.45	0.04
8283	0.167	0.15	0.92	0.09	0.78	0.12	0.25	0.06
8278	0.186	0.09	0.64	0.06	0.13	0.10	0.10	0.07
8273	0.204	0.01	0.94	0.09	0.03	0.02	0.06	0.06

5.3 A simple portfolio problem

The two previous subsections showed the effectiveness of the LR approach in the context of knapsack problems. In fact, these problems are very well suited for the LR models, as the slack variables in the model correspond to empty space in the the knapsacks, so encouraging large slacks has a clear impact on the robustness of the final solution. There are however other contexts where the correlation between slacks and robustness is more subtle, hence the LR approach is less likely to be effective. However, it is important to stress that the LR performance depends heavily on the *model* used (rather than on the problem itself), in the sense that different models can lead to drastically different results in terms of robustness—a property shared by other approaches to robustness, including the BS one.

To illustrate this point, we consider a simplified portfolio problem taken again from [6]. Given a set $N = \{1, \ldots, n\}$ of stocks, the i-th having an estimated return p_i, a simplified portfolio problem requires to select the fraction x_i of wealth invested in stock i so as to maximize the portfolio value equal to $\sum_{i \in N} p_i x_i$. In real applications, the return value for stock i ($i \in N$) is subject to uncertainty, i.e., it can differ by at most σ_i from the nominal value.

A linear formulation for the portfolio problem can be obtained as follows:

$$\max \; z \tag{43}$$

$$z \leq \sum_{i \in N} p_i \; x_i \tag{44}$$

$$\sum_{i \in N} x_i = 1 \tag{45}$$

$$x_i \geq 0 \qquad i \in N \tag{46}$$

We generated a portfolio instance as done in [6], using $n = 150$, and generating p_i and σ_i values as follows:

$$p_i = 1.15 + i\frac{0.05}{150} \qquad \text{and} \qquad \sigma_i = \frac{0.05}{450}\sqrt{2n(n+1)i}$$

so that stocks with higher return are also more risky. Note that the only uncertain constraint in the above model is (44).

Table 7 gives the results on this problem, providing the same information as in the previous tables; computing times are negligible for all approaches and are omitted. Column HLR* refers to HLR applied to a different model, to be described later. Note that `BinBS` fails in finding a feasible solution for the first two threshold values, for which the value Γ^* is so small to be below our binary search precision (of course, one could modify the binary search procedure so as to deal with case $\Gamma^* = 0$).

According to the table, LR provides results (in terms of robustness) somehow worse than `BinBS`, which suggests that the LR slack variables are not effective in this context. This is confirmed by the very bad performance of HLR, that returns solutions whose robustness seems to be independent of the threshold. A closer look to the portfolio model clarifies the situation. Given a threshold value $\bar{z}$, an optimal solution of model (27)–(32) is given by $x_i = x_i^*, z = \bar{z}, s = z^* - \bar{z}$, where x^* denotes the optimal solution of the nominal problem and z^* its value. Hence, HLR will always keep the same solution x^* and use the slack variable s to absorb the allowed worsening of the objective function.

The above considerations would suggest that HLR is not applicable to the the portfolio application. However this is not true, in that one can derive an alternative model where the slack variables do play a role in terms of robustness (see also Bienstock [7] for a recent paper based on a similar idea). Indeed, consider the alternative LP model

$$\max \; z \tag{47}$$

$$z = \sum_{i \in N} z_i \tag{48}$$

$$z_i \leq p_i \; x_i \qquad i \in N \tag{49}$$

$$\sum_{i \in N} x_i = 1 \tag{50}$$

$$x_i \geq 0 \qquad i \in N \tag{51}$$

By applying HLR to the above model one gets the results in column HLR* of Table 7, showing that our heuristic LR approach produces much better solutions than those found by `BinBS` with the original formulation.

Table 7: Results on a portfolio instance

	`BinBS`		$LR_{(\Gamma = 15)}$	HLR	HLR*
$\bar{z}$	Γ^*	% Infeas	% Infeas	% Infeas	% Infeas
1.1994	–	–	50.09	50.09	47.96
1.1971	–	–	49.97	49.60	42.88
1.1942	0.15	46.92	48.94	49.12	36.97
1.1855	1.03	37.79	49.10	47.63	19.03
1.1710	4.83	18.38	47.06	45.24	0.00
1.1652	7.18	11.23	22.72	44.30	0.00
1.1595	10.25	5.04	7.61	43.11	0.00
1.1537	14.06	2.02	1.98	42.19	0.00
1.1479	19.34	0.26	0.46	41.20	0.00

6 A real-world application: the train timetabling problem

In order to illustrate a possible application of the LR idea in a real world context, in this section we review the approach recently proposed by Fischetti, Salvagnin and Zanette [10] for finding robust railway timetables. We only give a brief sketch of the method and of the corresponding computational results; the reader is addressed to [10] for details.

The Train Timetabling Problem (TTP) consists in finding an effective train schedule on a given railway network. The schedule needs to satisfy some operational constraints given by capacities of the network and security measures. Moreover, one is required to exploit efficiently the resources of the railway infrastructure. In practice, however, the maximization of some objective function is not enough: the solution is also required to be robust against delays/disturbances along the network. Very often, the robustness of optimal solutions of the original problem turns out to be not enough for their practical applicability, whereas easy-to-compute robust solutions tend to be too conservative and thus unnecessarily inefficient. As

a result, practitioners call for a fast yet accurate method to find the most robust timetable whose efficiency is only slightly smaller than the theoretical optimal one.

Fischetti, Salvagnin and Zanette (FSZ) [10] proposed and analyzed computationally alternative methods to find robust and efficient solutions to the TTP, in its aperiodic (non cyclic) version described in [9]. Their method is based on an event-based MIP model for the nominal TTP, akin to the formulation proposed in [15] for the periodic (cyclic) case, and will be outlined briefly in the sequel.

6.1 Measuring timetable robustness

FSZ implemented an external simulation-based validation module that is independent from the optimization model itself, so that it can be of general applicability and allows one to compare solutions coming from different methods. The module is required to simulate the reaction of the railways system to the occurrence of delays, by introducing small adjustments to the planned timetable (received as an input parameter). The underlying assumption here is that timetabling robustness is not concerned with major disruptions (which are to be handled by the real time control system and require human intervention) but is a way to control delay propagation, i.e., a robust timetable has to favor delay compensation without heavy human action. As a consequence, at validation time no train cancellation is allowed, and event precedences are fixed with respect to the planned timetable.

The validation model analyzes a single delay scenario at a time. As all event precedences are fixed according to the input solution to be evaluated, the nominal TTP constraints simplify to linear inequalities of the form:

$$t_i - t_j \geq d_{i,j} \tag{52}$$

where t_i and t_j are time variables associated with significant events (typically, arrival and departure of a train from a certain station), and $d_{i,j}$ is a minimum trip time or minimum rest/headway time. Let $\mathcal{P}$ denote the set of ordered pairs (i, j) for which a constraint of type (52) can be written, and E denote the set of events.

The problem of adjusting the given timetable t under a certain delay scenario δ^ω can thus be rephrased as the following simple LP model with decision variables t^ω describing the best possible adjustment of the published timetable t for the considered delay scenario:

$$\min \sum_{j \in E} \left(t_j^\omega - t_j \right) \tag{53}$$

$$t_i^\omega - t_j^\omega \geq d_{i,j} + \delta_{i,j}^\omega \qquad (i,j) \in \mathcal{P} \tag{54}$$

$$t_i^\omega \geq t_i \qquad i \in E \tag{55}$$

Constraints (54) correspond to linear inequalities just explained, in which the nominal right-hand-side value $\delta_{i,j}$ is updated by adding the (possibly zero) extra-time $\delta_{i,j}^\omega$ from the current scenario ω.

Constraints (55) are non-anticipatory constraints stating the obvious condition that one is not allowed to anticipate any event with respect to its published value in the timetable.

The objective function is to minimize the "cumulative delay" on the whole network.

Given a feasible solution t, the validation tool keeps testing it against a large set of scenarios, one at a time, gathering statistical information on the value of the objective function and yielding a concise figure (the average cumulative delay) of the robustness of the timetable.

6.2 Finding robust solutions

Different techniques to enforce robustness were implemented by FSZ.

A fat stochastic model The first attempt to solve the robust version of the TTP was to use a standard scenario-based SP formulation whose structure can informally be sketched as follows:

$$\min \ \frac{1}{|\Omega|} \sum_{j \in E, \omega \in \Omega} \left(t_j^\omega - t_j \right) \tag{56}$$

$$\sum_{h \in T} \rho_h \geq (1 - \delta) z^* \tag{57}$$

$$t_i^\omega - t_j^\omega \geq d_{i,j} + \delta_{i,j}^\omega \qquad (i,j) \in \mathcal{P}, \omega \in \Omega \tag{58}$$

$$t_i^\omega \geq t_i \qquad i \in E, \omega \in \Omega \tag{59}$$

$$t_i - t_j \geq d_{i,j} \qquad (i,j) \in \mathcal{P} \tag{60}$$

$$l_i \leq t_i \leq u_i \qquad i \in E \tag{61}$$

The model is similar to that used in the validation tool, but takes into account several scenarios $\omega \in \Omega$ at the same time. Moreover, the nominal timetable values t_j are now viewed as decision variables to be optimized—their optimal value will define the final timetable to be published. The model keeps a copy of the original (linear) model with a modified right hand side for each scenario, along with the original model; the original variables and the correspondent second-stage copies in each scenario are linked through non-anticipatory constraints.

The objective is to minimize the cumulative delay over all events and scenarios. The original objective function (namely, the total train profit $\sum_{h \in T} \rho_h$, to be maximized, where T is the set of trains) is taken into account through constraint (57), where $\delta \geq 0$ is the tradeoff parameter and z^* is the objective value of the reference solution. As to the single-train profit variables ρ_h that appear in (57), they are linked to the timetable variables through appropriate constraints (not shown in the model); see [10] for a complete model.

For realistic instances and number of scenarios this model becomes very time consuming (if not impossible) to solve–hence we called it "fat". On the other hand, also in view of its similarity with the validation model, the fat model plays the role of a kind of "perfect model" in terms of achieved robustness, hence it will be used for benchmark purposes.

A slim stochastic model Given the computing time required by the full stochastic model, the following alternative SP model was designed, which is simpler yet meaningful for the TTP problem.

$$\min \sum_{(i,j) \in \mathcal{P}, \omega \in \Omega} w_{i,j}^\omega s_{i,j}^\omega \tag{62}$$

$$\sum_{h \in T} \rho_h \geq (1 - \delta) z^* \tag{63}$$

$$t_i - t_j + s_{i,j}^\omega \geq d_{i,j} + \delta_{i,j}^\omega \qquad (i,j) \in \mathcal{P}, \omega \in \Omega \tag{64}$$

$$s_{i,j}^\omega \geq 0 \qquad (i,j) \in \mathcal{P}, \omega \in \Omega \tag{65}$$

$$t_i - t_j \geq d_{i,j} \qquad (i,j) \in \mathcal{P} \tag{66}$$

$$l_i \leq t_i \leq u_i \qquad i \in E \tag{67}$$

In this model there is just one copy of the original variables, plus the recourse variables $s^\omega_{i,j}$ that account for the unabsorbed extra times $\delta^\omega_{i,j}$. It is worth noting that the above "slim" model is inherently smaller than the fat one. Moreover, one can drop all the constraints of type (64) with $\delta^\omega_{i,j} = 0$, a situation that occurs very frequently in practice since most extra-times in a given scenario are zero.

As to the objective function, it involves a weighted sum of the the recourse variables. Finding meaningful values for the weights $w^\omega_{i,j}$ turns out to be very important. Indeed, we will shown in the sequel how to define the weights so as to produce solutions whose robustness is comparable with that obtainable by solving the (much more time consuming) fat model.

Light Robustness A LR approach was used in [10] to generate robust timetables. The resulting method is related to the *adjustable robustness* paradigm used by Ben-Tal, El Ghaoui, and Nemirovski [1] in the context of project management.

In our TTP model, a typical constraint reads

$$t_i - t_j \geq d_{i,j}$$

where $d_{i,j}$ is the coefficient affected by uncertainty, and its LR counterpart is simply defined as

$$t_i - t_j + \gamma_{i,j} \geq d_{i,j} + \Delta_{i,j} \quad \gamma_{i,j} \geq 0$$

where $\Delta_{i,j}$ is a parameter fixing the desired (overconservative) protection level, and $\gamma_{i,j}$ are the slack variables whose weighted sum has to be minimized.

6.3 Computational Results

Computational tests were performed on four single-line medium-size TTP instances provided by the Italian railway company, Trenitalia. An almost-optimal heuristic solutions for each of these instances was computed through the algorithm described in [9], and used as a reference solution to freeze the event precedences and to select the trains to schedule.

The overall framework was implemented in C++ and tested on a AMD Athlon64 X2 4200+ computer with 4GB of RAM running Linux 2.6. The MIP solver used was `ILOG-Cplex` 10.1.

As far as scenarios are concerned, for each train on the line and for each scenario FSZ generated the corresponding extra-time, 5% on average, drawn from an exponential distribution, and distributed it proportionally to its train segments.

For each reference solution, a set of experiments was performed to compare the different methods for different values of the tradeoff parameter δ giving the allowed percentage of worsening of the nominal objective function, namely 1%, 5%, 10%, 20% and 40%. In particular, we compared the following alternative methods:

- *fat*: fat stochastic model (50 scenarios only)
- *slim1*: slim stochastic model with uniform objective function–all weights equal (400 scenarios)

- *slim2*: slim stochastic model with enhanced objective function (400 scenarios), where events arising earlier in each train sequence receive a larger weight in the objective function. More specifically, if the i-th event of train h is followed by k events, its weight in the objective is set to $k + 1$. The idea beyond this weighing policy is that early extra-times in a train sequence are likely to propagate to the next ones, so they are more important.

- *LR*: light robustness model, with objective function as in *slim2* and protection level parameters set to $\Delta = -\mu \ln \frac{1}{2}$, where μ is the mean of the exponential distribution. This is the protection level required to absorb a delay of such distribution with probability $\frac{1}{2}$.

The results are reported in Table 8, where for each tradeoff parameter δ and railway line we give, for each method, the level of robustness of the corresponding solution (measured by the validation tool in terms of cumulative delay, in minutes—the smaller, the better) and the required computing time (in CPU seconds). According to the table, *fat, slim2 and LR* models produce solutions of comparable robustness (at least when the tradeoff parameter δ is not unrealistically large), whereas *slim1* is clearly the worst method. As to computing times, the *fat* model is one order of magnitude slower than *slim1* and *slim2*, although it uses only 50 scenarios instead of 400. *LR* is much faster than any other method, more than two orders of magnitude w.r.t the fast stochastic models, and qualifies as the method of choice to attack even larger instances.

7 Conclusions

In this paper we addressed optimization problems in which input data is affected by uncertainty. Although many robust and/or stochastic programming models have been proposed in the literature to handle such a situation, their applicability is sometimes far from satisfactory. We proposed to deal with uncertainty by means of a new modelling framework that we called Light Robustness (LR). Light Robustness couples robust optimization with a simplified two-stage stochastic programming approach based on the introduction of suitable slack variables, and has a number of important advantages in terms of flexibility and ease to use. Experiments on both random and real word problems show that LR is often able to produce solutions whose quality is comparable with than obtained through stochastic programming or robust models, though it requires much less effort in terms of model formulation and solution time—even if this latter aspect appears to be less important in many applications.

According to our computational results, the LR approach is mostly successful when the slack variables have a direct impact on robustness. For the cases where the correlation between slacks and robustness is more subtle, the LR approach is less likely to be effective, though an appropriate reformulation of the model can be highly beneficial. We have illustrated this behavior on a simplified portfolio problem, where a simple LR scheme applied to a suitable reformulation of the initial model produces extremely good results in term of robustness.

In our view, the LR framework is not a rigid technique, but a modelling framework where robustness is achieved by first enforcing a demanding robustness/optimality goal, and then by allowing for local violations of the constraints (absorbed by the slack variables) to deal with possible infeasibility issues. As such, effective LR variants can be designed for specific problems, such as the train timetabling problem recently addressed in [10].

Table 8: Comparison of different methods w.r.t. computing time and robustness (cumulative delay in minutes), for different lines and tradeoff δ

δ		Fat		Slim1		Slim2		LR	
	Line	Delay	Time (s)	Delay	Time (s)	Delay	Time (s)	Delay	Time (s)
0%	BZVR	16149	9667	16316	532	16294	994	16286	2.27
0%	BrBO	12156	384	12238	128	12214	173	12216	0.49
0%	MUVR	18182	377	18879	88	18240	117	18707	0.43
0%	PDBO	3141	257	3144	52	3139	63	3137	0.25
	Tot:	49628	10685	50577	800	49887	1347	50346	3.44
1%	BZVR	14399	10265	15325	549	14787	1087	14662	2.13
1%	BrBO	11423	351	11646	134	11472	156	11499	0.48
1%	MUVR	17808	391	18721	96	17903	120	18386	0.48
1%	PDBO	2907	250	3026	57	2954	60	2954	0.27
	Tot:	46537	11257	48718	836	47116	1423	47501	3.36
5%	BZVR	11345	9003	12663	601	11588	982	12220	1.99
5%	BrBO	9782	357	11000	146	9842	164	10021	0.51
5%	MUVR	16502	385	18106	86	16574	107	17003	0.45
5%	PDBO	2412	223	2610	49	2508	57	2521	0.28
	Tot:	40041	9968	44379	882	40512	1310	41765	3.23
10%	BZVR	9142	9650	10862	596	9469	979	10532	2.01
10%	BrBO	8496	387	10179	132	8552	157	8842	0.51
10%	MUVR	15153	343	17163	84	15315	114	15710	0.43
10%	PDBO	1971	229	2244	50	2062	55	2314	0.25
	Tot:	34762	10609	40448	862	35398	1305	37398	3.20
20%	BZVR	6210	9072	7986	538	6643	1019	8707	2.04
20%	BrBO	6664	375	8672	127	6763	153	7410	0.52
20%	MUVR	13004	384	15708	91	13180	116	13576	0.42
20%	PDBO	1357	230	1653	55	1486	60	1736	0.28
	Tot:	27235	10061	34019	811	28072	1348	31429	3.26
40%	BZVR	3389	10486	4707	578	3931	998	5241	2.31
40%	BrBO	4491	410	6212	130	4544	166	6221	0.53
40%	MUVR	10289	376	13613	95	10592	108	11479	0.45
40%	PDBO	676	262	879	55	776	57	1010	0.28
	Tot:	18845	11534	25411	858	19843	1329	23951	3.57

Future research should investigate the applicability of the LR paradigm to other real life problems, so as to highlight its pros and cons in various contexts.

Acknowledgments

This work was supported by the Future and Emerging Technologies unit of the EC (IST priority), under contract no. FP6-021235-2 (project "ARRIVAL") and by MiUR, Italy (PRIN 2006 project "Models and algorithms for robust network optimization"). We thank Domenico Salvagnin and Arrigo Zanette for the implementation of the timetabling models.

References

[1] L. El Ghaoui A. Ben-Tal and A. Nemirovski. *Robust Optimization*. 2008. (in preparation) preliminary draft available at `http://www2.isye.gatech.edu/~nemirovs/RBnew.pdf`.

[2] A. Ben-Tal and A. Nemirovski. Robust solutions to uncertain linear programs. *Operations Research Letters*, 25:1–13, 1999.

[3] A. Ben-Tal and A. Nemirovski. Robust solutions of linear programming problems contaminated with uncertain data. *Mathematical Programming*, 88:411–424, 2000.

[4] A. Ben-Tal and A. Nemirovski. Robust optimization - methodology and applications. *Mathematical Programming*, 92:453–480, 2002.

[5] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical Programming*, 98:49–71, 2003.

[6] D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52:35–53, 2004.

[7] D. Bienstock. Histogram models for robust portfolio optimization. *The Journal of Computational Finance*, 11(1), 2007.

[8] J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer Series in Operations Research and Financial Engineering. Springer, 2000.

[9] A. Caprara, M. Fischetti, and P. Toth. Modeling and solving the train timetabling problem. *Operations Research*, 50:851–861, 2002.

[10] M. Fischetti, D. Salvagnin, and A. Zanette. Fast approaches to improve the robustness of a railway timetable. Research paper, DEI, University of Padova, 2007.

[11] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer, Berlin, Germany, 2004.

[12] J.T. Linderoth, A. Shapiro, and S.J. Wright. The empirical behavior of sampling methods for stochastic programming. *Annals of Operations Research*, 142:219–245, 2006.

[13] S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Chichester, 1990.

[14] A. Ruszczynski and A. Shapiro. *Stochastic Programming.* Hanbooks in Operations Research and Management Science. Elsevier, 2003.

[15] P. Serafini and W. Ukovich. A mathematical model for periodic scheduling problems. *SIAM Journal on Discrete Mathematics*, 2:550–581, 1989.

[16] A.L. Soyster. Convex programming with set-inclusive constraints and applications to inexact linear programming. *Operations Research*, 21:1154–1157, 1973.