# Toward Scalable, Parallel Progressive Hedging for Stochastic Unit Commitment

Sarah M. Ryan
Iowa State University
Ames, IA, USA
smryan@iastate.edu

Roger J.-B. Wets and David L. Woodruff
University of California Davis
Davis, CA, USA
{dlwoodruff,rjbwets}@ucdavis.edu

César Silva-Monroy and Jean-Paul Watson
Sandia National Laboratories
Albuquerque, NM, USA
{casilv,jwatson}@sandia.gov

***Abstract*—Given increasing penetration of variable generation units, there is significant interest in the power systems research community concerning the development of solution techniques that directly address the stochasticity of these sources in the unit commitment problem. Unfortunately, despite significant attention from the research community, stochastic unit commitment solvers have not made their way into practice, due in large part to the computational difficulty of the problem. In this paper, we address this issue, and focus on the development of a decomposition scheme based on the progressive hedging algorithm of Rockafellar and Wets. Our focus is on achieving solve times that are consistent with the requirements of ISO and utilities, on modest-scale instances, using reasonable numbers of scenarios. Further, we make use of modest-scale parallel computing, representing capabilities either presently deployed, or easily deployed in the near future. We demonstrate our progress to date on a test instance representing a simplified version of the US western interconnect (WECC-240).**

***Index Terms*—Computation time, Optimization methods, Parallel algorithms, Power generation scheduling, Wind energy.**

## I. Introduction

The uncertainty of power output associated with variable generation units has significant impact on power system operations. Specifically, ISOs and related entities must address the issue of unknown outputs in the day-ahead and reliability unit commitment (UC) problems. The issue also arises in the context of uncertainty in load and unforced outages, which has historically been managed by employing deterministically derived reserve margins. However, the increasingly large penetration rates of variable generation necessitates a more direct approach to uncertainty management, in order to minimize the need for significantly larger reserve margins – which in turn mitigate the potential cost and environmental benefits of renewables.

One approach to managing uncertainty in unit commitment is via stochastic programming [10]. A stochastic program is an extension of standard deterministic mathematical programming, in which the space of possible outcomes (e.g., load and variable generation output uncertainty) is represented by a probability-weighted scenario tree [11]. The objective in a stochastic unit commitment (SUC) is to minimize the expected total operational cost – including startup, fuel, and shutdown costs for thermal units – such that load is satisfied in all scenarios, subject to operational constraints such as ramp rate limits, minimum startup and shutdown times, and power flow limits on transmission lines. Since it considers several scenarios, the solution to the SUC better positions the generation fleet to handle variations in load and variable generation. Although sometimes the operational cost incurred might be higher than in the deterministic counterpart, over an extended period of time (weeks, months) costs savings are achieved [10,12].

While there is a significant body of power systems literature on stochastic unit commitment, this research has not yet been successfully transferred to real-world industrial contexts. This is due in large part to the well-known computational difficulty of stochastic unit commitment – where even small problems with a handful of scenarios can take hours to solve [13]. A survey of prior algorithmic approaches to SUC is provided in [12]. An analysis of this survey indicates that the current state-of-the-art for SUC can tackle approximately 50 scenarios on instances with approximately a hundred thermal generation units, achieving solutions in one or two hours of run time. Further, those studies consider short (24-hour) time horizons, which greatly simplifies the SUC problem.

The purpose of this paper is to detail an research effort dedicated to developing a SUC solver capable of achieving solutions in tractable (e.g., less than an hour) run-times, given realistic numbers of time periods (e.g., 48 hour-long periods)

on realistic-scale power systems (e.g., 1000 generation units). We demonstrate significant progress toward this goal, focusing on achieving the first two criteria on the WECC-240 test instance. We leverage modest-scale parallelism to achieve the required run-times, leveraging commodity computing capabilities that an ISO / utility either presently possesses or is likely to acquire in the near future. Fundamentally, our goal is to demonstrate the viability of SUC in industrial contexts.

The remainder of this paper is organized as follows. In Section II, we briefly survey the existing literature on stochastic unit commitment; we focus on scalability of proposed approaches and overall deployment goals. In Section III, we introduce the progressive hedging algorithm, placing it in the broader context of decomposition algorithms for stochastic programming. We document our experimental methodology and test models/instances in Section IV. In Section V, we discuss experimental results on a stochastic reduced model of the Western Interconnection (WECC-240). Finally, we conclude in Section VI with a summary of our results and in-progress follow-on research.

## II. STOCHASTIC UNIT COMMITMENT

A detailed review of SUC is beyond the scope of the present paper. Relevant recent literature is surveyed in [12]. Our work is most closely related to that of Takriti et al. [10], who document the first use of progressive hedging in the context of unit commitment. As discussed subsequently in Section III, we introduce several customizations to the configuration of progressive hedging, based on research conducted over the past few years, in order to accelerate convergence of the core algorithm. Mirroring most previous work, we consider a two-stage variant of the SUC, in which i) the first-stage decision variables are unit on/off and related state variables; ii) the second-stage (scenario-specific) decision variables include generator power output levels and transmission power flows; and iii) the optimization objective is to minimize the first stage cost plus the expected second stage cost.

## III. DECOMPOSITION AND PROGRESSIVE HEDGING

In general, it is widely accepted that the extensive form (i.e., a single, large mathematical program consisting of all scenarios – discussed subsequently) of the SUC problem is insoluble via direct methods. To achieve operational run-time requirements for a SUC, decomposition techniques must be leveraged. Two general classes of decomposition techniques for SUC are time stage-based and scenario-based. The exemplar stage-based technique is the L-shaped method, or Benders decomposition [8]. Exemplars of scenario-based decomposition include progressive hedging (PH) [6] and dual decomposition [7]. One advantage of scenario-based decomposition techniques over their stage-based counterparts is a more uniform distribution of sub-problem difficulty. In particular, the computational difficulty of the master problem in the L-shaped method can grow significantly as the number of iterations grows, while the sub-problems are typically comparatively easy.

In order to write the PH algorithm, we consider an abstract class of optimization problems that contains the unit commitment problem. We model the future as being described by a finite set of scenarios indexed by s, each of which gives a full realization of problem data (e.g., demand, wind power, generator outages) and each of which has an associated probability $p_s$. If we could know the future, the problem would be written as

$$\text{Min } f(x) + g_s(y;x) \mid x,y \in Q_s \qquad (1)$$

where x represents the vector of first stage decisions that must be made before the scenario is known (e.g., unit commitments) and $y_s$ represents the decisions that are made after, or as a result of, the scenario realization (e.g., dispatch). The function f gives the first stage costs and the function g gives the second stage costs, which depend on i) the scenario, ii) the first stage decisions, and iii) the scenario-specific second-stage decisions. We use the symbol | to mean "subject to" and the symbol $Q_s$ to summarize all constraints on the decision variables such as those imposed by the laws of physics and also policies. Since the future cannot be known, it is common to minimize the expected cost, which is given as

$$\text{Min } f(x) + \textstyle\sum_s [p_s g_s(y_s;x)] \mid x,y_s \in Q_s \qquad (2)$$

If the sum is expanded and the constraints are written for all scenarios, the formulation is known as the extensive form (EF), which can be solved directly for modest-sized (mixed-integer) linear problems. The PH algorithm, on the other hand, temporarily allows the x values to depend on the scenario and then, by estimation of appropriate multipliers, requiring that they be the same for all scenarios.

The algorithm proceeds as follows:

Input: ρ, a scalar or vector of the same length as x; initialize the iteration counter k=0; and the vector of weights $w^{(k=0)}=0$ with the same dimension as x.

0. For each s: $x_s^{(0)} = \text{argmin}_{x,y}\ f(x) + g_s(y;x) \mid x,y \in Q_s$
1. $\mu = \sum_s [p_s x_s]$ ; k = k+1
2. For each s: $w_s^{(k)} = w_s^{(k-1)} + \rho(x_s^{(k-1)} - \mu)$
3. For each s: $x_s^{(k)} = \text{argmin}_{x,y}\ f(x) + g_s(y;x) + w_s^{(k)} x + \rho/2 \| x - \mu \|^2 \mid x,y \in Q_s$
4. If x has not converged sufficiently to μ then goto 1.

In step 0, the scenario sub-problems are solved. In step 1, the algorithm forms its best current guess at a solution that is *non-anticipative*; i.e., the value of μ does not depend on s. In step 2, estimates of multipliers needed to enforce non-anticipativity are updated. In step 3, these multipliers are used along with a squared *proximal* term to solve a problem designed to result in x values that are converging to a non-anticipative, optimal solution.

As is obvious from the pseudo-code above, PH can be easily parallelized. Specifically, a barrier synchronization point occurs after Steps 0 and 3, i.e., after each set of sub-problem solves. As the number of sub-problems grows, the presence of this barrier synchronizer can lead to poor parallel efficiency, due to the variability in sub-problem solve times. This issue is not a primary concern when the number of sub-problems is $O(100)$ or less, such that the results presented in

Section V are not severely impacted. However, we do note that this barrier synchronization point can be relaxed under certain general conditions, yielding improved and more scalable parallel efficiency.

In contrast to the strictly line case, PH in the mixed-integer case is not provably convergent. In particular, the presence of integer decision variables can induce cycling behavior. However, effective techniques for detecting and breaking cycles have been recently introduced (e.g., see [9]). Further, accelerators are typically necessary to improve PH convergence (even in the linear case), as the linear convergence rates are typically too slow to achieve practical run-times. Specifically, we employ variable fixing (freezing the values of variables that have converged over the past *n* PH iterations) and slamming (forcing early convergence of specific variables that have minimal impact on the objective). Both of these techniques are described fully in [9].

The performance of PH is known to be critically dependent upon the value of the ρ parameter. Poor choices can lead to non-convergence, or extremely slow convergence times. In our PH configuration, we use variable-specific ρ values for unit on/off variables (the only first stage variables), which we compute based on the LMPs (Locational Marginal Prices) of the economic dispatches of scenario sub-problems following PH iteration 0. This strategy is a case of "cost-proportional" rho setting, which has been shown to be an effective technique for PH parameterization [3,9].

A promising method for further accelerating the PH sub-problem solve times involves the formation of scenario "bundles", each yielding a small-scale extensive form stochastic program. These sub-problems can be solved directly by commercial mixed-integer solvers. Intuitively, aggregation of scenarios should yield more rapid agreement in solutions, which in turn reduces the number of PH iterations required for convergence. The down-side involves the complexity of the sub-problems; solve times can grow dramatically as the number of scenarios in a bundle is increased. The question of an optimal number of scenarios in each bundle is an empirical one, which we address in Section V.

Finally, we briefly detail an additional modification of the basic PH configuration, which are required in practice to yield tractable run-times on large-scale problems. Due to the use of variable fixing and variable-specific ρ values, our PH for SUC typically converges quickly over the course of approximately the first 20 iterations. At the end of 20 PH iterations, we observe that between 90% and 95% variables are fixed, yielding a relatively small extensive form (EF) problem, which can be solved directly by a commercial solver such as CPLEX. These strategies minimize the number of PH iterations employed, in turn allowing for tractable run times.

## IV. MODEL AND EXPERIMENTAL ENVIRONMENT

We consider the specific problem of reliability unit commitment, primarily because we want to initially avoid issues relating to the impact of uncertainty on market mechanisms. The reliability UC differs from the day-ahead market in that the ISO integrates its forecasts of load and renewable sources, as well as reserve requirements to schedule generation in excess of day-ahead market commitments in order to maintain the reliability of the system. However, our core approach is extensible to day-ahead unit commitment contexts. We use Carrion and Arroyo's [4] unit commitment model as our core deterministic optimization model, based on our prior computational experience relative to other alternatives [5]. We have extended this baseline to include renewables generation units, which we model as must-take resources.

As our test system, we choose the WECC-240 instance introduced in [1], which provides a simplified description of the western US interconnection. This instance consists of 85 generators and 348 transmission lines. Because it was originally introduced to assess market design alternatives, we have modified this instance to capture characteristics more relevant to reliability assessment, including generator power and ramping limits. The full set of modifications can be obtained by contacting one of the authors.

To model uncertainty, we construct scenarios based on a sampled constant offset from the base load at each bus. Specifically, given a base load time series *l*, we construct a modified load by sampling a shift factor uniformly from the interval [0.9, 1.1], and applying the multiplicative factor to each element in the series *l*. To induce non-correlated noise, we further apply a multiplicative factor, sampled independently for each element, from the interval [0.99, 1.01]. Our objective is to generate scenarios with high (and largely unrealistic) variability in load (which implicitly includes renewables, modeled as negative load) – achieved by uniform sampling over the interval. High variability in scenarios typically poses the biggest challenge for the convergence of PH. We are not concerned with realism in the present study, but rather solver convergence. In contrast, we have developed rigorous stochastic process models for use in our studies on cost-benefit analysis of SUC [14].

We implemented all of our models within the Coopr open-source Python-based library for modeling and optimization (http://www.software.sandia.gov/trac/coopr), specifically using the Pyomo algebraic modeling language [2] and the PySP extension package for stochastic programming [3]. We use CPLEX 12.3 as our sub-problem solver, with default parameter settings unless otherwise noted. Parallelization of PH is performed using the Python Remote Objects library (http://pypi.python.org/pypi/Pyro/), which is integrated with PySP.

All serial experiments are performed using a dual quad-core workstation, with 96GB of RAM and 2.2GHz AMD processors. Parallel experiments were performed on Sandia National Laboratories' Red Sky cluster, whose individual blades consist of two quad-core 2.3GHz Intel X5570 processors and 12GB of RAM. All parallel jobs are allocated a number of processes equal to the number of sub-problems, plus additional processes for executing the PH master algorithm and for coordinating communication among the sub-problem solution processes. We observe that the dependency of our results on this particular architecture is negligible, in that a small-scale cluster could be substituted to achieve qualitatively identical performance.

## V. Preliminary Results

Our objective in this paper is to provide preliminary evidence as to the scalability of PH on modest-sized systems, using WECC-240 as a demonstration case. We consider both run-times and overall solution quality in our analysis; these two factors are critical in assessing viability of our approach in industrial contexts. Further, we constraint ourselves to modest-scale parallel computing environments, representative of the systems that ISOs currently possess or are likely to obtain in the near future.

To assess our progress toward this objective, we now consider the results of our parallel PH implementation on a stochastic modified WECC-240 instance with 100 scenarios, constructed using the generative process described in Section IV. Because our objective is to demonstrate run-time and quality scalability of our proposed PH configuration, we do not discuss off-line simulation validation of the resulting solution. However, we have performed this validation, and observe that 100 scenarios in this particular instance are more than sufficient to obtain reliable out-of-sample behavior.

The results of our experiment are shown in Table I. The first column in this table describes the algorithm configuration, while the second and third columns respectively report the run-time required by the algorithm and the optimality gap (computed off-line) of the resulting solution. Run-times are reported in minutes, rounded to the nearest 15 minute increment. Solution quality is quantified as the optimality gap relative to a provably optimal solution to our test instance, computed off-line using a multi-day run of CPLEX. All experiments were executed using the computational environment described in Section IV.

Table I. Run-time and solution quality results for different algorithmic configurations on modified WECC-240 stochastic instance with 100 scenarios

| *Algorithm Configuration* | *Run-time (minutes)* | *Optimality Gap (%)* |
|---|---|---|
| Extensive Form | 1440 | N/A |
| PH-Serial | 840 | 2.5% |
| PH-Parallel (No Bundling) | 15 | 2.5% |
| PH-Parallel (Bundling) | 15 | 1.5% |

We first consider the results observed while solving the extensive form (EF) of our stochastic WECC-240 instance, using CPLEX 12.3 on the workstation described in Section IV. Here, we imposed a limit of a day of wall clock time on the run. However, despite the availability of 8 cores (16 threads) for CPLEX, no feasible incumbent solution was obtained by the end of the run. This result clearly illustrates the difficult of stochastic unit commitment, when tackled via direct (non-decomposition-based) methods. Ultimately, we re-ran this instance without the time limit, and obtained a provably optimal solution (within a mipgap tolerance equal to 0.1%) in over a week. We further note that the memory consumptions are significant, requiring tens of GB of RAM to store the corresponding branch-and-cut tree.

Next we consider the results obtained by executing our PH algorithm configuration in serial, on the same workstation as our extensive form runs. In our experimental context, "serial" implies serial execution of CPLEX sub-problem solves, during each PH iteration. However, all 8 machine cores were dedicated to each sub-problem solve, implying parallelism in the branch-and-cut process. In contrast to the extensive form results, PH obtains a solution within 2.5% of optimal after 16 hours of wall clock time. This result demonstrates the relative effectiveness of PH, in terms of time-versus-quality tradeoff. However, the absolute wall clock time is still too large for deployment in real-world operational contexts.

We now change experimental contexts in terms of hardware, shifting from our 8-core workstation to Sandia's Red Sky cluster. Although times are not directly comparable, we do observe that each node in Red Sky corresponds roughly (in terms of CPU power, not RAM) to our experimental workstation.

Next, we consider results for PH in parallel, utilizing 100 nodes of the Red Sky cluster. Each sub-problem is allocated a single 8-processor node (executing CPLEX 12.3), along with its corresponding memory. As expected, the solution quality is identical to that obtained by PH in serial. However, the run-time is significantly reduced, to a total of 15 minutes. This represents a reduction factor of 56. The primary source of reduction in parallel efficiency is due to the combination of variability in PH sub-problem solve times and the presence of a barrier synchronization point in the core PH algorithm. As indicated before, we have developed techniques for mitigating this issue (via asynchronous PH), but a detailed discussion is beyond the scope of the present paper. Overall, the results are quite promising. In particular, the use of modest-scale parallelism enables solution of this complex stochastic unit commitment problem in tractable run-times.

Finally, we consider the results for parallel PH with scenario bundling. We experimented with various numbers of scenarios per bundle. Overall, we obtained the best time versus quality performance tradeoff using bundles with 2 scenarios. In such PH runs, each of the 50 bundles is allocated a single Red Sky node. While the resulting sub-problems are more difficult than individual scenario sub-problems, 2-scenario bundles force more rapid convergence in terms of the PH gap, and yield more rapid variable fixing. The reduction in sub-problem size yielded by variable fixing then balances the increase in sub-problem solve times. Overall, we observe similar run-time performance to parallel PH with no bundling, but obtain an improvement in overall solution quality.

Overall, through the careful use of parallelization and algorithm configuration, we observe that PH can obtain near-optimal solutions to very difficult stochastic unit commitment problems, in tractable run-times. While serial PH yields significantly better results than straightforward solution of the extensive form problem, modest degrees of parallel computing are necessary to achieve operational response times. Scenario bundling results in higher-quality solutions, but no significant difference in PH run-time.

## VI. Conclusions

Driven by the need to incorporate more direct uncertainty analysis as renewables penetration rates increase, researchers have conduced a significant amount of research into the development of algorithms for solving the stochastic unit commitment problem. Yet, these advances have not yet impacted practice, primarily due to the computational challenge of the problem. In this paper, we propose a decomposition-based strategy for solving the stochastic unit commitment problem, based on the progressive hedging (PH) algorithm of Rockafellar and Wets. Leveraging various advances over the past decade in PH configuration and tuning, we demonstrate tractable (sub-hour) solve times on the modest-scale WECC-240 instance, with a large number of scenarios. We leverage small-scale commodity clusters to achieve this performance, representing computing capabilities either currently deployed or likely in the near future to be deployed at ISOs and utilities. We are engaged in future research efforts, working to further advance PH acceleration mechanisms in order to achieve the same solve times on instances an order of magnitude larger, e.g., representing a full-scale ISO or utility.

## Acknowledgments

This research was sponsored by the US Department of Energy's ARPA-e Green Energy Network Integration (GENI) program. 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.

## References

[1] Price, J.E. *Reduced Network Modeling of WECC as a Market Design Prototype*. Proceedings of the 2011 IEEE Power and Energy Society General Meeting.

[2] W.E. Hart, J.P. Watson, and D.L. Woodruff. *Pyomo: Modeling and Solving Mathematical Programs in Python.* Mathematical Programming Computation 3(3). 2011

[3] J.P. Watson, D.L. Woodruff, and W.E. Hart. *PySP: Modeling and Solving Stochastic Programs in Python*. Mathematical Programming Computation 4(2). 2012.

[4] M. Carrion and J.M. Arroyo. *A Computationally Efficient Mixed-Integer Linear Formulation for the Thermal Unit Commitment Problem*. IEEE Transactions on Power Systems 21(3). 2006.

[5] J. Ostrowski, M.F. Anjos, A. Vanneli. *Tight Mixed Integer Linear Programming Formulations for the Unit Commitment Problem*. IEEE Transactions on Power Systems 27(1). 2012.

[6] R.T. Rockafellar and R.J.B. Wets. *Scenarios and Policy Aggregation in Optimization Under Uncertainty*. Mathematics of Operations Research 16(1). 1999.

[7] C.C. Caroe and R. Schultz. *Dual Decomposition in Stochastic Integer Programming*. Operations Research Letters 24(1-2). 1999.

[8] R. Van Slyke and R.J.B. Wets. *L-Shaped Linear Programs with Applications to Optimal Control and Stochastic Programming*. SIAM Journal of Applied Mathematics 17. 1969.

[9] J.P. Watson and D.L. Woodruff. *Progressive Hedging Innovations for a Class of Stochastic Mixed-Integer Resouce Allocation Problems*. Computational Management Science 8(4). 2011.

[10] S. Takriti, J.R. Birge, E. Long, "*A stochastic model for the unit commitment problem,*" IEEE Transactions on Power Systems, vol.11, no.3, pp.1497-1508, Aug. 1996.

[11] A.J. King and S.W. Wallace. *Modeling with Stochastic Programming*. Springer. 2012.

[12] A. Papavasiliou. Coupling Renewable Energy Supply with Deferrable Demand. PhD Thesis, University of California Berkeley. 2011.

[13] P.A. Ruix, R.C. Philbrick, E. Zack, K.W. Cheung, and P.W. Sauer. *Uncertainty Management in the Unit Commitment Problem*. IEEE Transactions on Power Systems 24(2). pp. 642-651. 2009.

[14] Y. Feng, D. Gade, S.M. Ryan, R.J.B. Wets, D.L. Woodruff, and J.P. Watson. *A New Approximation Method for Day-Ahead Load Modeling*. Submitted to IEEE Power and Energy Society General Meeting, 2013.