Mathematical Programming manuscript No.
(will be inserted by the editor)

Obtaining Lower Bounds from the Progressive Hedging Algorithm for Stochastic Mixed-Integer Programs

Dinakar Gade · Gabriel Hackebeil ·
Sarah M. Ryan · Jean-Paul Watson ·
Roger J-B Wets · David L. Woodruff

Received: date / Accepted: date

Abstract We present a method for computing lower bounds in the Progressive Hedging Algorithm (PHA) for two-stage and multi-stage stochastic mixed-integer programs. Computing lower bounds in the PHA allows one to assess the quality of the solutions generated by the algorithm contemporaneously. The lower bounds can be computed in any iteration of the algorithm by using dual prices that are calculated during execution of the standard PHA. We show that the best possible lower bound obtained using dual prices is as tight as the lower bound obtained using the Dual Decomposition method. We report computational results on stochastic unit commitment and stochastic server location problem instances, and explore the relationship between key PHA parameters and the quality of the resulting lower bounds.

This research was sponsored in part by the US Department of Energy's ARPA-e Green Energy Network Integration (GENI) program, and by the Department of Energy's Office of Science, Advanced Scientific Computing Research program. Thanks to Ge Guo for assistance with the numerical results. Sandia is a multi-program laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94-AL85000.

Dinakar Gade
Sabre Holdings
Southlake, TX 76092

Jean-Paul Watson
Sandia National Laboratories, Albuquerque, NM 87185

Gabriel Hackebeil
Texas A&M University, College Station, TX 77843

Sarah M. Ryan
Iowa State University, Ames, IA, 50011

Roger J-B Wets, David L. Woodruff
University of California Davis, Davis, CA 95616

Keywords Stochastic mixed-integer programming · Decomposition algorithms · Lower bounding

1 Introduction

Stochastic (mixed-) integer programs arise in a variety of situations in which discrete decisions combine with uncertainty in the data. Examples have been reported in the literature for some time, and include server location [23], batch sizing [18], electricity generation unit commitment [32], supply chain design [29], network interdiction [5,12,14], and many others [33]. The general combinatorial, NP-hard nature of integer and mixed-integer problems makes them difficult to solve even when all the data are known, but special structure may allow for easier solution. A common approach to representing uncertainty in data is to formulate a finite number of discrete scenarios for the values of uncertain parameters together with associated probabilities. Methods for obtaining scenarios do not concern us here, but often take the form of sampling from, or approximating, some stochastic process [8,15,25]. Decisions are classified into two or more stages according to which parameter values are assumed to be known to the decision-maker when the decisions must be made. Those decisions that can be delayed until some parameter values are revealed are (1) modeled as scenario-dependent, (2) required to satisfy constraints using that scenario's data, and (3) incur scenario-dependent costs. Implementability (or non-anticipativity) constraints are introduced to require that decisions not depend on data not yet revealed. When these model components are combined with an objective to minimize expected cost (where "cost" may include some measure of risk), the resulting extensive form of the stochastic program becomes a very large mixed-integer program in which the underlying structure of the deterministic combinatorial problem has been obscured.

The progressive hedging algorithm (PHA) has emerged as an effective method for solving multi-stage stochastic programs, particularly those with discrete decision variables in every stage. The PHA mitigates the computational difficulty associated with large problem instances by decomposing the extensive form according to scenario, and iteratively solving penalized versions of the sub-problems to gradually enforce implementability. Solving individual scenario problems separately may allow a solver to exploit any special combinatorial structure that may be present. The PHA is especially easy to implement in applications, such as the unit commitment problem we address in Section 5.3, where sophisticated computational infrastructure already exists for solving the deterministic version of the problem. In each iteration of the PHA, an aggregated solution that satisfies the implementability constraints is formed and penalties are applied in the next iteration based on deviations from that solution. In this primal-dual method, the penalties are based on estimates of the dual prices of the implementability constraints, and are updated in each iteration. While convergence to a globally optimal solution is not guaranteed in the case of mixed-integer problems, computational studies have shown that the PHA can find high-quality solutions within a reasonable numbers of iterations [35]. Moreover, the time expended for each iteration can be dramatically reduced by a very straightforward parallelization. Further, the PHA applies equally well to multi-stage stochastic programs with discrete decision variables in any stage or combination of stages.

A limitation in applying the PHA to stochastic mixed-integer programs is the historical lack of information it provides regarding solution quality relative to the optimal objective function value. In other words, without lower bound information, PHA has served as a heuristic, albeit a high-quality one. In contrast, solution methods that use branch-and-bound [1] or branch-and-price [18] rely on lower bounds on the optimal cost to form termination criteria as well as to eliminate regions of the solution space from consideration. Thus, they provide a built-in upper bound on the deviation of the incumbent solution's cost from global optimality.

In this paper, we correct this deficiency of the PHA in the mixed-integer case, and report a lower bound result. We use the estimates of the dual prices of the implementability (non-anticipativity) constraints to compute a lower bound on the optimal objective function value in any iteration of the PHA so that upper and lower bounds can be provided simultaneously. The bound computation at each iteration requires approximately the same effort as executing a standard iteration of the PHA. We also show that in theory, the lower bound from the PHA can be as tight as the lower bound from the Lagrangian dual problem, which is used in the dual decomposition method [3]. We empirically study the convergence of lower bounds from the PHA and use them to evaluate the quality of the primal solutions obtained in two different problem domains. The empirical results expose how the tradeoff between solution time and solution quality can be managed by appropriate parameterization of the PHA.

The remainder of this paper is organized as follows. We begin in Section 2 by developing our notation for stochastic mixed-integer programs and formally describing the PHA algorithm. Our lower bounding results are developed in Section 3 for the two-stage SMIP case, and are subsequently extended to the multi-stage case in Section 4. We empirically assess the quality of the PHA lower bounds in Section 5, and analyze the relationship between PHA parameters and bound quality. We then conclude in Section 6 with a summary of our results.

2 Preliminaries

We begin by considering a simple two-stage stochastic mixed-integer program of the form

$$\min \quad c^\top x + \mathbb{E}[f(x, \tilde{\xi})] \tag{1}$$

$$\text{s.t. } Ax \geq b \tag{2}$$

$$x \in \mathbb{Z}_+^{p_1} \times \mathbb{R}^{n_1 - p_1} \tag{3}$$

where $\tilde{\xi}$ is a random vector defined on a probability space $(\Xi, \mathcal{A}, \mathcal{P})$ and for a particular realization ξ of $\tilde{\xi}$, $f(x, \xi)$ is defined as:

$$f(x,\xi) = \min g(\xi)^\top y \tag{4}$$
$$\text{s.t. } Wy \geq r(\xi) - T(\xi)x \tag{5}$$
$$y \in \mathbb{Z}_+^{p_2} \times \mathbb{R}^{n_2 - p_2}. \tag{6}$$

Here, $c \in \mathbb{R}^{n_1}$, $A \in \mathbb{R}^{m_1 \times n_1}$, $b \in \mathbb{R}^{m_1}$, $g \in \mathbb{R}^{n_2}$, $W \in \mathbb{R}^{m_2 \times n_2}$, $T \in \mathbb{R}^{m_2 \times n_1}$ and $r \in \mathbb{R}^{m_2}$ comprise the data of the stochastic mixed-integer program. In two-stage stochastic programming, the decision maker chooses values for the first-stage decisions x before uncertainty is revealed and then takes recourse actions in response to a particular realization of the random vector $\tilde{\xi}$. The objective (1) is to minimize the sum of first-stage cost and the expectation of the second-stage costs. The first-stage decisions must satisfy the constraint set defined by (2). Constraint (3) enforces the mixed-integer restrictions on the first-stage variables. The second-stage decisions are subject to a cost $g(\xi)$ and are restricted by Constraint (5). First-stage decisions constrain second-stage decisions through the matrix $T(\xi)$. Constraints (6) enforce the mixed-integer requirements on the second-stage variables. In general, the expectation can be of a utility function and can include risk measures.

Stochastic programs of the form (1)-(6) are, in general, infinite dimensional optimization problems. To cope with this difficulty, one usually constructs an approximation of the problem by considering only finitely many realizations ξ of $\tilde{\xi}$. In this paper, we assume that the random vector $\tilde{\xi}$ has a finite support in Ξ, and restrict Ξ to denote the set of realizations of $\tilde{\xi}$ with corresponding probabilities p_ξ. With this assumption, one can express the expectation in (1) as a weighted sum, and write a large-scale deterministic mixed-integer programming formulation of the stochastic program called its *extensive form* (EF), as follows:

$$\min \quad c^\top x + \sum_{\xi \in \Xi} p_\xi g(\xi)^\top y(\xi) \tag{7}$$
$$\text{s.t. } Ax \geq b \tag{8}$$
$$Wy(\xi) \geq r(\xi) - T(\xi)x(\xi), \qquad \forall \xi \in \Xi \tag{9}$$
$$x \in \mathbb{Z}_+^{p_1} \times \mathbb{R}^{n_1 - p_1} \tag{10}$$
$$y(\xi) \in \mathbb{Z}_+^{p_2} \times \mathbb{R}^{n_2 - p_2} \qquad \forall \xi \in \Xi \tag{11}$$

The condition that the first-stage decision variables x must not depend on a particular realization of $\tilde{\xi}$ is implicit in the formulation (7)-(11). This condition was explicitly stated as a constraint in [36]. Writing the constraints explicitly leads to the so-called *scenario* formulation of the stochastic program:

$$\min \quad \sum_{\xi \in \Xi} p_\xi \left[c^\top x(\xi) + g(\xi)^\top y(\xi) \right] \tag{12}$$

$$\text{s.t. } Ax(\xi) \geq b \tag{13}$$

$$Wy(\xi) \geq r(\xi) - T(\xi)x(\xi) \qquad \forall \xi \in \Xi \tag{14}$$

$$p_\xi x(\xi) - p_\xi \hat{x} = 0 \qquad \forall \xi \in \Xi \tag{15}$$

$$\hat{x}, x(\xi) \in \mathbb{Z}_+^{p_1} \times \mathbb{R}^{n_1 - p_1} \qquad \forall \xi \in \Xi \tag{16}$$

$$y(\xi) \in \mathbb{Z}_+^{p_2} \times \mathbb{R}^{n_2 - p_2} \qquad \forall \xi \in \Xi \tag{17}$$

In the formulation (12)-(17), which we call the extensive form of the scenario formulation (EFS), copies of the first-stage variables are created for each realization or *scenario* of $\tilde{\xi}$. In addition, the EFS includes constraints (15), which are known as *non-anticipativity* or implementability constraints. Non-anticipativity constraints in a two-stage stochastic program stipulate that in all feasible solutions, the first-stage decisions are not allowed to depend on the scenario.

Solving the EFS directly (e.g., using a commercial solver such as CPLEX or Gurobi) is difficult for most practical problems, as large numbers of scenarios can yield extremely large-scale mixed-integer programs. However, the scenario formulation without the complicating non-anticipativity constraints (15) has a well-known block diagonal structure that decomposes the problem by block or scenario. Decomposing stochastic programs thus allows one to manage problem complexity. The progressive hedging algorithm (PHA) due to Rockafellar and Wets [26] is a decomposition algorithm that operates by decomposing a stochastic program by scenarios, and then coordinates a search for a $\hat{x}$ that satisfies (15). The PHA is related to other decomposition algorithms, e.g., alternating direction methods [2]. For $\xi \in \Xi$, let

$$X(\xi) := \{x \in \mathbb{Z}_+^{p_1} \times \mathbb{R}^{n_1 - p_1}, y \in \mathbb{Z}_+^{p_2} \times \mathbb{R}^{n_2 - p_2} : Ax \geq b, Wy \geq r(\xi) - T(\xi)x\}.$$

The statement of the PHA for two-stage stochastic mixed integer programs (SMIP) is then given in Algorithm 1.

The PHA is initialized by solving the individual scenario problems (Step 1). Each iteration of the PHA involves an *aggregation* operation (Step 3), which corresponds to a projection of the individual scenario solutions onto the subspace of non-anticipative policies [26]. The dual prices $w^\nu(\xi)$ are then updated (Step 4), using the sole external parameter associated with the basic PHA: ρ. The decomposition step of each PHA iteration (Step 5) involves solving scenario problems whose first-stage costs have been perturbed by the dual prices. Further, the objective function in this step is modified to include a proximal term that measures the deviation of the scenario solution from the aggregated first-stage policy $\hat{x}^\nu$. In practical applications, the test for convergence in Step 6 of the algorithm requires only convergence to within a tolerance for non-integer variables.

Algorithm 1 The Progressive Hedging Algorithm for Two-Stage SMIPs

1: **Initialization**: Let $\nu \leftarrow 0$ and $w^\nu(\xi) \leftarrow 0, \forall \xi \in \Xi$. For each $\xi \in \Xi$, compute:

$$(x^{\nu+1}(\xi), y^{\nu+1}(\xi)) \in \underset{(x,y)\in X(\xi)}{\arg\min}\ c^\top x + g(\xi)^\top y$$

2: **Iteration Update**: $\nu \leftarrow \nu + 1$

3: **Aggregation**: $\hat{x}^\nu \leftarrow \sum_{\xi\in\Xi} p_\xi x^\nu(\xi)$

4: **Price Update**: $w^\nu(\xi) \leftarrow w^{\nu-1}(\xi) + \rho(x^\nu(\xi) - \hat{x}^\nu)$

5: **Decomposition** : For each $\xi \in \Xi$, compute:

$$(x^{\nu+1}(\xi), y^{\nu+1}(\xi)) \in$$
$$\arg\min_{(x,y)\in X(\xi)} \left\{c^\top x + g(\xi)^\top y + w^\nu(\xi)^\top x + \frac{\rho}{2}||x - \hat{x}^\nu||^2\right\}$$

6: If all scenario solutions $x(\xi)$ are equal, stop. Else, go to step 2.

3 Mixed-Integer Lower Bounds from Progressive Hedging

We now show that the dual prices of the non-anticipativity constraints in two-stage stochastic MIPs define implicit lower bounds. We additionally demonstrate an equivalence between the best lower bounds obtained by the PHA and the lower bounds from the dual decomposition algorithm [3]. Finally, we prove that our results hold when the PHA proceeds in the context of bundles of scenarios in the course of decomposition.

3.1 Computing Lower Bounds

We now state our lower bounding result for the PHA. Let z^* denote the optimal objective function value of the SMIP defined by (12)-(17). In the following, we assume that the SMIP is feasible and has an optimal solution with $-\infty < z^* < +\infty$, and $X(\xi) \neq \emptyset$, $\forall \xi \in \Xi$. The following result shows that the dual prices $w(\xi), \xi \in \Xi$, define *implicit* lower bounds on z^*.

Proposition 1 *Let* $w = (w(\xi))_{\xi\in\Xi}$*, where* $w(\xi) \in \mathbb{R}^{n_1}$ *satisfy* $\sum_{\xi\in\Xi} p_\xi w(\xi) = 0$ *(component-wise). Let*

$$D_\xi(w(\xi)) := \min_{(x,y)\in X(\xi)} \left(c^\top x + g(\xi)^\top y + w(\xi)^\top x\right). \tag{18}$$

Then $D(w) := \sum_{\xi\in\Xi} p_\xi D_\xi(w(\xi)) \leq z^*$.

Proof : Let $(\hat{\bar{x}}, \{(\bar{x}(\xi), \bar{y}(\xi)), \forall \xi \in \Xi\})$ be an optimal solution to the SMIP defined by (12)-(17). Feasibility implies $(\bar{x}(\xi), \bar{y}(\xi)) \in X(\xi)$ for each $\xi \in \Xi$. Thus:

$$D_\xi(w(\xi)) \leq c^\top \bar{x}(\xi) + g(\xi)^\top \bar{y}(\xi) + w(\xi)^\top \bar{x}(\xi).$$

Then

$$\begin{aligned} D(w) &\leq \sum_{\xi\in\Xi} p_\xi \left(c^\top \bar{x}(\xi) + g(\xi)^\top \bar{y}(\xi) + w(\xi)^\top \bar{x}(\xi)\right) \\ &= \sum_{\xi\in\Xi} p_\xi \left(c^\top \hat{\bar{x}} + g(\xi)^\top \bar{y}(\xi)\right) + \sum_{\xi\in\Xi} p_\xi w(\xi)^\top \hat{\bar{x}} \\ &= \sum_{\xi\in\Xi} p_\xi (c^\top \hat{\bar{x}} + g(\xi)^\top \bar{y}(\xi)) = z^*. \end{aligned}$$

The next-to-last equality follows from the assumption that $\sum_{\xi\in\Xi} p_\xi w(\xi) = 0$. □

In every iteration of the PHA, the price update rule maintains the condition that $\sum_{\xi\in\Xi} p_\xi w^\nu(\xi) = 0$. Indeed, this is true for $\nu = 1$ since $w^1(\xi) = \rho(x^1(\xi) - \sum_{\xi\in\Xi} p_\xi x^1(\xi))$ and thus $\sum_{\xi\in\Xi} p_\xi w^1(\xi) = 0$. By induction, it is straightforward to see that $\sum_{\xi\in\Xi} p_\xi w^\nu(\xi) = 0$ for all ν. Proposition 1 demonstrates that a lower bound on z^* may be computed in *any* iteration of the PHA by solving an optimization problem that decomposes by scenario. We further observe that the scenario sub-problems are nearly identical in structure to those solved by the standard PHA, with the exception that the quadratic proximal terms are absent. This observation has significant practical implications for efficiently implementing lower bounding with PHA, including the availability of warm starts when solving the lower bounding scenario sub-problems.

In summary, our result demonstrates that the dual prices define implicit lower bounds for the PHA. Note that the non-anticipativity constraints (15) define a subspace $\mathcal{N}$ and the optimality conditions in the convex case [26] require that the dual prices lie in the subspace orthogonal to $\mathcal{N}$, i.e., the requirement $\sum_{\xi\in\Xi} p_\xi w(\xi) = 0$ can be interpreted as "dual feasibility" constraints for the primal constraint (15). Note that even in the SMIP case, the PHA maintains this requirement on dual variables.

3.2 Lower Bound Convergence

We next consider the ordinary Lagrangian for the SMIP defined by (12)-(17), which is obtained by dualizing the non-anticipativity constraints (15) using multipliers $\lambda(\xi)$. Let U be the feasible set defined by (13), (14), (16), and (17). For $u = (\hat{x}, (x(\xi), y(\xi))_{\xi\in\Xi}) \in U$ we define:

$$L(u, \lambda) := \sum_{\xi\in\Xi} p_\xi (c^\top x(\xi) + g(\xi)^\top y(\xi) + \lambda(\xi)^\top x(\xi) - \lambda(\xi)^\top \hat{x}).$$

The corresponding Lagrangian relaxation is given by:

$$F(\lambda) = \min_{u\in U} L(u, \lambda)$$

and its dual problem is given by:

$$z_{LD} := \sup_{\lambda} F(\lambda). \tag{19}$$

It is well known that the value of the Lagrangian dual problem in the mixed-integer case is equal to the optimal objective function value of a certain "primal" problem [3,10,22]. We summarize this result as follows:

Theorem 1

$$z_{LD} = \min \left\{ \sum_{\xi \in \Xi} p_\xi [c^\top x(\xi) + g(\xi)^\top y(\xi)] : \right.$$
$$\left. \{(x(\xi), y(\xi)) \in \operatorname{clconv}(X(\xi)), p_\xi x(\xi) - p_\xi \hat{x} = 0, \forall \xi \in \Xi\} \right\}. \tag{20}$$

when clconv$(X(\xi))$ *– the closure of the convex hull of* ξ *– is a closed, polyhedral set.*

The conditions under which clconv$(X(\xi))$ is a closed, polyhedral set are satisfied in a broad range of practical contest, such as when (a) the set determined by the linear constraints is bounded, or (b) the coefficients are rationals (see [19]). Under such conditions, the best bound obtained from the Lagrangian dual can be obtained by solving the linear program (20).

There are several methods for solving the dual problem (19), including subgradient methods [30], cutting plane, and bundle-type methods [13]. The Dantzig-Wolfe column generation method [7] is an approach to solve the primal linear program (20). In [17], the authors show the duality between a cutting plane model of $F(\lambda)$ and the primal linear program (20), and provide a method to recover a primal solution to (20) by solving the dual problem in the context of stochastic mixed integer programs. They also suggest warm-starting the branch-and-price method using the bound and primal solution obtained using this implementation of the proximal bundle method.

Because we are dealing with SMIPs, there is typically a duality gap between (20) and (12)-(17). The duality gap can be closed by branch-and-bound algorithms, where the bounding can be done by either solving the dual problem (19) or the primal problem (20). The first approach is developed in [3] under the name Dual Decomposition, where the authors employ a conic bundle method to solve the dual problem within the branch-and-bound. In [18], the authors develop a branch-and-price method for SMIPs by solving the primal problem for bounding.

We now show that by applying the PHA to the primal problem, one can recover both primal and dual optimal solutions to (20) and (19), respectively. Further and moreover, the lower bound $D(w)$ from (18) is equal to z_{LD}.

Proposition 2 *Suppose the PHA is applied to the primal problem* (20)*, where each iteration involves solving scenario sub-problems for each scenario* $\xi \in \Xi$ *of the following form:*

$$(x^{\nu+1}(\xi), y^{\nu+1}(\xi)) \in$$
$$\arg\min_{(x(\xi),y(\xi)) \in \operatorname{clconv}(X(\xi))} \left\{ c^\top x(\xi) + g(\xi)^\top y(\xi) + w^\nu(\xi)^\top x(\xi) + \tfrac{\rho}{2} ||x(\xi) - \hat{x}^\nu||^2 \right\}.$$

Then in the limit, one obtains a solution $(\hat{x}^*, w^*(\xi))$*, where* $\hat{x}^*$ *solves the primal problem* (20) *and* $w^*(\xi), \forall \xi \in \Xi$ *solves the dual problem* (19)*. Moreover, in the limit, the lower bound obtained from* (18) *is equal to* z_{LD}*.*

Proof : Because $\text{clconv}(X(\xi)), \xi \in \Xi$, are closed convex polyhedral sets, the optimization problem (20) is a linear program. The proof of this result then follows from the proof of Proposition 5.2 in [26]. □

Thus, the previous application of the PHA can be interpreted as a primal-dual algorithm in which sequences of primal solutions $\{\hat{x}^\nu\}_{\nu=1}^\infty$ and dual solutions $\{w^\nu(\xi)\}_{\nu=1}^\infty, \xi \in \Xi$ are generated during the course of execution. Further, these sequences converge to a saddle point of the ordinary Lagrangian.

3.3 Scenario Bundling

One proven way to accelerate PHA convergence is to decompose by bundles of scenarios, rather than individual scenarios. Bundles allows Step 1 and Step 5 of the algorithm to solve small extensive forms of the SMIP rather than single-scenario problems [37,16,6]. Simultaneous consideration of multiple scenarios enforces non-anticipativity among the composite scenarios, which in turn accelerates convergence at the master PHA level. The number of scenarios in each bundle must be balanced with the increased computational difficulty of the resulting bundles.

Here, we formalize the bundle version of PHA and show that Proposition 1 readily extends to this context. Our computational results presented in Section 5 indicate that the quality of PHA bounds can be improved dramatically by bundling scenarios.

Suppose the set of scenarios Ξ is partitioned into bundles, β, of K scenarios each. We denote the set of bundles by $\mathcal{B}$, with $\beta \in \mathcal{B}$. Let $P_\beta = \sum_{\xi \in \beta} p_\xi$. We then specify the extensive form of an SMIP given $\mathcal{B}$ as:

$$\min \quad c^\top x + \sum_{\xi \in \beta} \frac{p_\xi}{P_\beta} g(\xi)^\top y(\xi) \tag{21}$$

$$\text{s.t. } Ax \geq b \tag{22}$$

$$Wy(\xi) \geq r(\xi) - T(\xi)x(\xi), \qquad \forall \xi \in \beta \tag{23}$$

$$x \in \mathbb{Z}_+^{p_1} \times \mathbb{R}^{n_1 - p_1} \tag{24}$$

$$y(\xi) \in \mathbb{Z}_+^{p_2} \times \mathbb{R}^{n_2 - p_2} \qquad \forall \xi \in \beta. \tag{25}$$

We extend the notation defining the solution set X introduced in Section 2 as follows:

$$X(\beta) := \{x \in \mathbb{Z}_+^{p_1} \times \mathbb{R}^{n_1 - p_1}, y = (y(\xi))_{\xi \in \beta} \in \mathbb{Z}_+^{Kp_2} \times \mathbb{R}^{K(n_2 - p_2)} : \\ Ax \geq b, Wy(\xi) \geq r(\xi) - T(\xi)x, \xi \in \beta\}.$$

The PHA with scenario bundles is then given in Algorithm 2.

Algorithm 2 Progressive Hedging for Two-Stage SMIP with Scenario Bundles

1: **Initialization**: Let $\nu \leftarrow 0$ and $w^\nu(i) \leftarrow 0, \forall \beta \in \mathcal{B}$. Compute for each β

$$(x^{\nu+1}(\beta), \left(y^{\nu+1}(\xi)\right)_{\xi\in\beta}) \in \underset{(x,y)\in X(\beta)}{\arg\min}\; c^\top x + \sum_{\xi\in\beta} \frac{p_\xi}{P_\beta} g(\xi)^\top y(\xi).$$

2: **Iteration Update**: $\nu \leftarrow \nu + 1$.
3: **Aggregation**: $\hat{x}^\nu \leftarrow \sum_i P_\beta x^\nu(\beta)$.
4: **Price Update**: $w^\nu(\beta) \leftarrow w^{\nu-1}(\beta) + \rho(x^\nu(\beta) - \hat{x}^\nu)$.
5: **Decomposition** : Compute for each $\beta \in \mathcal{B}$

$$\left(x^{\nu+1}(\beta), \left(y^{\nu+1}(\xi)\right)_{\xi\in\beta}\right) \in$$
$$\arg\min_{(x,y)\in X(\beta)} \left\{ c^\top x + \textstyle\sum_{\xi\in\beta} \frac{p_\xi}{P_\beta} g(\xi)^\top y(\xi) + w^\nu(\beta)^\top x + \frac{\rho}{2}||x - \hat{x}^\nu||^2 \right\}.$$

6: If all bundle solutions $x(\beta)$ are equal, stop. Otherwise go to step 2.

The extension of Proposition 1 to the bundle version of the PHA is straightforward and is stated here for completeness. As in the single-scenario decomposition, the proof follows directly from the fact that in every iteration ν, $\sum_{\beta\in\mathcal{B}} P_\beta w^\nu(\beta) = 0$.

Proposition 3 *Let* $w = (w(\beta))_{\beta\in\mathcal{B}}$*, where* $w(\beta) \in \mathbb{R}^{n_1}$ *satisfy* $\sum_{\beta\in\mathcal{B}} P_\beta w(\beta) = 0$ *(component-wise). Let*

$$D_\beta(w(\beta)) := \min_{(x,y)\in X(\beta)} \left(c^\top x + \sum_{\xi\in\beta} \frac{p_\xi}{P_\beta} g(\xi)^\top y(\xi) + w(\beta)^\top x \right). \tag{26}$$

Then $D(w) := \sum_{\beta\in\mathcal{B}} P_\beta D_\beta(w(\beta)) \le z^*$.

4 The Multi-Stage Case

In the multi-stage case, $\tilde{\xi} = \left\{\tilde{\xi}_t\right\}_{t=1}^{\mathcal{T}}$ is defined on a probability space $(\Xi, \mathcal{A}, \mathcal{P})$, where $\mathcal{T}$ is the number of stages. We organize realizations, ξ, into a tree with the property that scenarios with the same realization up to stage t share a node at that stage. We use $\xi_{\le t}$ to refer to a realization up to time t – i.e., a node in the scenario tree – and $\xi_{<t}$ to refer to the parent node.

With this notation, we restate problem (1)-(3) as

$$\min \quad c^\top x_1 + \mathbb{E}_{\tilde{\xi}_2}[f_2(x_1, \xi_2)] \tag{27}$$
$$\text{s.t. } Ax_1 \ge b \tag{28}$$
$$x_1 \in \mathbb{Z}_+^{p_1} \times \mathbb{R}^{n_1-p_1}, \tag{29}$$

For $t \in 2, \ldots, \mathcal{T},\ f_t(x_{t-1}, \xi_{\leq t})$ is defined as:

$$f_t(x_{t-1}, \xi_t) = \min g(\xi_{<t})^\top x_t + \mathbb{E}_{\tilde{\xi}_{t+1}|\xi_{\leq t}}[f_{t+1}(x_{t+1}, \xi_{\leq t+1})] \tag{30}$$

$$\text{s.t. } W_t(\xi_{<t})x_t \geq r_t(\xi_{<t}) - T_t(\xi_{<t})x_{t-1} \tag{31}$$

$$x_t \in \mathbb{Z}_+^{p_t} \times \mathbb{R}^{n_t - p_t}. \tag{32}$$

Here, $c \in \mathbb{R}^{n_1}$, $A \in \mathbb{R}^{m_1 \times n_1}$, $b \in \mathbb{R}^{m_1}$, $g(\xi_{<t}) \in \mathbb{R}^{n_t}$, $W_t(\xi_{<t}) \in \mathbb{R}^{m_t \times n_t}$, $T(\xi_{<t}) \in \mathbb{R}^{m_t \times n_{t-1}}$ and $r_t(\xi_{<t}) \in \mathbb{R}^{m_t}$ comprise the data of the stochastic mixed integer program. To compress the problem statement, we define $\mathbb{E}_{\tilde{\xi}_{\mathcal{T}+1}|\xi_{\leq \mathcal{T}}}[f_{\mathcal{T}+1}(\cdot)]$ to be zero.

To re-write formulation (12)-(17) for the multi-stage case, it is useful to introduce some notation for the scenario tree. Let $\mathcal{G}_t(\xi)$ be the scenario tree node for ξ at stage t. We write the multi-stage scenario formulation as

$$\min \quad \sum_{\xi \in \Xi} p_\xi \left[c^\top x_1(\xi) + \sum_{t=2}^{\mathcal{T}} g_t(\mathcal{G}_t(\xi))^\top x_t(\xi) \right] \tag{33}$$

$$\text{s.t. } Ax_1(\xi) \geq b \tag{34}$$

$$W_t(\mathcal{G}_{t-1}(\xi))x_t(\xi) \geq r_t(\mathcal{G}_t(\xi)) - T(\mathcal{G}_t(\xi))x_{t-1}(\xi), \quad \xi \in \Xi,\ t = 2, \ldots, \mathcal{T} \tag{35}$$

$$p_\xi x_t(\xi) - p_\xi \hat{x}_t = 0,$$

$$t = 1, \ldots, \mathcal{T},\ \mathcal{D} \in \mathcal{G}_t,\ \xi \in \mathcal{D}^{-1} \tag{36}$$

$$\tag{37}$$

$$\hat{x}_t, x_t \in \mathbb{Z}_+^{p_t} \times \mathbb{R}^{n_t - p_t} \quad . \tag{38}$$

For the multi-stage case, let

$$X(\xi) := \{ x_1 \in \mathbb{Z}_+^{p_1} \times \mathbb{R}^{n_1 - p_1},\ x_t \in \mathbb{Z}_+^{p_t} \times \mathbb{R}^{n_t - p_t} : \tag{39}$$

$$Ax_1 \geq b,\ \ W_t(\mathcal{G}_{t-1}(\xi))x_t(\xi) \geq r_t(\mathcal{G}_t(\xi)) - T(\mathcal{G}_t(\xi))x_{t-1},\ \ t = 2, \ldots, \mathcal{T} \} . \tag{40}$$

For the statement of the algorithm, let $\mathcal{G}_t$ be the set of all scenario tree nodes for stage t. For a particular node $\mathcal{D}$ let $\mathcal{D}^{-1}$ be the set of scenarios that define the node. The statement of the multi-stage PHA is given in Algorithm 3. Some of the steps can be implemented to get the same result with a little less computational effort.

The main thing to notice about the multi-stage case is that that everything remains from the two-stage case for the first stage because all ξ share a single $\mathcal{G}_1(\xi)$. Further note that each non-leaf node behaves like the first stage for its own tree.

Algorithm 3 Progressive Hedging for Multi-Stage SMIP

1: **Initialization**: Let $\nu \leftarrow 0$ and $w^\nu(\mathcal{G}_t(\xi)) \leftarrow 0, \ \forall \xi \in \Xi, \ t = 1, \ldots, \mathcal{T}$. Compute for each $\xi \in \Xi$:

$$x^{\nu+1}(\xi) \in \arg\min_{x \in X(\xi)} c^\top x_1 + \sum_{t=2}^{\mathcal{T}} g_t(\mathcal{G}_t(\xi))^\top x_t$$

2: **Iteration Update**: $\nu \leftarrow \nu + 1$

3: **Aggregation**: Compute for each $t = 1, \ldots, \mathcal{T} - 1$ and each $\mathcal{D} \in \mathcal{G}_t$:

$$\hat{x}_t^\nu(\mathcal{D}) \leftarrow \sum_{\hat{\xi} \in \mathcal{D}^{-1}} \pi_{\hat{\xi}} x_t^\nu(\hat{\xi}) / \sum_{\hat{\xi} \in \mathcal{D}^{-1}} \pi_{\hat{\xi}}$$

4: **Price Update**: Compute for each $t = 1, \ldots, \mathcal{T} - 1$ and each $\xi \in \Xi$

$$w^\nu(\mathcal{G}_t(\xi)) \leftarrow w^{\nu-1}(\mathcal{G}_t(\xi)) + \rho \left[x^\nu(\mathcal{G}_t(\xi)) - \hat{x}^\nu(\mathcal{G}_t(\xi))\right]$$

5: **Decomposition**: Compute for each $\xi \in \Xi$

$$x^{\nu+1}(\xi) \in \arg\min_{x \in X(\xi)} c^\top x_1 + \sum_{t=2}^{\mathcal{T}} g_t(\mathcal{G}_t(\xi))^\top x_t + \sum_{t=1}^{\mathcal{T}-1} \left[w^\nu(\mathcal{G}_t(\xi))^\top x_t + \frac{\rho}{2} ||x_t - \hat{x}_t^\nu(\mathcal{G}_t(\xi))||^2\right]$$

6: If all scenario solutions $x(\xi)$ are equal, Stop. Otherwise goto step 2

5 Impact of ρ on Lower Bound Quality

In this section, we empirically study the impact of strategies for choosing the PH ρ parameter on the convergence of lower bounds in the mixed integer programming case. Note that in Proposition 1 the minimization is over the convex closure of the constraints, but in these experiments we minimize over the $X(\xi)$ defined in Section 2 or $X(\beta)$ as defined in Section 3 in the bundling case (i.e., we solve the MIP rather than optimizing over the convex closure). We consider different classes of two-stage stochastic mixed-integer programs. Previous experience [21,20] indicates that larger values of ρ can accelerate the convergence of the PHA to a primal feasible solution. In [35], the authors study the impact of ρ for obtaining fast primal solutions and give recommendations for choosing ρ for a general class of stochastic resource allocation problems. Here, we instead focus on the relationship between ρ and the convergence of lower bounds. We show that as in the primal case, the quality of lower bounds obtained by the PHA in the case of SMIPs is significantly impacted by the choice of ρ. However, the relationship between ρ and bound quality differs in key aspects from the primal case, as we will illustrate below.

We begin in Section 5.1 with a brief discussion of the theoretical motivation for differences in the empirical behavior of PHA convergence in the primal and lower bound iterates, as a function of the value of the ρ parameter. We then define our computational environment in Section 5.2. Our empirical analyses on the conference of PHA lower bounds in the SMIP case are then investigated on a pair of two-stage test cases: stochastic unit commitment (Section 5.3) and stochastic server location (Section 5.4).

5.1 Motivation

Rockafellar and Wets [26] provide a characterization of an iteration of the PHA in terms of a certain "proximal saddle function." They show that in each iteration of the algorithm, $\hat{x}^{\nu+1}$ and $w^{\nu+1}(\xi), \forall \xi \in \Xi$, is the saddle point of the following function:

$$l(v,w) + \frac{\rho}{2}||v - \hat{x}|| + \sum_{\xi \in \Xi} p_\xi \frac{1}{2\rho}||w(\xi) - \hat{w}^\nu||$$

where

$$\begin{aligned} l(v,w) := \inf & \sum_{\xi \in \Xi} p_\xi \{c^\top x(\xi) + g(\xi)^\top y(\xi) + w(\xi)^\top x(\xi)\} \\ \text{s.t. } & (x(\xi), y(\xi)) \in X(\xi), \forall \xi \in \Xi \\ & \sum_{\xi \in \Xi} p_\xi w(\xi) = 0, \\ & \hat{x} = v. \end{aligned}$$

The iterates being saddle points of the proximal saddle function suggests a tradeoff in choosing ρ for the convergence of the primal and dual sequences, which we will now empirically demonstrates.

5.2 Computational Environment

We encode the stochastic programming model and corresponding instance data for both the unit commitment and server location problems examined below in PySP [34]. PySP is an open-source modeling and optimization framework for stochastic programming, co-developed by Sandia National Laboratories and the University of California Davis. The Pyomo [11] algebraic modeling language is the basis for PySP; both packages are in turn embedded in the Coopr software library (`https:software.sandia.gov/trac/coopr`). The resulting models and associated data are available from the authors upon request.

The PySP library provides a generic and customizable implementation of progressive hedging, specifically focusing on capabilities such as cycle detection and variable fixing that are commonly employed in the case of stochastic MIPs [35], and scenario bundling. Using the supplied extension framework, we have developed a generic implementation of the lower bounding scheme for progressive hedging, as described in Section 3. This extension, presently integrated into PySP, is coded to ensure that the lower bounding computations (i.e., the sub-problem solves performed to compute $D(w)$) do not interfere with the core progressive hedging functionality. In particular, the extension ensures that (1) proper warm-starting of sub-problems between primary PH iterations are maintained (2) indicator status of fixed variables is restored following lower bounding sub-problem solves.

In the interest of brevity, we do not report timing results for the various algorithm, and consequently do not provide details of our hardware environment here. Rather, the focus of our experiments is to investigate the relationship between ρ and the quality of lower bounds generated by progressive hedging.

5.3 Stochastic Unit Commitment

Unit commitment is a tactical planning problem faced by power grid operators world-wide, and is solved on a daily basis. For each thermal generation unit (e.g., coal or natural gas plant) in the system, unit commitment determines the time periods for the next day during which each unit will be operating, in addition to the corresponding power output level, given predictions for the next-day load (demand). Complicating constraints are incurred by generator operational requirements. For example, large thermal generating units such as coal plants cannot cycle on and off frequently, and must be kept on or off for a minimum number of time units once started up or shut down. Similarly, thermal units are limited in their ability to rapidly change their power output levels.

Traditionally, unit commitment has been formulated and solved as a deterministic mixed-integer program, which includes reserve constraints to provide a buffer of extra capacity in case of inaccurate demand forecasts or renewable generation output. With the incorporation of increasing amounts of generation from unpredictable and variable sources such as wind and solar power, uncertainty in the demand placed on thermal generators has increased dramatically. Stochastic programming formulations of the unit commitment problem have been proposed in order to reduce the reserve levels, by explicitly accounting for the uncertainty via probabilistic scenarios [32]. Binary variables in the first stage constitute the on/off decisions for the thermal generators. Some formulations [24, 27] additionally include binary variables for committing fast-responding "peaker" units in the second stage. Other second stage variables include scenario-specific power generation levels. In [28], the authors report the results of a parallel implementation of the PHA to rapidly obtain solutions to large-scale stochastic unit commitment problems.

First, we report the results of the PHA when executed on a 5 bus test case of the AMES wholesale power market test bed system [31], augmented with additional unit commitment extensions [9]. The stochastic unit commitment problem formulation is based on the deterministic formulation of [4]. The problem instance includes 5 generators and 5 buses, with a scheduling horizon of 24 hours. We consider 10 load scenarios. The extensive form of this instance has 16,194 variables (1,200 binary) and 24,092 constraints.

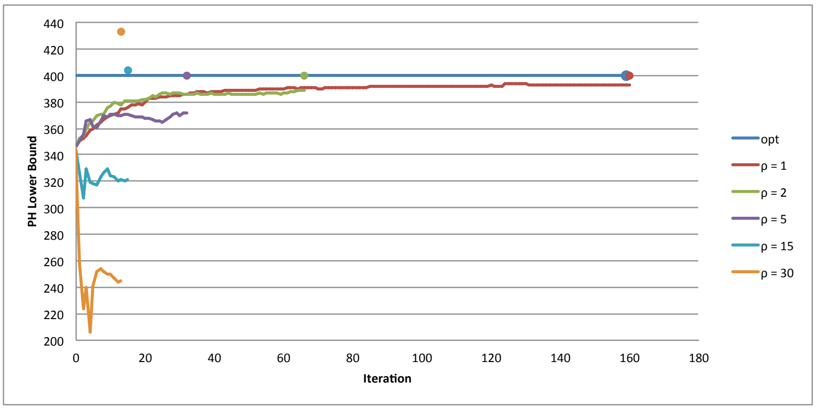

Fig. 1: Primal and lower bound quality for a 5-bus unit stochastic unit commitment instance, obtained by progressive hedging under different values of the penalty parameter ρ.

We perform multiple runs of PHA on this instance, varying the strategy used to compute values for the penalty parameter ρ. Specifically, we consider fixed $\rho \in \{1, 2, 5, 15, 30\}$. During each run, we record the value of the final primal incumbent solution and the time-series of the lower bound $D(w)$ obtained at each iteration of the PHA. The results are shown in Figure 1, which additionally displays the optimal solution value. We observe a significant trade-off in terms of the quality of *both* primal and lower bound quality as ρ is varied. Using large values of ρ leads to runs in which small numbers of iterations of the PHA are required to achieve primal convergence. However, the quality of the final solution can be relatively poor. Further, large ρ values can lead to oscillations of the dual prices, leading to poor convergence of not only the dual variables but also the lower bounds. In contrast, while low values of ρ lead to increased numbers of PHA iterations required for primal convergence, the quality of the resulting primal solutions and the corresponding lower bounds is significantly improved. In particular, when $\rho = 1$ the primal solution is optimal, and the lower bound is very tight. Overall, these results clearly demonstrate that the choice of ρ is of critical importance in determining not only high-quality primal solutions, but also tight lower bounds, via the PHA.

5.4 Stochastic Server Location

Next, we consider the interaction between PHA lower bound quality and ρ considering the stochastic server location problem (SSLP) [23]. Like the stochastic unit commitment problem considered above, this is a two-stage SMIP. Scenario-independent instance data specifies the set of potential server and customer locations, server capacities, installation costs, and revenues.

Scenario-dependent instance data additionally specifies the set of customers that are actually realized in that specific scenario. Binary first-stage decision variables indicate whether to invest in a server at each of the potential locations, while binary second-stage decision variables control the assignment of customers to servers. Constraints enforce limits on server capacity, and the objective is to minimize the difference between the investment costs associated with server siting and the revenue obtained by serving material customers.

We now examine illustrative empirical results associated with the SSLP. Publicly available SSLP instances (available in the SIPLIB – `http://www2.isye.gatech.edu/~sahmed/siplib`) are denoted as SSLP$m.n.s$, where m is the number of potential server locations, n is the number of potential clients, and s is the number of scenarios; instances having common values of (m, n) differ only in the scenario sets). We converted the SIPLIB instances into the PySP data format, and validated the resulting solutions relative to reported results.

First, we consider the interaction between the ρ and PHA bound quality. For a given SSLP instance, we vary ρ, and for each experiment we allow PHA to converge to a fully non-anticipative solution. At each iteration of the PHA, we compute a lower bound on the objective function value using the procedure described in Section 3. An illustrative example is shown in Figure 2, for the instance SSLP.5.25.50. Here, we vary ρ and display the corresponding time-series in the computed lower bound $D(w)$. We omit the primal solution objective value in this case, as the optimal value is achieved in each run. As in the stochastic unit commitment case, we observe an inverse relationship between lower bound quality and the value of ρ, i.e., lower values of ρ lead to improved lower bounds, albeit at the expense of increased numbers of PHA iterations required to achieve primal convergence. Further, for large ρ, we observe unstable, oscillatory behavior in the lower bounds computed by the PHA.

Next, we consider the result obtained for PHA lower bounds on the SSLP when bundling scenarios. Specifically, we vary the number of scenarios in each bundle considered by PHA, while holding ρ constant. The scenario-to-bundle assignment is fixed during the course of execution, although this is not strictly required. An illustrative example in shown in Figure 3, for the instance SSLP.5.25.100. In this case, we fix $\rho = 2$. This result clearly demonstrates the effectiveness of bundling as a method for simultaneously improving lower bound quality and reducing the number of primal iterations required for convergence. In particular, even small bundles can yield dramatic improvements in PHA lower bound quality.

Finally, we consider summary results of SSLP instance behavior under the PHA, shown in Table 1. In this experiment, we consider PHA behavior with no bundling, under two strategies for setting ρ. In the first strategy, we simply fix $\rho = 1$. In the second strategy, we employ the variable-specific ρ strategy described in [35]. This strategy has proved useful in quickly obtaining high-quality primal solutions to network design problems. As shown in Table 1, the average value of ρ obtained under this strategy is significant, ranging

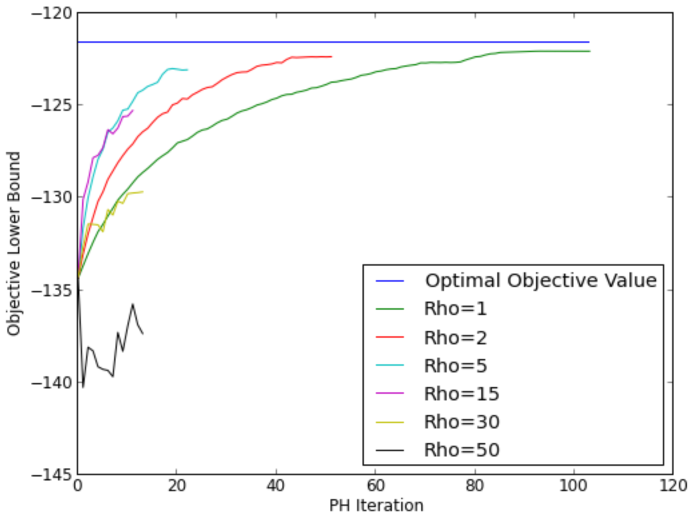

Fig. 2: Lower bound quality for the PHA, obtained for the SSLP.5.25.50 instance under various values of the ρ penalty parameter.

from 27.5 to 34.3. Under $\rho = 1$, the primal solutions obtained using PHA are extremely high-quality, with very tight corresponding lower bounds. However, the number of iterations required to achieve primal convergence is comparably large. In contrast, while we observe no significant difference in primal solution quality under the ρ_{sep} strategy, the lower bounds are significantly worse than those obtained using ρ. However, the total number of iterations is remarkably smaller. Overall, these results further reinforce the fundamental finding of our empirical investigations presented above and in Section 5.3: smaller values of ρ lead to higher-quality lower bounds, at the expense of increased numbers of PHA iterations. Fundamentally, there is a clear trade-off in bound quality and run time, analogous to case for PHA primal solution quality.

6 Conclusions

We have presented a lower bound for cost minimization stochastic mixed integer programs that can be computed in any iteration of the PHA using the dual prices on non-anticipativity as they are updated by the algorithm. The bound computed from optimal values of the dual prices is as tight as possible given the duality gap caused by integer-valued decision variables when computed using the convex closure of the constraints. We show computational evidence that it is very tight when computed over the constraints for the original problem, which results in an effort to compute the bound that is similar to one PHA iteration. The bounding procedure is applicable to any number of

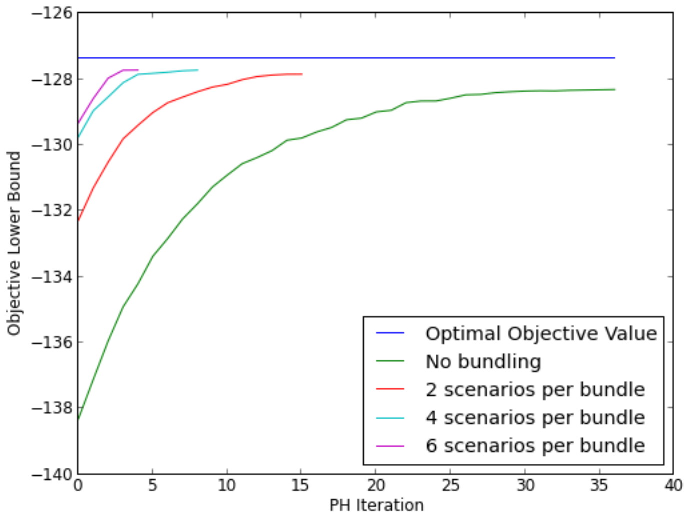

Fig. 3: Lower bound quality for the PHA obtained, for the SSLP-5-25-100 instance under $\rho = 2$ and variable numbers of scenarios per bundle.

	$\rho = 1$			$\rho = \rho_{sep}$			
Instance	Iters	Primal	L.B.	ρ_{sep}	Iters	Primal	L.B.
5.25.50	98	-121.60	-122.25	34.3	11	-121.60	-128.36
5.25.100	76	-127.37	-127.78	34.3	20	-127.37	-134.80
5.50.50	40	-337.12	-337.30	27.5	2	-337.12	-341.98
5.50.100	50	-323.70	-323.91	27.5	2	-323.70	-327.84
10.50.50	446	-369.94	-370.64	25.9	6	-369.94	-382.90
10.50.100	556	-359.33	-360.03	25.9	18	-359.33	-374.04
10.50.500	958	-354.09	-354.87	25.9	20	-354.09	-369.21
15.45.5	31	-262.50	-262.52	28.5	6	-261.20	-269.20

Table 1: Convergence, primal quality, and lower bound quality statistics for SSLP instances under PHA using $\rho = 1$ and the strategy ρ_{sep}. Columns record the instance, the number of iterations required for primal convergence, the primal solution objective function value, the best lower bound obtained by PHA, and – in the case of the strategy ρ_{sep} – the mean value of ρ for the first-stage decision variables.

decision stages with integer decisions in any stages. It also applies when the problem is decomposed into subproblems for bundles of scenarios rather than single-scenario subproblems.

Computing lower bounds for the PHA allows one to assess the quality of the solutions generated by the algorithm contemporaneously. This fills an important need when one applies PHA to mixed integer programs in practice or uses it in conjunction with other algorithms based on dual prices. The latter application remains as a promising area for potential future research.

Numerical results indicate that the quality of the bound from the PHA is inversely related to speed of convergence via the progressive hedging parameter, ρ. Bundling scenarios can dramatically improve its quality in both two- and multi-stage formulations. In two different types of problems, the computed bounds confirm that the PHA can quickly converge to good mixed-integer solutions.

References

1. Ahmed, S., Tawarmalani, M., Sahinidis, N.V.: A finite branch and bound algorithm for two-stage stochastic integer programs. Mathematical Programming, Series A **100**(2), 355–377 (2004)
2. Boyd, S., Parihk, N., Chu, E., Peleato, B., Eckstein, J.: Distributed optimization and statistical learning via the alternating direction method of multipliers. Foundations and Trends in Machine Learning **3** (2011)
3. Carøe, C.C., Schultz, R.: Dual decomposition in stochastic integer programming. Operations Research Letters **24**(1), 37–45 (1999)
4. Carrión, M., Arroyo, J.M.: A computationally efficient mixed-integer linear formulation for the thermal unit commitment problem. Power Systems, IEEE Transactions on **21**(3), 1371–1378 (2006)
5. Cormican, K.J., Morton, D.P., Wood, R.K.: Stochastic network interdiction. Operations Research **46**(2), 184–197 (1998)
6. Crainic, T., Hewitt, M., Rei, W.: Scenario grouping in a progressive hedging-based meta-heuristics for stochastic network design. Computers and Operations Research **43**, 90–99 (2014)
7. Dantzig, G.B., Wolfe, P.: Decomposition principle for linear programs. Operations research **8**(1), 101–111 (1960)
8. Eichhorn, A., Heitsch, H., Römisch, W.: Stochastic optimization of electricity portfolios: Scenario tree modeling and risk management. In: S. Rebennack, P.M. Pardalos, M.V.F. Pereira, N.A. Iliadis (eds.) Handbook of Power Systems II, Energy Systems, pp. 405–432. Springer Berlin Heidelberg (2010). DOI 10.1007/978-3-642-12686-4_15. URL `http://dx.doi.org/10.1007/978-3-642-12686-4_15`
9. Ela, E., Milligan, M., O'Malley, M.: A flexible power system operations simulation model for assessing wind integration. In: Power and Energy Society General Meeting, 2011 IEEE. IEEE (2011)
10. Guignard, M., Kim, S.: Lagrangean decomposition: a model yielding stronger lagrangean bounds. Mathematical programming **39**(2), 215–228 (1987)
11. Hart, W., Watson, J., Woodruff, D.: Pyomo: Modeling and solving mathematical programs in Python. Mathematical Programming Computation **3**(3) (2011)
12. Held, H., Hemmecke, R., Woodruff, D.L.: A decomposition algorithm applied to planning the interdiction of stochastic networks. Naval Research Logistics (NRL) **52**(4), 321–328 (2005)
13. Hiriart-Urruty, J.B., Lemaréchal, C.: Convex analysis and minimization algorithms. Springer (1993)
14. Janjarassuk, U., Linderoth, J.: Reformulation and sampling to solve a stochastic network interdiction problem. Networks **52**(3), 120–132 (2008)
15. Kaut, M., Wallace, S.: Evaluation of scenario-generation methods for stochastic programming. Pacific Journal of Optimization **3**(2), 257–271 (2007)
16. Løkketangen, A., Woodruff, D.L.: Progressive hedging and tabu search applied to mixed integer (0,1) multistage stochastic programming. Journal of Heuristics **2**, 111–128 (1996)
17. Lubin, M., Martin, K., Petra, C., Sandıkçı, B.: On parallelizing dual decomposition in stochastic integer programming. Operations Research Letters **41**(3), 252–258 (2013)

18. Lulli, G., Sen, S.: A branch-and-price algorithm for multistage stochastic integer programming with application to stochastic batch-sizing problems. Management Science **50**(6), 786–796 (2004)
19. Meyer, R.R.: On the existence of optimal solutions to integer and mixed-integer programming problems. Mathematical Programming **7**(1), 223–235 (1974)
20. Mulvey, J.M., Vladimirou, H.: Applying the progressive hedging algorithm to stochastic generalized networks. Annals of Operations Research **31**, 399–424 (1991)
21. Mulvey, J.M., Vladimirou, H.: Solving multistage stochastic networks: an appplication of scenario aggregation. Networks **21**, 619–643 (1991)
22. Nemhauser, G.L., Wolsey, L.A.: Integer and combinatorial optimization, vol. 18. Wiley New York (1988)
23. Ntaimo, L., Sen, S.: The million-variable "march" for stochastic combinatorial optimization. Journal of Global Optimization **32**, 385–400 (2005)
24. Papavasiliou, A., Oren, S.S.: Multiarea stochastic unit commitment for high wind penetration in a transmission constrained network. Operations Research **61**(3), 578–592 (2013)
25. Pflug, G.C., Pichler, A.: Approximations for probability distributions and stochastic optimization problems. In: M. Bertocchi, G. Consigli, M.A. Dempster (eds.) Stochastic Optimization Methods in Finance and Energy, *International Series in Operations Research and Management Science*, vol. 163. Springer (2011)
26. Rockafellar, R.T., Wets, R.J.B.: Scenarios and policy aggregation in optimization under uncertainty. Mathematics of Operations Research **16**(1), 119–147 (2004)
27. Ruiz, P.A., Philbrick, R.C., Zack, E., W., C.K., Sauer, P.: Uncertainty management in the unit commitment problem. IEEE Transactions on Power Systems **24**(2), 642–651 (2009)
28. Ryan, S.M., Wets, R.J.B., Woodruff, D.L., Silva-Monroy, C., Watson, J.P.: Toward scalable, parallel progressive hedging for stochastic unit commitment. In: Power and Energy Society General Meeting, 2013 IEEE. IEEE (2013)
29. Santoso, T., Ahmed, S., Goetschalckx, M., Shapiro, A.: A stochastic programming approach for supply chain network design under uncertainty. European Journal of Operational Research **167**(1), 96–115 (2005)
30. Shor, N.Z., Kiwiel, K.C., Ruszczynski, A.: Minimization methods for non-differentiable functions, vol. 3. Springer-Verlag Berlin (1985)
31. Sun, J., Tesfatsion, L.: Dynamic testing of wholesale power market designs: An open-source agent-based framework. Computational Economics **30**(3), 291–327 (2007)
32. Takriti, S., Birge, J.R., Long, E.: A stochastic model for the unit commitment problem. IEEE Transactions on Power Systems **11**(3), 1497–1508 (1996)
33. van der Vlerk, M.H.: Stochastic integer programming bibliography. World Wide Web, `http://www.eco.rug.nl/mally/biblio/sip.html` (1996-2007)
34. Watson, J., Woodruff, D., Hart, W.: PySP: Modeling and solving stochastic programs in Python. Mathematical Programming Computation **4**(2) (2012)
35. Watson, J.P., Woodruff, D.L.: Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems. Computational Management Science **8**(4), 355–370 (2011)
36. Wets, R.J.B.: On the relation between stochastic and deterministic optimization. In: Control Theory, Numerical Methods and Computer Systems Modelling, pp. 350–361. Springer Berlin Heidelberg (1975)
37. Wets, R.J.B.: The aggregation principle in scenario analysis and stochastic optimization. In: S.W. Wallace (ed.) Algorithms and Model Formulations in Mathematical Programming, pp. 91–113. Springer-Verlag (1989)