International Journal of Education and Science Research Review
E-ISSN 2348-6457
Volume-9, Issue-1 Jan-Feb-2022
P-ISSN 2349-1817
www.ijesrr.org
Email- editor@ijesrr.org

WARM-START STRATEGIES FOR STOCHASTIC LINEAR PROGRAMS

Dr. Rajni
Assistant Professor Mathematics
F.L.T.M.S.B.P.GOVT.GIRLS COLLEGE
REWARI, HARYANA

Abstract

In the framework of stochastic programming, we provide an explanation of a method for producing a warm-start point for interior point methods. Because our method makes use of the structural information contained inside the stochastic problem, it is possible to interpret it as an initial point generator that exploits structural information. We first establish an improved starting point for the whole problem by solving a scaled-down version of the problem that corresponds to a smaller event tree, and then we utilise that answer to solve the original problem. When we create a reduced tree, we do it in a way that aims to extract the most relevant information from the scenario space while while maintaining a manageable level of dimensionality in the corresponding reduced deterministic counterpart. We arrive at requirements that the reduced tree has to fulfil in order to ensure a successful warm start of the entire issue, and we derive these conditions here. The implementation inside the OOPS and HOPDM interior point solvers demonstrates a number of significant benefits.

Keywords: *Warm-start, stochastic, programs*

Introduction

Modeling uncertainty through consideration of a number of potential outcomes in the future is what stochastic programming does (scenarios). The judgments that are made tend to be more reliable when there is more depth in the description. In order to do this, extremely large scenario trees and, as a direct result of this, very large scale deterministic equivalent matrices need to be created. It is anticipated that the requirement for finding solutions to very big issue instances would increase in proportion to the rising industry recognition of the advantages afforded by taking into account uncertainty for the purposes of planning.

When dealing with issues of increasing complexity, depending on interior point solvers offers more obvious advantages in terms of their applicability in practise. Problems of this size can be solved by exploiting the structure that is already present in the matrix. This leads to an additional advantage that comes from assigning the computational work to more than one processing unit through the parallelization of the linear algebra. However, general purpose solvers have a very difficult time solving very large scale problems. Structure-exploiting parallel solvers such as OOPS [3] perform exceptionally well in this context. In addition, structure-exploiting interior point approaches are applicable to not just linear programming issues, but also quadratic and nonlinear problems [2]. These methods may be utilised to solve linear programming

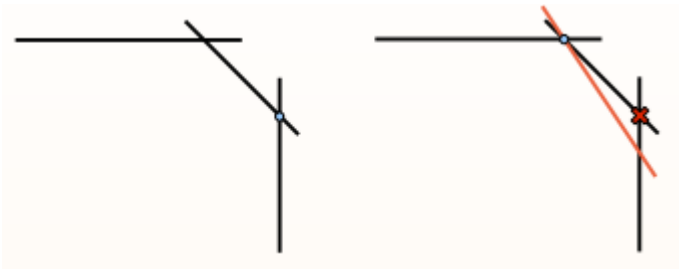
problems. Because there may be very little variation between scenarios in a large scenario tree, the large-scale problem may be able to provide a fine-grained solution to a problem that could have been solved more coarsely by employing a much smaller tree. In other words, the large-scale problem may be able to provide a solution to a problem that could have been solved more coarsely by employing This discovery provides support for a warm-start approach that may be utilised within the framework of interior point techniques. In order to acquire a warm-start solution, one must first solve the stochastic optimization problem for a reduced event tree. The dimension of the reduced event tree is significantly less than that of the whole event tree. In order to develop an advanced iteration for the whole formulation, the solution to the reduced issue is employed as a starting point. We present evidence that this unique technique of utilising the issue structure to generate an initial iteration gives a starting point that is superior to the one produced by a generic strategy in terms of centrality, feasibility, and proximity to optimality. We want to underline that the suggested warm-start method is not reliant in any way on the specifics of the linear algebra implementation that the solver chooses to use. The following is the structure of this paper: In the second section, we provide a measure of the distance between different scenarios and go through some of the fundamental ideas behind stochastic programming. In Section 3, we take a look at a few papers that discuss warm-start procedures for interior point methods. In the next section, "Section 4," we will discuss a way for creating the warm-start iterate and generating a reduced event tree. In Section 5, we do an analysis of the method and establish the boundaries that the reduced tree has to fulfil in order to ensure that the warm start is successful. In the next section, "Section 6," we will talk about the implementation as well as the numerical results. In the final part of this chapter, Section 7, we offer our conclusions.

Warm-start strategies

- A warm-start technique is one that begins working on the next problem by applying the solution that was just found to the previous one.
- Important if we are going to be working through a series of issues.
- Many times, we may anticipate that the answer to one problem will be somewhat near to the answer to the following problem.
- A more developed starting point might result in a shorter amount of computing time than beginning from scratch to solve the problem.

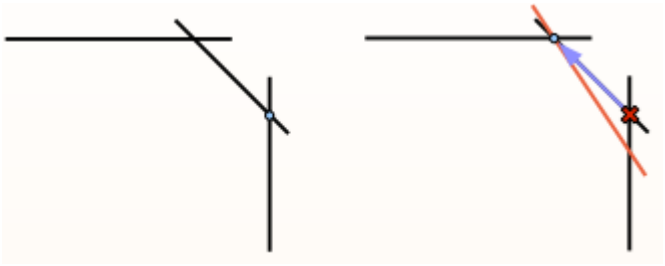
Warm-start with the simplex method

The solution of a problem is a vertex:



Warm-start with the simplex method

The solution of a problem is a vertex:



- The perfect place to begin with the customised instance
- Optimality was achieved in a short amount of iterations (if there are not too many changes in the problems)

Stochastic programming

Forecasts of future data are typically developed using econometric models that factor in data from the past. While this method is useful for identifying trends and the variations that accompany them, it is, unfortunately, inapplicable to products and services that have only recently been introduced because the relevant data may not be readily available. The use of deterministic models is regarded insufficient for decision making when the uncertainty cannot be readily forecasted. In these kinds of circumstances, the capability to characterize and model the unknowable characteristics becomes an absolute necessity for making sound decisions. The field of stochastic programming [4] investigates the approaches to modelling uncertainty and offers the relevant tools. The utility of stochastic programming stems from the fact that it provides a plethora of tools that make it possible to deal with uncertainty in a manner that is both realistic and actionable. Stochastic programming's growing popularity may be attributed to the fact that its underlying paradigm is ideally suited for modelling a wide variety of real-world issues that arise in a variety of contexts (finance, energy production and planning, telecommunications, logistics, etc). In stochastic programming, the uncertain environment is often estimated from historical data or conjectured according to required features, and this procedure is used to construct the stochastic process that is used to characterise the environment. In order to generate a description that can be processed by a computer, it is common practise to first obtain an approximation of the continuous process using a discrete distribution. In such a circumstance, the most prevalent methodologies [5] create a finite number of scenarios that offer an approximate description of the various outcomes. On average, the number of scenarios generated is rather big.

Deterministic equivalent formulation for stochastic programs

Recursion is essential to the natural formulation of a stochastic programming issue because it allows for an accurate description of the dynamics of the process being modelled. When we talk about having recourse, we imply that the decision variables will adjust to the various results that the random parameters provide at each and every time period. In the context of a planning strategy, the progression of uncertainty may be seen as an alternating series of decisions and random realisations that take place at various times throughout the planning process (stages).

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

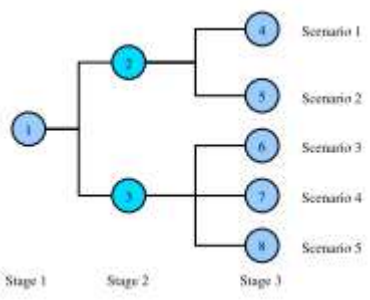


Figure 1: An event tree.

The instant at which it is determined whether or not a choice should be made after it has become apparent that the procedure was random. We assign a set of constraints, an objective function, and the conditional probability of visiting a node of the event tree based on the node that served as its parent in the stage before this one to each individual node of the tree. A scenario is represented by a path that travels from the event tree's root node to one of its leaf nodes. The likelihood of each scenario may be calculated by multiplying the conditional probabilities associated with travelling to each node along the path. We use a breadth-first ordering, which means that we start from the root node corresponding to the initial stage (stage 1) and end with leaf nodes corresponding to the final stage. Enumerating all of the nodes of the event tree is necessary for us to be able to express the deterministic equivalent of the multi-stage stochastic programming problem in node formulation. To do this, we need to enumerate all of the nodes of the event tree (stage tf). Let's designate the stages with the notation t = 1, 2, etc., and the set of nodes at stage t with the notation Lt. When we use the notation a(l), we are referring to the direct ancestor of the node l Lt, which is a node that is part of stage t 1. In this language, the decision variables are prefaced by the node number l, and a notation very similar to this is used for the accompanying matrix and vector blocks.

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

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

The deterministic equivalent formulation of the multi-stage problem has the following general form. 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 is as follows:

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

It is important to keep in mind that the probabilities used in the objective function of issue (1) are the unconditional route probabilities: The chance that a path travels via node l, denoted by the symbol p l, is equal to the product of the probabilities that are conditional on the path's existence. I for I along the route leading from the root to the node e l, in order to ensure that

$$p^i = \delta^i p^{a(i)}$$

During the development of the programme, if the event tree is traversed with depth-first ordering of the nodes, then the related constraint matrix will exhibit a nested dual block-angular structure. Figure 2 illustrates the two distinct structures that may have been created for the event tree shown in Figure 1 based on the particular ordering of the nodes. Despite the fact that the multiple orderings of the blocks inside the matrix are irrelevant for

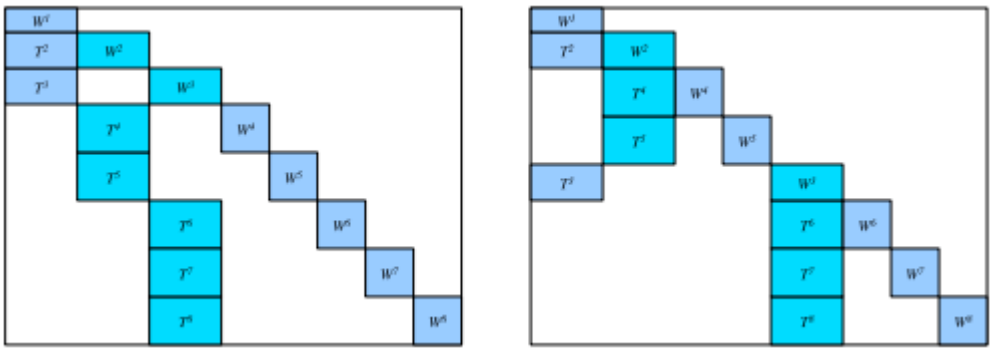


Figure 2: Deterministic equivalent corresponding to the event tree of Figure 1, with nodes listed in breadth-first order (left) and depth-first order (right).

The structure-exploiting programme OOPS [6] is able to make full use of the nested dual block-angular structure that is the consequence of the depth-first ordering in its internal object-oriented linear algebra representation. This allows general-purpose solvers to take full advantage of the structure. In the scholarly research on stochastic linear programming, several approaches to problem solving with solutions have been developed. These often rely, among other methods, on some kind of decomposition strategy [7]. Instead, in this piece of work, we take a look at the possibility of directly solving the deterministic equivalent issue using an interior point technique.

Warm-start with interior point methods

Take, for example, the issue of linear programming in its conventional form.

$$\min\ c^\top x \qquad \text{s.t. } Ax = b,\ x \geq 0, \qquad \text{.....(2)}$$

Where $A \in \mathcal{R}^{m \times n}$ is full rank $x, c \in \mathcal{R}^n$ and $b \in \mathcal{R}^m$. We shall refer to issue (2), which corresponds to the deterministic equivalent derived from a given event tree T, as the entire problem for the purposes of this work.

The non-negativity constraints are replaced by a logarithmic term in the context of interior point techniques, which results in the generation of the barrier problem.

$$\min\ c^\top x - \mu \sum_{i=1}^{n} \ln x_i \qquad \text{s.t. } Ax = b, \qquad \text{.....(3)}$$

Where μ > 0 is the barrier parameter the first-order optimality conditions (Karush-Kuhn Tucker conditions) corresponding to problem (3) can be expressed as

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

Whereas s Rn is the vector of dual slacks, X and S are diagonal matrices with elements xi and si respectively, and e Rn is a vector of ones, the expression is as follows: The solution of the perturbed Karush-Kuhn-Tucker conditions follows a unique path toward the optimal set as is lowered at each iteration. This path is commonly referred to as the centre path. Methods that follow a path via an inner point network [8] Make use of Newton's approach to look for a solution to the nonlinear system F(x, y, s) = 0 and take into consideration the Newton system.

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^\top & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} b - Ax \\ c - A^\top y - s \\ -XSe + \mu e \end{bmatrix} = \begin{bmatrix} \xi_b \\ \xi_c \\ \xi_\mu \end{bmatrix}, \quad \text{……..(4)}$$

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 techniques focus on maintaining the iterates in close proximity to the centre path in order to follow that path and get closer to the best possible answer. In this part of our investigation, we focus on the symmetrical area [6] surrounding the centre route.

$$\mathcal{N}_s(\gamma) = \{(x,y,s) : Ax = b,\ A^\top y + s = c,\ (x,s) > 0,\ \gamma\mu \le x_i s_i \le \mu/\gamma\}, \quad \text{……..(5)}$$

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

It is common practise to handle the challenge of locating a starting point by employing Mehrotra's starting point heuristic, which is recognised for its efficiency from a computing standpoint. The beginning point is determined using this heuristic by first shifting the point that was determined by solving two problems involving the least squares that aim to meet primal and dual constraints. This point is then moved inside the positive orthogonal. However, a large number of practical applications rely on the resolution of a series of issues that are closely connected to one another, but the specific examples differ in some way. This occurs inside algorithms that are sequential in their nature; also, it is a fairly regular occurrence in (mixed) integer programming, when the problems are addressed with some branching strategy, when new cuts are made, etc. In these kinds of circumstances, we anticipate that the answer to one instance will be quite similar to the answer to the following one. As a result, beginning the optimization of one problem from the solution of the problem that came before it should result in a reduction in the amount of computing work required to solve the perturbed instance. When utilised in conjunction with a simplex solver, warm-start strategies have a high rate of success (see, for example, [9]). Instead, when it comes to interior point methods, it is far more difficult to correctly implement them due to the reasons that we will discuss below. When solving a linear programming problem with the path-following interior point method, the optimal solution is located in close proximity to a vertex of the feasible polytope; alternatively, when there are multiple optimal solutions, the optimal solution is located in close proximity to the analytic centre of the optimal set of solutions. If the polytope is altered, the previously optimum solution may now be located a significant distance from the path that leads to the centre of the instance that has been disturbed. In addition, if an iterate goes too near to

a boundary before optimality is attained, an interior point algorithm could get stuck. Hipolito analysed such a case and shown that if the iterate is close to a boundary, the search direction may be parallel to the surrounding restrictions. He showed this by showing that if the iterate is close to a boundary.

Analysis of the warm-start iterate

In this part, we will investigate how the warm-start iterate that was constructed using the processes that were described earlier in this article fits the constraints that were stated by Gondzio and Grothey [10]. In contrast to what can be seen

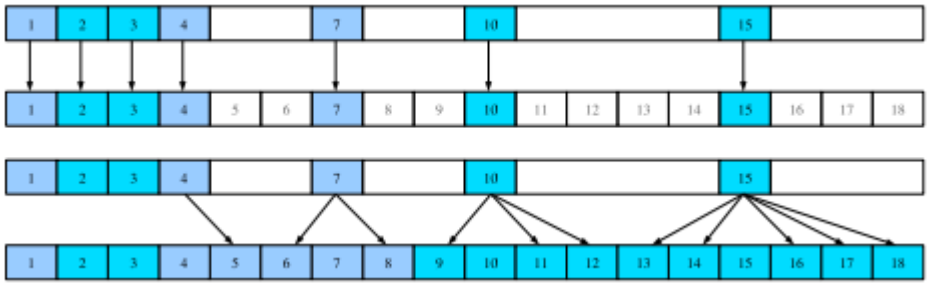


Figure 3: Generation of the warm-start iterate.

Given that the size of the reduced tree problem is, by definition, significantly lower than that of the full problem, it is assumed that the dimension of the problem shifts in both of these approaches, as well as in our own. However, similarly to what we did with the solution vector, we are able to expand the reduced problem to one that has the same dimension as the full problem (3) by replicating the blocks in the coefficient matrix and in the objective and right-hand side vectors, as shown in Figure 4. This allows us to expand the reduced problem to one that has the same dimension as the full problem. Creating the (artificially) enlarged problem is equivalent to this action.

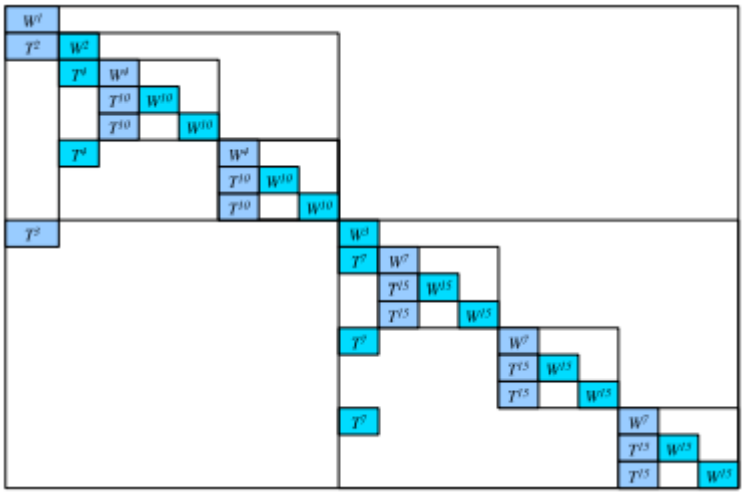


Figure 4: The expanded system for the complete event tree of Figure 3.

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

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 ˆ.

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 $(\hat{x}, \hat{y}, \hat{s}) \in \mathcal{N}_s(\hat{\gamma})$. the second part of the process is where we may leverage these iterates to get a head start on the entire challenge. Because the size of the issue does not change when moving from the extended to the entire problem, the methods that were described in [11] may be utilised to analyse the warm-start iterate.

The investigation starts out with a purely technical result.

Lemma Let l ∈ 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 following two findings demonstrate that the reduced tree solution may be used to the extended problem in order to obtain a point that is primal–dual viable and central to that problem.

Theorem If (xR, yR, sR) is primal and dual feasible for the reduced problem then the warm-start solution (x^, y^, s^) obtained from is primal and dual feasible for the expanded problem.

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

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

Now let's look at the possibility of dual feasibility. Assuming this to be true, the solution to the simplified issue meets the following dual constraints:

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

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 and Lemma 1, becomes?

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

So (yˆ, sˆ) satisfies the dual constraints in the expanded problem.

Theorem 3. If (xR, yR, sR) ∈ N R s (γ) for som $\gamma \in (0,1)$,

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 $j = 1,\ldots,n_R$, using (10) we have

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

The upper bound xˆ l j sˆ l j ≤ μ/ˆ (ργ) can be derived similarly.

Implementation and numerical results

Within the HOPDM solver, we began by putting into action the technique of constructing a reduced tree in conjunction with the associated warm start iterates. We examined a number of publicly accessible stochastic issues in the SMPS format. These problems originated from the POSTS collection, which can be accessed [13] It is important to highlight that in order to maintain the size of the problems and achieve appropriate warm-start points, we disabled the presolve feature of the HOPDM algorithm. However, in practise, we found that the reduced-tree warm-start strategy is effective even with a much sparser tree than what is suggested by the theory. This is in contrast to the results of the analysis presented in Section 5, which are very conservative in their estimates of the absorbable perturbations. During the course of our studies, a

variety of options for the smaller tree size were investigated; nevertheless, there was no discernible impact on the efficiency of the warm-start technique as a result. In the results that are provided below, the reduced tree was constructed using only two different possible outcomes. We solved the reduced issue using a tolerance for optimality of 5.0 x 101, whereas the optimality tolerance for the whole problem was set at 5.0 x 108. The computations were carried out using a personal computer running Linux and equipped with a 3.0GHz Intel Pentium processor and 1 gigabyte of random access memory (RAM). In Table 1, we present the dimensions of the issues in terms of the number of stages and scenarios for the whole tree, as well as the number of iterations and the amount of time it takes to compute the problem (in seconds) with a cold start and a warm start respectively. The latter entails the creation of the warm-start iteration in addition to the development and solution of the reduced issue. The issues that were resolved point to a generally positive performance of our warm-start method, which resulted in time savings of up to 59 percent (for problem pltexpA5 6) The production of the reduced tree and the solution of the related issue (11) are often quick processes, and the time it takes for either of these processes to complete becomes insignificant as the size of the problem increases. However, it is evident and eats up the savings provided by utilising an advanced iteration for the three smallest instances of our test set (fxm2 16, fxm3, and fxm4) 6.

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

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

Effectiveness with respect to the VSS

We investigated how well the warm-start strategy worked in comparison to the value of the stochastic solution (VSS) [12]. The VSS is a metric that evaluates how much of an improvement there is in the objective function when a stochastic issue is solved as opposed to an expected value problem. Since this is the case, formulating and solving a stochastic problem may not be worth the effort for low values of the VSS. On the other hand, when applied to greater values of the VSS, the stochastic solution results in judgments that are much improved. The VSS may therefore be viewed as a measure of how much new knowledge about the problem is contained in the extra scenarios, as well as how significantly we anticipate the first-stage judgments for the whole tree to differ from the ones reached on the reduced tree. Within the framework of the telecommunications issue presented in Section 6.1, we came to the realisation that we might adjust the VSS by taking into account a variety of alternative values for the budget M. The first-stage judgments are practically independent of the stochasticity for very small values of the budget since we would resort to buying the cheapest arcs for any value of the unknown demand. However, for larger values of the budget, the stochasticity plays a significant role. Stochasticity, on the other hand, has an effect on the decisions made in the first stage when it is applied to greater values of the budget; as a result, the stochastic

measure rises. We give the results that we achieved by using a variety of values for the budget in the issues mnx-200 and jlg-200 in Figure 5. These findings may be found below. As we can see, there is no discernable association between the severity of the VSS and the efficacy of the warm-start method for these difficulties. This is the case regardless of the fact that the warm-start technique was utilised.

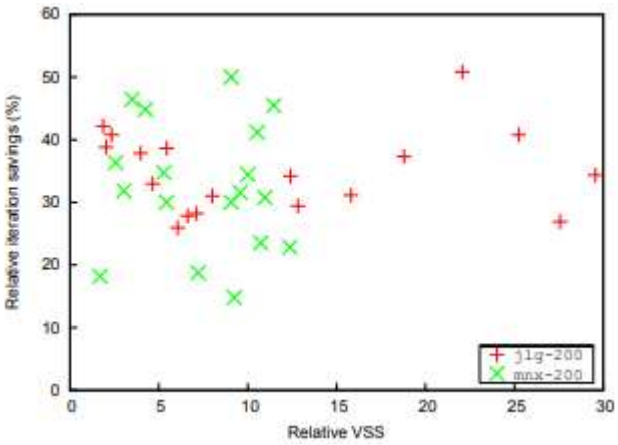


Figure 5: Plot of the relative savings in number of iterations against the relative VSS.

Conclusions

We came up with a method that makes use of the nearly optimum solution to a stochastic linear programme that is associated with a reduced scenario tree in order to get a head start on a significantly more difficult issue that involves the entire scenario tree. Our strategy for decreasing the size of the scenario tree involved the presumption that we are unaware of the stochastic process that lies beneath it. As a result, we devised a makeshift method for determining the amount of space that separates each of the possibilities. We recommended shortening the distance to a variety of sample case studies; however, alternative possibilities might be conceived of and may be the topic of more research in the future. We made the discovery that the iterate that was created from the reduced issue offers a more developed starting point for the solution of the full problem, which, in most cases, results in a reduction in the number of iterations that are required. This results in significant time savings in the computing process due to the fact that the computational cost of producing such an iteration is almost nothing.

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. Dupaco˘ va, ´ N. Growe-Kuska, ¨ and W. Romisch ¨ , 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.