A Fast Penalty-Based Gauss-Seidel Method for Stochastic Unit Commitment With Uncertain Load and Wind Generation

ANANTH M. PALANI (Member, IEEE), HONGYU WU (Senior Member, IEEE), AND MEDHAT M. MORCOS (Life Senior Member, IEEE)
Department of Electrical and Computer Engineering, Kansas State University, Manhattan, KS 66506 USA
CORRESPONDING AUTHOR: H. WU (HongyuWu@ksu.edu)

This work was supported in part by the U.S. National Science Foundation under Grant 1929147.

ABSTRACT Stochastic network-constrained unit commitment (S-NCUC) can be used to manage the uncertainty of an increasing penetration level of renewable energy effectively. However, the drawbacks of the progressive hedging algorithm (PHA) based solutions are that they are not provably convergent due to the non-convexity of S-NCUC. Additionally, the solution obtained is usually fractional-valued (non-binary) and therefore not readily implementable. In this paper, we apply a novel Penalty-Based Gauss-Seidel (PBGS) algorithm in solving S-NCUC using an exact augmented Lagrangian representation with proven convergence. To improve the computational efficiency of the PBGS, we further propose an accelerating technique with rigorous proof to skip solving scenarios when certain conditions are met during iterations. We numerically validate the proposed algorithms on the IEEE 118-bus system and a practically-sized ERCOT-like system, both with variable wind generation. Numerical results demonstrate the efficacy of the proposed algorithms in yielding high-quality S-NCUC solutions. The merits of the proposed algorithms are also revealed in comparison with PHA and extensive-form (EF) based mixed-integer programming solutions.

INDEX TERMS Exact augmented Lagrangian, Frank-Wolfe algorithm, Gauss-Seidel method, mixed-integer programming, progressive hedging algorithm, and stochastic programming.

NOMENCLATURE

A. PARAMETERS

| Symbol | Description |
|--------|-------------|
| NT | Number of time periods |
| N | Number of buses |
| NI | Number of generation units |
| S | Number of scenarios |
| NG | Number of segments in production cost curve |
| L | Number of transmission branches |
| t | Index for time periods |
| i | Index for units: 1, 2, ..., NI |
| s | Index for scenarios: 1, 2, ..., S |
| d | Index for cost curve segments: 1, 2, ..., NG |
| Pr^s | Probability of scenario s |
| NL_i | No-load cost of unit i [$/h] |
| SU_i,t | Startup cost of unit i at time t in scenario s [$] |
| D^s_{n,t} | Demand at bus n at time t scenario s [MW] |
| IC_{d,i} | Incremental cost of unit i, seg d [$/MW] |
| β | PBGS parameter |
| γ | PBGS Step size |
| ρ, ρ, ρ | Penalty factors, initially ρ = ρ = ρ in PBGS |
| Φ | Lower bound |
| φ | Production cost inside inner iteration of PBGS |
| ε | PBGS tolerance limit for Δφ |
| P_G^i/P_G^0 | Max/ min power output of unit i [MW] |
| RU_i/RD_i | Unit i ramp up/down limit [MW/h] |
| TU_i/TD_i | Unit i min.up/down time [h] |
| LF_l/LF_l | Branch flow limit [MW] for branch l |
| VOLL | Value of lost load [$/MWh] |
| VOOB | Value of overloaded branch [$/MWh] |
| SF_l,n | Shift factor of line l at bus n |

B. VARIABLES

| Symbol | Description |
|--------|-------------|
| I^s_{l,t} | State of unit i at time t in scenario s, 1 for ON and 0 for OFF |
| IU^s_{l,t} | Startup indicator of unit i at time t in scenario s; superscript s is omitted in EF |
| ID^s_{l,t} | Shutdown indicator of unit i at time t in scenario s; superscript s is omitted in EF |
| P^s_{d,i,t} | Dispatch of unit i at segment d at time t in scenario s [MW] |
occur. This local minimum is an exact augmented Lagrangian, also known as a sub-optimal solution. The convergence of the augmented Lagrangian method depends on the implementation of the penalty function. The augmented Lagrangian method can be used to solve constrained optimization problems in a finite number of iterations, provided that the penalty function is chosen appropriately.

A detailed analysis of the PHA application to stochastic unit commitment (UC) is given in [19]. Obtaining a lower bound for MIP with an application to stochastic UC is discussed in [20]. Nevertheless, PHA is originally devised for a continuous convex program and is not provably convergent for the non-convex S-NCUC problem. The solution to the dual problem is generally primal infeasible and the once relaxed system-wide constraints may not be satisfied. An additional algorithm is required to restore the primal feasibility from a Lagrangian dual solution [21]. Therefore, it is desirable to directly obtain a primal feasible solution from the Lagrangian dual iterations. This gives rise to exact augmented Lagrangian, a class of exact penalty methods [22], [23] whose objective is to solve a constrained optimization (primal) problem through an unconstrained optimization problem that has the same local (global) solutions as the primal problem. In a non-convex problem like S-NCUC, only the local minimum is often found. This local minimum is an exact augmented Lagrangian solution and is referred to as an exact solution hereinafter.

Once the local minimum is obtained, it is imperative to measure the quality of that local minimum. Nevertheless,
the Lagrangian dual solution obtained from PHA is very sensitive to the penalty factor chosen [24]. The sensitivity adversely affects the convergence rate of PHA and may even result in cyclic behaviors [18], [24]. Heuristic methods were developed within PHA to choose an effective penalty factor, detect and break cyclic behaviors, as well as select terminating criteria [18]. A Frank-Wolfe algorithm combined with PHA (FW-PHA) has been recently applied in [25] to obtain better Lagrangian dual bounds of S-NCUC. However, the following critical questions remain unresolved for S-NCUC:

1) How can we devise an exact augmented Lagrangian function to acquire an exact solution?

2) If an exact solution is attained, how can we find a high-quality lower bound capable of measuring the quality of the exact solution efficiently?

To address the above questions, this paper applies a novel Penalty-Based Gauss-Seidel (PBGS) algorithm [26] with an exact augmented Lagrangian representation to solve S-NCUC within a scenario-based decomposition framework. To improve the computational efficiency of PBGS, we propose an accelerating technique that skips solving scenarios as long as certain conditions are satisfied. We numerically validate these algorithms on the IEEE 118-bus and an ERCOT-like large-scale systems. Numerical results demonstrate the high quality of the PBGS solution and the efficacy of the proposed algorithms. Additionally, we compare the proposed algorithms with other prevailing S-NCUC methods such as PHA and extensive-form-based MIP solutions. The main contributions of the paper are three-fold.

- We apply the PBGS algorithm for the first time to S-NCUC, including a positive-basis exact penalty representation and a binary implementable calculation to produce an exact solution. The exactness guarantees the primal feasibility of a Lagrangian dual solution.
- We propose a novel accelerating technique with a mathematical proof to skip solving scenarios when certain conditions are met during the PBGS iteration.
- The proposed method is validated and compared with other algorithms, especially on a practically sized ERCOT-like system.

The remainder of the paper is organized as follows. In Section II, we briefly describe the S-NCUC formulation, followed by a detailed discussion on the PBGS algorithm followed by Fast PBGS. We validate the proposed algorithms and present the numerical test results in Section III. Conclusions and future work are provided in Section IV.

II. PROBLEM FORMULATION AND PROPOSED SOLUTIONS

In this section, we first lay out an S-NCUC formulation and then introduce the PBGS method, where a definition of an exact penalty representation is given. Finally, we discuss our proposed acceleration feature for PBGS which we call Fast PBGS.

A. STOCHASTIC-NETWORK CONSTRAINED UNIT COMMITMENT

We use an S-NCUC formulation that originated from Flexible Energy Scheduling Tool for Integrating Variable Generation (FESTIV) [27], [28]. The objective of the two-stage S-NCUC is to minimize the expected system operating cost across scenarios as follows:

$$
\zeta = \minimize_{i} \sum_{t=1}^{NT} \sum_{l=1}^{NI} \left( NL_{i,t} \cdot Z_{i,t} + SU_{i,t} \cdot IU_{i,t} \right) \\
+ \sum_{s=1}^{S} \sum_{d=1}^{NG} \left( P_{d,s,t} \cdot IC_{d,s} \right) \\
+ \sum_{s=1}^{S} \left( AL_{s,t} \cdot VOLL + \sum_{l \in L} \left( BrSI_{l,s,t} \right) \cdot VOOB \right) 
$$

The first row in the objective function (1) represents the first-stage no-load cost and unit startup cost. The second row in (1) shows the second-stage expected production cost for all possible scenarios considered. This expected production cost is the summation of scenario production cost multiplied by the probability of each scenario. Slack variables on the power balance constraint (2) and power flow constraint (3) are also considered in (1) to maintain the feasibility of S-NCUC solutions, which are penalized as shown in the third and fourth row of (1), respectively. The objective function (1) is subject to the following individual scenario constraints (2)-(8):

1) Power balance constraint

$$
\sum_{i=1}^{NI} P_{g,s,t,i} = \sum_{n=1}^{N} D_{n,t} - LL_{s,t} + AL_{s,t} \quad \forall t \in NT, \forall s \in S
$$

2) Power flow constraints

$$
\begin{align*}
LF_{s,t,l} & \leq LF_{s,t} + BrSI_{l,s,t} \\
LF_{s,t,l} & \geq -LF_{s,t} + BrSI_{l,s,t} \\
\forall l & \in L, \forall t \in NT, \forall s \in S
\end{align*}
$$

3) Power flow equation

$$
LF_{s,t,l} = \sum_{n=1}^{N} SF_{s,t,n} \cdot \left( \sum_{i=1}^{NI} P_{g,s,t,i,n} - D_{n,t} - LL_{s,t} + AL_{s,t} \right) \\
\forall l & \in L, \forall t \in NT, \forall s \in S
$$

4) Generator minimum and maximum constraints

$$
Z_{i,t} \cdot P_{g,s,t} \leq P_{g,s,t} \leq \overline{P}_{g,s,t} \cdot Z_{i,t} \quad \forall i \in NI, \forall t \in NT, \forall s \in S
$$
5) Generator ramp up and ramp down constraints

\[ P_{g,i,t}^6 - P_{g,i,t-1}^6 \leq RU_i \]
\[ P_{g,i,t}^d - P_{g,i,t-1}^d \leq RD_i \]
\[ \forall i \in NI, \forall t \in NT, \forall s \in S \] (6)

6) Generator minimum up and down constraints

\[ T + TU_{i-1} \]
\[ \sum_{t=T}^{T + TU_{i-1}} I_{i,t} \geq TU_i \]
\[ \forall i \in NI, \forall t \in NT \] (7)

Constraints (2)-(4) are system-wide constraints whereas constraints (5)-(7) are individual unit constraints in each scenario. Other prevailing constraints such as segment generation limits are also considered here. In addition, contingency and reserves constraints can be taken into consideration in the S-NCUC formulation [28]. The feasible region of decision variables is given as:

\[ \{ Z_{i,t}, p_{d,i,t} \} \in \Lambda, \forall i \in NI, \forall t \in NT, \forall d \in NG \] (8)

where \( \Lambda \) is determined by all the above constraints.

Problem (1)-(8) is also known as the extensive form (EF) of the two-stage S-NCUC, in which even a moderate number of scenarios can result in a computational burden that quickly exceeds the capability of any state-of-the-art MIP solver. In addition, the computational burden increases exponentially with the size of the problem using the branch-and-cut method [26]. This is why scenario-based decomposition frameworks such as PHA are used to solve a large-scale S-NCUC problem [21]. It should be noted that we restrict the scope of this paper to the scenario-based decomposition method, which is one of the widely used approaches in solving large-scale S-NCUC problems.

**B. PROPOSED FAST PENALTY-BASED GAUSS-SEIDEL ALGORITHM**

The scenario-based decomposition framework separates the first-stage decisions into each individual scenario (e.g., \( l_{i,t}^s, UI_{i,t}^s \)) and introduces a coupling constraint that enforces the first-stage S-NCUC decisions of non-quick-start generators (NQGs) to be the same across all scenarios. This constraint for NQGs, termed as a non-anticipativity constraint (NAC), is given below:

\[ l_{i,t}^s = Z_{i,t}, \forall i \in NI, \forall t \in NT, \forall s \in S \] (9)

As UCs of quick-start generators (QSGs) can vary across the scenarios considered, constraint (9) is not imposed on quick-start units. By relaxing constraint (9), the augmented Lagrangian function is defined as:

\[ \varphi^+(I, p, \omega, \rho) = \min_{s \in S} \sum_{i=1}^{NI} \sum_{t=1}^{NT} \left( NL_{i,t}^s \cdot l_{i,t}^s + SU_{i,t}^s \cdot IU_{i,t}^s \right) \]
\[ + \sum_{d=1}^{NG} \left( P_{d,i,t}^s \cdot IC_{d,i} + \omega_t^d \cdot (l_{i,t}^d - Z_{i,t}) \right) \]
\[ + \alpha \cdot \phi_{\rho} \left( I_{i,t}^s - Z_{i,t} \right) \cdot \left( VOLL \right) \] (10)

where \( I \in \{0, 1\}^{NI \times NT \times S}, p \in \mathbb{R}^{NG \times NI \times NT \times S}, \) and \( \omega, \rho \in \mathbb{R}^{NI \times NT \times S}; \)

\( \phi_{\rho} \left( I_{i,t}^s - Z_{i,t} \right) \) is an augmenting function associated with NAC (9) and a penalty factor \( \rho; \alpha \) is a scalar. In (10), the terms regarding the unit no-load cost, startup cost, and expected production cost across all scenarios are similar to (1) with the exception that the first-stage costs are written in the form of each individual scenario by using the split first-stage decision variables, i.e., \( l_{i,t}^s, UI_{i,t}^s \). In addition, the NAC constraint (9) is relaxed using an augmented Lagrangian function.

We enforce \( \sum_{i \in S} \sum_{t=1}^{NT} p_{d,i,t}^s \cdot \omega_t^d = 0 \) for the augmented Lagrangian function (10). The probability-weighted sum of the dual variable equaling zero is a necessary condition [16]. This condition makes the problem bounded from below. Under this imposed condition, \( Z_{i,t} \) associated with \( \omega_t^d \) in the third line of (10) vanishes and the augmented Lagrangian function (10) is decomposed into \( S \) subproblems, each representing an individual-scenario NCUC problem to be solved. Note that the proposed PBGS and widely used PHA solve the augmented Lagrangian function (10) using different augmenting functions \( \phi_{\rho}. \)

**Definition 1** (Exact Penalty Representation [29]): A given augmenting function \( \phi_{\rho} \) in (10) is an exact penalty representation, if there exists a dual vector \( \omega \), for all \( \rho \) sufficiently large, such that:

\[ \zeta = \varphi^+(I, p, \omega, \rho) \] (11)

and

\[ \argmin_{\omega} \zeta = \argmin_{\omega} \varphi^+(I, p, \omega, \rho) \] (12)

Definition 1 indicates that, with an exact augmenting function \( \phi_{\rho} \), an optimal solution to the augmented Lagrangian (10) is also a (local) optimal solution to the extensive form (1) of S-NCUC. The exactness implies that the optimal Lagrangian dual solution to (10) is directly primal feasible with all the relaxed NAC constraints being satisfied. Hence, no additional algorithm is required to restore the primal feasibility from a Lagrangian dual solution. It is worth mentioning that Definition 1 emphasizes the feasibility rather than the optimality of a solution to augmented Lagrangian (10). Specifically, an optimal solution to (10) is a local minimum to
extensive form (1) due to the non-convexity of S-NCUC [23]. Therefore, a quantitative assessment of the quality of a solution to (10) is needed.

An important question is how to construct an exact augmenting function $\psi^s_\beta$. We employ an $l_1$-norm-like function based on a semi-Lagrangian approach [26]. Unlike the semi-Lagrangian approach wherein an equality constraint is reformulated as a pair of inequality constraints and the Lagrangian relaxation is applied to either of the pair [30], here we reformulate NAC (9) as two inequality constraints and then relax both constraints using two penalty factors ($\rho_s, \bar{\rho}$) as follows:

$$\psi^s_\beta (l^s_{i,t} - Z_{i,t}) := \rho^s_{i,t} \cdot [I^s_{i,t} - Z_{i,t}]^- + \bar{\rho}^s_{i,t} \cdot [Z_{i,t} - I^s_{i,t}]^-$$ (13)

where $[-]$ represents a positive basis, which is defined as $-\min \{0, \cdot \}$. Let $u = [I^s_{i,t} - Z_{i,t}]^-$ and $\bar{u} = [Z_{i,t} - I^s_{i,t}]^-$. Equation (13) can be implemented as follows:

$$\psi^s_\beta (l^s_{i,t} - Z_{i,t}) := \rho^s_{i,t} \cdot u + \bar{\rho}^s_{i,t} \cdot \bar{u}$$

$$u \geq 0, \quad \bar{u} \geq 0, \quad \bar{u} \geq (l^s_{i,t} - Z_{i,t})$$ (14)

Augmenting function (13) is an exact penalty representation since it meets the following three criteria:

- $\psi^s_\beta (0) = 0$
- $\psi^s_\beta (l^s_{i,t} - Z_{i,t}) \geq \delta > 0, \forall \left( l^s_{i,t} - Z_{i,t} \right) \notin V$
- $\psi^s_\beta (l^s_{i,t} - Z_{i,t}) \geq v \left\| l^s_{i,t} - Z_{i,t} \right\|_\infty, \forall \left( l^s_{i,t} - Z_{i,t} \right) \in V$

for some open neighborhood $V$ of 0 and positive scalars $\delta, v > 0$. The exactness proof of augmenting function (13) is given in Theorem 5 of [29].

It is worth noting that a common choice for the augmenting function $\psi^s_\beta$ in the literature is the use of norms. For example, the PHA algorithm [17] uses the square of $l_2$ norm, i.e.,

$$\frac{\rho}{2} \left\| l^s_{i,t} - Z_{i,t} \right\|^2_2$$

as the augmenting function. As indicated in the third criterion above, the value of the augmenting function must be greater than or equal to an infinity norm in the neighborhood of zero. The PHA algorithm does not satisfy this criterion and therefore is not an exact penalty representation. For any norm used as penalty function, or any function that satisfy exact penalty representation criteria listed above, for any $\omega \in \mathbb{R}^{N_I \times N_T \times S}$, there exists a finite $\rho^s_\omega$ such that $\psi^s_\omega (\omega) = \zeta$, allows us to choose $\omega = 0$. This is detailed in proposition 8 of [29] and in definition 11.60 of [22]. Therefore, we set $\omega = 0$ and omit $\omega$ from subsequent discussions for the simplicity of representation.

The second important consideration is in the calculation of implementable $Z_{i,t}$. Since UC takes a binary value, it is imperative to maintain $Z_{i,t}$ binary. Calculation of $Z_{i,t}$ can be accomplished by minimizing (14) with respect to $Z_{i,t}$ using fixed $l^s_{i,t}$ over all the scenarios. Per [26], if the problem is restricted to take only binary values, we can calculate $Z_{i,t}$ as follows at every iteration:

$$Z_{i,t} := \begin{cases} 1, & \text{if } \sum_{s \in S} (1 - l^s_{i,t}) \cdot \rho^s_{i,t} < \sum_{s \in S} l^s_{i,t} \cdot \bar{\rho}^s_{i,t} \\ 0, & \text{if } \sum_{s \in S} (1 - l^s_{i,t}) \cdot \rho^s_{i,t} > \sum_{s \in S} l^s_{i,t} \cdot \bar{\rho}^s_{i,t} \\ 0 \text{ or } 1, & \text{otherwise} \end{cases}$$ (15)

As shown, the implementable takes a UC consensus on the majority of scenarios weighted by the penalty factors, leading to a binary value. In contrast, the PHA algorithm calculates the implementable via a weighted average of UC, resulting in a fractional value that requires further conversion to a binary value.

After $Z_{i,t}$ is obtained, $\rho$ at the $k$th iteration is updated as:

$$\rho^s_{i,t,k} := \rho^s_{i,t,k-1} + \gamma \cdot [l^s_{i,t,k} - Z_{i,t,k}]^- \quad \forall s \in S$$

$$\bar{\rho}^s_{i,t,k} := \bar{\rho}^s_{i,t,k-1} + \gamma \cdot [Z_{i,t,k} - I^s_{i,t,k}]^- \quad \forall s \in S$$ (16)

In (16), we update the two penalty factors separately based on which direction NAC (9) is violated. Unlike the PHA algorithm, which uses the square of $l_2$ norm as the augmenting function to penalize NAC violations uniformly, the penalty factor updated in (16) offers more granular modifications to satisfy the NAC constraints. Note that scalar $\alpha$ in (10) is increased exponentially over iterations to improve the convergence, which is defined as follows:

$$\alpha := \beta^{k-1} - 1$$ (17)

where $\beta \in (1, 2]$. A detailed explanation for this acceleration factor is given in Chapter 4 of [26].

The third critical factor to be considered is how we can accelerate the PBGS solution in view that an individual scenario may have to be solved many times, each time corresponding to one PBGS iteration. The idea behind the fast PBGS algorithm is to skip an individual scenario that is implementable always take binary values in each PBGS iteration.

Corollary 1: For scenario $s$ in PBGS, $(l^s_k, \bar{l}^s_k)$ is also the optimal solution at the $k + 1$th iteration if the following two conditions are satisfied at the $k$th iteration:

1. The optimal commitment decisions obtained are the same as the pre-update implementable, i.e.,

$$l^s_k = Z_{k-1}$$

2. The post-update implementable remains unchanged, i.e.,

$$Z_k = Z_{k-1}$$ (19)

The proof of Corollary 1 is provided in Appendix. Note that Corollary 1 is made possible in the Fast PBGS through how the implementable is calculated in (15). In the first condition (18), both the UC decision variables and the implementable always take binary values in each PBGS iteration.
However, this is not the case in PHA as its implementable is fractional-valued during iterations. We further propose the following proposition to generalize Corollary 1 to the entire iterative process of the Fast PBGS.

**Proposition 1**: Scenario \( s \) does not need to be solved as long as equations (18) and (19) hold in successive iterations.

Proposition 1 can be easily proven by induction. Based on the above discussion, we show the proposed Fast PBGS algorithm in A1, in which indices \( i \) and \( t \) are omitted for brevity.

It is worthwhile to discuss the initialization of the Fast PBGS as shown in Line 5 of A1. The PBGS algorithm in [26] suggests initializing \( Z \) as follows:

\[
Z := 0
\]

or rounding the weighted average of all unit statuses across scenarios for the respective unit and time period as follows:

\[
Z_{i,t} := \left\lfloor \sum_{s \in S} Pr^s \cdot I_{i,t}^s \right\rfloor, \quad \forall t \in NT, \forall i \in NI
\]

Here, we propose a third initialization method by using the unit statuses in a scenario from Line 3 of A1 that has the maximum number of online unit statuses.

\[
Z_{i,t} := \hat{I}_{i,t}^s, \quad \forall t \in NT, \forall i \in NI
\]

where \( \hat{s} \in S \) denotes a scenario such that

\[
\sum_{i=1}^{NI} \sum_{t=1}^{NT} \hat{I}_{i,t}^s \geq \sum_{i=1}^{NI} \sum_{t=1}^{NT} \hat{I}_{i,t}^j, \forall j \neq \hat{s}, j \in S
\]

According to our numerical experience, the third initialization method could yield results faster than the first two methods.

### III. NUMERICAL RESULTS

In this section, we validate the proposed algorithms on two systems as follows:

1. The IEEE 118-bus system [31]: The modified IEEE 118-bus system has 118 buses, 186 branches and 54 generators that include 10 WTGs. The peak load of the system is 3,974 MW. We generate 10 scenarios and each scenario contains 57,048 constraints, 51,577 continuous variables, and 4,032 integer variables.

2. The ERCOT-like system: The ERCOT-like system has 7226 buses, 8853 branches and 725 generators, including 178 WTGs. The total installed capacity is over 80,000 MW including 21,582 MW of wind. We generate 30 scenarios on this system. Each scenario has 1,595,953 constraints, 1,837,921 variables, and 51,240 first-stage UC decisions. After using the line-screening technique, the number of non-zero coefficients in CPLEX is reduced from 1,013,282,556 to 10,439,695.

The autoregressive moving average (ARMA) is used to generate a number of scenarios with uncertain wind and load in each system. Specifically, a standard deviation (STD) of 3% in load and 6% in the wind is used on the IEEE 118-bus systems (this scenario set is called IEEE118-10-s1), whereas an STD of 6% and 10% for load and wind on the ERCOT-like system. Note that the focus of this work is to propose and validate a novel PBGS solution for S-NCUC. The proposed solution can accommodate any state-of-the-art forecasting and scenario generation techniques.

All algorithms, including EF, PBGS, and FW-PHA are implemented in MATLAB within the FESTIV framework [27]. The MIP solver used is CPLEX 12.9 in GAMS 28.1. We set the CPLEX MIP optimality gap to 0.1% on the IEEE 118-bus systems and 1% on the ERCOT-like system. The computing platform is Intel Xeon CPU E5-2640 dual processors with 256 GB RAM.

#### Algorithm A1 Fast Penalty-Based Gauss-Seidel Algorithm

1. **Initialize** \( \rho, \gamma, \beta, \epsilon, l_{\max}, k, \omega \)
2. **for each** \( s \) **in** \( S \)
   3. \( (I^s, p^s) \leftarrow \arg\min_{p^s, \omega^s} \{ \psi^+(I^s, p^s, \omega^s, \rho^s = 0) \} \)
4. **end for**
5. **Calculate** \( Z^0 \leftarrow \text{round}(\sum_{s \in S} Pr^s \cdot I^s) \)
6. **while**
   7. **for** \( l \leftarrow 1, \ldots, l_{\max} \)**
     8. **for each** \( s \) **in** \( S \)**
       9. If \( (Z^l \neq Z^{l-1}) \) or \( (Z^l \neq I^s) \)
       10. \( (I^s, p^s, \phi^s) \leftarrow \arg\min_{p^s, \phi^s} \{ \phi^+(I^s, p^s, \omega^s, \rho^s) \} \)
     11. **end If**
   12. **end for**
   13. **Calculate** \( Z^l \) **using** (15)
   14. \( \Gamma \leftarrow \phi^l - \phi^{l-1} \)
   15. if \( ||\Gamma|| \leq \epsilon \) or \( l = l_{\max} \)
   16. \( Z \leftarrow Z^l \)
   17. **break**
   18. **end if**
   19. \( l \leftarrow l + 1 \)
20. **end for**
21. if \( \sum_{s \in S} ||I^s - Z||^2 = 0 \)
22. **return** \( Z \)
23. else
   24. **Update** \( \rho^k \) and \( \rho^l \) **using** (16)
   25. **Update** \( \alpha \) **using** (17)
   26. \( k \leftarrow k + 1 \)
27. **end if**
28. **end while**

### A. EVALUATION OF FAST PBGS RESULTS

In this subsection, we evaluate the quality of the proposed Fast PBGS solutions using the following methods for comparison:

1. **EF**: a primal method, which solves the EF of S-NCUC by directly using CPLEX MIP solver;
TABLE 1. Solution comparison on the IEEE 118-bus system.

| \(\rho, \beta\) | Operating cost by PBGS | Diff w.r.t EF | Gap w.r.t FW-PHA | Fast PBGS (PBGS) solution time in sequence (min) |
|-----------------|-------------------------|---------------|------------------|-----------------------------------------------|
| 500             | $851,149                | 0.00%          | 1.35%            | 120 (149)                                    |
| 1000            | $851,097                | 0.01%          | 1.34%            | 89 (106)                                     |
| 5,000           | $851,366                | 0.03%          | 1.37%            | 45 (53)                                      |
| 10,000          | $854,284                | 0.37%          | 1.72%            | 27 (49)                                      |

1) THE IEEE 118-BUS SYSTEM

The operating cost by PBGS under different penalty factor \(\rho\), and that of EF and FW-PHA are listed in Table 1.

In Table 1, the PBGS solution under all \(\rho\) values except for 10000 is very close to the EF solution with the gap calculated is less than 0.05\%. When \(\rho = 1000\), the PBGS difference with respect to EF is \(-0.01\%\), signifying that its operating cost is slightly less (better) than that of EF. The negative gap can be attributed to the MIP optimality gap of 0.1\% set in CPLEX. The solution obtained from EF is not true optimal due to the CPLEX optimality gap. This is why the solution is minutely higher than the PBGS solution. For all \(\rho\), the PBGS gap with respect to EF is less than 0.4\%. The EF solution, based on the state-of-the-art commercial MIP solver, is one of the most successful approaches in solving small- to medium-scale S-NCUC problems. Here, we use the EF solution as a benchmark to compare the PBGS solution wherever the EF solution is available. It is observed that the difference in the operating cost between the proposed Fast PBGS method and EF is very small on the IEEE-118 bus system. The results demonstrate the effectiveness of the proposed PBGS solution on the medium-scale S-NCUC problem.

The last column in Table 1 provides two values, i.e., one for the Fast PBGS and the other in parentheses for the PBGS without the proposed accelerating technique. Both methods yield the same solution but with different solution time. It is seen the Fast PBGS is roughly 15\% to 45\% faster than the PBGS without the proposed accelerating technique. Simulations for different \(\rho\) values were carried out while fixing \(\gamma = 1.0\) and \(\beta = 1.1\).

2) THE ERCOT-LIKE SYSTEM

It is important to note that EF is not implemented in the ERCOT-like system due to an enormous computational burden of EF (computationally infeasible). Therefore, only FW-PHA is executed as the lower bound method. Three simulations are carried out with different penalty factors \(\rho\). Table 2 shows the PBGS operating cost and the gap with respect to FW-PHA. It is seen the PBGS yields an exact solution with a gap of less than 0.5\% for the first two penalty factors chosen (\(\rho = 5000\) and \(\rho = 10,000\)). For \(\rho = 50,000\), the PBGS solution results in a higher operating cost and a larger gap (i.e., 1.48\%) than those with a smaller \(\rho\). However, the duality gap of 1.48\% is acceptable on such a large-scale S-NCUC.

It is fascinating to discuss the computational performance of PBGS as this is a common concern in large-scale applications. In Table 2, the column labeled “PBGS solution time in sequence” shows the solution time of stochastic DA-NCUC on the ERCOT-like system with 30 scenarios, where we solve each scenario sequentially without parallelization. Analogous to Table 1, two values, i.e., one for the Fast PBGS and the other in parentheses for the PBGS without the accelerating technique, are shown in the last column of Table 2. It is seen the Fast PBGS is 34\% to 39\% faster than PBGS. In addition, we find that the time saving is more significant when the number of scenarios is increased.

With an increase in \(\rho\), we observe a similar trend as that on the IEEE 118-bus systems. Apparently, if one were to implement the proposed algorithm in the day-ahead operation, the time taken for sequentially solving each scenario of the ERCOT-like system is far from practical. The solution time can be drastically shortened using parallel computing, as shown in the column labeled “estimated PBGS solution time in parallel” in Table 2. The time estimation is based on the longest time taken by a scenario in each iteration plus the time between MATLAB and GAMS interaction through the file I/O activities. The FESTIV environment we use is in an academic setting and currently does not support the parallel execution. We understand that the parallel computing environment is not a standard in ISOs or utilities, which may need hardware upgrades and adjustments to the existing day-ahead market timeline if the stochastic method is implemented.

Computational efficiency is a grand challenge in the power system community when stochastic NCUC is applied in an extremely large-scale system such as the ERCOT-like system. Since the EF is not likely to be solvable for a reasonable number of scenarios, decomposition techniques are required. To the best of our knowledge, the state of art solutions for such a large-scale real-world application are heuristic, such as terminating it prematurely for saving iterations and use a rule-based method to coordinate the implementable afterward. This paper is not intended to solve this grand challenge completely. Instead, we provide an alternate method to get the exact solution of S-NCUC (primal solution). Our simulation
results demonstrate that Fast PBGS is a worthy alternative to the PHA since it results in a lower objective value with faster convergence. In tandem with the proposed scenario-skipping technique, the superior convergence leads to a significant amount of savings in computational time. Fast PBGS can still be used for short-term such as week-ahead stochastic planning studies and in DA NCUC for systems smaller than ERCOT-like System. The Fast PBGS algorithm is also beneficial to parallel computing, especially when the number of scenarios is more than the number of computing cores available.

**B. COMPARISON OF FAST PBGS WITH PHA**

Since the scope of this paper is restricted to the scenario-based decomposition methods, we also confine the comparison within this approach. In the literature, there exists a stage-wise decomposition approach, such as Benders decomposition. However, the stage-wise decomposition approach may eventually grow into a computationally intractable problem as the iteration progresses for the size of the problem we considered in this paper. Therefore, the cross-comparison between the scenario-based and stage-based decomposition methods is out of this paper’s scope and will be systematically conducted in future work.

1) **THE IEEE 118-BUS SYSTEM**

Figure 1 shows a comparison of the Fast PBGS with the PHA for the IEEE 118-bus system. In Fig. 1, the second y-axis is the percentage difference in the objective value per (24). A negative percentage difference indicates that the F-PBGS leads to a better solution in terms of the objective value, and vice versa. Table 3 shows the data values of Fig. 1.

![FIGURE 1. Comparison of Fast PBGS with PHA on IEEE 118-bus system with 10 scenarios.](image)

The comparison between the Fast PBGS and PHA is carried out with five different $\rho$ values ranging from 500 to 20,000. The Fast PBGS has one additional tuning parameter $\beta$ that is used in the calculation of convergence accelerator $\alpha$ in (17). For each of these five $\rho$ values, we use two $\beta$ values, i.e., 1.11 and 1.25. It is seen in Fig. 1 when $\beta$ is 1.25, the Fast PBGS solves faster than it does when $\beta$ is set to 1.11, but with a slightly higher objective value. The difference in the solution time between the Fast PBGS and the PHA is much more apparent with a lower $\rho$. Overall, the Fast PBGS yields a much faster solution than the PHA. By using the Fast PBGS, we observe an average of 42% and 61% saving in solution time with $\beta$ values equal to 1.11 and 1.25, respectively.

In Fig. 1, all Fast PBGS simulations converge with zero violations. In contrast, none of the PHA solutions converges except for the case when $\rho = 10000$. The convergence tolerance $\epsilon$ for PHA is set to 0.01. At a lower $\rho$, we set a solution time limit of 180 minutes when PHA does not converge. At a higher $\rho$ (i.e., 5000 and 20000), the PHA algorithm is terminated if no progress in violation is made for five consecutive iterations. Additionally, a terminated PHA algorithm requires rounding-off the fractional implementable values obtained, followed by UC to ensure its primal feasibility.

In Fig. 1, for a comparatively small $\rho$, i.e., when $\rho = 500$ and 1000, the Fast PBGS outperforms the PHA by 2% to 4% in terms of the objective value, respectively. This difference is attributed to the rounding of the PHA implementable values, resulting in overcommitment of generators. When $\rho$ is increased, the difference in the objective values decreases. When $\rho = 10000$, the PHA converges and no rounding is required. The PHA performs slightly, i.e., around 0.5%, better than the Fast PBGS. The above results in Fig. 1 suggest that the significant improvement in computational efficiency makes the Fast PBGS a better choice than the PHA on the medium-scale system.

2) **THE ERCOT-LIKE SYSTEM**

For the ERCOT-like large-scale system, we set $\beta = 1.11$ and focus exclusively on the sensitivity studies with respect to $\rho$ values. Comparative results between the Fast PBGS and the PHA are shown in Fig. 2. It is seen that the Fast PBGS outperformed the PHA in terms of the solution time, whereas the objective value of PHA is slightly better than that of the Fast PBGS in all three cases. The average execution time saved by the Fast PBGS is 50%, at an average cost increase of 0.48% in the objective value across the three cases. None of the PHA solutions converges. The closest is when $\rho = 50000$, the violation reaches 0.06 before it triggers the stopping criterion. Note that the convergence tolerance $\epsilon$ for PHA is set to 0.01. The PHA simulation is terminated if no progress in violation is made for five consecutive iterations. Additionally, a terminated PHA simulation requires rounding-off of the fractional implementable obtained, followed by UC to ensure its primal feasibility.

**TABLE 3. Algorithm comparison for IEEE 118-bus system with 10 scenarios.**

| $\rho$ | PHA Obj. Value | Ef Obj. Value | EF Time in min | Ef Time in min |
|-------|----------------|---------------|----------------|----------------|
| 1000  | $884,333$      | $851,152$     | $852,711$      | $851,259$      |
| 5000  | $861,043$      | $855,976$     | $854,985$      | $854,284$      |
| 10000 | $851,490$      | $851,366$     | $854,284$      | $854,284$      |

| $\rho$ | PHA Time in min | F-PBGS Time in min | F-PBGS Obj. Value |
|-------|-----------------|-------------------|--------------------|
| 1000  | 85              | 89                | $851,152$          |
| 5000  | 37              | 60                | $851,152$          |
| 10000 | 27              | 27                | $851,152$          |
In contrast, these additional heuristics and steps are not needed at all for the Fast PBGS since its convergence is guaranteed and the solution obtained is directly feasible. It is worth mentioning that in the case study of this paper, we never experience the cyclic behaviors of the PHA as reported in [33]. Therefore, the performance of the PHA may be illustrated on the high side if potential cyclic behaviors are considered. Furthermore, some heuristic enhancements [33], such as using a large MIP gap for initial iterations, and initializing successive iterations from the previous solution can be applied to both the Fast PBGS and the PHA to further improve their computational efficiency. The results in Fig. 2 show that the significant improvement in computational efficiency also makes the Fast PBGS a better choice on the practically-sized large-scale system.

As mentioned before, Fast PBGS skips a scenario from solving it if it meets certain conditions. Figure 3 shows the actual number of scenarios solved at each iteration. All 30 scenarios are solved in the first three iterations. Any changes in the implementable between the iterations require solving all 30 scenarios, as shown at Iterations 7, 9, 11, 16, 22, 24 and 40.

For the out-of-sample testing, we create 25 sets, each containing 50 scenarios, and conduct simulations using EF, Fast PBGS, and PHA. We used ARMA with 4% STD for load and 8% STD for wind in creating these scenarios. Unlike the simulations in the previous section that used (21) to initialize the implementable, here we use the initialization method (22). For both Fast PBGS and PHA, we did not include any heuristic-based improvements suggested in the literature [33]. All the Fast PBGS solutions converge to zero violations, while only 10 out of 25 sets converge under the PHA. The PHA termination threshold was set at 0.1 or 60 iterations, whichever comes first.

Fig. 4 compares the objective values of the three methods using the box-whiskers plot. As seen, the median production cost obtained by the EF method is $885,667, which is the lowest among all three methods. In contrast, the median production cost of the Fast PBGS and the PHA are $893,280 and $906,628, respectively. The standard deviations for the EF and the Fast PBGS are $20,339 and $23,367, respectively, whereas the standard deviation of the PHA is $29,836, which is higher than the other two methods.

Fig. 5 compares the computational performance of the three methods. As seen, the computational performance of the Fast PBGS is far better than the PHA by several folds. The mean time taken to obtain a solution by the Fast PBGS is 135 minutes with a standard deviation of 58 minutes, as opposed to the mean time of 661 minutes with a standard deviation of 123 minutes by the PHA algorithm.

The results in Figs. 4 and 5 demonstrate that the Fast PBGS is a worthy alternative to the PHA since it yields a better
production cost with much superior computational efficiency. When comparing the Fast PBGS with the EF at 50 scenarios, the Fast PBGS leads to a slightly higher production cost but better computational efficiency. The advantage in computational efficiency is even more apparent with a larger number of scenarios. This is because the computation time taken by the EF increases exponentially with an increased number of scenarios, whereas that by the Fast PBGS increases linearly [21].

D. DISCUSSION ON PARAMETERS

The Fast PBGS uses parameters $\beta$, $\rho$, and $\gamma$, and therefore it is necessary to select the proper values for them. The parameter $\beta$ is used in (17) to accelerate the convergence. The impact of $\beta$ on the IEEE 118-bus system when it is increased from 1.11 to 1.25 is shown in Fig.1. A high value of $\beta$ makes the convergence faster at the cost of a decrease in the solution quality. As the iteration number gets high, $\alpha$ in (17) grows exponentially and makes the penalty function higher. This may result in loss of load or additional load slack. This situation is can be mitigated by increasing the VOLL or increasing $\alpha$ adaptively only when needed instead of every iteration based on the progress toward the convergence.

We have conducted simulations with a wide range of $\rho$ values for the IEEE 118-bus system and the ERCOT-like system. Like $\beta$, a high value of $\rho$ makes the convergence faster with a compromise on the solution quality. While the $\rho$ is used in the initial penalty, the parameter $\gamma$ is (step size) used in every iteration (16). Although we have kept both $\rho$ and $\gamma$ the same in the results of this paper, we experiment them with different values. A large $\rho$ with a small $\gamma$ took more iterations to eliminate the violation. A small $\rho$ value with a large $\gamma$ did not make that much difference, compared to setting both $\rho$ and $\gamma$ equal. According to our simulations, the suggested initial value of $\rho$ for a medium-sized system is from 5,000 to 10,000. For a large system, it is from 10,000 to 50,000. For the step size $\gamma$, we recommend setting it equal to $\rho$. The acceleration factor $\beta$ can be set to 1.11 to 1.25. However, the above parameter selection is empirical and not conclusive. More theoretical analyses will be performed in our future work.

IV. CONCLUSION

In this paper, we propose a Fast PBGS algorithm that uses an exact augmented Lagrangian function to obtain a primally feasible solution to S-NCUC. For a quantitative evaluation of the PBGS, we use an FW-PHA technique initialized by the PBGS solution to calculate a computationally efficient, high-quality lower bound. Numerical results on the IEEE 118-bus system show the Fast PBGS yields high-quality exact solutions as good as that of solving the EF of S-NCUC using the commercial MIP solver. Numerical results on the ERCOT-like large-scale system validate that the proposed method can effectively solve a real-world S-NCUC problem with a small optimality gap. In both systems, an increase in the chosen penalty factor produces a faster solution but with a higher objective value. Therefore, a fine-tuning of the penalty factor is needed for achieving a trade-off between the solution quality and computational efficiency.

Through a comprehensive comparison between the Fast PBGS and the PHA, it is shown that the proposed PBGS algorithm saves a significant amount of computational time while yielding comparably good solutions. As opposed to the PHA, the Fast PBGS does not require additional heuristics and steps at all since its convergence is guaranteed and the solution obtained is directly feasible. This represents a salient feature of the Fast PBGS algorithm, which is shown as a worthy alternative to the PHA. Additionally, the proposed Fast PBGS algorithm is easy to be implemented as a wrapper over the existing deterministic UC in the ISO’s current practice. Our future work will focus on the cross-comparison between the scenario-based and stage-based decomposition methods. Furthermore, we will improve the computational efficiency of the Fast PGBS algorithm by using expediting techniques [33] and [34].

APPENDIX

PROOF OF SCENARIO-SKIPPING TECHNIQUE IN FAST PBGS

The objective function of PBGS in abbreviated form is given as follows. All the constraints (1)-(7) are applicable.

$$\varphi^+(I,p,\rho) = C(I,p) + \rho\psi(I - Z)$$ (A1)

where $C(I,p)$ is scenario-wise decomposed NCUC objective function and $\rho\psi(I - Z)$ is the augmented Lagrangian term.

**Proof 1:** We use proof by contradiction. Since $(I_k^*,p_k^*)$ denotes the optimal solution obtained in scenario $s$ at the $k^{th}$ iteration of PBGS, i.e.,

$$\{I_k^*,p_k^*\} = \arg\min_{I,p} [C(I_k,p) + \rho_k \cdot \psi(I_k^* - Z_k)]$$ (A2)

Similarly, $(I_{k+1}^*,p_{k+1}^*)$ denotes the optimal solution obtained in scenario $s$ at the $(k+1)^{th}$ iteration of PBGS.

Note that $\rho_{k+1} \geq \rho_k \geq 0$ in PBGS. It is apparent that the optimal solution obtained in $k^{th}$ iteration is a feasible solution to $(k+1)^{th}$ iteration since the solution is from the feasible region $\Delta$ defined in (8). This is defined as follows:

$$\{I_{k+1}^*,p_{k+1}^*\} = \arg\min_{I,p} [C(I_{k+1}^*,p_{k+1}^*) + \rho_{k+1} \cdot \psi(I_{k+1}^* - Z_k)]$$ (A3)

Assume (A2) is not equal to (A3). According to this assumption, in view of its feasibility, we have

$$\varphi^+(I_{k+1}^*,p_{k+1}^*,\rho_{k+1}) < \varphi^+(I_k^*,p_k^*,\rho_k)$$ (A4)

Spelling out (A4)

$$C(I_{k+1}^*,p_{k+1}^*) + \rho_{k+1} \cdot \psi(I_{k+1}^* - Z_k) < C(I_k^*,p_k^*) + \rho_k \cdot \psi(I_k^* - Z_k)$$ (A5)

220  
VOLUME 8, 2021
Applying Corollary 1, conditions (17) and (18) to (A5) we get
\[ C \left( I_{k+1}^{s}, p_{k+1}^{s} \right) + \rho_{k+1} \cdot \psi \left( I_{k+1}^{s} - Z_{k-1} \right) < C \left( I_{k}^{s}, p_{k}^{s} \right) \]
(A6)

For any given solution \( \left( I_{k}, p_{k} \right) \), a higher \( \rho \) lead to a higher objective function value in terms of a higher penalty term due to the structure of \( \phi^{+} \). We have
\[ C \left( I_{k}^{s}, p_{k}^{s} \right) + \rho \cdot \psi \left( I_{k}^{s} - Z_{k-1} \right) \leq C \left( I_{k+1}^{s}, p_{k+1}^{s} \right) + \rho_{k+1} \cdot \psi \left( I_{k+1}^{s} - Z_{k-1} \right) \]
(A7)

Combining (A6) and (A7) results in the following
\[ C \left( I_{k+1}^{s}, p_{k+1}^{s} \right) + \rho_{k} \cdot \psi \left( I_{k+1}^{s} - Z_{k-1} \right) < C \left( I_{k}^{s}, p_{k}^{s} \right) \]
(A8)

Left-hand side of (A8) is exactly the objective function in \( k \)-th iteration of PBGS (A2). Therefore, (A8) cannot be true. Therefore,
\[ C \left( I_{k}^{s}, p_{k}^{s} \right) + \rho \cdot \psi \left( I_{k}^{s} - Z_{k-1} \right) \leq C \left( I_{k+1}^{s}, p_{k+1}^{s} \right) + \rho_{k+1} \cdot \psi \left( I_{k+1}^{s} - Z_{k-1} \right) \]
(A9)

According to (17) in corollary 1, we have
\[ C \left( I_{k}^{s}, p_{k}^{s} \right) = C \left( I_{k}^{s}, p_{k}^{s} \right) + \rho \cdot \psi \left( I_{k}^{s} - Z_{k-1} \right) \]
(A10)

Combining (A8) – (A10) we have
\[ C \left( I_{k}^{s}, p_{k}^{s} \right) = C \left( I_{k}^{s}, p_{k}^{s} \right) + \rho \cdot \psi \left( I_{k}^{s} - Z_{k-1} \right) \]
(A11)

Since (A11) cannot hold, the assumption (A2) not equal to (A3) does not hold.

**APPENDIX B**

**PROOF OF PBGS CONVERGENCE**

**Proof 2:** The PBGS algorithm uses an exact augmenting Lagrangian. According to Theorem 5 of [29], there exists a finite \( \rho^{*} > 0 \) such that for any \( \rho \in \left[ \rho^{*}, \infty \right) \), the following equation holds regarding (A1).
\[ \psi^{+} \left( I, p, \rho \right) = C \left( I, p \right) + \rho \psi \left( I - Z \right) = C \left( I, p \right) \]
(A12)

Equation (A12) holds if and only if
\[ \rho \psi \left( I - Z \right) = 0 \]
(A13)

Equation (A13) implies \( I = Z \) fulfilling the termination criteria in Line 21 of the PBGS algorithm. Due to monotonically increasing \( \rho \), \( \rho \in \left[ \rho^{*}, \infty \right) \) is satisfied within a finite number of iterations.

**ACKNOWLEDGMENT**

The authors would like to thank Dr. Erik Ela of the Electric Power Research Institute (EPRI), Palo Alto, CA, USA, Dr. Mohammad Feizollahi of Georgia State University, Atlanta, GA, USA, and Dr. Fabrizio Oliveira of Aalto University, Aalto, Finland for their valuable suggestions.

**REFERENCES**

[1] E. Ela et al., “Future electricity markets: Designing for massive amounts of zero-variable-cost renewable resources,” IEEE Power Energy Mag., vol. 17, no. 6, pp. 58–66, Nov. 2019.

[2] W. van Ackooij, I. D. Lopez, A. Frangioni, F. Lacalandra, and M. Tahanan, “Large-scale unit commitment under uncertainty: An updated literature survey,” Ann. Oper. Res., vol. 271, no. 1, pp. 11–85, Dec. 2018.

[3] K. De Vos and J. Driesen, “Dynamic operating reserve strategies for wind power integration,” IET Renew. Power Gener., vol. 8, no. 6, pp. 598–610, Aug. 2014.

[4] M. A. Matos and R. J. Bessa, “Setting the operating reserve using probabilistic wind power forecasts,” IEEE Trans. Power Syst., vol. 26, no. 2, pp. 594–603, May 2011.

[5] L. Wu, M. Shahidehpour, and Z. Li, “Comparison of scenario-based and interval optimization approaches to stochastic SCUC,” IEEE Trans. Power Syst., vol. 27, no. 2, pp. 913–921, May 2012.

[6] D. Bertsimas, E. Litvinov, X. A. Sun, J. Zhao, and T. Zheng, “Adaptive robust optimization for the security constrained unit commitment problem,” IEEE Trans. Power Syst., vol. 28, no. 1, pp. 52–63, Feb. 2013.

[7] C. Dai, L. Wu, and H. Wu, “A multi-bundle uncertainty set based robust SCUC with spatial and temporal budget constraints,” IEEE Trans. Power Syst., vol. 31, no. 6, pp. 4998–5008, Nov. 2016.

[8] H. Wu, M. Shahidehpour, Z. Li, and W. Tian, “Chance-constrained day-ahead scheduling in stochastic power system operation,” IEEE Trans. Power Syst., vol. 29, no. 4, pp. 1583–1591, Jul. 2014.

[9] L. Wu, M. Shahidehpour, and T. Li, “Stochastic security-constrained unit commitment,” IEEE Trans. Power Syst., vol. 22, no. 2, pp. 800–911, Apr. 2007.

[10] A. Tuohy, P. Meibom, E. Deney, and M. O’Malley, “Unit commitment for systems with significant wind penetration,” IEEE Trans. Power Syst., vol. 24, no. 2, pp. 592–601, Mar. 2009.

[11] H. Wu and M. Shahidehpour, “Stochastic SCUC solution with variable wind energy using constrained ordinal optimization,” IEEE Trans. Sustain. Energy, vol. 5, no. 2, pp. 379–388, Apr. 2014.

[12] K. Hreinsson, A. Scaglione, and B. Analui, “Continuous time multi-stage stochastic unit commitment with storage,” IEEE Trans. Power Syst., vol. 34, no. 6, pp. 4476–4489, Nov. 2019.

[13] J. Zou, S. Ahmed, and X. A. Sun, “Multistage stochastic unit commitment using stochastic dual dynamic integer programming,” IEEE Trans. Power Syst., vol. 34, no. 3, pp. 1814–1823, May 2019.

[14] Y. Chen, Y. Wang, D. Kirschen, and B. Zhang, “Model-free renewable scenario generation using generative adversarial networks,” IEEE Trans. Power Syst., vol. 33, no. 3, pp. 3265–3275, May 2018.

[15] A. Solis and I. Andrés, “Analysis of Renewable Energy Integration in Transmission Constrained Electricity Markets Using Parallel Computing,” Accessed: May 14, 2021. [Online]. Available: https://hdl.handle.net/2078.1/203018

[16] Q. P. Zheng, J. Wang, and A. L. Liu, “Stochastic optimization for unit commitment—A Review,” IEEE Trans. Power Syst., vol. 30, no. 4, pp. 1913–1924, Sep. 2015.

[17] R. T. Rockafellar and R. J.-B. Wets, “Scenario and policy aggregation in optimization under uncertainty,” Math. Oper. Res., vol. 16, no. 1, pp. 119–147, 1991.

[18] J.-P. Watson and D. L. Woodruff, “Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems,” Comput. Manage. Sci., vol. 8, no. 4, pp. 355–370, Nov. 2011.

[19] C. Ordoudis, P. Pinson, M. Zagnio, and J. M. Morales, “Stochastic unit commitment via progressive hedging—Extensive analysis of solution methods,” in Proc. IEEE Eindhoven PowerTech, Eindhoven, The Netherlands, Jun. 2015, pp. 1–6.

[20] D. Gade, G. Hackebeil, S. M. Ryan, J.-P. Watson, R. J.-B. Wets, and D. L. Woodruff, “Obtaining lower bounds from the progressive hedging algorithm for stochastic mixed-integer programs,” Math. Program., vol. 157, no. 1, pp. 47–67, May 2016.

[21] H. Wu, X. Gao, X. Zhang, and H. Ye, “A systematic method for constructing feasible solution to SCUC problem with analytical feasibility conditions,” IEEE Trans. Power Syst., vol. 27, no. 2, pp. 526–534, Feb. 2012.

[22] R. T. Rockafellar and R. J. B. Wets. Variational Analysis. Accessed: May 14, 2021. [Online]. Available: https://sites.math.washington.edu/~rtr/papers/tr169-VarAnalysis-RockWets.pdf

[23] M. V. Dolgospolik, “A unifying theory of exactness of linear penalty functions,” 2018, arXiv:1802.00404. [Online]. Available: https://arxiv.org/abs/1802.00404
[24] N. Boland et al., “Combining progressive hedging with a Frank–Wolfe method to compute lagrangian dual bounds in stochastic mixed-integer programming,” SIAM J. Optim., vol. 28, no. 2, pp. 1312–1336, Jan. 2018.

[25] A. M. Palani, H. Wu, and M. M. Morcos, “A Frank–Wolfe progressive hedging algorithm for improved lower bounds in stochastic SCUC,” IEEE Access, vol. 7, pp. 99398–99406, 2019.

[26] F. Oliveira, J. Christiansen, B. Dandurand, and A. Eberhard, “Combining penalty-based and gauss-seidel methods for solving stochastic mixed-integer problems,” Int. Trans. Oper. Res., vol. 27, no. 1, pp. 494–524, Jan. 2020.

[27] E. Ela and M. O’Malley, “Studying the variability and uncertainty of variable generation at multiple timescales,” IEEE Trans. Power Syst., vol. 27, no. 3, pp. 1324–1333, Mar. 2012.

[28] E. Ela, “Scheduling and pricing of power systems with increased variable, uncertain, and nonsynchronous resources,” Ph.D. dissertation, School Elect. Electron. Eng., Univ. College Dublin, Dublin, Ireland, 2014.

[29] M. J. Feizollahi, S. Ahmed, and A. Sun, “Exact augmented lagrangian duality for mixed integer linear programming,” Math. Program., vol. 161, nos. 1–2, pp. 365–387, Jan. 2017.

[30] C. Beltran, C. Tadonki, and J.-P. Vial. (Sep. 14, 2004). Semi-Lagrangian Relaxation. Accessed: Jul. 14, 2019. [Online]. Available: https://www.optimization-online.org/DB_FILE/2004/09/959.pdf

[31] IEEE118bus_Data_Figure.Xls. Accessed: Jun. 3, 2019. [Online]. Available: http://motor.ece.iit.edu/Data/

[32] X. Guan, Q. Zhai, and A. Papalexopoulos, “Optimization based methods for unit commitment: Lagrangian relaxation versus general mixed integer programming,” in Proc. IEEE Power Eng. Soc. Gen. Meeting, Jul. 2003, pp. 1095–1100.

[33] K. Cheung et al., “Toward scalable stochastic unit commitment: Part 2: Solver configuration and performance assessment,” Energy Syst., vol. 6, no. 3, pp. 417–438, Sep. 2015.

[34] X. Li, Q. Zhai, J. Zhou, and X. Guan, “A variable reduction method for large-scale unit commitment,” IEEE Trans. Power Syst., vol. 35, no. 1, pp. 261–272, Jan. 2020.