

A warm-start approach for large-scale stochastic linear programs*

Marco Colombo[†] Jacek Gondzio[‡] Andreas Grothey[§]

School of Mathematics and Maxwell Institute
The University of Edinburgh
Mayfield Road, Edinburgh EH9 3JZ
United Kingdom

MS-2006-004

29 August 2006, revised 12 March 2009

Abstract

We describe a way of generating a warm-start point for interior point methods in the context of stochastic programming. Our approach exploits the structural information of the stochastic problem so that it can be seen as a structure-exploiting initial point generator. We solve a small-scale version of the problem corresponding to a reduced event tree and use the solution to generate an advanced starting point for the complete problem. The way we produce a reduced tree tries to capture the important information in the scenario space while keeping the dimension of the corresponding (reduced) deterministic equivalent small. We derive conditions which should be satisfied by the reduced tree to guarantee a successful warm-start of the complete problem. The implementation within the HOPDM and OOPS interior point solvers shows remarkable advantages.

1 Introduction

Stochastic programming [4, 18] models uncertainty through the analysis of possible future outcomes (scenarios). The more detailed the description is, the more robust the decisions taken are. This involves the generation of very large scenario trees and, consequently, of very-large scale deterministic equivalent matrices. With the growing industrial acknowledgement of the benefits of considering uncertainty for planning purposes, it is expected that the need for solving very large problem instances will grow as well.

The practical advantages of relying on interior point solvers become more and more evident as the dimensions of the problems increase. Very-large scale problems, however, are very difficult

*This research has been supported by France Télécom.

[†]Email: M.Colombo@ed.ac.uk

[‡]Email: J.Gondzio@ed.ac.uk, Homepage: <http://maths.ed.ac.uk/~gondzio/>

[§]Email: A.Grothey@ed.ac.uk, Homepage: <http://maths.ed.ac.uk/~agr/>

to solve with general purpose solvers: problems of these sizes can be solved by exploiting the structure present in the matrix; this leads to a further advantage that comes from assigning the computational work to more than one processing unit through the parallelisation of the linear algebra. This is where structure-exploiting parallel solvers such as OOPS [13] excel. Moreover, structure-exploiting interior point methods can be used not only for linear programming problems, but also for quadratic and nonlinear problems [12].

In a large scenario tree there may be very little difference among scenarios, and so the large-scale problem can provide a fine-grained solution to a problem that could have been solved more coarsely by using a much smaller tree. This observation suggests a warm-start technique that can be applied in the context of interior point methods. A warm-start solution is obtained by solving the stochastic optimization problem for a reduced event tree, the dimension of which is much smaller than that of the complete one. The solution to the reduced problem is used to construct an advanced iterate for the complete formulation. We provide evidence that this novel way of exploiting the problem structure to generate an initial iterate provides a better starting point (in terms of centrality, feasibility, and closeness to optimality) than the one produced by a generic strategy. We emphasize that the proposed warm-start strategy is independent of the details of the linear algebra implementation adopted by the solver.

This paper is organised as follows. In Section 2 we outline some basic concepts of stochastic programming and introduce a measure of the distance between scenarios. In Section 3 we review some studies on warm-start techniques for interior point methods. In Section 4 we present a method of generating a reduced event tree and constructing the warm-start iterate. In Section 5 we analyse the approach and derive bounds that the reduced tree has to satisfy to guarantee a successful warm start. Considerations on the implementation and numerical results are discussed in Section 6. Finally, in Section 7 we draw our conclusions.

2 Stochastic programming

Data forecasts are usually made through econometric models that take into account historical data: this helps to determine trends and their variations, but unfortunately it is not applicable to recently introduced products and services, as this data may not be available. When the uncertainty cannot be conveniently forecast, the use of deterministic models is considered inadequate for decision making. In these situations, being able to describe and model the uncertain parameters becomes a requirement for robust decision making. Stochastic programming [4, 18] studies the methods and provides the tools for modelling uncertainty.

The relevance of stochastic programming lies in the fact that it allows the handling of uncertainty in a practical way, through a rich set of tools. The popularity of stochastic programming is on the increase, as its paradigm is well-suited to modelling many real-life problems in several different areas (finance, energy production and planning, telecommunications, logistics, etc). In stochastic programming, the uncertain environment is described through a stochastic process which is usually estimated from historical data or conjectured according to prescribed properties. The continuous process is usually further approximated by a discrete distribution in order to obtain a computationally amenable description. In such a case, the most common techniques [16, 25] generate a finite, but usually very large, number of scenarios that represent an approximate description of the possible outcomes.

2.1 Deterministic equivalent formulation for stochastic programs

A natural formulation of a stochastic programming problem relies on recursion to describe the dynamics of the modelled process. The term *recourse* means that, at each time period, the decision variables adapt to the different outcomes of the random parameters. In a planning approach, the evolution of uncertainties can be described as an alternating sequence of decisions and random realisations that occur at different points in time (stages).

The discrete stochastic process can be represented as an event tree (Figure 1). A node denotes a

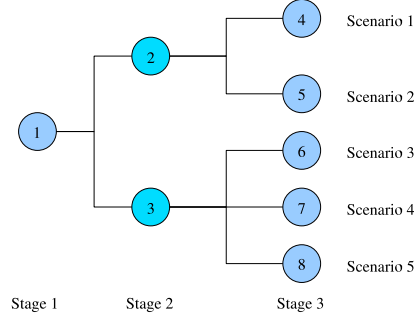


Figure 1: An event tree.

point in time when a realisation of the random process becomes known and a subsequent decision is taken. To each node of the event tree we associate a set of constraints, an objective function, and the conditional probability of visiting the node from its parent node in the previous stage. A path from the root to a leaf node of the event tree represents a scenario. The probability of each scenario is the product of the conditional probabilities of visiting each of the nodes on the path.

To express the deterministic equivalent of the multi-stage stochastic programming problem in node formulation we need to enumerate all nodes of the event tree: we use a breadth-first ordering, i.e. we start from the root node corresponding to the initial stage (stage 1) and end with leaf nodes corresponding to the final stage (stage t_f). Let $t = 1, 2, \dots, t_f$ denote the stages and let \mathcal{L}_t be the set of nodes at stage t . With $a(l)$ we denote the direct ancestor of node $l \in \mathcal{L}_t$ (which is a node that belongs to stage $t - 1$). The decision variables are superscripted with the node number l ; similar notation is used for the corresponding matrix and vector blocks.

In the case of one-period recourse, the main constraint that describes the dynamics of the system has the form

$$T^l x^{a(l)} + W^l x^l = h^l, \quad l \in \mathcal{L}_t, t = 2, \dots, t_f,$$

where T^l is the technology matrix that varies with the node in the event tree, and W^l is the recourse matrix that, in general, may depend on realisations within the same stage, but often varies only with time. The deterministic equivalent formulation of the multi-stage problem has the following general form:

$$\begin{aligned} \min \quad & \sum_{t=1}^{t_f} \sum_{l \in \mathcal{L}_t} p^l (q^l)^\top x^l \\ \text{s.t.} \quad & W^1 x^1 = h^1, \\ & T^{l_t} x^{a(l_t)} + W^{l_t} x^{l_t} = h^{l_t}, \quad l_t \in \mathcal{L}_t, t = 2, \dots, t_f, \\ & x^{l_t} \geq 0, \quad l_t \in \mathcal{L}_t, t = 1, \dots, t_f. \end{aligned} \tag{1}$$

Note that the probabilities in the objective function of problem (1) are the unconditional path probabilities: p^l is the probability that a path goes through node l , which equals the product of the conditional probabilities δ^i , for i along the path from the root to node l , so that $p^i = \delta^i p^{a(i)}$.

If the event tree is traversed with depth-first ordering of the nodes during the generation of the program, the corresponding constraint matrix displays a nested dual block-angular structure. Figure 2 displays the two possible structures for the event tree of Figure 1 according to the chosen ordering of nodes. While the different ordering of blocks within the matrix is not relevant for

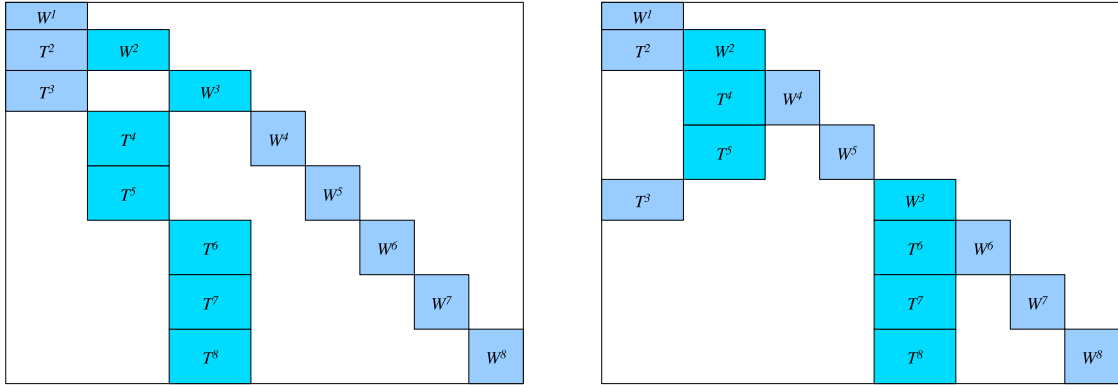


Figure 2: Deterministic equivalent corresponding to the event tree of Figure 1, with nodes listed in breadth-first order (left) and depth-first order (right).

general-purpose solvers, the structure-exploiting software OOPS [12, 13] can take full advantage of the nested dual block-angular structure resulting from the depth-first ordering in its internal object-oriented linear algebra representation.

Several solution methods for stochastic linear programs have been presented in the literature. These often rely on some decomposition approach [2, 19, 23], among others. In this paper, instead, we consider solving the deterministic equivalent problem directly through an interior point method.

2.2 Scenario distance

For the purposes of this paper (see Section 4.1), we need to introduce a measure of distance between scenarios. Let the total number of scenarios be N . Recalling that a scenario is a path in the event tree from the root to a leaf node, we can encode a scenario s_k , $k = 1, \dots, N$, as an ordered set of nodes $s_k = \{l_1, \dots, l_{t_f} : l_t = a(l_{t+1}), t = 1, \dots, t_f - 1\}$. To each node l_t of the tree we associate the 4-tuple $\eta^{l_t} = \{T^{l_t}, W^{l_t}, h^{l_t}, q^{l_t}\}$ of matrices, right-hand side and objective coefficients.

We first define the distance between two nodes i_t and j_t that belong to the same stage t as

$$d(\eta^{i_t}, \eta^{j_t}) = \|T^{i_t} - T^{j_t}\|_\infty + \|W^{i_t} - W^{j_t}\|_\infty + \|h^{i_t} - h^{j_t}\|_\infty + \|q^{i_t} - q^{j_t}\|_\infty. \quad (2)$$

Hence, we compute the distance between scenarios s_i and s_j as

$$D(s_i, s_j) = \sum_{t=1}^{t_f} d(\eta^{i_t}, \eta^{j_t}), \quad i_t \in s_i, j_t \in s_j.$$

Scenarios that belong to the same branch of the tree will have smaller distance in general, as they share some of the nodes. Conversely, scenarios are likely to be farther away if they do not share nodes apart from the root.

3 Warm-start with interior point methods

Consider the linear programming problem in standard form

$$\min c^\top x \quad \text{s.t.} \quad Ax = b, \quad x \geq 0, \quad (3)$$

where $A \in \mathcal{R}^{m \times n}$ is full rank, $x, c \in \mathcal{R}^n$ and $b \in \mathcal{R}^m$. For the purposes of this paper, problem (3) corresponds to the deterministic equivalent generated from a given event tree \mathcal{T} , and we will refer to it as the *complete problem*.

In the context of interior point methods, the non-negativity conditions are replaced by a logarithmic term, thus generating the barrier problem

$$\min c^\top x - \mu \sum_{i=1}^n \ln x_i \quad \text{s.t.} \quad Ax = b, \quad (4)$$

where $\mu > 0$ is the barrier parameter. The first-order optimality conditions (Karush-Kuhn-Tucker conditions) corresponding to problem (4) can be expressed as

$$F_\mu(x, y, s) = \begin{bmatrix} Ax - b \\ A^\top y + s - c \\ XSe - \mu e \end{bmatrix} = 0, \quad (x, s) > 0,$$

where $s \in \mathcal{R}^n$ is the vector of dual slacks, X and S are diagonal matrices with elements x_i and s_i respectively, and $e \in \mathcal{R}^n$ is a vector of ones. As μ is decreased at each iteration, the solution of the perturbed Karush-Kuhn-Tucker conditions traces a unique path toward the optimal set, generally referred to as the central path. Path-following interior point methods [26] seek a solution to the nonlinear system $F_\mu(x, y, s) = 0$ by using Newton's method, and consider the Newton system

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^\top & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} b - Ax \\ c - A^\top y - s \\ -XSe + \mu e \end{bmatrix} = \begin{bmatrix} \xi_b \\ \xi_c \\ \xi_\mu \end{bmatrix}, \quad (5)$$

which needs to be solved with a specified μ for a search direction $(\Delta x, \Delta y, \Delta s)$. To guarantee the positivity of the x and s components when moving along the search direction, the maximum stepsize α is computed such that $(x + \alpha \Delta x, s + \alpha \Delta s) > 0$. Path-following methods rely on keeping the iterates in a neighbourhood of the central path, thus follow it in approaching the optimal solution. In our analysis we work with the *symmetric neighbourhood* [6] of the central path

$$\mathcal{N}_s(\gamma) = \{(x, y, s) : Ax = b, A^\top y + s = c, (x, s) > 0, \gamma\mu \leq x_i s_i \leq \mu/\gamma\}, \quad (6)$$

where $0 < \gamma < 1$. In the authors' experience, such a neighbourhood best describes the desired properties of a "well-centered" interior point iterate.

The problem of finding a starting point is usually solved by using Mehrotra’s starting point heuristic [20], which is considered to be computationally effective. In this heuristic, the starting point is found by solving two least squares problems which attempt to satisfy primal and dual constraints; this point is then shifted inside the positive orthant. However, many practical applications rely on solving a sequence of closely related problems, where the instances differ by some perturbation. This happens within algorithms that are sequential in their nature; also it is very common in (mixed) integer programming, when the problems are solved with some branching strategy, when new cuts are added, etc. In these situations, we expect the solution of an instance to be close to the solution of the next one. Therefore, (warm) starting the optimization of one problem from the solution of the previous problem should reduce the computational effort of solving the perturbed instance. Warm-start techniques are very successful when implemented with a simplex solver (see, for example, [5]). Instead, in the context of interior point methods, they are much more difficult to implement successfully, for the reasons we outline below.

The optimal solution of a linear programming problem found with a path-following interior point method is very close to a vertex of the feasible polytope or, in the case of multiple solutions, it is close to the analytic center of the optimal set of solutions. If the polytope changes, the previously optimal solution may now be very far away from the central path of the perturbed instance. Moreover, an interior point algorithm may get stuck when an iterate gets too close to a boundary before optimality is reached. Hipolito [15] analysed such situation and showed that if the iterate is close to a boundary, the search direction may be parallel to the nearby constraints.

The required features for a good warm-start candidate for an interior point algorithm are somewhat contradictory. The point should not be too close to the boundary of the feasible region in order to be able to absorb larger perturbations in the problem data. Also, it should be sufficiently advanced to provide computational savings over a cold-start iterate. The theory and practice of warm-start techniques for interior point methods is a relatively new and still open field of study. In the remainder of this section, we present a review of some of the warm-start approaches proposed in the interior point literature.

3.1 Literature review

Mitchell [21] and Mitchell and Todd [22] analyse the potential reduction interior point method within a cutting plane algorithm. They exploit the fact that the primal feasible point can be constructed after a set of new columns is added to the problem. They use this strategy with success in column generation scheme and more generally in the solution of combinatorial optimization problems.

Gondzio [9] presents a warm-start procedure for primal–dual interior point methods in the context of a cutting plane method. The interior point method is used to solve a sequence of restricted master problems, which differ by one or more cutting planes. The idea proposed in [9] is to store a nearly optimal point (3–4 digits of accuracy) to be employed as a warm-start point. As one requirement for a good iterate is centrality, it is of interest to perform a few centering steps based on centrality correctors [8] on the stored iterate. An auxiliary feasibility recovery procedure may be needed as, due to the addition of cuts, large infeasibilities are often produced. The warm-start approach proposed in [9] is extended in [14] to the case of solving a sequence of problems with the same dimensions but changing problem data (the objective function or the right-hand side) which arise in the context of decomposition approaches for large structured

linear programs.

Yildirim and Wright [27] consider again the case of solving a sequence of problems in fixed dimensions, and analyse the number of iterations required to converge to a solution of the perturbed instance from the warm-start point and obtain worst-case estimates. They show that these estimates depend on the size of the perturbation as well as on the conditioning of the problem instances. Thus they obtain conditions under which the complexity of the warm-start approach is better than for the cold start case. The strategy proposed in [27] aims at absorbing the primal and dual infeasibilities introduced by the perturbation in just one step. This strategy requires to backtrack to an iterate for which μ is large enough to allow a full step for the correction direction they produce. The amount of necessary backtracking depends on the magnitude of the perturbation (as measured by the change in the problem data): this is intuitively justified considering that a large perturbation will produce a large adjustment. To ensure the availability of an approximate μ -center from which the perturbation can be absorbed in one step, a subset of iterates for different values of μ is stored. When the size of the perturbation becomes known, the closest μ available is retrieved, and the corresponding iterate is used as a warm-start point for the next problem in the sequence. In [27], two different corrections for the perturbation are studied: one is based on least squares, the other on a Newton step correction. A detailed computational comparison of these strategies has been carried out by John and Yildirim [17].

Gondzio and Grothey [10] assess perturbations by a relative measure of implied primal and dual infeasibilities, and analyse recovery steps in the primal and the dual spaces independently. This reoptimization procedure is based on two phases: first, an attempt is made to absorb the infeasibilities caused by the perturbation with a full Newton step; second, the centrality of the iterate is improved. Another key feature of Gondzio and Grothey's approach [10] is that the primal search direction is governed only by the primal perturbation, and similarly for the dual space. They produce bounds on the magnitude of primal perturbation ξ_b and dual perturbation ξ_c that can be absorbed in a single Newton step; as opposed to the results of [27], these bounds are easy to compute and thus can be used in practice. An approximate μ -center is stored for a tolerance level that depends on the magnitude of the expected perturbation. The absorption of infeasibilities may be spread across a few iterations whenever the stepsizes fall below a predefined level. As this strategy does not make assumptions upon the centrality of the warm-start iterate, it can be initialised with any iterate. Gondzio and Grothey [10] apply this warm-start strategy successfully to structured problems for crash-start points that come from a cross-decomposition scheme, and thus may lack centrality. In subsequent work, Gondzio and Grothey [11] develop a variety of heuristics based on sensitivity analysis according to which the warm-start iterate is perturbed with the aim of allowing a longer stepsize in the search direction.

A different approach has been studied by Benson and Shanno [1]. They investigate how to improve the efficiency of interior point methods in a reoptimization context by the use of a primal-dual penalty approach. While standard penalty techniques are effective only in one space, the introduction of penalty parameters in both the primal and the dual problems allows to capture perturbations in both spaces. The strategy relaxes the non-negativity constraints for the decision variables, penalising the violation in the objective, for both the primal and dual problems. The penalised problem allows the variables to become negative: this provides more freedom of movement for the variables, with the immediate advantage of accepting larger stepsizes along the computed search direction. This favours a faster progress especially in the first few iterations, when the perturbation needs to be absorbed. Benson and Shanno [1] also provide some computational evidence of the effectiveness of their strategy.

4 A reduced-tree warm-start iterate

While the strategies mentioned in the previous section apply to general linear problems, we introduce an approach tailored to stochastic programming. In particular, we propose to exploit the structure inherent to a stochastic programming problem to generate a good warm-start iterate.

In the event tree corresponding to a large multistage program, the numerous leaf nodes descend from a relatively small number of branches in the first few stages. Two “neighbouring” scenarios, that is two scenarios that have common nodes, may display large differences concerning later stage decisions, but the decisions taken in the earlier stages are identical (nonanticipativity).

Techniques for reducing the size of the scenario tree have been studied before from a probabilistic perspective; in some cases considerable savings can be obtained with such methods. Among others, Dupačová et al. [7] discuss an optimal scenario reduction technique that couples a large reduction of the scenario tree with a small loss in accuracy. In their example, a reduction by 50% of the scenario tree still maintains about 90% of the original accuracy. In this paper, we are interested in capturing some aspects of the stochasticity of the event tree without assuming further knowledge on the underlying stochastic process that generated it. Given this difference from what is required for example by [7], we will use less sophisticated arguments in finding a reduced tree. We remark that if we had knowledge of the underlying stochastic process, then we could exploit it in the generation of the reduced tree.

We first study how to build a reduced tree, \mathcal{T}_R , by choosing just some of the available scenarios. We provide some insight on how to make this selection, so that our choice performs better than an arbitrary one. Then we discuss how to obtain a warm-start solution from the reduced tree that corresponds to the chosen scenarios. Our aim is to generate a warm-start iterate that allows the complete problem to be solved to optimality in fewer iterations (and less computing time) than an iterate chosen with a standard starting point heuristic [20]. With these aims, we propose a way of choosing a subset of scenarios that we believe to be sufficiently representative of the whole tree. The approach can be summarised in the steps of Algorithm 1.

Algorithm 1 Reduced-tree warm-start algorithm

Require: The complete event tree \mathcal{T} .

1. Generate a reduced event tree $\mathcal{T}_R \subset \mathcal{T}$;
 2. Solve the deterministic equivalent corresponding to \mathcal{T}_R with a loose tolerance;
 3. Use this solution to construct a warm-start iterate for the complete problem;
 4. Solve the complete problem to optimality.
-

In the rest of this section we define our method of generating a reduced tree and describe the construction of the complete warm-start iterate.

4.1 Reduced tree generation

We generate the reduced tree by taking into account both the structural and the stochastic information available from the problem formulation. By structural information we mean the shape of the event tree, i.e. how the tree branches at the various stages. By stochastic information

we mean the probabilities associated to each node in the tree, and consequently to each scenario. Hence we adopt two complementary strategies. First we choose a subset of branches of the event tree; then, in each branch, we choose the most representative leaf nodes.

We try to capture the structure of the complete tree by making sure that a sufficient number of different early stage decisions will appear in the reduced tree. In some sense, we look for a way to span the breadth of the complete tree. For a defined small value $\kappa < t_f$, where t_f is the number of stages in the problem, we choose some of the nodes at the κ -th stage, together with all their ancestors up to the root, to appear in the subtree. The choice of nodes to appear in the reduced tree should be guided by probabilities. The rationale for this strategy is to ensure that our warm-start iterate is a good representation of the decisions to be taken in the first few stages, as getting early decisions right is fundamental for easier optimization of the later stages.

To illustrate this idea, suppose we deal with a multistage setting where there are $t_f = 4$ stages, such as in the tree of Figure 3: for $\kappa = 2$ we choose nodes 1, 2 and 3 to be in the reduced tree.

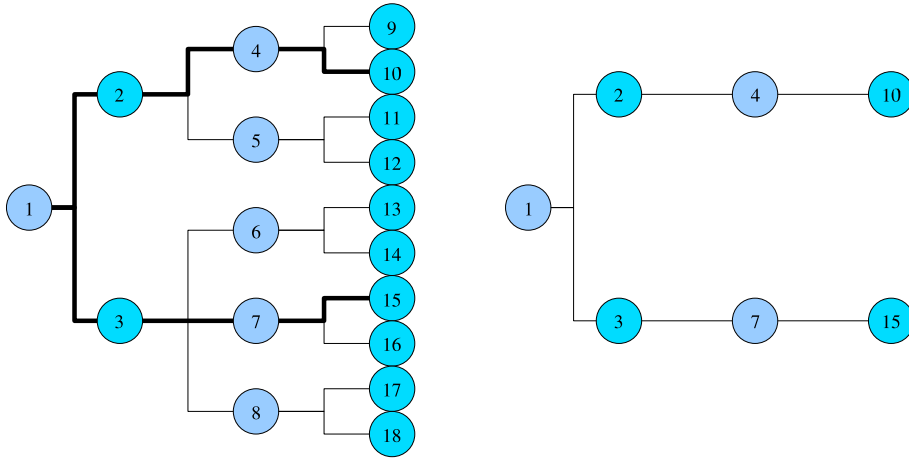


Figure 3: Complete tree and the reduced tree corresponding to the chosen scenarios (in bold).

Each of the chosen nodes in the κ -th stage now becomes the root of a branch of the tree, which we call a subtree. In each subtree we choose the scenario that minimises the distance to an average scenario in the same subtree. Let S_t be the set of nodes in the subtree \mathcal{S} at stage t , and $|S_t|$ its cardinality. For each stage t within subtree \mathcal{S} , we determine an artificial node n_t by averaging the data associated to all the subtree nodes at this stage:

$$n_t = \frac{1}{|S_t|} \sum_{l_t \in S_t} (T^{l_t}, W^{l_t}, h^{l_t}, q^{l_t}), \quad \kappa < t \leq t_f.$$

We define the average scenario for subtree \mathcal{S} as an ordered set of nodes $\{l_\kappa, n_{\kappa+1}, \dots, n_{t_f}\}$. Therefore, the average scenario \bar{s} (in the complete tree) is obtained by listing the nodes from the root of the tree to the root of the subtree \mathcal{S} , and then by appending the artificial (averaged) nodes. We define it as

$$\bar{s} = \{l_1, \dots, l_\kappa, n_{\kappa+1}, \dots, n_{t_f}\},$$

where $l_t = a(l_{t+1})$ for $t = 1, \dots, \kappa - 1$. Scenario \bar{s} is completely artificial, and there is no guarantee that it is feasible; hence, we cannot use it directly as our representative scenario. Instead, we use it as a reference point which we compare all other scenarios to, and thus find the closest scenario

among the existing ones. In this way we do not introduce spurious infeasibilities. Hence, in the subtree \mathcal{S} we choose the representative scenario s^* as

$$s^* = s_k, \quad k = \arg \min_{i \in \mathcal{S}} \{(1 - p^i)D(s_i, \bar{s})\}, \quad (7)$$

where, since our ultimate goal is to find the most representative scenario in the subtree, we use the term $(1 - p^i)$ to “bring closer” scenarios that have a higher probability of occurring.

The reduced tree selection induces a function $r : \mathcal{T} \rightarrow \mathcal{T}_R$ that maps each node l of the complete tree to a corresponding node in the reduced tree in the following way: if $l \in \mathcal{T}_R$, then $r(l) = l$; if $l \notin \mathcal{T}_R$, then we choose as $r(l)$ the node in the representative scenario corresponding to the same stage as node l . In other words, to get from node $l \in \mathcal{T}$ to $r(l) \in \mathcal{T}_R$ we walk up the tree \mathcal{T} until we find a node that is also in \mathcal{T}_R , and from there walk back down the reduced tree until we arrive at the same stage as the original node.

The mapping between the two trees is used to decide how to initialise the warm-start iterate for the complete tree, as presented in the next section. We remark that our generation process guarantees that for each $l \in \mathcal{T}$, the following properties hold:

$$a(r(l)) \in \mathcal{T}_R, \quad \text{and} \quad a(r(l)) = r(a(l)), \quad (8)$$

that is, if a node is in the reduced tree then so is its parent, and the mapping $r(\cdot)$ preserves the parent-child relationship.

Continuing the example started above, we consider two subsets of scenarios, corresponding to nodes 9–12 (for the subtree rooted at node 2) and to nodes 13–18 (for the subtree rooted at node 3). Within each subset we build the scenario of average nodes and then find the representative scenario. The resulting reduced tree is shown in the right of Figure 3.

Before proceeding further, we introduce some notation. We adopt the convention that symbols referring to the reduced tree carry the subscript R . Given a node $l \in \mathcal{L}_t$, we define $\mathcal{D}^l \subset \mathcal{L}_{t+1}$ to be the set of direct descendants of node l . We introduce the set \mathcal{I}^k of nodes in the complete tree that are mapped to the same reduced-tree node k , that is $\mathcal{I}^k = \{l \in \mathcal{T} : r(l) = k\}$ for $k \in \mathcal{T}_R$. In what follows, we will exploit the fact that

$$\mathcal{D}^l = \bigcup_{k \in \mathcal{D}_R^{r(l)}} \mathcal{I}^k \cap \mathcal{D}^l, \quad (9)$$

that is, the set of descendants can be partitioned according to which nodes of the reduced tree they map to.

As will be seen later it is advantageous if the aggregation of nodes is balanced throughout the tree. On average, every node in the reduced tree corresponds to n/n_R nodes in the full tree, so for a totally balanced aggregation we would expect

$$\frac{p^l}{p_R^{r(l)}} \frac{n}{n_R} \approx 1.$$

We define the deviation from this as

$$\rho = \min_{l \in \mathcal{T}} \left\{ \frac{p^l}{p_R^{r(l)}} \frac{n}{n_R}, \frac{p_R^{r(l)}}{p^l} \frac{n_R}{n} \right\}. \quad (10)$$

4.2 Construction of the warm-start iterate

From the reduced tree we build the reduced deterministic equivalent problem

$$\min c_R^\top x_R \quad \text{s.t.} \quad A_R x_R = b_R, \quad x_R \geq 0, \quad (11)$$

with $A_R \in \mathcal{R}^{m_R \times n_R}$, $x_R, c_R \in \mathcal{R}^{n_R}$ and $b_R \in \mathcal{R}^{m_R}$. We call problem (11) the *reduced problem*. As we expect that $(m_R, n_R) \ll (m, n)$, the reduced problem is much smaller than the complete formulation, and hence much easier to solve.

We solve problem (11) with an interior point method. For the reasons presented in Section 3, we do not aim for optimality, but instead we aim for a sufficiently advanced primal–dual feasible point. Therefore we stop at an iterate $(x_R^*, y_R^*, s_R^*) \in \mathcal{N}_s(\gamma)$ for a barrier parameter corresponding to merely few digits of accuracy in the solution. This iterate is used to construct the warm-start point $(\hat{x}, \hat{y}, \hat{s})$ for the complete problem on a node-by-node basis.

It should be noted that the reduced-tree generation process can be interpreted as a node-aggregation process, in which all nodes in $\mathcal{I}^k \subset \mathcal{T}$ are aggregated into a single node $k \in \mathcal{T}_R$. The node aggregation determines the node probabilities p_R^k associated with each node $k \in \mathcal{T}_R$; we use:

$$p_R^k = \sum_{i \in \mathcal{I}^k} p^i, \quad k \in \mathcal{T}_R. \quad (12)$$

Denote by $(\hat{x}^l, \hat{y}^l, \hat{s}^l)$ the part of the vectors $(\hat{x}, \hat{y}, \hat{s})$ corresponding to node $l \in \mathcal{T}$, and likewise $(x_R^{r(l)}, y_R^{r(l)}, s_R^{r(l)})$ for components of the reduced problem solution. We construct the starting point for the complete problem in the following manner:

$$\hat{x}^l = x_R^{r(l)}, \quad (\hat{y}^l, \hat{s}^l) = \frac{p^l}{p_R^{r(l)}} (y_R^{r(l)}, s_R^{r(l)}), \quad l \in \mathcal{T}, \quad (13)$$

where $p_R^{r(l)}$ is computed according to (12). This means that the dual reduced-tree solution is spread among the nodes it initialises, as can be seen here:

$$\sum_{i \in \mathcal{I}^k} (\hat{y}^i, \hat{s}^i) = \sum_{i \in \mathcal{I}^k} \frac{p^i}{p_R^k} (y_R^k, s_R^k) = (y_R^k, s_R^k) \frac{1}{p_R^k} \sum_{i \in \mathcal{I}^k} p^i = (y_R^k, s_R^k), \quad k \in \mathcal{T}_R.$$

Considering again the example of Figure 3, suppose that in the reduced tree we accepted only the scenarios that end at node 10 and node 15, so that the reduced tree consists of nodes 1, 2, 3, 4, 7, 10 and 15. By solving the corresponding reduced problem, we obtain the parts of the solution vector associated to such nodes. These can be used directly in the complete iterate (Figure 4 top). We fill in the missing elements by reproducing the solution from the nodes in the same subtree and the same stage (Figure 4 bottom). The proposed way of constructing the complete iterate is easy to implement and its execution time is negligible.

5 Analysis of the warm-start iterate

In this section we study how the warm-start iterate generated with the procedures presented above satisfies the conditions expressed by Gondzio and Grothey [10]. Contrary to what is

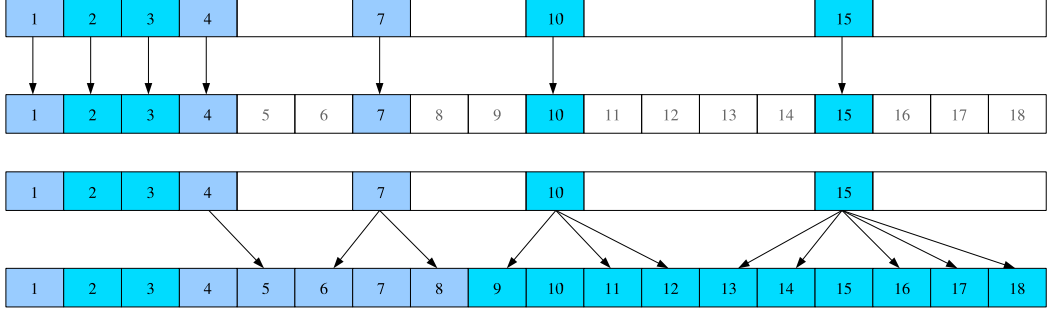


Figure 4: Generation of the warm-start iterate.

assumed in both [27] and [10], in our approach the dimension of the problem changes, as the reduced tree problem is, by construction, much smaller than the complete problem.

However, similarly to what we did with the solution vector, we can expand the reduced problem to one which has the same dimension as the complete problem (3) by replicating the blocks in the coefficient matrix and in the objective and right-hand side vectors, as shown in Figure 5. This corresponds to creating the (artificial) *expanded problem*

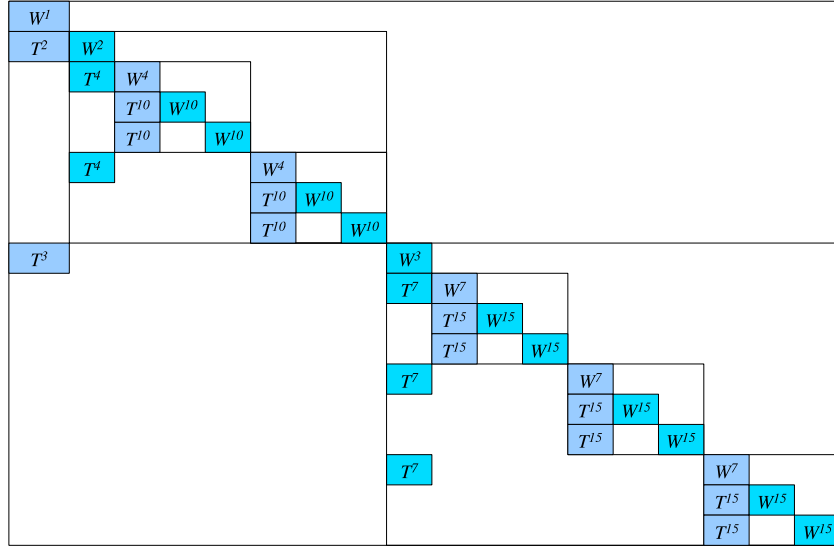


Figure 5: The expanded system for the complete event tree of Figure 3.

$$\min \hat{c}^\top x \quad \text{s.t.} \quad \hat{A}x = \hat{b}, \quad x \geq 0, \quad (14)$$

the dimension of which, $\hat{A} \in \mathcal{R}^{m \times n}$, $\hat{c}, x \in \mathcal{R}^n$ and $\hat{b} \in \mathcal{R}^m$, corresponds to the dimension of the complete problem (3). Using the notation introduced earlier, we will denote all symbols referring to the expanded problem with a hat $\hat{\cdot}$.

To analyse the warm-start iterate we can now follow a two-step procedure. First we note that from an advanced iterate $(x_R, y_R, s_R) \in \mathcal{N}_s^R(\gamma)$ for the reduced problem the procedure in (13) constructs a primal–dual feasible point $(\hat{x}, \hat{y}, \hat{s})$ for the expanded problem. Indeed, in Theorem 3 we will show that $(\hat{x}, \hat{y}, \hat{s}) \in \hat{\mathcal{N}}_s(\hat{\gamma})$. In the second step we can use this iterate to warm-start the complete problem. Since from the expanded to the complete problem the problem size does not change, the methods developed in [10, 27] can be used to analyse the warm-start iterate.

We start the analysis with a technical result.

Lemma 1. *Let $l \in \mathcal{T}$, then*

$$\sum_{i \in \mathcal{D}^l} T^{r(i)\top} \hat{y}^i = \frac{p^l}{p_R^{r(l)}} \sum_{k \in \mathcal{D}_R^{r(l)}} T^{k\top} y_R^k.$$

Proof. We have this chain of identities:

$$\sum_{i \in \mathcal{D}^l} T^{r(i)\top} \hat{y}^i = \sum_{i \in \mathcal{D}^l} T^{r(i)\top} y_R^{r(i)} \frac{p^i}{p_R^{r(i)}} = \frac{p^l}{p_R^{r(l)}} \sum_{i \in \mathcal{D}^l} T^{r(i)\top} y_R^{r(i)} \frac{\delta^i}{\delta_R^{r(i)}} = \frac{p^l}{p_R^{r(l)}} \sum_{k \in \mathcal{D}_R^{r(l)}} \frac{T^{k\top} y_R^k}{\delta_R^k} \sum_{i \in \mathcal{I}^k \cap \mathcal{D}^l} \delta^i,$$

where the first equality follows from (13) and the second from $p^i = p^l \delta^i$ and $p_R^{r(i)} = p_R^{r(l)} \delta_R^{r(i)}$ for $i \in \mathcal{D}^l$. The last equality is obtained observing that we can partition \mathcal{D}^l according to (9). The claim then follows noting that for a node l at stage $t < \kappa$:

$$\sum_{i \in \mathcal{I}^k \cap \mathcal{D}^l} \delta^i = \sum_{i \in \mathcal{I}^k} \delta^i = \delta^k = \delta_R^k,$$

while for a node l at stage $t \geq \kappa$:

$$\sum_{i \in \mathcal{I}^k \cap \mathcal{D}^l} \delta^i = \sum_{i \in \mathcal{D}^l} \delta^i = \delta_R^k = 1.$$

□

The next two results show that the reduced tree solution can be used to generate a point that is primal–dual feasible and central for the expanded problem.

Theorem 2. *If (x_R, y_R, s_R) is primal and dual feasible for the reduced problem (11), then the warm-start solution $(\hat{x}, \hat{y}, \hat{s})$ obtained from (13) is primal and dual feasible for the expanded problem (14).*

Proof. As $\hat{x}^l = x_R^{r(l)}$, primal feasibility is trivially satisfied:

$$T^{r(l)} \hat{x}^{a(l)} + W^{r(l)} \hat{x}^l = h^{r(l)}, \quad l \in \mathcal{T}. \quad (15)$$

Now we consider dual feasibility. By assumption, the reduced problem solution satisfies the dual constraints:

$$W^{r(l)\top} y_R^{r(l)} + \sum_{k \in \mathcal{D}_R^{r(l)}} T^{r(k)\top} y_R^{r(k)} + s_R^{r(l)} = p_R^{r(l)} q^{r(l)}, \quad r(l) \in \mathcal{T}_R.$$

Multiplying both terms by $p^l/p_R^{r(l)}$ we obtain

$$\frac{p^l}{p_R^{r(l)}} \left(W^{r(l)\top} y_R^{r(l)} + \sum_{k \in \mathcal{D}_R^{r(l)}} T^{r(k)\top} y_R^{r(k)} + s_R^{r(l)} \right) = p^l q^{r(l)},$$

which, according to (13) and Lemma 1, becomes

$$W^{r(l)\top} \hat{y}^l + \sum_{i \in \mathcal{D}^l} T^{r(i)\top} \hat{y}^i + \hat{s}^l = p^l q^{r(l)}, \quad l \in \mathcal{T}, \quad (16)$$

so (\hat{y}, \hat{s}) satisfies the dual constraints in the expanded problem. □

Theorem 3. If $(x_R, y_R, s_R) \in \mathcal{N}_s^R(\gamma)$ for some $\gamma \in (0, 1)$, then $(\hat{x}, \hat{y}, \hat{s}) \in \hat{\mathcal{N}}_s(\rho\gamma)$, with ρ defined in (10).

Proof. From Theorem 2, the warm-start iterate $(\hat{x}, \hat{y}, \hat{s})$ is feasible in the reduced system. Hence, here we only need to prove centrality. We observe that

$$\begin{aligned} \hat{\mu} &= \frac{\hat{x}^\top \hat{s}}{n} = \frac{1}{n} \sum_{l \in \mathcal{T}} (\hat{x}^l)^\top \hat{s}^l = \frac{1}{n} \sum_{k \in \mathcal{T}_R} \sum_{i \in I_k} (\hat{x}^i)^\top \hat{s}^i \\ &= \frac{1}{n} \sum_{k \in \mathcal{T}_R} \sum_{i \in I_k} \frac{p^i}{p_R^k} (x_R^k)^\top s_R^k \\ &= \frac{1}{n} \sum_{k \in \mathcal{T}_R} \frac{1}{p_R^k} (x_R^k)^\top s_R^k \sum_{i \in I_k} p^i \\ &= \frac{n_R}{n} \mu_R, \end{aligned}$$

where we used (13) and (12), and $\mu_R = x_R^\top s_R / n_R$. Hence, since $(x_R, y_R, s_R) \in \mathcal{N}_s(\gamma)$ implies $(x_R)_j (s_R)_j \geq \gamma \mu_R$, for $j = 1, \dots, n_R$, using (10) we have

$$\hat{x}_j^l \hat{s}_j^l = (x_R^{r(l)})_j (s_R^{r(l)})_j \frac{p^l}{p_R^{r(l)}} \geq \gamma \mu_R \frac{p^l}{p_R^{r(l)}} = \gamma \hat{\mu} \frac{n}{n_R} \frac{p^l}{p_R^{r(l)}} \geq \rho \gamma \hat{\mu}, \quad l \in \mathcal{T}.$$

The upper bound $\hat{x}_j^l \hat{s}_j^l \leq \hat{\mu} / (\rho \gamma)$ can be derived similarly. \square

5.1 Absorbing perturbations

We argue that the difference between the data of the expanded problem (14) and that of the original (complete) problem (3) can be interpreted as a perturbation between two problem instances of identical dimension. Clearly the expanded system has merely a theoretical interest, as we use it to evaluate the magnitude of the perturbation introduced, and we never generate it in practice.

We assume that a feasible long-step path-following algorithm based on the symmetric neighbourhood $\mathcal{N}_s(\gamma)$ [6] is used to solve the warm-started complete problem. Although the constructed warm-start iterate $(\hat{x}, \hat{y}, \hat{s})$ from (13) is feasible in the expanded problem, it is not feasible in the complete problem. As in [27] and [10] we derive conditions that guarantee to absorb these infeasibilities with one *modification step*. For this, consider the following Newton system:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^\top & I \\ \hat{S} & 0 & \hat{X} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} \xi_b \\ \xi_c \\ 0 \end{bmatrix}, \quad (17)$$

where $\xi_b = b - A\hat{x}$ and $\xi_c = c - A^\top \hat{y} + \hat{s}$ are the infeasibilities incurred by using the expanded iterate $(\hat{x}, \hat{y}, \hat{s})$ to warm-start the complete problem. Gondzio and Grothey [10] analyse the same system, but are concerned with absorbing primal and dual infeasibility separately by splitting (17) into two separate directions. We will give a more general result and apply it to the situation of warm start for stochastic programming problems. To avoid overburdening the notation, we will drop the hat from the warm-start vectors. We will keep it in the neighbourhoods to make

a clear distinction between $\widehat{\mathcal{N}}_s(\gamma)$ and $\mathcal{N}_s(\gamma)$ denoting the symmetric neighbourhoods for the expanded problem (14) and the complete problem (3), respectively.

After some straightforward manipulation following the arguments of [10], the Newton direction (17) can be expressed in terms of the primal and dual residuals ξ_b, ξ_c as

$$\begin{aligned}\Delta x &= (XS^{-1}A^\top(AXS^{-1}A^\top)^{-1}AXS^{-1}-XS^{-1})\xi_c+XS^{-1}A^\top(AXS^{-1}A^\top)^{-1}\xi_b, \\ \Delta y &= (AXS^{-1}A^\top)^{-1}(AXS^{-1}\xi_c+\xi_b), \\ \Delta s &= (I-A^\top(AXS^{-1}A^\top)^{-1}AXS^{-1})\xi_c-A^\top(AXS^{-1}A^\top)^{-1}\xi_b.\end{aligned}\tag{18}$$

We consider matrix $Q = I - S^{-1}A^\top(AXS^{-1}A^\top)^{-1}AX$, and restate Lemma 3.2 of [10], which provides a bound on the norm of Q , in terms of the symmetric neighbourhood $\mathcal{N}_s(\gamma)$.

Lemma 4. *If $(x, y, s) \in \mathcal{N}_s(\gamma)$, then $\|Q\|_2 \leq 1/\gamma$.*

Proof. For a point $(x, y, s) \in \mathcal{N}_s(\gamma)$, the following inequalities hold:

$$(x_i s_i)^{-1/2} \leq (\gamma\mu)^{-1/2}, \quad \text{and} \quad (x_i s_i)^{1/2} \leq (\mu/\gamma)^{1/2}.$$

With some manipulations, we can express matrix Q as

$$Q = X^{-1/2}S^{-1/2} \left[I - X^{1/2}S^{-1/2}A^\top(AXS^{-1}A^\top)^{-1}AX^{1/2}S^{-1/2} \right] X^{1/2}S^{1/2},$$

where the term in square brackets is an orthogonal projection on the null space of $AX^{1/2}S^{-1/2}$, so its Euclidean norm is 1. As desired, we obtain

$$\|Q\|_2 = \|X^{-1/2}S^{-1/2}\|_2 \|X^{1/2}S^{1/2}\|_2 \leq 1/\gamma. \quad \square$$

In the next Lemma we state sufficient conditions for the perturbations to guarantee a full Newton step.

Lemma 5. *Let $(x, y, s) \in \widehat{\mathcal{N}}_s(\gamma)$ be the warm-start iterate and define the scaled residuals*

$$\tilde{\xi}_b = X^{-1}A^\top(AA^\top)^{-1}\xi_b \quad \text{and} \quad \tilde{\xi}_c = S^{-1}\xi_c.\tag{19}$$

If for $\beta < 1$ we have

$$\|\tilde{\xi}_b\|_\infty + \|\tilde{\xi}_c\|_\infty \leq \beta (1 + \sqrt{n}/\gamma)^{-1},$$

then the full Newton step (17) from the warm-start iterate can be taken and absorbs the complete infeasibilities.

Proof. Using the definitions of the matrix Q and of the relative residual vectors (19), the relations (18) simplify to

$$X^{-1}\Delta x = -Q\tilde{\xi}_c + (I - Q)\tilde{\xi}_b = -S^{-1}\Delta s,$$

yielding the bound

$$\|X^{-1}\Delta x\|_\infty \leq \|Q\|_\infty \|\tilde{\xi}_c\|_\infty + (1 + \|Q\|_\infty) \|\tilde{\xi}_b\|_\infty \leq (1 + \|Q\|_\infty) (\|\tilde{\xi}_b\|_\infty + \|\tilde{\xi}_c\|_\infty).\tag{20}$$

As $(x, y, s) \in \widehat{\mathcal{N}}_s(\gamma)$, using Lemma 4 we obtain $\|Q\|_\infty \leq \sqrt{n}\|Q\|_2 \leq \sqrt{n}/\gamma$. Substituting it into (20), we get

$$\|X^{-1}\Delta x\|_\infty \leq (1 + \sqrt{n}/\gamma) (\|\tilde{\xi}_b\|_\infty + \|\tilde{\xi}_c\|_\infty),$$

which, under the condition of the Lemma, implies

$$\|X^{-1}\Delta x\|_\infty = \|S^{-1}\Delta s\|_\infty \leq \beta, \quad (21)$$

that is the full Newton step is feasible, as $\beta < 1$. \square

The following theorem establishes that the search direction obtained from (17) brings the iterate inside a symmetric neighbourhood of the central path for the complete problem.

Theorem 6. *Let $(x, y, s) \in \widehat{\mathcal{N}}_s(\gamma)$ and $\beta < 1$. Under the conditions of Lemma 5, the new point $(\tilde{x}, \tilde{y}, \tilde{s}) = (x + \Delta x, y + \Delta y, s + \Delta s) \in \mathcal{N}_s((1 - \beta^2)\gamma)$.*

Proof. The barrier parameter at the new point $(\tilde{x}, \tilde{y}, \tilde{s})$ is

$$n\tilde{\mu} = \sum_{i=1}^n \tilde{x}_i \tilde{s}_i = \sum_{i=1}^n (x_i + \Delta x_i)(s_i + \Delta s_i) = \sum_{i=1}^n (x_i s_i + \Delta x_i \Delta s_i), \quad (22)$$

as, according to the last equation of (17), $s_i \Delta x_i + x_i \Delta s_i = 0$, $i = 1, \dots, n$; the latter also implies that $\Delta x_i \Delta s_i \leq 0$. Using (21) from Lemma 5, we have $\|X^{-1}\Delta x\|_\infty \|S^{-1}\Delta s\|_\infty \leq \beta^2$, and so

$$-\beta^2 x_i s_i \leq \Delta x_i \Delta s_i \leq 0. \quad (23)$$

By summing up all products and adding $n\mu = \sum_i x_i s_i$ to all terms, and by using (22), we obtain

$$(1 - \beta^2)n\mu \leq n\tilde{\mu} \leq n\mu. \quad (24)$$

We now study whether the new iterate is still in (some) symmetric neighbourhood of the central path by checking the pairwise complementary products

$$\tilde{x}_i \tilde{s}_i = x_i s_i + \Delta x_i \Delta s_i = \left(1 + \frac{\Delta x_i \Delta s_i}{x_i s_i}\right) x_i s_i.$$

Using (23) and (24) we obtain

$$\tilde{x}_i \tilde{s}_i \geq (1 - \beta^2)\gamma\mu \geq (1 - \beta^2)\gamma\tilde{\mu} \quad \text{and} \quad \tilde{x}_i \tilde{s}_i \leq \frac{1}{\gamma}\mu \leq \frac{1}{(1 - \beta^2)\gamma}\tilde{\mu},$$

which proves the statement of the theorem. \square

5.2 Conditions on the warm-start iterate

We use Lemma 5 to obtain conditions that the reduced tree has to satisfy in order for a warm start of the complete problem to be successful. In order to prove this result, we need to assume that the primal–dual solution (x_R^*, y_R^*, s_R^*) to the reduced stochastic programming problem is uniformly bounded, say,

$$\max\{\|x_R^*\|_\infty, \|y_R^*\|_\infty, \|s_R^*\|_\infty\} \leq B, \quad \max\{\|(X_R^*)^{-1}e\|_\infty, \|(S_R^*)^{-1}e\|_\infty\} \leq B, \quad (25)$$

where $B > 1$. It is worth noting that since we work with the symmetric neighbourhood (6), we actually need only the first inequality to hold. Indeed, if $x_j^* \leq B$ then $1/s_j^* \leq x_j^*/(\gamma\mu) \leq B/(\gamma\mu)$ and, similarly, if $s_j^* \leq B$ then $1/x_j^* \leq s_j^*/(\gamma\mu) \leq B/(\gamma\mu)$. In other words, the boundedness of the iterate (x_R^*, y_R^*, s_R^*) implies the boundedness of the component-wise inverses of x_R^* and s_R^* .

The reduced problem solution is in a neighbourhood of the central path for the reduced problem. In particular, this is the case if additional centering steps are computed once the desired tolerance level has been attained [9]. Using the feasibility result of Theorem 2, the residuals for the complete problem at the warm-start point $(\hat{x}, \hat{y}, \hat{s})$ are:

$$\begin{aligned}\xi_b &= b - A\hat{x} &= (b - \hat{b}) - (A - \hat{A})\hat{x}, \\ \xi_c &= c - A^\top \hat{y} - \hat{s} &= (c - \hat{c}) - (A - \hat{A})^\top \hat{y}.\end{aligned}$$

It is crucial to ensure that the primal and dual residuals ξ_b and ξ_c are small. By construction, the elements of the vectors $(b - \hat{b})$ and $(c - \hat{c})$ that correspond to nodes in the reduced tree are zero; for the same reason, the corresponding blocks of $(A - \hat{A})$ are zero as well. The elements corresponding to the nodes not considered in the reduced tree will be, in general, non zero. However, as the scenarios in the reduced tree were chosen according to (7) in order to minimize the distance from the average case, we expect the perturbations to be small.

We can now state the following result, in which we obtain some bounds on the size of the primal and dual perturbations.

Lemma 7. *Let the reduced tree be chosen in such a way that for every node $i \in \mathcal{T}$ the node distance (2) is $d(r(i), i) < \varepsilon$, for an $\varepsilon > 0$. If the reduced problem solution is primal and dual feasible and satisfies (25), then $\|\xi_b\|_\infty \leq \varepsilon B$ and $\|\xi_c\|_\infty \leq \varepsilon B |\mathcal{T}_R|$, where $|\mathcal{T}_R|$ is the number of nodes in the reduced tree.*

Proof. Using the form of the stochastic programming problem (1) we can write the primal residual of the complete problem as

$$\|\xi_b\|_\infty = \|b - A\hat{x}\|_\infty = \max\{\|h^l - T^l \hat{x}^{a(l)} - W^l \hat{x}^l\|_\infty : l \in \mathcal{L}_t, t = 1, \dots, t_f\}.$$

The contribution of a node $l \in \mathcal{T}$ to ξ_b is

$$\begin{aligned}\|\xi_b^l\|_\infty &= \|h^l - T^l \hat{x}^{a(l)} - W^l \hat{x}^l\| \\ &= \|h^l - h^{r(l)} - (T^l - T^{r(l)})\hat{x}^{a(l)} - (W^l - W^{r(l)})\hat{x}^l\| \\ &\leq (\|h^l - h^{r(l)}\| + \|T^l - T^{r(l)}\| + \|W^l - W^{r(l)}\|)B \\ &\leq d(l, r(l))B \leq \varepsilon B,\end{aligned}$$

where the step from the first to the second line uses (15), and all norms here are infinity norms. This clearly implies that $\|\xi_b\|_\infty \leq \varepsilon B$.

The dual residual for the complete problem at the warm-start point can be written as

$$\|\xi_c\|_\infty = \|c - A^\top \hat{y} - \hat{s}\|_\infty = \max\{\|p^l q^l - W^{l^\top} \hat{y}^l - \sum_{i \in \mathcal{D}^l} T^{i^\top} \hat{y}^i - \hat{s}^l\|_\infty : l \in \mathcal{L}_t, t = 1, \dots, t_f\}.$$

The contribution of a node $l \in \mathcal{T}$ to ξ_c is

$$\begin{aligned}\xi_c^l &= p^l q^l - W^{l\top} \hat{y}^l - \sum_{i \in \mathcal{D}^l} T^{i\top} \hat{y}^i - \hat{s}^l \\ &= p^l (q^l - q^{r(l)}) - (W^l - W^{r(l)})^\top \hat{y}^l - \sum_{i \in \mathcal{D}^l} (T^i - T^{r(i)})^\top \hat{y}^i \\ &= p^l (q^l - q^{r(l)}) - \frac{p^l}{p_R^{r(l)}} \left[(W^l - W^{r(l)})^\top y_R^{r(l)} + \sum_{i \in \mathcal{D}^l} (T^i - T^{r(i)})^\top \frac{\delta^i}{\delta_R^{r(i)}} y_R^{r(i)} \right],\end{aligned}$$

where the step from the first to the second line uses (16) and the next step uses (13) together with $p^i = p^l \delta^i$, $p_R^{r(i)} = p_R^{r(l)} \delta_R^{r(i)}$. Taking norms (all norms here are infinity norms) and using the partitioning defined in (9) we obtain

$$\begin{aligned}\|\xi_c^l\|_\infty &\leq \|q^l - q^{r(l)}\| + \|W^l - W^{r(l)}\| \|y_R^{r(l)}\| + \sum_{k \in \mathcal{D}_R^{r(l)}} \|y_R^k\| \sum_{i \in \mathcal{I}^k \cap \mathcal{D}^l \setminus \{k\}} \|T^i - T^k\| \frac{\delta^i}{\delta_R^k} \\ &\leq \|q^l - q^{r(l)}\| + \|W^l - W^{r(l)}\| \|y_R^{r(l)}\| + \sum_{k \in \mathcal{D}_R^{r(l)}} \|y_R^k\| \varepsilon \sum_{i \in \mathcal{I}^k \cap \mathcal{D}^l \setminus \{k\}} \frac{\delta^i}{\delta_R^k} \\ &\leq \left(\|q^l - q^{r(l)}\| + \|W^l - W^{r(l)}\| + \sum_{k \in \mathcal{D}_R^{r(l)}} \left(1 - \frac{\delta^k}{\delta_R^k}\right) \varepsilon \right) B \leq \varepsilon B |\mathcal{T}_R|. \quad \square\end{aligned}$$

The following result combines the findings of Lemmas 5 and 7.

Theorem 8. *Let the assumptions of Lemma 7 be satisfied and*

$$\varepsilon B^2 \max\{\|A^\top (AA^\top)^{-1}\|_\infty, |\mathcal{T}_R|\} \leq \frac{1}{2} \beta (1 + \sqrt{n}/\gamma)^{-1}.$$

Then the full Newton step (17) from the warm-start iterate is feasible and restores primal and dual feasibility.

Proof. Using the definition of $\tilde{\xi}_b$ from (19), the bounds (25), and Lemma 7, we get

$$\|\tilde{\xi}_b\|_\infty = \|X^{-1} A^\top (AA^\top)^{-1} \xi_b\|_\infty \leq \varepsilon B^2 \|A^\top (AA^\top)^{-1}\|_\infty \leq \frac{1}{2} \beta (1 + \sqrt{n}/\gamma)^{-1}.$$

In a similar way, we obtain

$$\|\tilde{\xi}_c\|_\infty = \|S^{-1} \xi_c\|_\infty \leq \varepsilon B^2 |\mathcal{T}_R| \leq \frac{1}{2} \beta (1 + \sqrt{n}/\gamma)^{-1}.$$

Now the result follows from Lemma 5. □

It is worth making a few remarks about these results. Theorem 8 implies that if we can choose the reduced scenario tree such that $\varepsilon = \max_i \{d(r(i), i)\}$ is small enough to satisfy the bound given in the Theorem, then the warm-start point constructed from the reduced scenario tree will be successful for the complete problem. Unfortunately we have only limited influence on ε . Indeed ε is the result of the variation of the problem data between the expanded and the

complete systems. However, we can reduce ε by having a denser reduced tree, although this would make the solution of the reduced problem more expensive.

While the analysis performed concerned a primal–dual interior point algorithm applied to a deterministic equivalent problem with node (rather than scenario) formulation, we expect that similar results could be adapted to a different interior point algorithm and problem formulation.

It is important to remember that these are theoretical bounds. There is a gap between theory and practice. In practice much larger infeasibilities $\|\tilde{\xi}_b\|$, $\|\tilde{\xi}_c\|$ can be absorbed. This is confirmed by our numerical results where even choosing just two scenarios in the reduced tree leads to a significant reduction in the number of interior point iterations required to solve the complete problem. We remark that this is problem dependent, and on other instances a larger number of scenarios in the reduced tree may be necessary.

6 Implementation and numerical results

We first implemented the strategy of generating a reduced tree and the corresponding warm-start iterate within the HOPDM [8] solver. We tested a series of publicly available stochastic problems in the SMPS format [3] coming from the POSTS collection available from: <http://users.iems.northwestern.edu/~jrberge/html/dholmes/post.html>.

It should be noted that we disabled HOPDM’s presolve in order to preserve the dimensions of the problems, and thus obtain sensible warm-start points.

While the analysis of Section 5 is very conservative in its estimates of the absorbable perturbations, in practice we noticed that the reduced-tree warm-start strategy is effective even with a much sparser tree than that suggested by the theory. In our experiments different choices for the reduced tree size have been explored, without any noticeable difference in the effectiveness of the warm-start strategy. In results presented below, the reduced tree was built with only two scenarios.

We solved the reduced problem with an optimality tolerance of 5.0×10^{-1} , while the optimality tolerance for the complete problem was set to 5.0×10^{-8} . Computations were performed on a Linux PC with a 3.0GHz Intel Pentium processor and 1GB of RAM. In Table 1 we report the dimensions of the problems in terms of the number of stages and scenarios for the complete tree, the number of iterations and the computing time (in seconds) with cold start and warm start. The latter includes the generation and solution of the reduced problem, and the construction of the warm-start iterate.

The problems solved show an overall good behaviour of our warm-start strategy, with time savings of up to 59% (for problem `pltexpA5_6`). The generation of the reduced tree and the solution of the corresponding problem (11) is generally fast, and becomes negligible as the problem sizes increase. However, for the smallest instances of our test set (`fxm2_6`, `fxm3_6` and `fxm4_6`), it is noticeable and consumes the savings produced by using an advanced iterate.

Problem data			Cold start		Warm start	
Name	Stages	Scens	Iters	Time	Iters	Time
fxm2_16	2	16	22	1.2	13	1.0
fxm3_6	3	36	30	1.5	17	1.3
fxm3_16	3	256	40	31.1	20	20.7
fxm4_6	4	216	30	8.2	22	8.3
fxm4_16	4	4096	41	218.3	27	182.6
pltexpA3_16	3	256	26	153.8	14	87.8
pltexpA4_6	4	216	36	55.8	16	27.5
pltexpA5_6	5	1296	81	772.0	30	311.5
storm27	2	27	41	95.4	22	53.2
storm125	2	125	73	107.3	36	69.1
storm1000	2	1000	107	1498.3	45	831.5

Table 1: Results obtained with HOPDM, 2 scenarios in the reduced tree.

6.1 Telecommunication problems

We implemented the same approach in OOPS [12, 13], where we were able to test larger problem instances. Since OOPS does not have features such as presolve and scaling, the accuracy requested in the solution has to be smaller. We set it to 5.0×10^{-4} which is sufficient for telecommunication applications. On the other hand, OOPS makes an effective use of its structure-exploiting capabilities allowing the solver to tackle large-scale problems and provides access to parallel computing techniques.

We applied our warm-start strategy to the capacity assignment problem with uncertain demand, a model relevant to the telecommunication industry [24]. The objective of this model is to find the optimal choice of capacities to be assigned to the links in the network in order to minimize unsatisfied customer demands. In our particular application we assume that the topology of the network and the sets of origin–destination pairs are given and are not going to change during the planning horizon.

We model this situation as a two-stage stochastic linear program with recourse. The general model has the following form:

$$\min_x E_d[f(x, d)] \quad \text{s.t.} \quad \sum_{l \in \mathcal{A}} c_l x_l \leq M, \quad x \geq 0,$$

where c_l and x_l are the cost and capacity of link $l \in \mathcal{A}$, respectively, and M is a bound on the budget. The objective here is to minimize the expected cost (conditional on the uncertain demand). This general model describes the first stage decision about the link capacities. The function $f(x, d)$ is defined in the following model, which describes the second stage decisions:

$$\begin{aligned} f(x, d) = \min & \sum_{k \in \mathcal{D}} (d_k - \sum_{p \in \mathcal{P}_k} z_p) \\ \text{s.t.} & \sum_{k \in \mathcal{D}} \sum_{p \in \mathcal{P}_k: l \in p} z_p \leq x_l \quad \forall l \in \mathcal{A} \\ & \sum_{p \in \mathcal{P}_k} z_p \leq d_k \quad \forall k \in \mathcal{D} \\ & z_p \geq 0, \end{aligned}$$

where d_k is the demand for the k -th origin–destination pair, \mathcal{P}_k is a given set of paths linking the k -th pair, and z_p is the flow on path p . While this problem has *relatively complete recourse*, that is, for any first-stage decision there is always a second-stage recourse that satisfies all constraints, our warm-start approach does not exploit this property. Indeed we do not attempt to construct a primal-feasible recourse decision for the incoming scenarios, rather we aim to construct an approximately primal-dual feasible and central point for the complete problem on the full tree.

To generate scenarios, we used the approach described in [24]. For each origin–destination pair k we need to have a demand estimate d_k , which can be determined from historic data or from an educated guess. The demand is assumed to be uniformly distributed around this estimate. Hence, the demand d_k^s for the k -th pair in scenario s is given by

$$d_k^s = (1 + \epsilon_k^s)d_k,$$

where ϵ_k^s is a random number generated in the interval $[-\varrho, \varrho]$. The value of $\varrho > 0$ determines the range in which we assume the demand to fluctuate. In our experiments we chose a value of $\varrho = 0.5$, thus allowing very large variations in the demand.

The relevant network characteristics of the problems solved are shown in Table 2, where we detail the size of the network, the number of demands considered, the overall number of paths and the average number of arcs in each path.

Name	Nodes	Arcs	Demands	Paths	Av.Length
mnx	12	50	66	189	2.6
jlg	26	84	264	697	5.6
mgntA	53	158	1378	3574	6.7
mgntB	70	210	2346	6137	6.4

Table 2: Characteristics of the telecommunication problems.

For a problem with N scenarios, the number of constraints and decision variables (including slacks) are

$$m = 1 + N \times (\#\mathcal{A} + \#\mathcal{D}), \quad \text{and} \quad n = 1 + \#\mathcal{A} + N \times (\#\mathcal{A} + \#\mathcal{D} + \#\mathcal{P}),$$

respectively, where $\#\mathcal{A}$ is the number of arcs, $\#\mathcal{D}$ the number of demands, and $\#\mathcal{P}$ the total number of paths.

In the second and third column of Table 3 we report the solution statistics for OOPS. Computations were performed on a Linux PC with 3.0GHz Intel Pentium processor and 2GB of RAM. In all cases the reduced tree was built with merely two scenarios. Therefore the computation time corresponding to the solution of the reduced problem (included in time reported in the table) was always negligible. The savings of warm start over cold start strategy vary between 40% and 80% in most cases.

We have also solved the smallest instances of these problems with Cplex 9.0 Barrier Solver. The problems `mnx-100`, `jlg-100`, `mgntA-100` and `mgntB-100` were solved in 1.1s, 7.1s, 4379.9s and 9030.4s, respectively. This means that Cplex was about 4 and 2 times faster than OOPS on `mnx-100` and `jlg-100` problems, respectively but it was about 28 times slower than OOPS on more difficult `mgntA-100` and `mgntB-100` problems.

Problem	Serial case				Parallel case			
	Cold start		Warm start		Cold start		Warm start	
	Iters	Time	Iters	Time	Iters	Time	Iters	Time
mnx-100	15	6.7	9	3.9	15	3.9	9	2.6
mnx-200	13	12.9	7	7.3	13	4.6	7	3.5
mnx-400	16	28.9	8	15.5	16	10.5	8	6.3
mnx-800	17	58.8	10	39.5	17	18.8	10	10.7
mnx-1600	19	131.1	10	68.8	19	50.3	10	31.4
jlg-100	21	38.3	6	15.5	21	11.0	6	6.1
jlg-200	45	164.9	17	39.5	45	49.9	17	20.7
jlg-400	44	255.2	18	103.1	43	83.2	19	39.7
jlg-800	27	353.4	10	152.9	29	130.5	10	50.1
jlg-1600	32	855.3	13	360.6	35	286.1	14	129.7
mgntA-100	28	260.0	14	156.2	28	76.9	14	51.6
mgntA-200	50	877.1	35	690.6	50	256.4	34	195.3
mgntA-400	40	1470.3	14	572.5	40	410.9	14	181.6
mgntB-100	23	511.1	14	318.0	23	137.5	14	103.9
mgntB-200	25	909.4	8	332.4	25	284.2	8	140.5
mgntB-400	29	2154.5	7	538.1	29	605.5	7	211.6

Table 3: Efficiency of the warm-start strategy in OOPS in the serial case (2 scenarios in the reduced tree) and in the parallel case (4 processors and 4 scenarios in the reduced tree).

In the fourth and fifth columns of Table 3 we report the parallel performance of OOPS on a cluster of four machines with a 3.0GHz Intel Pentium processor and 2GB of RAM each. In this case, we choose the size of the reduced tree to be equal to the number of processors employed for two complementary reasons. First, it is preferable to assign to OOPS a balanced number of blocks on each processor, so we needed to guarantee that each processor gets at least one block; second, we obtain a more refined starting solution at no additional computational cost. However the analysis of the parallel results collected in Table 3 indicates that the use of a slightly larger reduced tree does not translate into any noticeable improvement in the warm start runs as measured with the number of warm start iterations. Obviously, the solution times are reduced but this is the effect of using more processors.

6.2 Effectiveness with respect to the VSS

We considered the effectiveness of the warm-start strategy with respect to the value of stochastic solution (VSS) [4]. The VSS measures the improvement in objective function obtained by solving a stochastic problem over solving an expected value problem. Therefore, for low values of the VSS, it may not be worthwhile formulating and solving a stochastic problem; however, for higher values of the VSS, the stochastic solution yields measurably better decisions. The VSS can therefore be seen as a measure of how much new problem information is contained in the additional scenarios, and how far we expect the first-stage decisions for the complete tree differ from the ones obtained on the reduced tree.

In the context of the telecommunication problem of Section 6.1, we noticed that we could modify the VSS by considering different values of the budget M . For very small values of the budget,

the first-stage decisions are virtually independent of the stochasticity, as we would resort to buying the cheapest arcs for any value of the unknown demand. However, for larger values of the budget, stochasticity has an impact on the first-stage decisions, and therefore the stochastic measure grows.

In Figure 6 we present the results we obtained by setting different values of the budget in problems `mnx-200` and `j1g-200`. As we can see, for these problems there is no discernible relationship between the magnitude of the VSS and the success of the warm-start strategy.

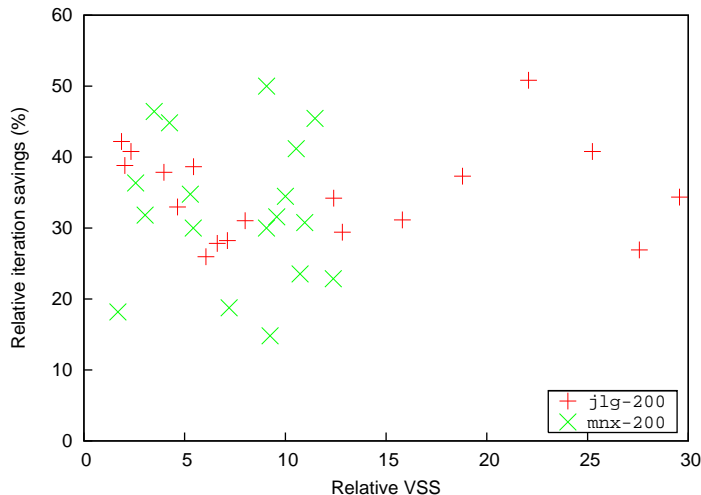


Figure 6: Plot of the relative savings in number of iterations against the relative VSS.

7 Conclusions

We introduced a technique that exploits the near-optimal solution to a stochastic linear program corresponding to a reduced scenario tree to warm-start a much larger problem that encompasses the complete scenario tree. Our way of reducing the dimension of the scenario tree was based on the assumption that we have no knowledge of the underlying stochastic process, and therefore we developed an ad-hoc measure of distance between the scenarios. We proposed to minimize the distance to a selection of representative scenarios; other possibilities can be devised, and may be the subject of future research. We observed that the iterate generated from the reduced problem provides an advanced starting point for the solution of the complete problem, in general resulting in a decrease of the number of iterations needed. As the computational cost of generating such an iterate is negligible, this produces consistent savings in computational time.

Acknowledgements: We thank one of the referees for pointing out a way to strengthen the statement of Theorem 6.

References

- [1] H. Y. BENSON AND D. F. SHANNO, *An exact primal-dual penalty method approach to warmstarting interior-point methods for linear programming*, Computational Optimization and Applications, 38 (2007), pp. 371–399.

- [2] J. R. BIRGE, *Decomposition and partitioning methods for multistage stochastic linear programs*, Operations Research, 33 (1985), pp. 989–1007.
- [3] J. R. BIRGE, M. A. H. DEMPSTER, H. I. GASSMANN, E. A. GUNN, A. J. KING, AND S. W. WALLACE, *A standard input format for multiperiod stochastic linear programs*, Committee on Algorithms Newsletter, 17 (1987), pp. 1–19.
- [4] J. R. BIRGE AND F. LOUVEAUX, *Introduction to stochastic programming*, Springer Series in Operations Research, New York, 1997.
- [5] R. E. BIXBY, *Solving real-world linear programs: a decade and more of progress*, Operations Research, 50 (2002), pp. 3–15.
- [6] M. COLOMBO AND J. GONDZIO, *Further development of multiple centrality correctors*, Computational Optimization and Applications, 41 (2008), pp. 277–305.
- [7] J. DUPAČOVÁ, N. GRÖWE-KUSKA, AND W. RÖMISCH, *Scenario reduction in stochastic programming*, Mathematical Programming, 95 (2003), pp. 493–511.
- [8] J. GONDZIO, *Multiple centrality corrections in a primal-dual method for linear programming*, Computational Optimization and Applications, 6 (1996), pp. 137–156.
- [9] ———, *Warm start of the primal-dual method applied in the cutting-plane scheme*, Mathematical Programming, 83 (1998), pp. 125–143.
- [10] J. GONDZIO AND A. GROTHEY, *Reoptimization with the primal-dual interior point method*, SIAM Journal on Optimization, 13 (2003), pp. 842–864.
- [11] ———, *A new unblocking technique to warmstart interior point methods based on sensitivity analysis*, Technical Report MS-06-005, School of Mathematics, The University of Edinburgh, December 2006. Accepted for publication in *SIAM Journal on Optimization*.
- [12] ———, *Solving non-linear portfolio optimization problems with the primal-dual interior point method*, European Journal of Operational Research, 181 (2007), pp. 1012–1029.
- [13] J. GONDZIO AND R. SARKISSIAN, *Parallel interior-point solver for structured linear programs*, Mathematical Programming, 96 (2003), pp. 561–584.
- [14] J. GONDZIO AND J.-P. VIAL, *Warm start and ε -subgradients in a cutting plane scheme for block-angular linear programs*, Computational Optimization and Applications, 14 (1999), pp. 17–36.
- [15] A. L. HIPOLITO, *A weighted least squares study of robustness in interior point linear programming*, Computational Optimization and Applications, 2 (1993), pp. 29–46.
- [16] K. HØYLAND, M. KAUT, AND S. W. WALLACE, *A heuristic for moment-matching scenario generation*, Computational Optimization and Applications, 24 (2003), pp. 169–185.
- [17] E. JOHN AND E. A. YILDIRIM, *Implementation of warm-start strategies in interior-point methods for linear programming in fixed dimensions*, Computational Optimization and Applications, 41 (2008), pp. 151–183.
- [18] P. KALL AND S. W. WALLACE, *Stochastic programming*, John Wiley and Sons, New York, 1994.

- [19] J. LINDEROTH AND S. J. WRIGHT, *Decomposition algorithms for stochastic programming on a computational grid*, Computational Optimization and Applications, 24 (2003), pp. 207–250.
- [20] S. MEHROTRA, *On the implementation of a primal-dual interior point method*, SIAM Journal on Optimization, 2 (1992), pp. 575–601.
- [21] J. E. MITCHELL, *Karmakar’s algorithm and combinatorial optimization problems*, PhD thesis, Cornell University, 1988.
- [22] J. E. MITCHELL AND M. J. TODD, *Solving combinatorial optimization problems using Karmarkar’s algorithm*, Mathematical Programming, 56 (1992), pp. 245–284.
- [23] J. M. MULVEY AND A. RUSZCZYŃSKI, *A new scenario decomposition method for large-scale stochastic optimization*, Operations Research, 43 (1995), pp. 477–490.
- [24] A. OUOROU, *Robust capacity assignment in telecommunications*, Computational Management Science, 3 (2006), pp. 285–305.
- [25] G. C. PFLUG, *Scenario tree generation for multiperiod financial optimization by optimal discretization*, Mathematical Programming, 89 (2001), pp. 251–271.
- [26] S. J. WRIGHT, *Primal-dual interior-point methods*, SIAM, Philadelphia, 1997.
- [27] E. A. YILDIRIM AND S. J. WRIGHT, *Warm-start strategies in interior-point methods for linear programming*, SIAM Journal on Optimization, 12 (2002), pp. 782–810.