Numerical Techniques in Applied Multistage Stochastic Programming

Karl Frauendorfer, Gido Haarbrücker

Institute of Operations Research, University of St. Gallen, Switzerland

Abstract

This contribution deals with the apparent difficulties when solving an optimization problem with random influences by the use of multistage stochastic linear programming. It names specific numerical solution techniques which are suitable for coping with the curse of dimensionality, with increasing scenario trees and associated large-scale LPs. The focus lies on classic decomposition methods which are natural candidates to apply parallelization techniques. In addition, an alternative approach is sketched where the replacement of optimization runs by optimality checks leads to an efficient handling of consecutive discretization steps given some structural requirements are fulfilled.

Keywords: Multistage Stochastic Linear Programming, Discretization, Large-Scale Linear Program, Numerical Techniques

1 Introduction

A huge number of operational and planning problems is characterized by sequences of decisions over time. As though this structure of acting over time – in the sense of taking optimal or, at least, “good” decisions w.r.t. some constraints – would not be complex enough, one often is additionally faced with a stochastic evolution of parts of the problem’s parameters: prices, sales, production costs or interest rates are only a few examples of relevant future information which is very rarely already known today. Whenever possible, the today’s and future decisions should take into account this uncertainty and respond to the realizations of the random variables involved.

Then, the resulting model for optimal decision making is a *multistage stochastic program* given the following assumptions are fulfilled:

- These distributions are unaffected by the decisions taken.
- Decisions have to be taken on past information only, i.e. decisions made today cannot depend on information received tomorrow or any day thereafter (so-called *nonanticipativity*).
- A finite number of decision stages is considered.

As to the third item, finitely many stages do not demand for a finite horizon of the problem to be formulated; for instance, the problem may in fact be characterized by an infinite planning horizon, but this circumstance has to be incorporated in the model by a finite number of stages.

It is possible that in some – perhaps most practical – situations a distribution law of the stochastic influences may not be known. In such cases, one can choose an approximating distribution to achieve results which normally are superior to a substitution of the random parameters by single best estimates or mean values. Anyway, at the end, one needs a concrete probability distribution (an a priori known one, a subjective idea, an approximating one or whatsoever) because otherwise one could just formulate the stochastic program without a chance to determine a solution. For this reason, in the following we will assume that the probability distributions of stochastic influences in all stages are known.

In general, it is the expected value of an interesting quantity which will be maximized or minimized subject to some constraints. Throughout this paper, we confine ourselves to multistage stochastic programs with fixed recourse for two reasons: first, these programs proved to be an adequate tool reflecting the complexity of real-life situations and, secondly, in the last decades a wide theory and powerful solution procedures have been developed. Furthermore, the objective function is assumed to be linear in the decision variables so that we are dealing with multistage stochastic *linear* programs (MSLP) with fixed recourse. For ease of simplicity, we take the *technology matrices* T^t to be known as well, but they can differ from stage to stage denoted by the superscript t .

When stochastic influence is given by continuous distributions, one of the most common – and in fact promising – approaches is to discretize the support of these distributions. Of course, although we concentrate on discretization approaches, we do not wish to conceal the existence of alternative methods: for instance, approximation of the original distributions can also be achieved by simpler – but still continuous ! – distributions (see e.g. Birge and Wallace [7], Birge and Wets [8], Wallace [34], and Birge and Qi [6]). However, discretization always leads to scenario trees depicting the stochastic evolution. On the one hand, this overcomes the obstacle of numerical integration in high dimensions corresponding to the random variables because an MSLP with a finite number of scenarios still has a *deterministic equivalent linear program*. On the other hand, such a MSLP in its extensive form mostly is given by a really large-scale linear program; normally, the latter does not appear accessible to manipulations such as *basis factorizations* for extreme or interior point methods (see e.g. Birge and Louveaux [4], Chapter 5, for factorization in the two-stage case). Up to a certain size, these large-scale LPs are solvable by *direct methods* like simplex-based or interior point methods with acceptable computational effort. When the size of an LP sets limits to the application of direct methods, promising *indirect* methods are given by methods which are based on *decompositions*, on some form of *Lagrangian relaxation*, and on uses of *separability*.

Anyway, the solutions based on scenarios and, hence, on approximating discrete probability measures are solutions of approximate programs; they can only serve as approximate solutions of the original ‘continuously distributed’ MSLP. Thus, one normally aims at a successive improvement of the approximate solution until an a priori approximation quality¹ is reached. This leads to a cycle of consecutive approximation steps ν which are – hopefully – tractable until the approximation quality desired is achieved.

To summarize, two aspects are crucial to “solve” an MSLP with continuous underlying distributions.

- (i) The discretization scheme is responsible for quality of the scenarios, and it determines the type of result one can get based on these scenarios.
Does one achieve probabilistic or deterministic bounds on the true² expected objective value ? And are the scenarios representative for the stochastic evolution of the uncertain components (in case of probabilistic bounds), or are they adequately determined to actually provide bounds, respectively ?
- (ii) Depending on methodological aspects of the approach utilized and/or the discretization scheme, one is forced to dispose of a more or less powerful (direct or indirect) solution method. In general, it holds: the higher the number of scenarios to be dealt with, or the better the demanded approxi-

¹Still to be defined!

²with regard to the original continuous joint distribution

mation quality, or the larger the single LPs to be solved in approximation step ν , the sooner one gets into trouble with one's solution method.

There are many ways to approach this two-sided medal: for instance, (1) use a large crude scenario sample in conjunction with a sophisticated – e.g. parallelized – method to solve the corresponding large-scale LP, or (2) select a few “good” scenarios so that one can cope with the arising LPs by the use of an ordinary ‘commercial’ direct method, or (3) something in between, e.g. embed an advanced sampling method into a well-trying decomposition method.

In order to treat more formally the alternate procedure consisting of observing random events and taking decisions, we state a mathematical formulation of a – slightly simplified – multistage stochastic program in the next section. This facilitates the introduction of some technical terms and helps to make the subject accessible to readers who are not familiar with multistage stochastic programming. Afterwards, Section 3 states informally well-known existing techniques (with a focus on decomposition) and deals with an alternative approach designed to handle sequences of approximating programs. The paper ends up with some summarizing conclusions.

2 Multistage Stochastic Linear Program with Fixed Recourse

Using a dynamic programming type of recursion, we can write an MSLP in the following form:

for $t = H, \dots, 2$ define backwards

$$\begin{aligned} Q^t(x^{t-1}, \boldsymbol{\xi}^t(\omega)) := & \min \quad \mathbf{c}^t(\omega)x^t(\omega) + E_{\boldsymbol{\xi}^{t+1}}[Q^{t+1}(x^t, \boldsymbol{\xi}^{t+1}(\omega))] \\ \text{s.t.} \quad & W^t x^t(\omega) = \mathbf{h}^t(\omega) - T^{t-1}x^{t-1} \\ & x^t(\omega) \geq 0, \end{aligned} \quad (1)$$

with terminal value function given by

$$Q^{H+1}(\cdot, \cdot) \equiv 0.$$

The interesting minimal objective value and the today's optimal decision can then be obtained by solving the problem

$$\begin{aligned} \min \quad & z(x^1) := c^1 x^1 + Q^2(x^1) \\ \text{s.t.} \quad & W^1 x^1 = h^1 \\ & x^1 \geq 0. \end{aligned} \quad (2)$$

In case of unboundedness below or infeasibility of the program (1) $Q^t(x^{t-1}, \boldsymbol{\xi}^t(\omega))$ is set to $-\infty$ or $+\infty$, respectively. The *expectation functional* $Q^2(\cdot)$ of the *value*

function $Q^2(\cdot, \cdot)$ in (2) is defined as

$$\mathcal{Q}^2(x^1) := E_{P^{\xi^2}}[Q^2(x^1, \cdot)] = \int_{\Xi^2} Q^2(x^1, \xi^2) dP^{\xi^2}(\xi^2) \quad (3)$$

$$= \int_{\Omega} Q^2(x^1, \xi^2(\omega)) dP(\omega) = E_P[Q^2(x^1, \xi^2)] \quad (4)$$

with

$$\begin{aligned} Q^2(x^1, \xi^2(\omega)) := \min \quad & \mathbf{c}^2(\omega)x^2(\omega) + \mathcal{Q}^3(x^2) \\ \text{s.t.} \quad & W^2x^2(\omega) = \mathbf{h}^2(\omega) - T^1x^1 \\ & x^2(\omega) \geq 0. \end{aligned} \quad (5)$$

Sometimes the expectation functionals $\mathcal{Q}^t$ are named *expected value functions*, too (see e.g. Kall and Wallace [23], p. 148). In the following, we refrain from a differentiation from 'normal' value functions; depending on the situation and notation it will become clear if it concerns the evaluation of an expectation or not. Occasionally, in literature the term *recourse function* is used instead of a value function; but the reader should be aware that sometimes *decision rules* or *policies* $\xi \mapsto x(\xi)$ are also named recourse functions (see e.g. Wets [35], p. 566, or Rockafellar and Wets [28], S. 171).

In (3), Ξ^2 denotes the support of ξ^2 ; concerning the definition of P , see below. Integrals in the form of $E_{P^{\xi^2}}[\cdot]$ are given by the sum of their positive and negative parts, with positive (negative) parts to be defined as $+\infty$ ($-\infty$) when the integral diverges or the integrand takes the value $+\infty$ ($-\infty$) on a set of strictly positive measure.³ $\mathbf{c}^1$ and $\mathbf{h}^1$ are known vectors out of $\mathbb{R}^{n_1}$ and $\mathbb{R}^{m_1}$, respectively. For all stages $t = 2, \dots, H$ the random vectors $\xi^t := (\mathbf{c}^t, \mathbf{h}^t)$ fulfill

$$\begin{aligned} \xi^t = (\mathbf{c}^t, \mathbf{h}^t) : (\Omega, \Sigma^t) &\longrightarrow (\mathbb{R}^{n_t+m_t}, \mathcal{B}^{n_t+m_t}) \\ \omega &\longmapsto \xi^t(\omega) = \begin{pmatrix} \mathbf{c}^t(\omega) \\ \mathbf{h}^t(\omega) \end{pmatrix} =: \begin{pmatrix} c^t \\ h^t \end{pmatrix} \end{aligned} \quad (6)$$

with $\Sigma^t \subseteq \Sigma^{t+1}$ σ -algebras relative to Ω ($t = 2, \dots, H-1$); herein, (6) stands for the $(\Sigma^t, \mathcal{B}^{n_t+m_t})$ -measurability of ξ^t .⁴ Each σ -algebra Σ^t consists of those events that are 'known' at time t . Let P be a known probability measure on the measurable space (Ω, Σ^H) . Because of the monotone increasing σ -algebras Σ^t – in the sense of set inclusion – the (Ω, Σ^t, P) are well-defined probability spaces for all $t = 2, \dots, H$.

³Cases of simultaneous occurrence of $-\infty$ and $+\infty$ are handled by the convention $(-\infty) + \infty = +\infty$ (see e.g. Wets [37], p. 312), i.e. infeasibility in one subproblem cannot be compensated by unboundedness below in another subproblem.

⁴ $\mathcal{B}^n$ ($n \in \mathbb{N}$) designates the Borel σ -algebra on $\mathbb{R}^n$, i.e. the σ -algebra which is generated by all open subsets of $\mathbb{R}^n$.

(The distribution of ξ^t is assumed to be independent of past realizations. If this assumption is given up,⁵ then the problem formulation must contain further information in the form of observed realizations of the random vectors up to time t . In order to keep the problems on a tractable level and to avoid an overloaded notation, herein we will assume stochastic independence between stages.

As to the decisions, we additionally demand the $(\Sigma^t, \mathcal{B}^{n_t})$ -measurability of x^t for all t ; this ensures the nonanticipativity of the decisions to be taken. x^{t-1} denotes the states of the system and $E_{P^{\xi^t}}[\cdot]$ the expected value w.r.t. the induced probability measure P^{ξ^t} ($t = 2, \dots, H$). The value functions $Q^t(\cdot, \cdot)$ and expectation functionals $\mathcal{Q}^t(\cdot)$ of stages $t > 2$ are defined analogously to $Q^2(\cdot, \cdot)$ and $\mathcal{Q}^2(\cdot)$, respectively; Ξ^t designates the support of ξ^t .

Since the most common way to treat such a multistage stochastic linear program is to solve (approximating) discretized versions of the MSLP under consideration, the next section starts with the formulation of a deterministic equivalent program given a finite number of random outcomes. Afterwards, various types of solution techniques are stated rather verbally with a focus on well-known decomposition methods and an alternative approach developed recently.

3 Solution Techniques

When deriving approximate problems by discretization, the multistage problem structure obviously results in the following effect: an increase of the number of decision stages leads to an exponential increase in problem size even if the number of realizations in each stage remains constant. To underline this effect, one could think of the following example: in a stochastic programming problem with 10 stages, 10 random variables per stage and only 10 random outcomes per stochastic dimension (i.e. a rather small sample of only 100 realizations of the 10-dimensional distribution on each stage, which can hardly obtain a sufficient estimation quality), one would have to deal with 10^{20} scenarios – a problem size which obviously is not tractable on a today's computer platform within acceptable time although the high-dimensional integration is reduced to a finite summation,

⁵Take note of the fact that relaxing the stochastic independence cannot be done without any compensation but only in favour of some weaker kind of regularity condition like, e.g., linear dependency (see Wets [36], pp. 202f, 205).

as is shown in the following formulation of problem (1) - (2) in a discretized form:

$$\begin{aligned}
\min \quad & c' x_o + \sum_{s=1}^{S_1} p_1^s (c_1^s)' x_1^s + \sum_{s=1}^{S_2} p_2^s (c_2^s)' x_2^s + \dots + \sum_{s=1}^{S_H} p_H^s (c_H^s)' x_H^s \\
\text{s.t.} \quad & A x_o = b \\
& T_1^s x_o + W_1 x_1^s = h_1^s \quad (s = 1, \dots, S_1) \\
& T_2^s x_1^{v(2,s)} + W_2 x_2^s = h_2^s \quad (s = 1, \dots, S_2) \\
& \ddots \quad \ddots \quad \vdots \\
& T_H^s x_{H-1}^{v(H,s)} + W_H x_H^s = h_H^s \quad (s = 1, \dots, S_H) \\
& x_o \geq 0, \quad x_t^s \geq 0, \quad 1 \leq t \leq H, \quad 1 \leq s \leq S_t
\end{aligned} \tag{7}$$

Herein, $x_o := x_o^1$ and x_t^s ($t = 1, \dots, H, s = 1, \dots, S_t$) indicate the today's decision and the decision in the future node (t, s) , respectively.⁶ $v(t, s)$ designates the state index of the antecedent node according to the underlying scenario tree, i.e. it holds $v(t, s) \in \{1, \dots, S_{t-1}\}$ for all t, s .⁷ But even for lower-dimensional problems with less stages, random variables and realizations, the above-mentioned *curse of dimensionality* has to be overcome by powerful solution techniques. As already said, up to a certain size of the DEP (7), commercial or academic direct methods can be applied successfully without taking advantage of the special problem structure, i.e. just using well-tried *sparse simplex methods* or interior point methods.

An alternative way to cope with larger DEPs is to exploit the special structure of multistage stochastic programs in the implementation of interior point algorithms.⁸ For instance, using such an approach Yang and Zenios [39] were able to solve test problems with up to 2.6 million constraints and more than 18 million variables. In general, interior point algorithms turned out to be very effective to solve large-scale stochastic programming problems, likewise whether implemented to directly exploit problem structures or when applied to solve *master- and subproblems* within a decomposition method.

Because of the road of success of decomposition methods and their widespread usage, Subsection 3.1 briefly deals with the idea and realization of these approaches. Afterwards, Subsection 3.2 presents another kind of 'decomposing' a

⁶The stage index $t = 1, \dots, H$ is now written as subscript in order to simplify the notation.

⁷with s used as state index, S_t indicating the total number of nodes (states) on stage t , and p_t^s denoting the probability of node (t, s) .

⁸Since a detailed treatise of this subject is out of the scope of this paper, the reader is referred to e.g. Birge and Qi [5], Yessup, Yang and Zenios [22], or Czyzyk, Fourer and Mehrotra [9].

large-scale program into smaller subproblems within the handling of subsequent approximation steps.

3.1 Decomposition Methods

Considering a two-stage linear recourse problem with a finite number of second-stage realizations, then one can always form the two-stage extensive form. Its primal (dual) formulation has a block (block angular) structure, so that it seems natural to take advantage of this structure by performing a Benders [1] decomposition of the primal or a Dantzig-Wolfe [12] decomposition of the dual, respectively. Benders' method has been extended in stochastic programming to take care of feasibility aspects and is known as *L-shaped method* by Van Slyke and Wets [33]: actually, it is a cutting plane technique which builds an outer linearization of the recourse function; the idea consists of sequentially adding *feasibility cuts* which determine the effective domain $\{x^1 \mid Q^2(x^1) < +\infty\}$, and *optimality cuts* which are linear approximations to Q^2 on its effective domain. For a detailed scheme of the L-shaped algorithm and enhancements like *multicut versions* or *bunching* of realizations, the interested reader is referred to Birge and Louveaux [4], Chapter 5. Furthermore, a variety of variants and extensions exist: e.g., adding a nonlinear regularized term to the objective (*regularized decomposition* by Ruszczyński [31]), or using sequential bounding approximations within the L-shaped method (see Birge [2] or Birge and Louveaux [4], pp. 296 ff).

As was already mentioned before, a potential disadvantage of sampling approaches is that some computational effort might be wasted on optimizing when the approximation is not accurate; an approach to avoid such difficulties is to combine sampling with another algorithm without complete optimization. An obvious candidate is to embed sampling into L-shaped method. Two well-known examples of such a combined approach are given by (1) an importance sampling based approach by Dantzig and Glynn [10] and Dantzig and Infanger [11], and (2) the *stochastic decomposition* approach by ⁹. The first approach uses importance sampling to reduce variance when deriving each cut based on a large sample. In the second one, a single sample is employed to derive many cuts; the latter are less accurate and eventually drop away with increasing iteration number.

Generalizations of the classic two-stage L-shaped method to the multistage case have been done by Louveaux [24] who performed this extension for multistage quadratic problems, and by Birge [3] for the multistage linear case. The latter is also known as *nested L-shaped* or *nested Benders decomposition*. The basic idea is – analogously to the two-stage case – to place feasibility and optimality cuts; but now, optimality cuts are placed on the period t value function $Q^{t+1}(x^t)$ for all t , and by feasibility cuts one achieves solutions x^t that have a feasible completion in all descendent scenarios. Again, the cuts represent consecutive

⁹see Higle and Sen [21]

(outer) linearizations of the value functions Q^{t+1} ; convergence to an optimal solution in a finite number of steps is due to the polyhedral convex structure of the Q^{t+1} . However, despite of many similar features between the nested and the two-stage L-shaped method, there are some differences and peculiarities; we will only mention two of them:

- (i) In the nested decomposition procedure, many alternative strategies are possible with regard to determining the next subproblem to solve, in particular on which stage t a subproblem will be solved. A so-called *fast-forward-fast-back*¹⁰ sequencing protocol proposed by Wittrock [38] seems to be superior to other sequencing protocols like “fast-back” or “fast-forward” (see Gassmann [17] and Morton [26]).
- (ii) The two-stage method always produces cuts which are supports of the value function Q^2 if the subproblem is solved to optimality. In contrast to this, the nested method working with the above-mentioned “fast-forward-fast-back” protocol does not automatically generate a true support, i.e. cuts may lie strictly below the value function to be approximated.

Similar to the two-stage case, significant enhancement is possible for the nested method by efficient bunching (or *sifting*); see Birge [3] or Gassmann [17].

Decomposition methods which are based on an *Augmented Lagrangian* can be found in Ruszczyński [32], Rosa and Ruszczyński [29] or Mulvey and Ruszczyński [27]. For parallelization aspects concerning decomposition see e.g. Ruszczyński [30].

It should be noticed, that a multistage stochastic program does not automatically demand for the application of the nested method. Even in the multistage case one can utilize the classic two-stage Benders decomposition as is done in Gondzio and Kouwenberg [19]: they divide 6-period problems into a “first stage” problem from today up to time $H_{sub} - 1$ and a set of “second stage” subproblems from time H_{sub} up to the horizon H . Applying the two-stage Benders decomposition which employs the interior point solver HOPDM¹¹ to optimize the master problem and the second stage subproblems, they were able to solve a problem with 25 million columns and 12.5 million rows in less than 5 hours – admittedly by massive parallelization using up to 13 processors.

On the one hand, this impressive result points out that the hope to solve truly large-scale deterministic equivalent linear programs became a certainty. On the other hand, this does not at all close the gap between demand and reality: the question still remains in how far this really helps to “solve” an original MSLP with a continuous underlying distribution. For instance, in the above-mentioned program with 25 million columns the corresponding scenario tree consisted of

¹⁰i.e. the algorithm proceeds from stage t in a specific direction – either $t + 1$ (“forward”) or $t - 1$ (“back”) – until it can no longer proceed in that direction

¹¹see Gondzio [18]

$13^6 = 4'826'809$ scenarios, namely a tree with 7 decision stages and 13 branches per node. Taking into account that 6 random coefficients are included in this special ALM model, the amount of 13 realizations does not appear that large. Of course, sophisticated methods can be employed to construct specific event trees which reproduce some properties of the underlying distributions; e.g. *tree-fitting* can be utilized to fit the first few moments of the underlying distributions (e.g. mean, standard deviation, skewness and kurtosis – depending on the number of realizations). But this only increases the confidence in the quality of the approximating discrete probability measure in the sense that one is “near” to the original distribution; one can at most expect that this helps to improve the quality of the solutions. It does not really help to quantify a solution quality in terms of probabilistic bounds (confidence intervals) or deterministic bounds on the true overall objective value. Statements on probabilistic bounds can only be given by statistical reasoning based on the sample size, and an evaluation of deterministic bounds requires special discrete support points. The next subsection deals with a specific discretization scheme – which has turned out to be an adequate technique given some structural properties of the multistage stochastic program are fulfilled – in conjunction with a numerical procedure to handle sequences of approximating deterministic equivalent linear programs: the choice of the discrete support points guarantees to evaluate upper and lower deterministic bounds on the true overall objective value, and the numerical procedure can cope effectively with the large-scale LPs which arise when one tries to successively improve these bounds.

3.2 Progressive Refinement Approach and Barycentric Approximation

Consider the general MSLP (1) - (2) stated in Section 2 with an underlying continuous joint distribution. When the problem has some – not very restrictive and rather technical – properties¹², then the value functions $Q^t(x^{t-1}, \xi^t(\omega))$ in all stages are *saddle functions* with a convex behaviour w.r.t. x^{t-1} and h^t , and a concave behaviour w.r.t. c^t . This saddle structure is exploited by the so-called *barycentric approximation scheme* in the following way: upper and lower approximations of the value functions at each stage are constructed leading to upper and lower scenario trees and associated large-scale LPs. The optimal objective values of these ‘upper’ and ‘lower’ LPs provide deterministic bounds on the true overall objective value of the original MSLP with its continuous distribution. Furthermore, they enable to state and localize the inaccuracy (in the sense of the distance between the upper and lower approximate objective values) w.r.t. the current approximations.

Usually, one’s aim is to improve the quality of the approximations built up so far which is done by successive refinements of the support of the underlying

¹²For a detailed treatise see e.g. Frauendorfer [13][14][15]

distributions; in doing so, these supports are more and more discretized which leads to more ramified scenario trees and gives a better depiction of the future stochastic evolution. On balance, the procedure of successively raising the number of discretization points is inevitably linked to sequences of ever increasing¹³ large-scale LPs which all have to be solved in order to track the development of the accuracy achieved.

Consider a scenario tree representing a deterministic equivalent program after having carried out a sequence of ν refinements of the support of the underlying probability distributions (where we confine ourselves to refinements in the root node only, i.e. partitioning always takes place w.r.t. the support of second-stage distributions). This refinement is restricted to the root node for two reasons: First, this kind of refinement results in a largest possible increase of the number of scenarios and thus represents the most challenging case since the corresponding LPs enlarge to the same extent. Secondly, this immense growth of the scenario tree stands for a very exhaustive incorporation of potential dynamics of the uncertain quantities and can be assumed to result in a strong improvement of the approximation accuracy.¹⁴ The upper part of Fig. 1 gives an exemplary image of such a refined scenario tree when the unrefined tree consisted of an ordinary binary tree. Having carried out a further refinement step $\nu + 1$ – in the sense that the chosen part of the support will be split – the subtree contained in the upper framed box will vanish and has to be replaced by new subtrees representing the new discretization points. If, for ease of simplicity, we think of a simple split of the partial support into two pieces, then the new parts are given by the two lower dashed boxes. Take note of the fact that, naturally, one can obtain new subtrees which are of exactly the same form as the vanishing part (for instance when each of the two new partial supports is represented by the same number of discretization points). We will come back to this fact when dealing with the need to solve these new subproblems on the operational level by means of a commercial LP solver. For every refinement step ν the scenario tree achieved is associated with a linear program LP_1^ν where the subscript 1 indicates the stage of the root node and the superscript ν the number of refinements carried out so far. Considering now the second-stage problems $LP_{2,s}^\nu$ (with $s = 1, \dots, S^\nu$ both numbering the nodes on the second stage from left to right and reflecting the different states of the system on that stage, i.e. the specific discretization points of the support of ξ^2), the next refinement step $\nu + 1$ leads to the following situation in our specific example of Fig. 1:

- The overall linear program LP_1^ν is replaced by $LP_1^{\nu+1}$. The new number of second-stage nodes is given by $S^{\nu+1} := S^\nu + 2$.

¹³At least, if no *scenario aggregation* procedures are applied or “important” scenarios are selected.

¹⁴E.g., as to the barycentric approximation scheme, this latter reasoning can be found in detail in Marohn [25], Chapter 5.

Figure 1: Changing parts of a scenario tree per refinement

- The two old second-stage subproblems, say LP_{2,s^*}^ν and LP_{2,s^*+1}^ν , are replaced by $LP_{2,s^*}^{\nu+1}, LP_{2,s^*+1}^{\nu+1}, LP_{2,s^*+2}^{\nu+1}$ and $LP_{2,s^*+3}^{\nu+1}$.
- All the other old second-stage subproblems $LP_{2,s}^\nu, s \in \{1, \dots, S^\nu\} \setminus \{s^*, s^* + 1\}$, remain unchanged and form the rest of the subproblems $LP_{2,s}^{\nu+1}$.

Usually, when employing whatever solver to optimize the large-scale LPs LP_1^ν , the following straightforward procedure is hardly suitable for coping the refinement cycle:

1. Build up LP_1^0 and convey problem to the solver. Solve LP_1^0 and get optimal solution $x^{1,0}$. Set $\nu = 1$.
2. Carry out refinement step ν . Build up LP_1^ν and convey problem to the solver. Solve LP_1^ν and get optimal solution $x^{1,\nu}$. Set $\nu = \nu + 1$.
3. IF $\nu < \text{Max}^{15}$ and $x^{1,\nu}$ does not fulfill a certain prechosen stopping-rule GOTO 2. ELSE STOP.

Naturally, this approach can be modified in the sense that the deterministic equivalent program is solved after every k^{th} ($k > 1$) refinement only. However, when the main interest is to track the improvement of single refinement steps or to stop as soon as the stopping criterion is fulfilled, one cannot avoid solving the large-scale LP associated with each refinement. Having in mind the so-called *curse of dimensionality*, the bottleneck of this approach obviously is given by solving these expanded large-scale LPs.

¹⁵Max indicating the maximal number of refinement steps

It is this computational burden which can be overcome by the *progressive refinement approach* (PRA), which reduces the numerical effort by trying to verify that the formerly obtained optimal solution is still optimal after the very last refinement step. In order to give a brief description of the course of events within the PRA, we will state a form of a meta-algorithm (i.e. only some modules are given with details hiding behind black-boxes termed, e.g., 'Evaluate' or 'Update'):

Progressive Refinement Approach

1. Set $\nu = 0$.
2. Build up LP_1^ν and convey problem to the solver. Solve LP_1^ν and get optimal solution $x^{1,\nu}$.
IF $\nu = \text{Max}$ or $x^{1,\nu}$ fulfills a certain prechosen stopping-rule STOP.
3. Solve all $LP_{2,s}^\nu$ with $x^{1,\nu}$ fixed in the RHS. Evaluate (sub-)gradient g^ν of z at $x^{1,\nu}$.
4. Set $\nu = \nu + 1$.
Carry out refinement step ν . Build up new subproblems $LP_{2,s}^\nu$ and convey subproblems to the solver. Solve subproblems.
5. Update last evaluated (sub-)gradient $g^{\nu-1}$ to the current one g^ν . Check optimality, i.e. try to show that $-g^\nu \in \mathcal{N}_{X^1}(x^{1,\nu-1})$.¹⁶
IF positive: Set $x^{1,\nu} = x^{1,\nu-1}$.
IF $\nu < \text{Max}$ and $x^{1,\nu}$ does not fulfill a certain prechosen stopping-rule GOTO 4.
ELSE STOP.
ELSE (i.e. negative) GOTO 2.

It should be noticed that the optimality check in step 5 utilizes an only *sufficient* optimality condition, i.e. a negative optimality check does not mean that the optimal first-stage decision changes from refinement step $\nu \rightarrow \nu + 1$. Hence, a negative optimality check can cause a wasted effort to carry through steps 2 and 3, only to find out that the optimal first-stage decision did not change at all. Fortunately, when applying the PRA to two different multistage problem formulations (coming from the area of running financial products), the empirical results showed that this last-mentioned case occurs very seldom, if ever.

The refinement method utilized must necessarily possess some rather heuristic properties: with the ongoing refinement process, the replaced parts of the scenario tree have to become smaller and smaller relative to the whole scenario tree; additionally, all parts besides the replaced part have to remain totally unchanged

¹⁶ $\mathcal{N}_{X^1}(x^{1,\nu-1})$ denotes the normal cone to the feasibility region X^1 of x^1 at the previously achieved optimal solution $x^{1,\nu-1}$.

(except the node probabilities of all second-stage nodes which are allowed to be changed). Otherwise the PRA cannot be expected to save much numerical effort and, hence, valuable computation time. The barycentric discretization scheme fits these requirements.

Altogether, the necessary handling of sequences of “refined” LPs – according to a barycentrically discretized support – could be shown to be manageable by the PRA with acceptable effort, at least with regard to two specific practical problems of running fixed-income products: Depending on the problem characteristics (number of stages and random variables, lower or upper approximation, etc.) up to several hundreds of partitions of the second-stage support can be carried out within an elapsed CPU time of half an hour.¹⁷ Quite contrary to the ‘traditional’ approach to solve every refined LP_1' as a whole: there, only a few refinement steps are possible within a comparable period of time.

The main advantage is given by a very small number of new subproblems $LP_{2,s}'$ which additionally have the same problem dimensions, so that within the simplex algorithm an effective start from an advanced basis can be employed. Because of the (mostly) small number of subproblems to be solved in a refinement step – and because of the hot-start possibility keeping the simplex algorithm time for the subproblems in an order of magnitude of some seconds –, parallelization is not so essential for a successful handling of the refinement cycle. Of course, to solve the subproblems in parallel could be implemented without further ado. However, we assume that a real acceleration of the computation will only be achieved in case of a negative optimality check (i.e. when the solution formerly achieved could not be verified to be still optimal after the very last refinement step) because then all subproblems arisen so far have to be solved again;¹⁸ but the latter situation was only rarely observed in the two problem statements considered. Comprehensive numerical results with regard to CPU time and a more detailed presentation of the practical problems (funding conventional Swiss mortgages and investing savings account deposits) can be found in Haarbrücker [20], and Frauendorfer and Haarbrücker [16].

4 Summary

A variety of suitable methodologies, numerical techniques and optimization tools exists to deal with multistage stochastic programs. For instance, deterministic equivalent programs – corresponding to approximating or originally discrete probability measures – can be solved with well-tried direct methods (commercial or academic solvers) or sophisticated indirect methods (decomposition, stochastic decomposition, Lagrangian-based approaches). In order to achieve such approx-

¹⁷even on small Unix workstation with a moderate memory of 256 MB RAM

¹⁸regardless of the difficulty that in those cases the (presumably very large) overall linear program LP_1' has to be solved, too

imating discrete probability measures and associated scenario trees, a multitude of discretization procedures can be applied: sampling (Monte Carlo – possibly in conjunction with variance-reducing techniques, EVPI-based importance sampling, and many others), selecting some kind of “good” scenarios” providing deterministic bounds (e.g. barycentric approximation, second-order scenario approximation), generating representative scenarios, and so on. In some cases, sampling/discretization procedures can be embedded in or at least combined with the above-mentioned direct or indirect methods which often leads to efficient numerical techniques. Another way to decrease computation time is given by techniques which aim at smaller scenario trees, like scenario aggregation or scenario reduction techniques.

As has been already mentioned, when one utilizes such a sophisticated method of a piece or combines some good modules in an efficient way, it is quite possible, for instance, to solve very large-scale deterministic equivalent programs with up to 25 million columns and nearly 13 million rows in a parallelized manner in a few hours, or to handle sequences of hundreds of successively discretized programs.

On balance, two aspects play – and will play! – a key role to cope successfully with MSLPs which model practical problem situations (i.e. rather high- than low-dimensional problems): to keep the gap between the original and the approximate distributions as close as possible, and in the meantime to keep an eye on the numerical solvability of the corresponding large-scale LPs.

References

- [1] BENDERS, J. Partitioning methods for solving mixed variables programming problems. *Numerische Mathematik* 4 (1962), 238–252.
- [2] BIRGE, J.R. *Using sequential approximations in the L-shaped and generalized programming algorithms for stochastic linear programs*. Technical Report 83-12, Department of Industrial and Operations Engineering, University of Michigan (Ann Arbor, MI), 1983.
- [3] BIRGE, J.R. Decomposition and partitioning methods for multistage stochastic linear programs. *Operations Research* 33 (1985), 989–1007.
- [4] BIRGE, J.R., LOUVEAUX, F. *Introduction to Stochastic Programming*. Springer-Verlag, New York, 1997.
- [5] BIRGE, J.R., QI, L. Computing block-angular Karmarkar projections with applications to stochastic programming. *Management Science* 34 (1988), 1472–1479.
- [6] BIRGE, J.R., QI, L. Continuous approximation schemes for stochastic programs. *Annals of Operations Research* 56 (1995), 15–38.
- [7] BIRGE, J.R., WALLACE, S.W. A separable piecewise linear upper bound for stochastic linear programs. *SIAM Journal on Control and Optimization* 26 (1988), 725–739.
- [8] BIRGE, J.R., WETS, R.J.-B. Sublinear upper bounds for stochastic programs with recourse. *Mathematical Programming* 43 (1989), 131–149.
- [9] CZYZYK, J., FOURER, R., MEHROTRA, S. Using a massively parallel processor to solve large sparse linear programs by an interior point method. *SIAM Journal on Scientific Computing* 19 (1998), 553–565.
- [10] DANTZIG, G.B., GLYNN, P. Parallel processors for planning under uncertainty. *Annals of Operations Research* 22 (1990), 1–21.
- [11] DANTZIG, G.B., INFANGER, G. Large-scale stochastic linear programs – Importance sampling and Benders decomposition. In *Computational and applied mathematics, I. Algorithms and theory, Sel. Rev. Pap. IMACS 13th World Congr., Dublin/Irel. 1991* (1992), pp. 111–120.
- [12] DANTZIG, G.B., WOLFE, P. The decomposition principle for linear programs. *Operations Research* 8 (1960), 101–111.

- [13] FRAUENDORFER, K. *Stochastic Two-Stage Programming*. Lecture Notes in Economics and Mathematical Systems, No. 392. Springer-Verlag, Berlin, 1992.
- [14] FRAUENDORFER, K. Multistage stochastic programming: Error analysis for the convex case. *Z. Oper. Res.* 39:1 (1994), 93–122.
- [15] FRAUENDORFER, K. Barycentric scenario trees in convex multistage stochastic programming. *Mathematical Programming* 75:2 (1996), 277–294.
- [16] FRAUENDORFER, K., HAARBRÜCKER, G. Solving Sequences of Refined Multistage Stochastic Linear Programs. Submitted for appearance in *Annals of Operations Research*.
- [17] GASSMANN, H. MSLIP, a computer code for the multistage stochastic linear programming problem. *Mathematical Programming* 47 (1990), 407–423.
- [18] GONDZIO, J. HOPDM (version 2.12) - A fast LP solver based on a primal-dual interior point method. *European Journal of Operational Research* 85, 221–225 (1995).
- [19] GONDZIO, J., KOUWENBERG, R. *High Performance Computing for Asset Liability Management*. Preprint MS-99-004, Department of Mathematics & Statistics, The University of Edinburgh, UK, 1999.
- [20] HAARBRÜCKER, G. *Sequentielle Optimierung verfeinerter Approximationen in der mehrstufigen stochastischen linearen Programmierung*. Doctoral thesis No. 2410, University of St. Gallen, 2000.
- [21] HIGLE, J., SEN, S. Stochastic decomposition: an algorithm for two stage linear programs with recourse. *Mathematics of Operations Research* 16 (1991), 650–669.
- [22] JESSUP, E.R., YANG, D., ZENIOS, S.A. Parallel factorization of structured matrices arising in stochastic programming. *SIAM Journal on Optimization* 4 (1994), 833–846.
- [23] KALL, P., WALLACE, S.W. *Stochastic Programming*. Wiley, Chichester, 1994.
- [24] LOUVEAUX, F. A solution method for multistage stochastic programs with recourse with application to an energy investment problem. *Operations Research* 28 (1980), 889–902.
- [25] MAROHN, C. *Stochastische mehrstufige lineare Programmierung im Asset & Liability Management*. Bank- und finanzwirtschaftliche Forschungen, Bd. 282. Paul Haupt Verlag, Bern, 1998.

- [26] MORTON, D.P. *An enhanced decomposition algorithm for multistage stochastic hydroelectric scheduling*. Technical Report NPSOR-94-001, Department of Operations Research, Naval Postgraduate School (Monterey, CA), 1994.
- [27] MULVEY, J.M., RUSZCZYŃSKI, A. A new scenario decomposition method for large-scale stochastic optimization. *Operations Research* 43:3 (1995), 333–353.
- [28] ROCKAFELLAR, R.T., WETS, R.J.-B. Nonanticipativity and $\mathcal{L}^1$ -martingales in stochastic optimization problems. *Mathematical Programming Study* 6 (1976), 170–187.
- [29] ROSA, C., RUSZCZYŃSKI, A. *On augmented Lagrangian decomposition methods for multistage stochastic programs*. Working Paper WP-94-125, IIASA International Institute for Applied Systems Analysis (Laxenburg, Austria), 1994.
- [30] RUSZCZYŃSKI, A. Parallel decomposition of multistage stochastic programming problems. *Mathematical Programming* 58 (1993), 201–228.
- [31] RUSZCZYŃSKI, A. A regularized decomposition method for minimizing a sum of polyhedral functions. *Mathematical Programming* 35 (1986), 309–333.
- [32] RUSZCZYŃSKI, A. An augmented Lagrangian decomposition method for block diagonal linear programming problems. *O. R. Letts.* 8 (1989), 287–294.
- [33] VAN SLYKE, R.M., WETS, R.J.-B. L-shaped linear programs with application to optimal control and stochastic programming. *SIAM J. Appl. Math.* 17 (1969), 638–663.
- [34] WALLACE, S.W. Solving stochastic programs with network recourse. *Networks* 16 (1986), 295–317.
- [35] WETS, R. J.-B. Stochastic Programming: Solution techniques and approximation schemes. In *Mathematical Programming: The state-of-the-art 1982* (1983), Bachem, A., Grötschel, M., Korte, B., Ed., Springer-Verlag, Berlin, pp. 566–603.
- [36] WETS, R.J.-B. Stochastic Programs with Recourse: A Basic Theorem for Multistage Problems. *Z. Wahrscheinlichkeitstheorie verw. Geb.* 21 (1972), 201–206.
- [37] WETS, R.J.-B. Stochastic programs with fixed recourse: The equivalent deterministic program. *SIAM Rev.* 16 (1974), 309–339.

- [38] WITTROCK, R.J. *Advances in a nested decomposition algorithm for solving staircase linear programs*. Technical Report SOL 83-2, Systems Optimization Laboratory, Stanford University (Stanford, CA), 1983.
- [39] YANG, D., ZENIOS, S.A. A scalable parallel interior point algorithm for stochastic linear programming and robust optimization. *Computational Optimization and Applications* 7:1 (1997), 143–158.