5-21-2020

Temporal Decomposition for Multi-Interval Optimization in Power Systems

Farnaz Safdarian
Louisiana State University and Agricultural and Mechanical College

TEMPORAL DECOMPOSITION FOR MULTI-INTERVAL OPTIMIZATION IN POWER SYSTEMS

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Electrical and Computer Engineering

by
Farnaz Safdarian
B.Sc., Shahid Beheshti University, 2011
M.Sc., Amirkabir University of Technology (Tehran Polytechnic), 2014
August 2020

ACKNOWLEDGEMENTS

I would like to express my gratitude to my advisor Dr. Amin Kargarian and all the committee members, Prof. Czarnecki, Dr. Mehraeen, Dr. Busch, Dr. Farasat, and Prof. Adkins.

I also express gratitude to all ECE staff and faculty members, specially Ms. Beth Cochran, and also Dr. Jerry Trahan as the chair of Division of Electrical and Computer Engineering, for being always kind and helpful, and would like to thank all my friends at Louisiana State University who have been supportive throughout all my Ph.D. study.

Finally, I am grateful to my lovely family for their invaluable support during all my studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Large optimization problems are frequently solved for power systems operation and analysis of electricity markets. Many of these problems are multi-interval optimization with intertemporal constraints. The size of optimization problems depends on the size of the system and the length of the considered scheduling horizon. Growing the length of the scheduling horizon increases the computational burden significantly and might make solving the problem in a required time span impossible. Many simplifications and approximation techniques are applied to reduce the computational complexity of multi-interval scheduling problems and make them solvable in a reasonable time span. Geographical decomposition is presented in the literature to divide optimization problems according to the power system geographical regions and solve them faster than centralized methods. These decompositions, however, do not relieve the computational complexity originated from intertemporal constraints.

In this dissertation, temporal decomposition strategies are proposed to decompose the overall scheduling horizon into several smaller subhorizons. The proposed strategies, which can be combined with geographical decomposition, relieve the computational complexities of the multi-interval scheduling problems originated from intertemporal constraints, such as ramp up/down limits and minimum on/off time of thermal generation units and storage systems energy balance constraints. An optimization subproblem is formulated for each subhorizon with respect to local variables and constraints inside that subhorizon and intertemporal connectivities between consecutive subhorizons. Several distributed optimization algorithms are developed to coordinate subproblems in a parallel manner and find a feasible solution that is also optimal from the perspective of the whole scheduling horizon. These coordination algorithms are based on analytical target cascading and auxiliary problem principles.

Since the number of subhorizons affects the solution time, a machine learning-based approach is proposed to decompose the scheduling horizon optimally with the goal of obtaining the most time-saving. The proposed approach, which uses XGBoost as a multi-class classifier, reads the load profile and determines the best temporal decomposition pattern.

CHAPTER 1
INTRODUCTION TO DISTRIBUTED OPTIMIZATION AND TEMPORAL PARTITIONING FOR POWER SYSTEM OPERATION

1.1. Introduction

Power system analysis and decision-making problems are based on formulation and solution of large-scale multi-interval optimization models. These decision-making problems include large multi-interval security-constrained economic dispatch (SCED) and security-Constrained unit commitment (SCUC) [1, 2]. The economic dispatching problem with $N-1$ security criteria is called SCED, and the unit commitment problem with security constraints is called SCUC [3]. An important factor that increases the size and computational burden of these problems drastically is the number of time intervals of the considered scheduling horizon. Depending on the application and type of analysis, the time horizon can be one day, three days, one week, one month, or one year [4]. In such optimization problems, the size of the search space is rather large, and as a result, the computational burden is high.

In addition, integrating renewable energy sources, storage devices, and demand response to power systems brings new challenges to system operation and planning problems. One of the main challenges is handling computational complexity brought to optimization problems by these emerging technologies. The size of optimization problems grows by increasing the number of uncertain sources, the size of the system, and the number of considered contingencies, and the number of time intervals. If the problem includes integers such as charging/discharging status of storage systems and on/off status of thermal units in unit commitment [4, 5], the solution time increase drastically since solving a mixed-integer problem is much more complicated. Conventional centralized methods and standard solver packages, such as CPLEX and Gurobi, may face difficulties in handling such large optimization problems within an acceptable time limit.

Many simplifying assumptions are applied to make these problems solvable. However, these simplifying assumptions decrease the accuracy of the solution. Reducing the solution time of these problems is highly desirable as this problem is at the heart of many power system analyses, and decision-makers prefer to obtain solutions to their problems as fast as possible with a high accuracy. Decomposition approaches and distributed optimization algorithms are presented in the literature as promising alternatives to decompose large problems into smaller subproblems and distribute computational burden on multiple computing machines with the aim of reducing the overall solution time [6, 7].

1.2. Motivation

Many of power system analysis and decision-making problems are based on the formulation and solution of multi-interval optimization models. Two of these problems are security-constrained economic dispatch (SCED) and security-constrained unit commitment (SCUC) [4] [1, 2, 8]. Depending on the application and type of analysis, the time horizon can be one day, three days, one week, one month, or one year. The size of and computational burden of such problems not only depends on the size of the considered system but also is proportional to the length of the considered scheduling horizon. The larger/more the system/number of time intervals is, the larger the size and computational costs of the optimization problem will be.

Distributed optimization approaches are presented in the existing papers to solve large SCED and SCUC problems [6, 7]. The majority of these papers have focused on decomposing power systems over geographical areas. The system is partitioned into smaller zones, a subproblem is formulated for each zone, and a coordination strategy is applied to coordinate the subproblems and find the optimal solution for the whole system. Although geographical decomposition is a promising method, it does not relieve the complexity originated from intertemporal constraints

that connect optimization variables and constraints through the scheduling horizon. Intertemporal constraints, such as ramp limitations of thermal units, minimum up and down time of units, and energy storage device constraints, add a significant complexity and computation time to SCED and SCUC problems. The lack of having a decomposition algorithm to handle intertemporal constraints is the primary motivation for the research activities conducted in this dissertation.

1.3. Literature Review

In this section, the literature related to the proposed models and algorithms in this dissertation is reviewed. The reviewed literature is categorized into three sections as follows.

1.3.1. Decomposition for Power System Problems

Approximations, relaxations, and decomposition techniques have been presented in the literature to alleviate the computational burden of power system problems and find high-quality solutions within an acceptable range of time. Distributed optimization algorithms are presented as promising techniques to decompose large problems into several subproblems and distribute computational burden on several computing machines [7]. Many papers have tried to reduce the solution time to enhance decision-making processes for power system analysis. Distributed algorithms, such as alternating direction method of multipliers [9], optimality condition decomposition [10], auxiliary problem principle [11], and analytical target cascading [12, 13], are applied to coordinate the geographical subproblems and reach a good-enough solution from the perspective of the whole power system.

Reference [7] reviews distributed and decentralized algorithms to solve the optimal power flow problem in electric power systems. Reference [14] provides a tutorial for decomposition methods for network utility maximization. Reference [15] surveys the literature of distributed algorithms with applications to optimization and control of power systems. In [16], the authors have

overviewed distributed approaches, all based on consensus +innovations, for three common energy management functions: state estimation, economic dispatch, and optimal power flow. Reference [17] studies iterative distributed algorithms to assess transfer capability for management of multi-area power systems. The authors of [18] have described a distributed implementation of optimal power flow on a network of workstations. An approach to paralleling optimal power flow is described in [19] that is applicable to large interconnected power systems. References [20] and [21] present a coordination method for distributed economic dispatch (ED) problem. Distributed algorithms were implemented in [22] to solve large-scale unit commitment problems.

1.3.2. Decomposition for Economic Dispatching

There are several ways to decompose such problems, and each way of decomposition follows a particular goal. Geographical-based decompositions, which divide a power system into several small zones, are the most common approaches used in the literature. A power system is decomposed into several smaller zones, and a local subproblem is formulated for each zone. Note that most of the existing geographical-based distributed algorithms in the literature aim at preserving information privacy of autonomous entities. These methods formulate a subproblem for each zone and solve the subproblems iteratively either in a sequential manner or a parallel fashion [20, 23-25]. Reference [7] presents a geographically distributed economic dispatch (ED), where every generator/load is modeled as an agent, and the related agents of an area form a cluster. In [24], the authors have described a transmission+distribution ED and proposed a decentralized method to solve this problem using multi-parametric quadratic programming. Lagrangian relaxation and augmented Lagrangian relaxation are used in [25] to solve ED in a distributed fashion. Reference [26] presents a distributed ED with second-order convergence, which is based on a parallel primal-dual interior-point algorithm with a matrix-splitting technique. Reference [23]

proposes a consensus-based distributed ED algorithm and analyses the influence of time delays. An attack-robust distributed ED strategy was developed in [27] assuming that every distributed generator can monitor the behavior of its neighbors. In [28], a distributed bisection algorithm is proposed for economic dispatch to minimize the aggregated cost of a network. Reconfiguration of smart distribution systems with time-varying loads using parallel computing is presented in [29]. In [30], a decentralized and self-organizing solution framework for economic dispatch (ED) is proposed. In [31], the distributed economic dispatch and demand response initiatives for grid-connected microgrids with high-penetration of wind power are studied. A transmission and distribution network coordinated model and an efficient decentralized method to solve ED using multi-parametric quadratic programming is presented in [32]. Reference [33] introduces a consensus-based control scheme for distributed power systems to solve the distributed economic dispatch problem.

Although geographical decompositions have shown desirable performance for large power system problems, they might not be effective for computation time reduction for optimization problems with multiple time intervals, especially when the focus is on a part of the system that has one owner. In such multi-interval scheduling problems, the computational burden not only depends on the size of the system but also is pertained to the complexity and the number of intertemporal constraints.

However, the dimensionality and complexity of the ramp-constrained ED not only depend on the size of the system but also depend on the number of scheduling time intervals. Although decomposing the system geographically potentially reduces the size and computation time of the ED problem, it does not deal with intertemporal constraints. The intertemporal constraints, which originate from limits of ramping capabilities of generating units, interconnect decisions made in a

time interval to decisions made in other intervals and increase the complexity of the decision-making process in the ramp-constrained ED. Another factor that increases the size and computational complexity of the ED problem is $N-1$ security criteria. A set of credible contingencies is considered for each interval of the SCED, and a new set of variables and constraints are added to each time interval. Considering contingencies and interdependencies between scheduling time intervals (i.e., intertemporal constraints) increases the computation time of the SCED problem. In such a situation, decomposing SCED over the considered time horizon is potentially a promising strategy to reduce the computation time.

Several decentralized approaches for solving the economic dispatch problem in smart grid systems are presented in [34-38]. However, these papers do not consider the complexities of modeling intertemporal constraints. In order to study a particular system with one owner, it would be beneficial to decompose the problem over the time horizon. Time decomposition can also be applied to a geographically decomposed system to further decrease the solution time.

In addition, uncertainties have been modeled in the literature using several methods such as chance-constrained programming [39], stochastic programming [40], robust optimization [41], heuristic optimization [42], two-stage optimization [43], and conditional value at risk [44]. Such approaches enhance system reliability and security and may reduce real-time rescheduling costs. Parametric chance-constrained methods have been widely used to model uncertainty in ED problems. Reference [45] presents adaptive robust optimization for the economic dispatch of power systems with a high level of wind penetration. In [46], a decentralized chance-constrained ED model is proposed, wherein economic model predictive control is adopted to take future uncertainties into account. However, a known probability distribution function (PDF) is assumed for modeling uncertainty using these methods. The problem is that the random parameter may not

follow any known distribution function. In addition, parameters and shape of PDFs might change depending on the weather condition, geographical location, and the type of load. PDF of wind might not belong to any class of known probability distribution. It is more realistic to use the historical wind data, estimate their PDFs with imposing no assumption on the class of PDFs and use the estimated PDFs to formulate the chance constraints. This process leads to a data-driven nonparametric chance-constrained model. The term data-driven refers to the case in which historical data are used for modeling. We presented a data-driven nonparametric chance-constrained optimization for microgrid energy management in our previous work. [47]

1.3.3. Decomposition for Unit Commitment

The literature on distributed SCUC is more limited than optimal power flow and economic dispatch. To the best of our knowledge, all existing distributed SCUC approaches are based on decomposition over geographical areas, uncertainty scenarios, and contingency scenarios. In [48], progressive hedging is presented to decompose problems over scenarios. Reference [49] decomposes hourly day-ahead unit commitment over stochastic scenarios. In [50], a scenario-based decomposition is presented for solving two-stage stochastic unit commitment. In [51], Benders decomposition is applied to decompose SCUC into a master problem and several subproblems that handle contingency constraints. The majority of recent publications focus on area decomposition. In [22, 52, 53], the system is decomposed into several smaller zones, an SCUC subproblem is formulated for each zone, and distributed algorithms are developed to coordinate subproblems. Reference [54] presents a decentralized solution algorithm for network-constrained unit commitment in multiregional power systems. Reference [53] presents a distributed unit commitment algorithm to accelerate the generation scheduling for large-scale power systems.

Although decompositions over geographical areas, uncertainty scenarios, and contingency

scenarios may reduce the computation time of SCUC, they are not capable of handling the computational complexity originated from temporal constraints (i.e., generators' ramping and minimum on/off time restrictions). Such complexity increases as the number of scheduling intervals grows. Considering contingencies and interdependencies between scheduling intervals increase the computation time of SCUC. To reduce the computation time in such a situation, decomposing SCUC over the considered scheduling time horizon is potentially a more promising strategy than a geographical decomposition. However, the most challenging part of time decomposition is dealing with ramping and minimum on/off time constraints of units that interconnect decisions made in consecutive intervals. In our previous work [55], we presented the idea of time decomposition for a simple economic dispatch problem without modeling minimum on/off time constraints. Also, the coordination algorithm in [55] was based on the classical auxiliary problem principle.

1.3.4. Classification in Power Systems

Recently, the applications of machine learning to solve various problems have seen increased interest [56]. Reference [57] reviews recent studies that have applied ML to solve OPF-related problems and suggest potential research directions. Reference [58] presents a combination of machine learning and analytical model-based methods to make a prediction on line congestions before solving optimal powe flow. The behavior of complex phenomena in power systems can be modeled using either simulation or historical datasets. Regression and classification learners read input data and project them to a set of output features. Classification methods have been applied to solve a variety of power system optimization problems. In [59], classification algorithms are used to predict optimal power flow solutions in real-time instead of solving an optimization problem. A classification-based method is presented in [60] to predict the parameters of an

environmental multi-objective economic dispatch problem. Reference [61] uses classification to predict on/off status of generators in unit commitment. An augmented Lagrangian Hopfield network is used in [62] to enhance the unit commitment solution procedure. Classification is used to de-commit extra spinning reserve units caused by minimum up/down time constraints. Although classification learners have been widely used for power systems optimization, their applications for optimal decomposition (neither temporal nor geographical) and distributed optimization has not been explored despite their potential advantages to enhance decomposition.

1.4. Contribution and Organization

In Chapter 2 of this dissertation, entitled *Temporal Decomposition Strategy for Security-Constrained Economic Dispatch*, we propose a distributed optimization algorithm to reduce the computation time of SCED problems. A horizontal time partitioning strategy is developed to divide the ramp-constrained SCED problem into several smaller optimization subproblems, each containing fewer intertemporal constraints and variables than the original centralized SCED. Unlike the geographical decomposition, the considered scheduling horizon is decomposed into several subhorizons (we call this strategy as a horizontal time decomposition since it decomposes the optimization over the time horizon with respect to intertemporal constraints). The concept of *overlapping time intervals* is presented to model intertemporal constraints, which are ramping constraints of generating units, for transition from one subhorizon to its neighboring subhorizons. The overlapping time intervals are mathematically modeled as a set of variables and constraints. An SCED subproblem is formulated for each subhorizon with respect to variables and constraints of internal intervals and the overlapping intervals of that subhorizon. Variables and constraints of an overlapping time interval can be controlled by two consecutive subhorizons (an overlapping interval is at the end of subhorizon N_{-} and at the beginning of subhorizons N). To coordinate

solutions of the SCED subproblems and ensure the feasibility and optimality of results from the perspective of the power system and its components, accelerated auxiliary problem principle (A-APP) is developed to solve subproblems distributedly. Furthermore, we present an initialization technique to enhance the convergence performance of the distributed optimization algorithm. The proposed algorithm is scalable and reduces the computation time of the SCED problem.

In Chapter 3, entitled *Temporal Decomposition-based Stochastic SCED for Smart Grid Energy Management,* we aim to decrease the solution time of operation problems for a modern system, which includes wind generation, controllable loads, and energy storage. We consider a spinning reserve for generating units to mitigate the uncertainties of wind generation. We propose to decompose the problem over time horizon and solve the achieved subproblems in parallel. The complexity arises from modeling the intertemporal constraints such as ramping and state of charge of batteries between subproblems when we want to avoid sub-optimality resulted from forcing the state of a subproblem that is solved earlier. To address this issue, we will define a new time interval between every two consecutive subproblems and will name it a dummy or overlapping time interval. The state of this dummy interval defines a set of shared variables between the two subproblems. Auxiliary problem principle (APP) with a suitable initialization technique is proposed for coordinating the consecutive subproblems in parallel.

In Chapter 4, entitled *Temporal Decomposition and Coordination for Security-Constrained Unit Commitment,* a novel temporal decomposition, along with a coordination strategy, is proposed to solve SCUC distributedly with the aim of computation time reduction. The overall scheduling horizon is decomposed into multiple smaller subhorizons, and a SCUC subproblem is formulated for each subhorizon. The concept of *coupling time intervals* is introduced to model ramping limitations of generating units between two consecutive subhorizons. Each subproblem has one or

two coupling intervals. Moreover, several nonlinear constraints and logical expressions, along with their linear equivalent models, are presented to model minimum on/off time constraints between subhorizons. In addition, to coordinate SCUC subproblems and ensure feasibility and optimality of results, accelerated analytical target cascading (A-ATC) is developed to solve subproblems distributedly. An initialization technique is presented to enhance the convergence performance of A-ATC.

In Chapter 5, we focus on developing methods to find the best strategy for partitioning the considered scheduling horizon and find the optimal number of subhorizons. We analyze partitioning of a SCED problem over the considered time horizon and study the effect of the number of subproblems on the performance of the time decomposition strategy, the solution time, the number of iterations, and accuracy of the obtained results. Since it is preferred to have equal subhorizons to take the most advantage of parallel computing, the number of subproblems determines the breaking intervals. The proposed model aims to classify the optimal number of subproblems based on previously tested data. Since the problem can be partitioned to a range of equal subproblems, the method is a multiclass classification. Multiclass classification is a classification method that consists of more than two classes. Once the model is trained, validated, and tested, it will be applied to each new predicted load to identify the class of the optimal number of subproblems.

Finally, concluding remarks and suggestions for future work are provided in Chapter 6.

CHAPTER 2
TIME DECOMPOSITION STRATEGY FOR SECURITY-CONSTRAINED ECONOMIC DISPATCH

2.1. Introduction

In this chapter, a horizontal time decomposition strategy to reduce the computation time of security-constrained economic dispatch (SCED) is presented. The considered scheduling horizon is decomposed into multiple smaller subhorizons. The concept of overlapping time intervals is introduced to model ramp constraints for the transition from one subhorizon to another subhorizon. A subhorizon includes several internal intervals and one or two overlapping time intervals that interconnect consecutive subhorizons. A local SCED is formulated for each subhorizon with respect to internal and overlapping intervals' variables/constraints. The overlapping intervals allow modeling intertemporal constraints between the consecutive subhorizons in a distributed fashion. An accelerated version of the auxiliary problem principle is developed to coordinate the subproblems and find the optimal solution for the whole operation horizon distributedly. Furthermore, we present an initialization strategy to enhance the convergence performance of the coordination strategy. The proposed algorithm is applied to three large systems, and promising results are obtained.

2.2. Symbols

A. Indices and Sets:

i, j	Index for buses.
t	Index for time intervals.
u	Index for units.
c	Index for contingencies.
k	Index for iterations.
ij	Index for lines.
N	Index for subhorizons.
t_o	Index for overlapping time intervals.

n_{to} Number of overlapping time intervals between a subhorizon and its neighboring subhorizons.

B. Parameters:

N_{-} The subhorizon (subproblem) before subhorizon N.

N_{+} The subhorizon (subproblem) after subhorizon N.

n Number of time intervals in a subhorizon

UR_u Ramping up of unit u.

DR_u Ramping down of unit u.

C. Variables:

$p_{ut_oN_{-}\to N}^{*(k-1)N_{-}}$ Power generated by unit u in overlapping time interval t_o between subhorizons N_{-} and N determined at iteration $k-1$ by subproblem N_{-} and sent to subproblem N.

$p_{ut_oN\to N_{+}}^{*(k-1)N_{+}}$ Power generated by unit u in overlapping time interval t_o between subhorizons N and N_{+} determined at iteration $k-1$ by subproblem N_{+} and sent to subproblem N.

$f(\cdot)$ Generation cost function.

2.3. Contributions

The main contributions of this chapter, that make it different from the existing literature, are summarized as follows:

- A *temporal decomposition* strategy is proposed to decompose SCED into multiple time-interdependent subproblems. This strategy can be applied to other multi-stage problems. To the best of our knowledge, such a temporal decomposition strategy is its first kind developed for SCED.
- Based on the concept of Nesterov momentum for gradient descent methods, an *accelerated* APP is presented to coordinate SCED subproblems distributedly.
- An initialization strategy is presented to enhance the performance of the proposed distributed SCED algorithm.

The proposed time decomposition and geographical decomposition are complements of each other. Geographical decomposition can be used to divide a system into several smaller subsystems. If the considered problem has multiple time intervals, time decomposition can be used to further

decompose each subsystem into several subhorizons to reduce the computational cost of each subsystem.

2.4. The Considered SCED Problem

The considered centralized SCED is formulated below. The objective function is to minimize generation costs subject to power flow, network, and equipment constraints under normal condition ((*1b*)-(*1g*)), and $N-1$ security criteria ((*1h*)-(*1l*)) [4].

$$\min_{(p_{ut},p_{ut}^{c},\delta_t)} \sum_t \sum_u \underbrace{a_u \cdot {p_{ut}}^2 + b_u \cdot p_{ut} + C_u}_{f(p_{ut})} \qquad (1a)$$

$$s.t. \qquad h(p,\delta): \begin{cases} p_{uit} - p_{dit} = \sum_j \dfrac{\delta_{it} - \delta_{jt}}{X_{ij}} & \forall i, \forall t \quad (1\text{b}) \\ \delta_{ref,t} = 0 & \forall t \quad (1c) \end{cases}$$

$$g(p,\delta): \begin{cases} \underline{P_{ut}} \le p_{ut} \le \overline{P_{ut}} & \forall u, \forall t \quad (1d) \\ p_{ut} - p_{u(t-1)} \le \mathrm{UR}_u & \forall u, \forall t \quad (1e) \\ p_{u(t-1)} - p_{ut} \le \mathrm{DR}_u & \forall u, \forall t \quad (1f) \\ \underline{P_{ij}} \le \dfrac{\delta_{it} - \delta_{jt}}{X_{ij}} \le \overline{P_{ij}} & \forall ij, \forall t \quad (1g) \end{cases}$$

$$h^c(p,\delta): \begin{cases} p_{uit}^{c} - p_{dit} = \sum_j \dfrac{\delta_{it}^{c} - \delta_{jt}^{c}}{X_{ij}} & \forall i, \forall t \quad (1\text{h}) \\ \delta_{ref,t}^{c} = 0 & \forall t \quad (1i) \end{cases}$$

$$g^c(p,\delta): \begin{cases} \underline{P_{ut}} \le {p^c}_{ut} \le \overline{P_{ut}} & \forall u, \forall t \quad (1j) \\ \underline{P_{ij}} \le \dfrac{\delta_{it}^{c} - \delta_{jt}^{c}}{X_{ij}} \le \overline{P_{ij}} & \forall ij, \forall t, \forall c \quad (1k) \\ \left|p_{ut} - {p^c}_{ut}\right| \le \Delta & \forall u, \forall t, \forall c \quad (1l) \end{cases}$$

We have used a DC power flow based SCED formulation. However, a convex relaxation of the AC power flow model can be integrated to formulate a more complex and computationally

expensive SCED formulation [63]. Although we have ignored uncertainties in the considered SCED formulation, chance constraints can be formulated to account for generation and demand uncertainties. In that case, line flow and power reserve inequality constraints should be modeled as chance constraints [64].

2.5. The Proposed Temporal Decomposition Strategy

Consider a scheduling horizon consisting of $n \times N$ intervals as shown in Fig 2.1. a. The horizon can be decomposed into N equal subhorizons, each consisting of n intervals as depicted in Fig. 1b. One strategy is to formulate an SCED subproblem for each subhorizon N regardless of other subhorizons $N_{\mp}$ and then independently solve the subproblems in parallel. Another strategy is to formulate an SCED for each subhorizon N by fixing the solution of interval n of SCED $N\_$ as the initial condition of SCED N and then solve SCED subproblems 1 to N sequentially. While the first strategy may violate system intertemporal constraints between the ending interval of subhorizon $N\_$ and the beginning interval of subhorizon N, the second strategy may provide feasible but suboptimal results for the whole scheduling horizon $n \times N$. Consider the SCED subproblems of two consecutive subhorizons $N\_$ and N (from now on, for the sake of brevity, we write SCED of (1) in a compact form).

$$\min_{\left(p_{N\_}, p^c_{N\_}, \delta_{N\_}, \delta^c_{N\_}\right)} \sum_{t=1}^{n} \sum_{u} f\left(p_{utN\_}\right) \qquad (2a)$$

$$s.t. \quad h_{N\_}\left(p_{utN\_}, p^c_{utN\_}, \delta_{itN\_}, \delta^c_{itN\_}\right) = 0 \quad \forall t, u, i, c \quad (2b)$$

$$g_{N\_}\left(p_{utN\_}, p^c_{utN\_}, \delta_{itN\_}, \delta^c_{itN\_}\right) \leq 0 \quad \forall t, u, i, c \quad (2c)$$

where equality constraint (*2b*) is the compact form of (*1b*), (*1c*), (*1h*) and (*1i*), and inequality constraint (*2c*) is the compact form of (*1d*)-(*1g*) and (*1j*)-(*1l*) for SCED corresponding to

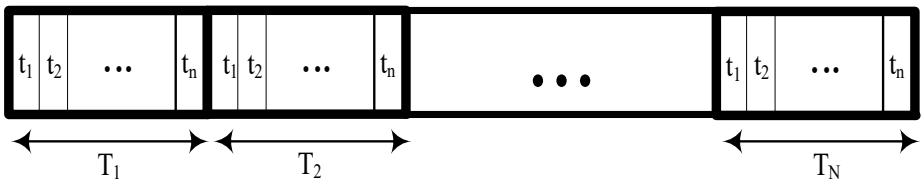
subhorizons $N\_$.

$$\min_{(p_N, p_N^c, \delta_N, \delta_N^c)} \sum_{t=1}^{n} \sum_{u} f(p_{utN}) \qquad (3a)$$

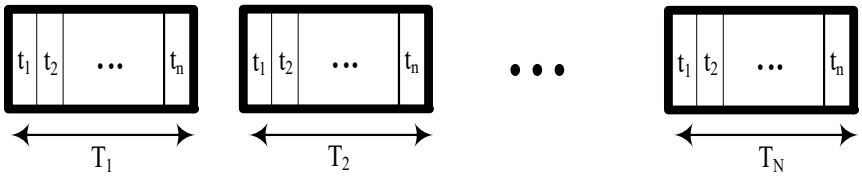
$$s.t. \quad h_N(p_{utN}, p_{utN}^c, \delta_{itN}, \delta_{itN}^c) = 0 \quad \forall t, u, i, c \qquad (3b)$$

$$g_N(p_{utN}, p_{utN}^c, \delta_{itN}, \delta_{itN}^c) \leq 0 \quad \forall t, u, i, c \qquad (3c)$$

Equality constraint (*3b*) is the compact form of (*1b*), (*1c*), (*1h*) and (*1i*), and inequality constraint (*3c*) is the compact form of (*1d*)-(*1g*) and (*1j*)-(*1l*) for SCED corresponding to subhorizons N.



(a)



(b)

Figure 2.1. (a) The whole $n \times N$ intervals, and b) the decomposed N subhorizons each with n intervals.

As shown in Fig. 2.1, the constraints of subhorizon $N\_$ are coupled with the constraints of subhorizon N through the generators' ramping constraints for transition from interval n of subhorizon $N\_$ to interval one of subhorizon N. The red (blue) line indicates that if the unit u is generating $p_{unN\_}$ at the end of subhorizon $N\_$, it cannot generate more (less) than $p_{unN\_} + UR_u$ ($p_{unN\_} - DR_u$) at the beginning of subhorizon N. The coupling constraints between subhorizons

$N\_$ and N are as follows:

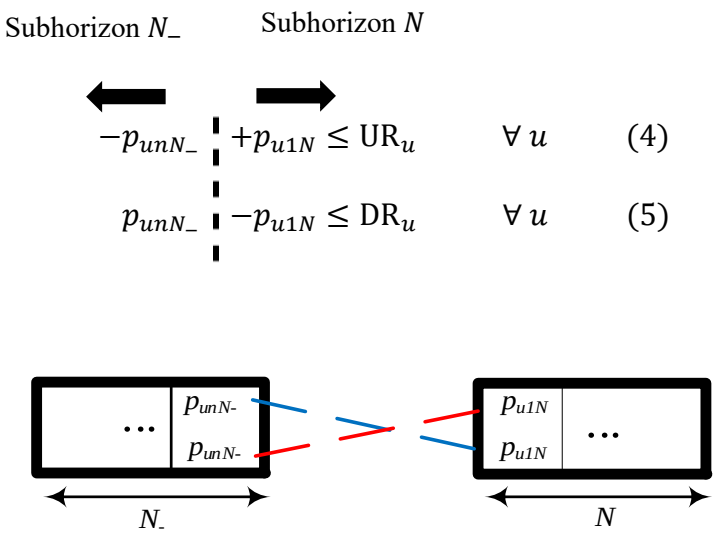
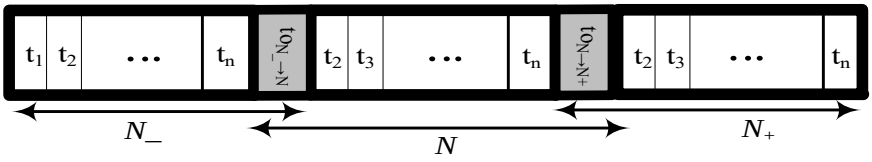


Figure 2.2. Coupling constraint (ramping limit) for transition from subhorizon $N\_$ to subhorizon N.

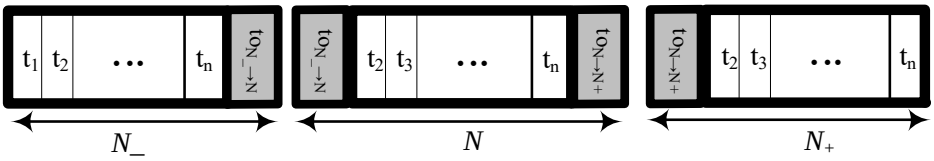
To handle these two sets of coupling intertemporal constraints, we present the concept of overlapping (or coupling) time intervals between the consecutive subhorizons as illustrated in Fig. 3a. We duplicate the first interval of each subhorizon N and consider that as interval $n+1$ of subhorizon $N\_$. All variables and constraints of the overlapping time intervals appear in both optimization subproblems corresponding to subhorizons $N\_$ and N. Coordinating the time intervals n and $n+1$ of subhorizon $N\_$ and the time intervals one and two of subhorizon N through intertemporal constraints inside each subhorizon leads to coordination between the consecutive subhorizons. We separate the subhorizons and duplicate the overlapping time intervals in each subhorizon as shown in Fig. 3b.

The optimization subproblem $N\_$ (i.e., (2a)-(2c)) with the overlapping time intervals is now formulated as follows:

$$\min_{\left(p_{N\_},p^{c}_{N\_},\delta_{N\_},\delta^{c}_{N\_},p_{t_oN\_\to N}\right)} \sum_{t=1}^{n+n_{to}} \sum_{u} f\left(p_{utN\_}, p^{N\_}_{ut_oN\_\to N}\right) \qquad (6a)$$



(a)



(b)

Figure 2.3. a) Three consecutive subhorizons with overlapping time intervals and b) subhorizons decomposition with overlapping time intervals.

$$s.t. \qquad (2b), (2c)$$

$$N_- \rightarrow N \begin{cases} h_{N_-}^{to}\left(p_{utN_-}, p_{utN_-}^{c}, \delta_{itN_-}, \delta_{itN_-}^{c}, p_{ut_oN_-\rightarrow N}^{N_-}\right) = 0 & \forall u, i, c \qquad (6b) \\ g_{N_-}^{to}\left(p_{utN_-}, p_{utN_-}^{c}, \delta_{itN_-}, \delta_{itN_-}^{c}, p_{ut_oN_-\rightarrow N}^{N_-}\right) \leq 0 & \forall u, i, c \qquad (6c) \end{cases}$$

where (2b) and (2c) are constraints corresponding to internal intervals of subhorizon N_-, and (6b) and (6c) are constraints corresponding to the overlapping time intervals for transition from subhorizon N_- to subhorizon N. The optimization subproblem N (i.e., (3a)-(3c)), which is a middle subproblem with two overlapping time intervals at the beginning and ending time intervals, is formulated as:

$$\min_{(p_{t_oN_-\rightarrow N}, p_N, p_N^c, \delta_N, \delta_N^c, p_{t_oN\rightarrow N_+})} \sum_{t=1}^{n+n_{to}} \sum_{u} f\begin{pmatrix} p_{ut_oN_-\rightarrow N}^{N}, p_{utN}, \\ p_{ut_oN_-\rightarrow N}^{N} \end{pmatrix} \qquad (7a)$$

$$s.t. \qquad (3b), (3c)$$

$$N_- \rightarrow N \begin{cases} h_N^{to}\left(p^N_{ut_oN_-\rightarrow N}, p_{utN}, p^c_{utN}, \delta_{itN}, \delta^c_{itN}\right) = 0 & \forall u, i, c \quad (7b) \\ g_N^{to}\left(p^N_{ut_oN_-\rightarrow N}, p_{utN}, p^c_{utN}, \delta_{itN}, \delta^c_{itN}\right) \leq 0 & \forall u, i, c \quad (7c) \end{cases}$$

$$N \rightarrow N_+ \begin{cases} h_N^{to}\left(p_{utN}, p^c_{utN}, \delta_{itN}, \delta^c_{itN}, p^N_{ut_oN\rightarrow N_+}\right) = 0 & \forall u, i, c \quad (7d) \\ g_N^{to}\left(p_{utN}, p^c_{utN}, \delta_{itN}, \delta^c_{itN}, p^N_{ut_oN\rightarrow N_+}\right) \leq 0 & \forall u, i, c \quad (7e) \end{cases}$$

where (3b) and (3c) are constraints corresponding to internal intervals of subhorizon N. Since subhorizon N has overlapping intervals with subhorizons N_- and N_+, its optimization subproblem includes two sets of overlapping constraint, one with N_- and another one with N_+. Expressions (7b) and (7c) are constraints corresponding to the overlapping time intervals for transition from subhorizon N_- to subhorizon N, and (7d) and (7e) are constraints corresponding to the overlapping time intervals for transition from subhorizon N to subhorizon N_+. The optimization subproblems (6) and (7) are still non-separable as they include shared variables $p_{ut_oN_-\rightarrow N}$. In addition, subproblems N and N_+ are non-separable because of shared variables $p_{ut_oN\rightarrow N_+}$. The shared variables are power produced by generating units in the overlapping time intervals. We duplicate $p_{ut_oN_-\rightarrow N}$ to create two sets of new variables with one belonging to subproblem N_- ($p^{N_-}_{ut_oN_-\rightarrow N}$) and another one belonging to subproblem N ($p^N_{ut_oN_-\rightarrow N}$). Similarly, we duplicate $p_{ut_oN\rightarrow N_+}$ and assign one set to subproblem N ($p^N_{ut_oN\rightarrow N_+}$) and the other set to subproblem N_+ ($p^{N_+}_{ut_oN\rightarrow N_+}$). These auxiliary variables make SCEDs of the subhorizons separable. That is, each SCED subproblem can be solved separately from other subproblems. To ensure the feasibility of the results from the perspective of the whole scheduling horizon, we convert each pair of the shared variables to a set of consistency constraint as:

$$CC_{N_-\rightarrow N}: p^{N_-}_{ut_oN_-\rightarrow N} - p^N_{ut_oN_-\rightarrow N} = 0 \quad \forall u \qquad (8)$$

$$CC_{N\to N_+}: p^{N}_{ut_o N\to N_+} - p^{N_+}_{ut_o N\to N_+} = 0 \quad \forall u \qquad (9)$$

$CC_{N_-\to N}$ is enforced in subproblems N_- and N, and $CC_{N\to N_+}$ is enforced in subproblems N and N_+. The SCED subproblem of subhorizon N_- is:

$$\min \ (6a)$$

$$s.t. \ (2b), (2c), (6b), (6c), (8)$$

and the SCED subproblem of subhorizon N is:

$$\min \ (7a)$$

$$s.t. \ (3b), (3c), (7b) - (7e), (8), (9)$$

While $p^{N_-}_{ut_o N_-\to N}$ is a decision variable in subproblem N_-, it is treated as a constant in subproblem N. Similarly, $p^{N_+}_{ut_o N_-\to N}$ is a decision variable in subproblem N_+ and a constant in subproblem N. Consistency constraints (8) and (9) are hard constraints that make the order of solving subproblems of importance. If subhorizon N_- solves its SCED first, $CC_{N_-\to N}$ forces subhorizon N to set its decision variables $p^{N}_{ut_o N_-\to N}$ equal to $p^{N_-}_{ut_o N_-\to N}$. Since the subhorizons are connected like a string, if one starts from the beginning of the string to solve the decomposed SCED problem, the solution of hour one of each subhorizon is dictated by the solution of its previous subhorizon. In the existing literature and current practice, the initial condition of every subhorizon (which is usually one day in ramp-constrained SCED) is set by its previous subhorizon. However, this may lead to a suboptimal solution for the whole scheduling horizon.

2.6. Proposed Coordination Algorithm

To avoid obtaining suboptimal results, we present an iterative coordination algorithm that allows a parallel solution of the SCED subproblems. The consistency constraints are relaxed in the objective function of each subhorizon using the concept of augmented Lagrangian relaxation. We

utilize auxiliary problem principle [11], which is a parallel coordination strategy, to coordinate the subhorizons. We further propose an accelerated APP (A-APP) and an initialization strategy to enhance the performance of the distributed coordination algorithm.

2.6.1. Normal Auxiliary Problem Principle

The APP is an iterative method that aims at finding the optimal solution of several coupled optimization problems in a distributed manner [11]. APP is a suitable method for parallel computing and reduces the idle time of subproblems as compared to sequential distributed approaches. This method uses the augmented Lagrangian relaxation to penalize the consistency constraints (i.e., hard constraints) with a set of penalty terms in the objective function. A sequence of auxiliary problems are solved to make the non-separable terms of the augmented Lagrangian (i.e., the quadratic term) separable and solve subproblems in parallel (we refer to [65] for more details).

Consider the SCED subproblems $N\_$ and N. We denote the iteration index of APP as k and penalize the violation of hard constraint (8) at each iteration k in the objective functions of subproblem $N\_$. The resultant SCED subproblem for subhorizon $N\_$ becomes as follows:

$$\min_{\left(p^{k}_{N\_},p^{kc}_{N\_},\delta_{N\_},\delta^{c}_{N\_},p^{kN\_}_{t_oN\_\to N}\right)} \sum_{t=1}^{n+n_{to}} \sum_{u} f\left(p^{k}_{utN\_},p^{kN\_}_{ut_oN\_\to N}\right)$$

$$+\sum_{u}\begin{pmatrix}\frac{\rho}{2}\left\|p^{kN\_}_{ut_oN\_\to N}-p^{*(k-1)N\_}_{ut_oN\_\to N}\right\|^{2}+\lambda^{k\,\dagger}_{N\_\to N}p^{kN\_}_{ut_oN\_\to N}\\ +\gamma p^{kN\_\dagger}_{ut_oN\_\to N}\left(p^{*(k-1)N\_}_{ut_oN\_\to N}-p^{*(k-1)N}_{ut_oN\_\to N}\right)\end{pmatrix} \quad (10)$$

$$s.t. \qquad (2b),(2c),(6b),(6c)$$

where $p^{k}_{utN\_}$, $p^{kc}_{utN\_}$, and $p^{kN\_}_{ut_oN\_\to N}$ are decision variables while $p^{*(k-1)N}_{ut_oN\_\to N}$ and $p^{*(k-1)N\_}_{ut_oN\_\to N}$ are known. Parameter $p^{*(k-1)N}_{ut_oN\_\to N}$ is the value of generated power by each unit u (i.e., the shared variable) in the overlapping time interval determined by subhorizon N at iteration $k-1$. Parameter

$p_{ut_oN_-\to N}^{*(k-1)N_-}$ denotes the value of generated power by each unit u in the overlapping time interval determined by subhorizon N_- at iteration $k-1$. The SCED subproblem N is penalized by relaxing two sets of hard constraints (8) and (9) in its objective function as follows:

$$\min_{\left(p_{t_oN_-\to N}^{k},p_N^k,p_N^{kc},\delta_N,\delta_N^c,p_{t_oN\to N_+}^{k}\right)} \sum_{t=1}^{n+n_{to}} \sum_{u} f\begin{pmatrix} p_{ut_cN_-\to N}^{kN}, \\ p_{utN}^{k}, p_{ut_oN\to N_+}^{kN} \end{pmatrix}$$

$$+\sum_{u}\left(\begin{pmatrix} \frac{\rho}{2}\left\|p_{ut_oN_-\to N}^{kN}-p_{ut_oN_-\to N}^{*(k-1)N}\right\|^2-\lambda_{N_-\to N}^{k\,\dagger}p_{ut_oN_-\to N}^{kN} \\ +\gamma p_{ut_oN_-\to N}^{kN\dagger}\left(p_{ut_oN_-\to N}^{*(k-1)N}-p_{ut_oN_-\to N}^{*(k-1)N_-}\right) \end{pmatrix}\right.$$

$$\left.+\begin{pmatrix} \frac{\rho}{2}\left\|p_{ut_oN\to N_+}^{kN}-p_{ut_oN\to N_+}^{*(k-1)N}\right\|^2+\lambda_{N\to N_+}^{k\,\dagger}p_{ut_oN\to N_+}^{kN} \\ +\gamma p_{ut_oN\to N_+}^{kN\dagger}\left(p_{ut_oN\to N_+}^{*(k-1)N}-p_{ut_oN\to N_+}^{*(k-1)N_+}\right) \end{pmatrix}\right) \qquad (11)$$

$$s.t. \qquad (3b),(3c),(7b)-(7e)$$

While the first penalty term penalizes any violations in the overlapping time interval between subhorizons N_- and N, the second term penalizes any violations in power produced by each unit u at the overlapping time interval between subhorizons N and N_+. Since in each iteration k, each subhorizon N uses values of the shared variables obtained by its neighboring subhorizons N_- and N_+ at iteration $k-1$ (i.e., $p_{ut_oN_-\to N}^{*(k-1)N_-}$ and $p_{ut_oN\to N_+}^{*(k-1)N_+}$), the SCED subproblems can be solved in parallel. Before starting each iteration k, the penalty multiplier λ is updated as follows:

$$\lambda_{N_-\to N}^{k+1}=\lambda_{N_-\to N}^{k}+\beta\left(p_{ut_oN_-\to N}^{*kN_-}-p_{ut_oN_-\to N}^{*kN}\right) \qquad (12)$$

$$\lambda_{N\to N_+}^{k+1}=\lambda_{N\to N_+}^{k}+\beta\left(p_{ut_oN\to N_+}^{*kN}-p_{ut_oN\to N_+}^{*kN_+}\right) \qquad (13)$$

where β should be a suitable positive step-size. If the step-size is selected too large, the convergence speed may go up; however, it increases the chance of losing optimality, oscillating around the optimal point, or even divergence. If the step-size is selected too small, although the

solution is more accurate and the chance of divergence reduces, the convergence speed degrades [66]. A user may solve the problem several times with different step-sizes to gain knowledge on the suitable values/ranges for the step-size. The value of the Lagrange multiplier λ in each iteration implies the cost to maintain the consistency constraints in the overlapping time intervals. The above formulation can be generalized for multiple subhorizons. The pseudocode for implementation of distributed SCED is given in Algorithm I. Since the considered SCED problem is convex, APP is proven to converge to the global optimal solution. We refer to [11] for discussions on the convergence rate/proof of APP.

Algorithm I Pseudocode for coordinating SCED subproblems with normal APP

1: Decompose the considered horizon into T_N equal subhorizons

2: Initialize $p^{*0N_-}_{ut_oN_-\to N}$, $p^{*0N}_{ut_oN_-\to N}$, $p^{*0N}_{ut_oN\to N_+}$, $p^{*0N_+}_{ut_oN\to N_+}$, λ, β, γ, ρ and set $k = 0$

3: **while** $\left|p^{*kN_-}_{ut_oN_-\to N} - p^{*kN}_{ut_oN_-\to N}\right| > \varepsilon$ & $\left|p^{*kN}_{ut_oN\to N_+} - p^{*kN_+}_{ut_oN\to N_+}\right| > \varepsilon$, $k = k + 1$ **do**

4: Solve SCED subproblems in parallel and determine the optimal values of $p^{kN_-}_{ut_oN_-\to N}$, $p^{kN}_{ut_oN_-\to N}$, $p^{kN}_{ut_oN\to N_+}$, and $p^{kN_+}_{ut_oN\to N_+}$

5: Exchange $p^{*kN_-}_{ut_oN_-\to N}$, $p^{*kN}_{ut_oN_-\to N}$, $p^{*kN}_{ut_oN\to N_+}$, and $p^{*kN_+}_{ut_oN\to N_+}$ between the neighboring subhorizons

6: Update λ^k by (12) and (13)

7: **end while**

2.6.2. Accelerated Auxiliary Problem Principle

The normal APP uses a simple fixed step size gradient-based method to update multipliers with respect to the difference between coupling variables ($p^{kN_-}_{ut_oN_-\to N} - p^{kN}_{ut_oN_-\to N}$ & $p^{kN}_{ut_oN\to N_+} - p^{kN_+}_{ut_oN\to N_+}$), which is the gradient. However, the convergence performance might degrade when more iterations are carried out, and the solution is getting close to the optimal point or when the

optimal point is in a shallow area (or a ravine). Near the optimal point, the terms $p_{ut_oN_-\to N}^{kN_-} - p_{ut_oN_-\to N}^{kN}$ and $p_{ut_oN\to N_+}^{kN} - p_{ut_oN\to N_+}^{kN_+}$ (or gradients) might become very small that leads to an updated multiplier in iteration $k+1$ that is almost the same as that in iteration k, i.e., $\lambda^{k+1} \approx \lambda^k$. This slows the convergence speed.

The local objective functions of the SCED subproblems are strongly convex. Taking advantages of this feature, we present an accelerated APP based on the technique that was first proposed by Nesterov to accelerate gradient descent methods and later was used to accelerate alternating direction method of multipliers [67, 68]. The suggested accelerated APP utilizes a prediction type acceleration step. The momentum of the algorithm is used to prevent the algorithm from deceleration while more iterations are carried out. After each iteration k, the cumulated gradient of the previous iterations (i.e., momentum) is calculated as the predicted direction of the algorithm in the next iteration, and a big jump is made in that direction. Then, the gradient is measured, and a correction is made to avoid going too fast. This procedure is shown in Fig. 4. The dashed line shows the jump in the direction of the previous accumulated gradient, the dotted line indicates the correction based on measuring the gradient in the new point, and the solid line is the accumulated gradient (predicted-correction direction), used in the accelerated APP algorithm [69].

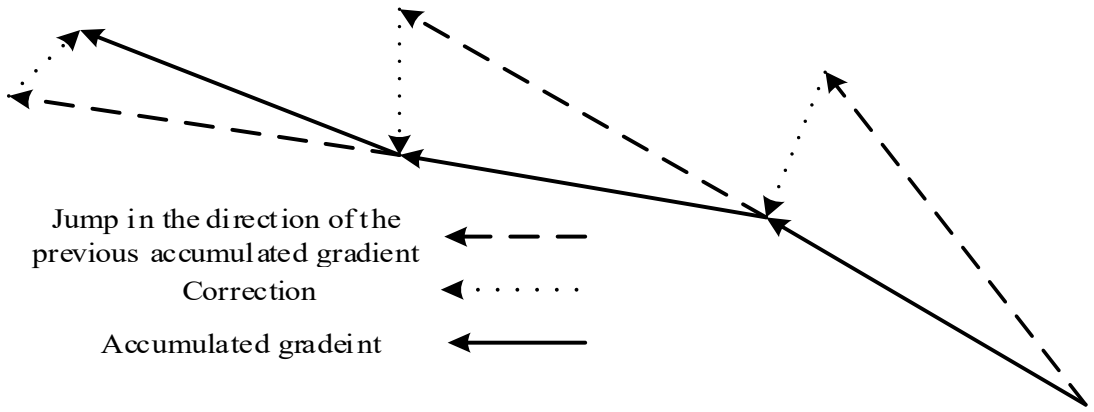


Figure 2.4. Nesterov update.

To implement the accelerated APP for SCED, the penalty term in objective function (10) at iteration k is modified as follows:

$$\sum_{u}\begin{pmatrix}\frac{\rho}{2}\left\|p^{kN_-}_{ut_oN_-\to N}-\phi^{*(k-1)N_-}_{ut_oN_-\to N}\right\|^2+\hat{\lambda}^{k\,\dagger}_{N_-\to N}p^{kN_-}_{ut_oN_-\to N}\\ +\gamma p^{kN_-\dagger}_{ut_oN_-\to N}\left(\phi^{*(k-1)N_-}_{ut_oN_-\to N}-\phi^{*(k-1)N}_{ut_oN_-\to N}\right)\end{pmatrix} \qquad (14)$$

in which $p^{*(k-1)N_-}_{ut_oN_-\to N}$, $p^{*(k-1)N}_{ut_oN_-\to N}$, and $\lambda_{N_-\to N}$ are replaced by $\phi^{*(k-1)N_-}_{ut_oN_-\to N}$, $\phi^{*(k-1)N}_{ut_oN_-\to N}$, and $\hat{\lambda}_{N_-\to N}$. The two penalty terms in objective function (11) at iteration k are modified by replacing $p^{*(k-1)N_-}_{ut_oN_-\to N}$, $p^{*(k-1)N}_{ut_oN_-\to N}$, $p^{*(k-1)N}_{ut_oN\to N_+}$, $p^{*(k-1)N_+}_{ut_oN\to N_+}$, $\lambda_{N_-\to N}$, and $\lambda_{N\to N_+}$ by $\phi^{*(k-1)N_-}_{ut_oN_-\to N}$, $\phi^{*(k-1)N}_{ut_oN_-\to N}$, $\phi^{*(k-1)N}_{ut_oN\to N_+}$, $\phi^{*(k-1)N_+}_{ut_oN\to N_+}$, $\hat{\lambda}_{N_-\to N}$, and $\hat{\lambda}_{N\to N_+}$, respectively.

$$\sum_{u}\left(\begin{pmatrix}\frac{\rho}{2}\left\|p^{kN}_{ut_oN_-\to N}-\phi^{*(k-1)N}_{ut_oN_-\to N}\right\|^2-\hat{\lambda}^{k\,\dagger}_{N_-\to N}p^{kN}_{ut_oN_-\to N}\\ +\gamma p^{kN\dagger}_{ut_oN_-\to N}\left(\phi^{*(k-1)N}_{ut_oN_-\to N}-\phi^{*(k-1)N_-}_{ut_oN_-\to N}\right)\end{pmatrix}\right.$$

$$\left.+\begin{pmatrix}\frac{\rho}{2}\left\|p^{kN}_{ut_oN\to N_+}-\phi^{*(k-1)N}_{ut_oN\to N_+}\right\|^2+\hat{\lambda}^{k\,\dagger}_{N\to N_+}p^{kN}_{ut_oN\to N_+}\\ +\gamma p^{kN\dagger}_{ut_oN\to N_+}\left(\phi^{*(k-1)N}_{ut_oN\to N_+}-\phi^{*(k-1)N_+}_{ut_oN\to N_+}\right)\end{pmatrix}\right) \qquad (15)$$

The new parameters marked by ($\hat{\cdot}$) are predictions of actual shared variables/penalty multipliers.

The predicted values that are used at iteration $k+1$ are calculated based on the actual shared variables and penalty multipliers obtained at iterations k and $k-1$. ϕ^{*k+1} and $\hat{\lambda}^{k+1}$ are updated according to (17) and (18), respectively.

$$\alpha_{k+1} = \frac{1+\sqrt{1+4\alpha_k^2}}{2} \tag{16}$$

$$\phi^{*k+1} = p^{*k} + \frac{\alpha_k - 1}{\alpha_{k+1}}\left(p^{*k} - p^{*(k-1)}\right) \tag{17}$$

$$\hat{\lambda}^{k+1} = \lambda^k + \frac{\alpha_k - 1}{\alpha_{k+1}}(\lambda^k - \lambda^{k-1}) \tag{18}$$

The solution procedure and the way that we calculate the predicted parameters are shown in Algorithm II.

Algorithm II Pseudocode for coordinating SCED subproblems with accelerated APP

1: Decompose the considered horizon into T_N equal subhorizons

2: Initialize $\phi^{*0N_-}_{ut_oN_-\to N}$, $\phi^{*0N}_{ut_oN_-\to N}$, $\phi^{*0N}_{ut_oN\to N_+}$, and $\phi^{*0N_+}_{ut_oN\to N_+}$, $\hat{\lambda}$, β, γ, ρ and set $k=0$

3: **while** $\left|\phi^{*kN_-}_{ut_oN_-\to N} - \phi^{*kN}_{ut_oN_-\to N}\right| > \varepsilon$ & $\left|\phi^{*kN}_{ut_oN\to N_+} - \phi^{*kN_+}_{ut_oN\to N_+}\right| > \varepsilon$, $k = k+1$ **do**

4: Solve SCED subproblems in parallel and determine the optimal values of $p^{*kN_-}_{ut_oN_-\to N}$, $p^{*kN}_{ut_oN_-\to N}$, $p^{*kN}_{ut_oN\to N_+}$, and $p^{*kN_+}_{ut_oN\to N_+}$

5: Exchange $p^{*kN_-}_{ut_oN_-\to N}$, $p^{*kN}_{ut_oN_-\to N}$, $p^{*kN}_{ut_oN\to N_+}$, and $p^{*kN_+}_{ut_oN\to N_+}$ between the SCED subproblems

6: Update λ^k by (12) and (13)

7: Calculate α_{k+1} by (16)

8: Update ϕ^{*k+1} and $\hat{\lambda}^{k+1}$ by (17) and (18)

9: **end while**

2.6.3. Initialization

One of the main drawbacks of distributed and decentralized optimization algorithms is their dependency on initial conditions. That is, the convergence performance (the number of iterations and the optimality gap) might be different if two different sets of initial conditions are used. If a set of good initial conditions is selected, the proposed APP-based distributed SCED potentially converges in a considerably fewer number of iterations compared to a case in which good initial conditions are not available. A good initial condition is system/problem dependent. This is ongoing research in power systems and operations research communities.

Since the main goal of the proposed time decomposition is to decrease the computation time, we need to choose a suitable starting point. For this purpose, we take advantage of power system characteristics and propose a technique to find a set of good initialize values for the shared variables at the overlapping time intervals. To initialize the problem, we ignore the overlapping time intervals and shared variables (i.e., the subhorizons are independent) and solve the SCED subproblems in parallel. Intuitively, since the load does not drastically change by transition from the last time interval of subhorizon N_{-} to the first interval of subhorizon N, ignoring ramp rates of the units (which are eliminated as the shared variables) does not impose a large error on the solution. Although the obtained results might not be feasible, they are close to the feasible solution, and the relative error might be small from the beginning. We use the results of this procedure to initialize the APP-based distributed SCED. The pseudocode for the distributed SCED with the initialization technique is given in Algorithm III.

Algorithm III Pseudocode for coordinating SCED subproblems with APP + initialization

1: Decompose the considered horizon into T_N equal subhorizons

2: Ignore the overlapping time intervals and shared variables

3: Solve the SCED subproblems in parallel

4: Use the obtained results to initialize $\phi^{*0N_-}_{ut_oN_-\rightarrow N}$, $\phi^{*0N}_{ut_oN_-\rightarrow N}$, $\phi^{*0N}_{ut_oN\rightarrow N_+}$, and $\phi^{*0N_+}_{ut_oN\rightarrow N_+}$

5: Set multipliers λ^0, β, γ, ρ and set $k = 0$

6: **Do** the while loop of Algorithm I for the normal APP or Algorithm II for the accelerated APP

2.6.4. Discussion on Convergence

The Nesterov momentum technique is proven to accelerate the convergence of gradient descent methods [30]. On the other hand, APP, which is a gradient descent based method, is proven to converge for convex problems (the considered SCED problem in this chapter is convex) [16]. Hence, intuitively, using the Nesterov accelerated technique instead of the ordinary gradient descent for updating Lagrange multipliers after each iteration of APP keeps the convergence proof of APP valid and enhances its convergence speed.

2.7. Case Study

The proposed algorithm is applied to solve a week-ahead SCED problem for the IEEE 118-bus system and a 472-bus system and a day-ahead problem for a 4720-bus test system. System and equipment characteristics are given in [103]. All simulations are carried out on Matlab using YALMIP [70] as modeling software and the QP solver of ILOG CPLEX 12.4 on a 3.7 GHz personal computer with 16GB of RAM. The solver default settings are used. Although the reported results are obtained by CPLEX, we have applied Gurobi and Mosek and achieved almost the same simulation times as those reported in this chapter.

To evaluate the performance of the proposed distributed SCED algorithm, we use a convergence index rel that shows the relative difference between the operating costs determined by the

distributed SCED (f^d) and the centralized SCED (f^*) (note that f^* is the benchmark cost):

$$rel = \frac{|f^* - f^d|}{f^*} \qquad (19)$$

The closer the convergence measure gets to zero, the more precise solution is obtained. We have found that the choice of 1% maximum mismatch between the shared variables yields a high-quality solution with a negligible rel index for all studied cases. This maximum mismatch is considered as the stopping criterion. Note that unlike the geographical decomposition, the nodal power balance is always satisfied at each iteration of the time decomposition strategy.

2.7.1. IEEE 118-Bus System

Six cases and sensitivity analysis are studied. The operation horizon is divided into seven subhorizons, each including 24 intervals. To have a fair comparison, we set $\beta, \rho, \gamma = 0.2$ and $\lambda = 1$ in all cases except for the sensitivity analysis in which we study the effect of multipliers on the performance of the algorithm.

Case 1: Four contingency scenarios are considered for each hour (note that one can consider a complete set of *N*-1 contingencies. As shown in further cases, the efficiency of the proposed distributed SCED as compared to the centralized SCED becomes better for larger problems). We use a flat start strategy to initialize the shared variables (power generated by units in overlapping intervals) as $p^{N-}_{t_o N-\rightarrow N} = P_{min}$ and $p^{N}_{t_o N-\rightarrow N} = P_{max}$. As shown in Table 2.1, the proposed distributed SCED converges after nine iterations within 11 seconds. The operation cost is \$11,099,286. Figure 2.5 shows the rel values over the course of iterations and time. The convergence measure decreases as more iterations are carried out. The rel index is 0.0004 upon the algorithm convergence (one may run the algorithm for more iterations to get a smaller rel index; however, we stop the algorithm at this point since we further apply the proposed

initialization strategy to significantly reduce error to 1e-8 as shown in Table 2.1).

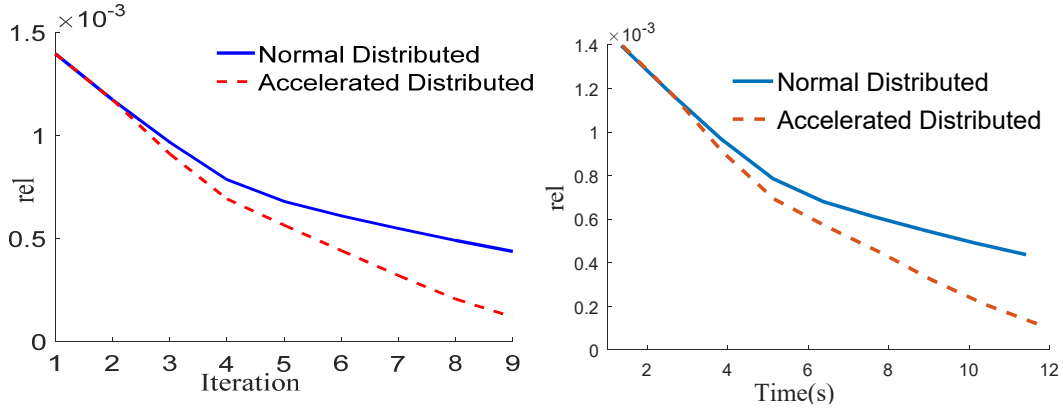
We implement the accelerated distributed algorithm. The results are shown in Table 2.1, and the rel index is depicted in Fig. 2.5. Although the number of iterations and the required time to converge are the same as the normal distributed algorithm, 75% of improvement is observed in rel. The rel index obtained by the accelerated algorithm is less than that for the normal algorithm over the course of iterations. If a user wants to reach a specific rel index (e.g., 0.0002), the normal APP takes more iterations and time to reach that target rel compared to the accelerated APP.

Table 2.1. Comparison of rel and simulation times obtained by distributed and centralized SCED approaches for cases 1 and 2

Case no.	Algorithm	Iteration	rel	Time (s)
	Centralized	-	-	14
Case 1	APP	9	0.0004	11
	A-APP	9	0.0001	11
Case 2	APP+ initialization	2+1	1e-8 (≈ 0)	5
	A-APP+ initialization	2+1	1e-8 (≈ 0)	5

Case 2: The proposed initialization strategy is implemented. All other values are the same as in Case 1. As shown in Table 2.1, the distributed SCED takes 5 seconds to converge after 2+1 iterations (note that 2+1 means that the algorithm takes two iterations in addition to the initialization step. We have counted the computation time of the initialization step in the reported times). The number of iterations is six less than that for case 1, and the computation time has been reduced by 55%. Also, the relative error has been considerably decreased to almost zero. The same

trend is observed for the accelerated SCED. Note that in this case, both normal and accelerated algorithms provide almost the same results. This is because of the very high accuracy of the solution because of the initialization step.



(a) (b)

Figure 2.5. a) Relative error and b) time for case 1 over the course of iterations.

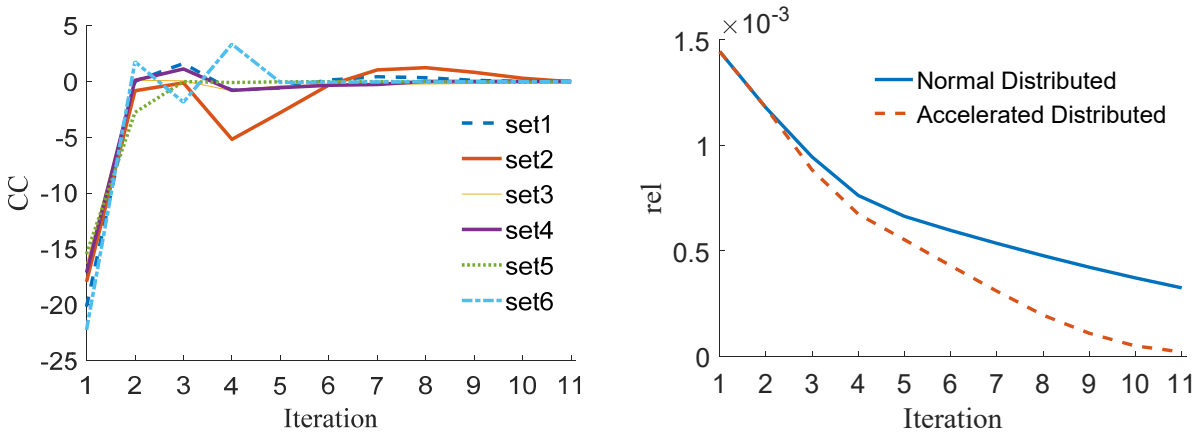
Case 3: This case is similar to case 1 except for the permissible ramping of several units that are purposely reduced to show that the proposed algorithm handles problems of which shared variables in the neighboring subhorizons have a considerable difference at initialization (this makes the SCED problem more computationally complex). Several intertemporal constraints are heavily binding. This condition affects Lagrange multipliers and can be interpreted as a situation in which lines are congested in a geographical decomposition. In such a case, the values of Lagrange multipliers are larger than those for case 1, and thus the sensitivity of the solution to the variations of shared variables is higher. Figure 6a shows the consistency constraint CC (difference between shared variables) corresponding to a sample unit (i.e., unit 25). Six curves are depicted, each of which refers to a shared variable corresponding to unit 25 at the overlapping time intervals. Figure 6b shows the rel index over the course of optimization. The rel and CC values decrease as more iterations are carried out. The convergence measure rel is 0.0003 upon the convergence. We also implement the accelerated APP and present the results in Table 2.2. Although the convergence

time is the same as the normal distributed algorithm, 93% of improvement is observed in rel upon the convergence. Figure 6b depicts that the rel index obtained by the accelerated algorithm is always less than that for the normal algorithm. Note that, since several intertemporal constraints are heavily binding, the CPU time of the centralized SCED increases by 43% and the distributed SCED takes two more iterations than case 1. However, the distributed SCED still shows good performance and is 25% faster than the centralized SCED.

Case 4: Case 3 is reconsidered, and the suggested initialization strategy is applied. The results are given in Table 2.2. Both normal and accelerated distributed SCED algorithms converge after 7+1 iterations within 11 seconds that is 27% and 45% less than the solution times of case 3 and the centralized SCED, respectively. Since the intertemporal constraints between consecutive days are ignored in the initialization step and ramping limits are binding heavily, the initialized solution is farther from the optimal point as compared with case 2 in which the ramp limits are not binding heavily. Thus, the distributed SCED takes five iterations more than that for case 2. Upon convergence, the rel values are less than those for case 3. The accelerated method provides a smaller error compared to the normal algorithm.

Table 2.2. Comparison of rel and simulation times obtained by A-APP, APP, and Centralized SCED for Cases 3 and 4

Case no.	Algorithm	Iteration	rel	Time (s)
	Centralized	-	-	20
Case 3	APP	11	3e-04	15
	A-APP	11	2e-05	15
Case 4	APP+ initialization	7+1	2e-06	11
	A-APP+ initialization	7+1	6e-07	11



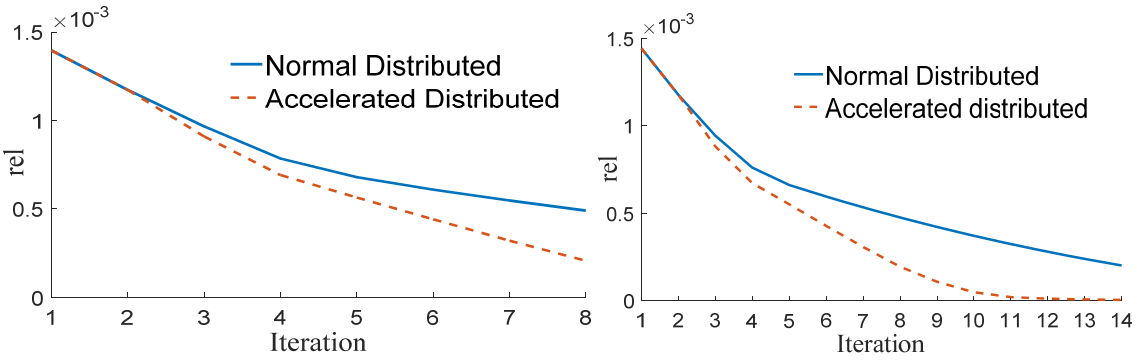
(a) (b)

Figure 2.6. a) Consistency constraints corresponding to unit 25 and b) relative error in case 3.

Case 5: We consider ten contingency scenarios and compare the results with and without the suggested initialization strategy. All other parameters are the same as in case 2. Table 2.3. shows the comparison of the proposed A-APP with APP and the centralized SCED. The suggested initialization strategy reduced the number of iterations (CPU time) from 8 (31 seconds) to 2+1 iterations (12 seconds). As observed in Fig. 7, the rel index obtained by the accelerated APP is always less than that for the normal APP. Increasing the number of credible contingencies from four to ten increases the computation time of the centralized SCED by 320%, while the distributed SCED is only 10% slower than that for case 2. The distributed SCED takes 12 seconds to converge that is almost five times faster than the centralized SCED that takes 59 seconds. The difference between each pair of the shared variables and the relative error are almost zero upon convergence.

Table 2.3. Comparison of rel and simulation times obtained by A-APP, APP, and Centralized SCED for Case 5

Algorithm	Iteration	rel	Time (s)
Centralized	-	-	59
APP	8	0.0005	31
A-APP	8	0.0002	31
APP + initialization	2+1	1e-07	12
A-APP + initialization	2+1	1e-07	12



(a) (b)

Figure 2.7. The rel index for normal and accelerated APP without initialization obtained in a) case 5 and b) case 6.

Case 6: This is similar to case 5, but we decrease the permissible ramping of several units. Therefore, the difference between the shared variables after the initialization step is much more than that in case 5. As shown in Table 2.4, the distributed SCED with (without) the suggested initialization strategy converges after 9+1 (14) iterations, seven (six) iterations more than that for case 5. The CPU time with (without) the suggested initialization is 38 (54) seconds, which is 46% (30%) faster than the centralized method. As observed in Table 2.4 and Fig. 2.7, the accelerated APP provides a smaller error than the normal APP. The maximum acceptable mismatch is selected 1%, but usually, upon the convergence, this mismatch for most generators is much smaller than

1%. Note that the power balance constraints are considered as hard constraints and are always satisfied regardless of the mismatch in generations. Considering the sign of mismatches, the summation of mismatches is zero at every interval.

Table 2.4. Comparison of rel and simulation times obtained by A-APP, APP, and Centralized SCED for Case 6

Algorithm	Iteration	rel	Time (s)
Centralized	-	-	70
APP	14	0.0002	54
A-APP	14	6e-06	54
APP + initialization	9+1	2e-06	38
A-APP +initialization	9+1	3e-07	38

Insight on selecting suitable multipliers: Although we have used the same multipliers for cases 1 to 6 to have a fair comparison, this set of multipliers may not be the best choice for all cases. We can change the multipliers to obtain better results. In this section, we perform a sensitivity analysis and give an insight on how to initialize multipliers β, ρ, and γ. We examine the effect of multipliers and found suitable sets for cases 1 to 4. By changing multipliers from 0.1 to 10, we plot the rel index versus the number of iterations required for convergence to determine the most suitable initial point for each case. Fig. 2.8 shows the simulation results for different multiplier values. The best solutions are highlighted by red color in Fig. 2.8 and their corresponding multipliers are shown in Table 2.5. Cases 1 and 3 are more sensitive to the choice of initial values, while cases 2 and 4 in which the suggested initialization strategy is implemented are less sensitive to the initial values. As expected, we have observed that selecting large multipliers potentially reduces the number of iterations; however, it increases error.

When a user wants to use the distributed SCED algorithm with no historical information, we recommend selecting small values for multipliers. The user will gradually obtain knowledge on suitable values of multipliers when more simulations are carried out. Evaluating the recorded results, the user can adjust the multipliers to enhance the solution performance for future simulations. In addition, it is proper to select ρ and γ equal to β [71].

Table 2.5. Best multipliers for cases 1 to 4

Case No.	β, ρ, γ	rel	iteration
Case 1	0.2	4e-4	9
Case 2	1.25	1e-10	2
Case 3	0.1	4e-5	11
Case 4	0.1	1e-8	7

2.7.2. 472-Bus System

In *Case 7*: We connect four IEEE 118-bus systems to make a larger (472-bus) system and compare changes in simulation times. Ten contingency scenarios are considered for each hour. The operation cost using both centralized and distributed SCED algorithms is $44,376,003. The number of iterations, rel, and solution time using the normal APP and the proposed A-APP are compared with the centralized SCED in Table 2.5. The convergence measure rel is almost zero upon convergence. The centralized SCED takes around 40 minutes, but both normal and accelerated distributed algorithms converge after 2 minutes that are around 94% faster than the centralized algorithm.

Although the size of the system is four times larger than the 118-bus system, the solution time of centralized SCED is increased around 40 times compared to Case 5, which has the same loading.

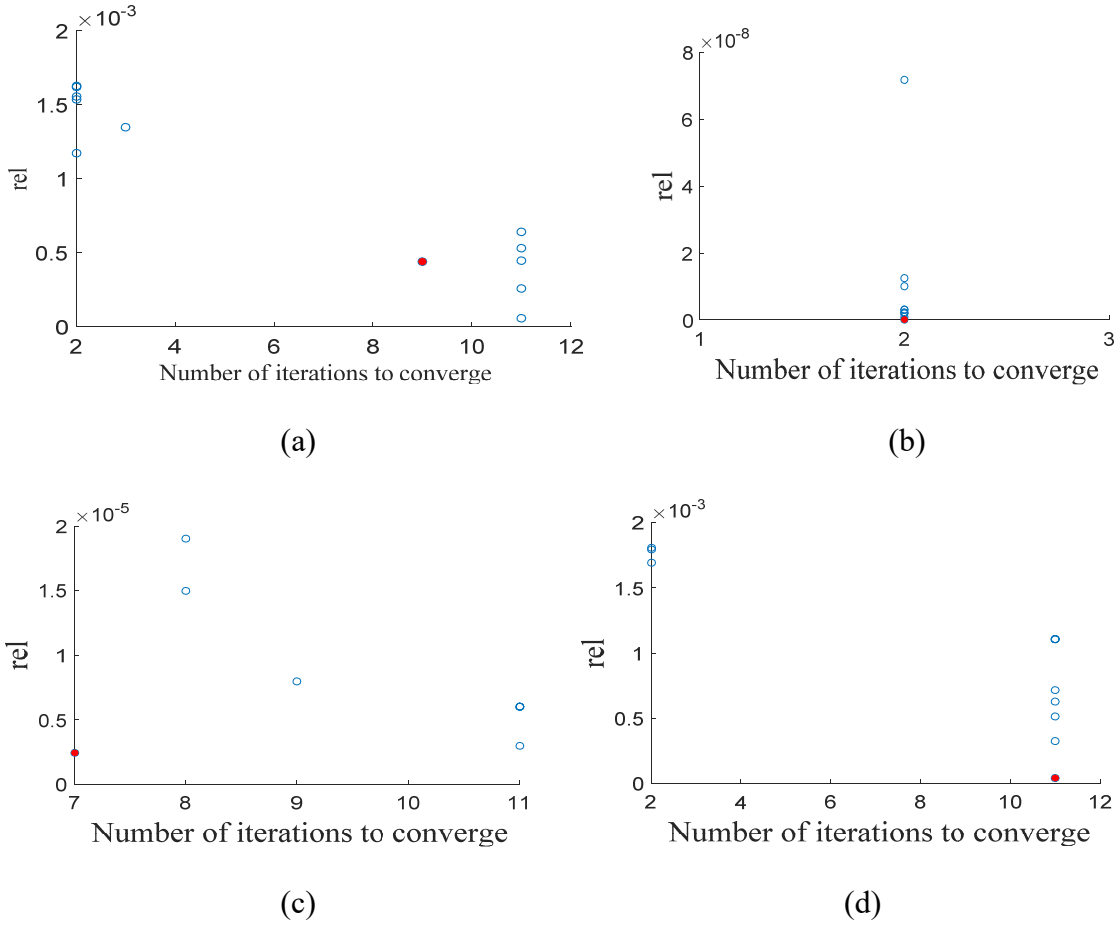


Figure 2.8. Relative error and the number of iterations to converge depending on the changes of multipliers β, ρ, and γ, a) case 1, b) case 2, c) case 3, and d) case 4.

Table 2.6. Comparison of rel and simulation times obtained by A-APP, APP, and Centralized SCED for Case 7

Algorithm	Iteration	rel	Time (s)
Centralized	-	-	2400
APP+ initialization	2+1	1e-08	140
A-APP+ initialization	2+1	1e-08	140

Case 8: We manipulate ramping capability of generating units to make ramping limits of several units heavily binding in several hours and make more consistency constraints active. In this situation, the shared variables of neighboring subproblems have a larger difference at the

initialization step, and the initial solution is farther from the feasible and optimal point. The results are summarized in Table 2.7. The operation cost obtained by the centralized SCED is $44,388,886. Although the distributed algorithm takes seven iterations more than case 8, it still is 72% faster than the centralized SCED.

Table 2.7. Comparison of rel and simulation times obtained by A-APP, APP, and Centralized SCED for Case 8

Algorithm	Iteration	rel	Time (s)
Centralized	-	-	1900
APP+ initialization	9+1	1e-06	532
A-APP+ initialization	9+1	1e-07	532

2.7.3. 4720-Bus System

In *Case 9*: A day-ahead scheduling problem is considered for a 4720-bus test system. The centralized SCED takes 550 seconds to provide a cost of $64,736,891. The time horizon is decomposed into four subhorizons, each including six hours. The proposed distributed SCED converges after 1+1 iterations within 141 seconds. The rel index is almost zero, and the solution time is improved by a factor around four.

Table 2.8. Comparison of proposed distributed and centralized SCED approaches for 4720-bus system

Algorithm	Iterations	rel	Time (s)
Centralized	-	1+1	550
Accelerated Distributed	1+1	3e-9	141

Case 10: We have studied a network constrained economic dispatch (NCED) for this system to analyze the effect of the number of subproblems on the overall solution time. Fig. 9 shows the

solution time variation versus the number of subhorizons. A considerable time saving is achieved even if the time horizon is decomposed into a few subhorizons. Increasing the number of subhorizons increases the number of shared variables and the required iterations for A-APP to converge. Hence, the time saving is not significant if the number of subhorizons goes beyond a certain number. Our simulations show that having four subhorizons for this test system provides the best time saving and rel index.

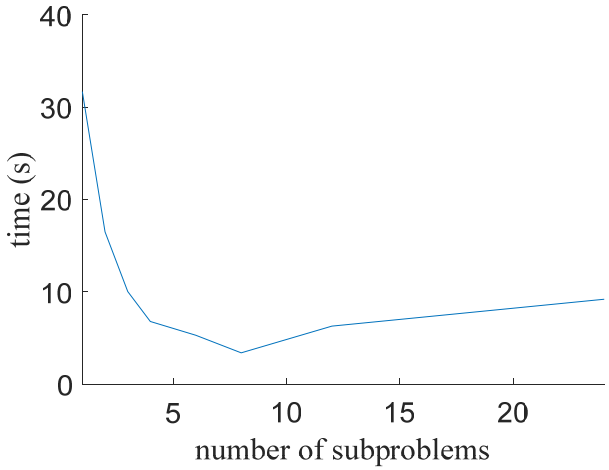


Figure 2.9. Time versus the number of subhorizons for the 4720-bus system.

2.8. Conclusion

In this chapter, a decomposition strategy is proposed to divide the ramp-constrained SCED problem. Since the optimization is decomposed over the scheduling time horizon, the proposed strategy is called a horizontal time decomposition. The concept of overlapping time intervals is introduced to model interdependencies (coupling constraints) between the subproblems, which originate from intertemporal (or ramping) constraints of generating units. An accelerated APP and an initialization strategy are proposed to enhance the convergence performance of the distributed SCED algorithm.

The simulation results show that the proposed algorithm reduces the computation time of SCED for the IEEE 118-bus system from 45% to 74% depending on the size of the considered problem.

For cases with more contingency scenarios and less limited generation ramping capabilities, more time-saving is obtained using the distributed SCED instead of the centralized method. The simulation results on large test systems show that as the size of the problem (that depends on the size of the system and the number of contingencies) increases, the distributed algorithm shows better performance.

CHAPTER 3
TEMPORAL DECOMPOSITION-BASED STOCHASTIC SCED FOR SMART GRID ENERGY MANAGEMENT

3.1. Introduction

This chapter presents a temporal decomposition strategy to decompose security-constrained economic dispatch (SCED) over the scheduling horizon with the goal of reducing its computational burden and enhancing its scalability. The problem presented in Chapter 2 was a simple SCED. However, the goal of this chapter is to solve SCED in a real modern network. A set of subproblems, each with respect to demand response, normal constraints, and N-1 contingency corrective actions at a subhorizon, is formulated. Unlike geographical decompositions, the proposed temporal decomposition relives computational complexities originated from intertemporal interdependencies of system equipment, i.e., generators' ramp constraints and state of change of storage devices. The concept of overlapping intervals is introduced to make SCED subproblems solvable in parallel. Intertemporal connectivity related to energy storage is also modeled in the context of temporal decomposition. Besides, reserve up and down requirements are formulated as data-driven nonparametric chance constraints to account for wind generation uncertainties. The concept of ϕ-divergence is used to convert nonparametric chance constraints to more conservative parametric constraints. A reduced risk level is calculated with respect to wind generation prediction errors to ensure the satisfaction of system constraints with a confidence level after the true realization of uncertainty. Auxiliary problem principle is applied to coordinate SCED subproblems in parallel. Numerical results on two test systems show the effectiveness of the proposed algorithm as compared to the centralized SCED.

3.2. Symbols

A. Indices and Sets:

c	Index for contingencies.
i, j	Index for buses.
ij	Index for lines.
k	Index for iterations.
m	Index for subproblems and related subhorizons.
s	Index for storage devices.
t	Index for time intervals.
u	Index for units.
w	Index for wind farms.
a, b	Cost coefficients for generating units
b_s	Storage operational costs.
b_{sh}	load shedding cost.

B. Parameters:

$PD_{i,t}$	Power demand at bus i at time t.
PD_t	Total power demand at time t.
SP_m	Subproblem m corresponding to subhorizon m
SP_m^+	Subproblem m with dummy time intervals.
$ICC_{m,m+1}$	Intertemporal consistency constraints between two consecutive subproblems m and $m+1$.
$PD_{i,t}$	Power demand at bus i at time t.
PD_t	Total power demand at time t.
SP_m	Subproblem m corresponding to subhorizon m
SP_m^+	Subproblem m with dummy time intervals.
$ICC_{m,m+1}$	Intertemporal consistency constraints between two consecutive subproblems m and $m+1$.
UR_u	Ramping up limit of unit u.
DR_u	Ramping down limit of unit u.
T	Overall time horizon.
Ξ_{m+1}^{*k-1}	Values of shared variables determined by subproblems SP_{m+1}^+ at iteration $k-1$.
λ	Vector of Lagrange multipliers.
β, γ	Tuning parameters.
ρ	Penalty factor.
X_{ij}	Reactance of line between buses i and j.
$\underline{z}, \overline{z}$	Minimum and maximum of a variable z.
z_t^c	Values of a variable z at time t after contingency c.
α	Risk level of chance constraints.

C. Variables:

$E_{s,t}$	State of charge of storage s at time t.

$I_{ch,s}, I_{dc,s}$	Binary variables for modeling storage charging and discharging modes.
$p_{u,t}$	Generation of unit u at time t.
$p_{ch,s,t}$	Charging power of storage s at time t.
$p_{dc,s,t}$	Discharging power of storage s at time t.
$\mathrm{p}_{\mathrm{ij,t}}$	Flow in line ij at time t.
$\mathrm{p}_{\mathrm{sh,i,t}}$	Load shedding at bus i at time t.
$\delta_{\mathrm{i,t}}$	Voltage angle of bus i at time t.
$\mathrm{r}^{\mathrm{d}}_{\mathrm{u,t}}$	Reserve down provided by unit u at time t.
$\mathrm{r}^{\mathrm{up}}_{\mathrm{u,t}}$	Reserve up provided by unit u at time t.
$\Xi_{\mathrm{m-1,m}}$	Set of shared decision variables corresponding to overlapping time intervals between su
$\Xi^{\mathrm{k}}_{\mathrm{m}}, \Xi^{\mathrm{k}}_{\mathrm{m}'}$	Shared variables of subproblem m with its previous and next neighbors at iteration k.
$\mathrm{z}^{\mathrm{c}}_{\mathrm{t}}$	Values of a variable z at time t after contingency c.

D. Uncertainty Related Parameters and Functions:

$f_{w,t}(\cdot)$	True wind power PDFs at interval t.
$\hat{f}_{w,t}(\cdot)$	Estimated wind power PDFs at interval t.
$E(\tilde{P}_{w,t})$	Expected value of wind power generation at time t.
$\tilde{P}_{w,t}$	Wind power generation at time t.
$\tilde{\xi}$	A random variable.
α	Confidence level of chance constraints.
$\mathbb{P}$	Probability measure.
$\widehat{\mathbb{P}}$	Estimated distribution.
$\widehat{CDF}^{-1}$	Estimated quantile function (i.e., inverse CDF).

3.3. Contributions

The main contributions of this chapter are summarized as follows:

- To decrease the solution time of operation problems for a modern system which includes wind generation, controllable loads and energy storage.
- Including a spinning reserve for generating units to mitigate the uncertainties of wind generation.
- We propose to decompose the problem over time horizon and solve the achieved subproblems in parallel. The complexity arises from modeling the intertemporal constraints such as ramping and state of charge of batteries between subproblems when

we want to avoid sub-optimality resulted from forcing the state of a subproblem which is solved earlier.

- We propose to define a new time interval between every two consecutive subproblems and we name it a "dummy time interval". The state of this dummy interval defines a set of shared variables between the two subproblems.
- Auxiliary problem principle (APP), with a suitable initialization technique is proposed for coordinating the consecutive subproblems in parallel.

3.4. Temporal Decomposition of SCED

Consider a centralized SCED problem (CP) with an overall time horizon of T. As shown in Fig. 1, we proposed to decompose CP over the time horizon to create n SCED subproblems (SPs), each including a subhorizon of T so that:

$$\{1, \dots, T_1\} \cup \dots \cup \{T_{n-1} + 1, \dots, T_n\} = \{1, \dots, T\} \qquad (1)$$

Each SCED SP contains fewer variables and constraints than the original CP, particularly intertemporal constraints. Hence, SPs are computationally less expensive than CP.

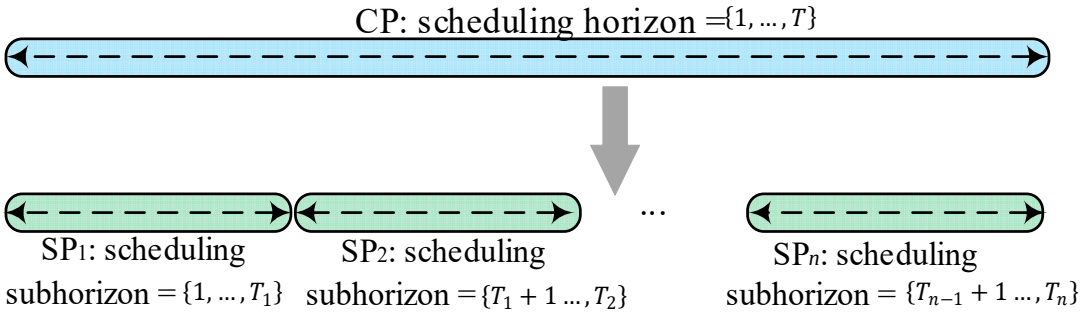


Figure 3.1. Time decomposition concept.

3.4.1. Deterministic Constraints of Subproblem m

Consider subhorizon m with the scheduling interval of $\{T_{m-1} + 1, \dots, T_m\} \subseteq \{1, \dots, T\}$. A single-stage nonparametric chance-constrained-based SCED is formulated for SP_m whose objective function is to minimize operational costs over subhorizon m. The first three terms of (2)

are generators cost function, the fourth term is the reserve cost, and the fifth term is the load shedding cost [72].

$$\min \sum_{t} \sum_{u} a_u {p_{u,t}}^2 + b_u p_{u,t} + o_{u,t} + b_u r_{u,t}^{up} + \sum_{i} b_{sh,i,t}\, p_{sh,i,t} \qquad (2)$$

The continuous decision variables are $\{E_{s,t}, p_{u,t}, p_{ch,s,t}, p_{dc,s,t}, p_{ij,t}, p_{sh,i,t}, \delta_{i,t}, r_{u,t}^{d}, r_{u,t}^{up}\}$ and the binary decision variables are $\{I_{ch,s}, I_{dc,s}\}$ under normal conditions and each contingency c. Constraints of SP_m include system and component restrictions under normal condition and $N-1$ security criteria $\forall t \in \{T_{m-1}+1, \dots, T_m\}$. Constraints under normal condition are power balance equalities (3)-(4) and transmission line limitation (6), load shedding limitation (7), upper and lower bounds of generating units (8)-(9), units ramping restriction (10)-(11), units spinning reserve up and down constraints (12)-(15), storage state of change at time t (16), storage charge and discharge power restrictions (17)-(18), constraints to avoid simultaneous storage charge and discharge (19), and upper and lower bounds of storage energy level (20) [72-74]. To ensure system security after the occurrence of a contingency, corrective actions should be scheduled by adjusting control variables (e.g., power generated by units and storage power charge/discharge) within a reasonable rage. A set of new variables are defined for each contingency c. System and equipment constraints corresponding to each contingency c, known as $N-1$ security criteria [72-74], include power balance constraints (21)-(22), line flow limits (24), load shedding constraints (25)-(26), thermal units upper and lower bounds considering reserve values (27)-(32), thermal units adjustment capabilities after occurrence of a contingency as compared to their generation under normal condition (33), and storage limitations after occurrence of a contingency (34)-(38). Expression (34) shows the amount of energy that a storage unit must provide to help the system after the occurrence of a contingency [74]. After an outage, fast-response storage units inject or extract

power instantly to bring line flows back down within their short-term emergency rating. The power injections or extractions from the storage units remain constant for a period of τ_1 (e.g., 5 minutes) until generators start ramping. During the ramping period τ_2 (e.g., 10 minutes), storage units reduce their injections or extractions until they reach zero, while generators ramp their output up or down. Flows in overloaded lines decrease until they reach their long-term emergency rating [74].

$$\sum_u p_{u,t} + \sum_s \left(p_{dc,s,t} - p_{ch,s,t}\right) + \sum_w E\left(\tilde{P}_{w,t}\right) = \sum_i PD_{i,t} - p_{sh,i,t} \qquad \forall t \qquad (3)$$

$$p_{i,t} - PD_{i,t} = \sum_j \frac{\delta_{i,t} - \delta_{j,t}}{X_{ij}} \qquad \forall i, \forall t \qquad (4)$$

$$\delta_{ref,t} = 0 \qquad \forall t \qquad (5)$$

$$\underline{P}_{ij} \le p_{ij,t} = \frac{\delta_{i,t} - \delta_{j,t}}{X_{ij}} \le \overline{P}_{ij} \qquad \forall ij, \forall t \qquad (6)$$

$$\underline{P}_{sh,i} \le p_{sh,i,t} \le \bar{P}_{sh,i} \qquad \forall i, \forall t \qquad (7)$$

$$\underline{P}_{u,t} \le p_{u,t} + r_{u,t}^{up} \le \overline{P}_{u,t} \qquad \forall u, \forall t \qquad (8)$$

$$\underline{P}_{u,t} \le p_{u,t} - r_{u,t}^{d} \le \overline{P}_{u,t} \qquad \forall u, \forall t \qquad (9)$$

$$\left(p_{u,t} + r_{u,t}^{up}\right) - \left(p_{u,t-1} - r_{u,t}^{d}\right) \le \mathrm{UR}_u \quad \forall u, \forall t \qquad (10)$$

$$\left(p_{u,t-1} + r_{u,t}^{up}\right) - \left(p_{u,t} - r_{u,t}^{d}\right) \le \mathrm{DR}_u \quad \forall u, \forall t \qquad (11)$$

$$0 \le r_{u,t}^{up} \le \mathrm{UR}_{u,10\,min} \qquad \forall u, \forall t \qquad (12)$$

$$0 \le r_{u,t}^{up} \le \overline{P}_{u,t} - p_{u,t} \qquad \forall u, \forall t \qquad (13)$$

$$0 \le r_{u,t}^{d} \le \mathrm{DR}_{u,10\,min} \qquad \forall u, \forall t \qquad (14)$$

$$0 \le r_{u,t}^{d} \le p_{u,t} - \underline{P}_{u,t} \qquad \forall u, \forall t \qquad (15)$$

$$E_{s,t} = E_{s,t-1} + \left(\mu p_{ch,s,t} - \frac{p_{dc,s,t}}{\mu}\right)\Delta t \qquad \forall s, \forall t \qquad (16)$$

$$I_{ch,s,t} \cdot \underline{P}_{ch,s} \le p_{ch,s,t} \le I_{ch,s,t} \cdot \overline{P}_{ch,s} \qquad \forall\, s, \forall t \qquad (17)$$

$$I_{dc,s,t} \cdot \underline{P}_{dc,s} \le p_{dc,s,t} \le I_{dc,s,t} \cdot \overline{P}_{dc,s} \qquad \forall\, s, \forall t \qquad (18)$$

$$I_{dc,s,t} + I_{ch,s,t} \le 1 \qquad \forall\, s, \forall t \qquad (19)$$

$$\underline{E}_s \le E_{s,t} \le \overline{E}_s \qquad \forall s, \forall t \qquad (20)$$

$$\sum_u p^c{}_{u,t} + \sum_s \left(p^c_{dc,s,t} - p^c_{ch,s,t}\right) + \sum_w E\left(\tilde{P}_{w,t}\right) = \sum_i D_{i,t} - p^c_{sh,i,t} \qquad \forall t, \forall c \qquad (21)$$

$$p^c_{i,t} - PD_{i,t} = \sum_j \frac{\delta^c_{i,t} - \delta^c_{j,t}}{X_{ij}} \qquad \forall i, \forall t, \forall c \qquad (22)$$

$$\delta^c_{ref,t} = 0 \qquad \forall t, \forall c \qquad (23)$$

$$\underline{P}_{ij} \le p^c_{ij,t} = \frac{\delta^c{}_i - \delta^c{}_j}{X_{ij}} \le \overline{P}_{ij} \qquad \forall ij, \forall c \qquad (24)$$

$$\underline{P}_{sh,i} \le p^c_{sh,i,t} \le \bar{P}_{sh,i} \qquad \forall i, \forall t, \forall c \qquad (25)$$

$$\left|p_{sh,i,t} - p^c_{sh,i,t}\right| \le \Delta \qquad \forall u, \forall t, \forall c \qquad (26)$$

$$\underline{P}_{u,t} \le p^{\mathrm{c}}_{u,t} + r^{up,c}_{u,t} \le \overline{P}_{u,t} \qquad \forall u, \forall t, \forall c \qquad (27)$$

$$\underline{P}_{u,t} \le p^{\mathrm{c}}_{u,t} - r^{d,c}_{u,t} \le \overline{P}_{u,t} \qquad \forall u, \forall t, \forall c \qquad (28)$$

$$0 \le r^{up,c}_{u,t} \le UR_{u,10\,min} \qquad \forall u, \forall t, \forall c \qquad (29)$$

$$0 \le r^{up,c}_{u,t} \le \overline{P}_{u,t} - p_{u,t} \qquad \forall\, u, \forall t, \forall c \qquad (30)$$

$$0 \le r^{d,c}_{u,t} \le DR_{u,10\,min} \qquad \forall\, u, \forall t, \forall c \qquad (31)$$

$$0 \le r^{d,c}_{u,t} \le p_{u,t} - \underline{P}_{u,t} \qquad \forall\, u, \forall t, \forall c \qquad (32)$$

$$\left|p_{u,t} - p^c_{u,t}\right| \le \Delta \qquad \forall u, \forall t, \forall c \qquad (33)$$

$$E^c_{s,t} = E_{s,t} + (\tau_1 + 0.5\tau_2)\left(\mu p^c_{ch,s,t} - \frac{p^c_{dc,s,t}}{\mu}\right)\Delta t \qquad \forall s, \forall t, \forall c \qquad (34)$$

$$I^c_{ch,s,t} \cdot \underline{P}_{ch,s} \le p^c_{ch,s,t} \le I^c_{ch,s,t} \cdot \overline{P}_{ch,s} \qquad \forall s, \forall t, \forall c \qquad (35)$$

$$I^c_{dc,s,t} \cdot \underline{P}_{dc,s} \le p^c_{dc,s,t} \le I^c_{dc,s,t} \cdot \overline{P}_{dc,s} \quad \forall s, \forall t, \forall c \quad (36)$$

$$I^c_{dc,s,t} + I^c_{ch,s,t} \le 1 \qquad \forall s, \forall t, \forall c \qquad (37)$$

$$\underline{E}_s \le E^c_{s,t} \le \overline{E}_s \qquad \forall s, \forall t, \forall c \quad (38)$$

3.4.2. Probabilistic Constraints of Subproblem m

In addition to (3)-(38), adequate reserve up and down should be provided to compensate wind generation forecast errors. Considering wind generation uncertainties, reserve requirements are formulated as probabilistic constraints. Chance-constrained programming is a suitable approach for modeling reserve requirements [64]. The majority of existing papers consider a known PDF for wind power and formulate parametric chance constraints. However, wind power depends on different factors and may not follow any known class of distribution functions. Chance constraints are sensitive to PDFs. If the true realizations of wind generation do not match with the assumed PDF, the probability of satisfaction of parametric chance constraints may violate the predetermined confidence level that further results in system security degradation.

We propose two data-driven nonparametric chance constraints (41) and (42) to ensure that generation and reserve down/up provided by thermal units and storage satisfy load with a confidence level of $1-\alpha$ if wind power goes above or falls below its forecast values. A confidence set for each interval t is specified by the ϕ-divergence function and tolerance $d_{w,t}$ representing the size of the confidence set [75]. We can use the worst distribution in the confidence set (i.e., the PDF with the largest distance from the true PDF within a tolerance d) to formulate nonparametric chance constraints (41) and (42). Using historical data, we define the confidence set for wind generation at time t as [75]:

$$\mathcal{D}_{w,t} = \left\{ \mathbb{P}_{w,t} \in \mathcal{M}_+ : D_{w,t}\left(f_{w,t} || \hat{f}_{w,t}\right) \le d_{w,t}, f_{w,t} = \frac{\mathrm{d}\mathbb{P}_{w,t}}{\mathrm{d}\tilde{\xi}_t} \right\} \qquad (39)$$

where $\mathbb{P}_{w,t}$ represents the ambiguous true distribution function for wind power at interval t, and $\mathcal{M}_+$ is the set of all distribution functions. The distance between the estimated and true wind power PDFs, or ϕ-divergence $D_{s,t}(\cdot \,||\, \cdot)$, is defined as [75-77]:

$$D_{w,t}\left(f_{w,t}||\hat{f}_{w,t}\right) = \int_{\mathbb{R}^m} \phi\left(\frac{f_{w,t}\left(\tilde{\xi}_t\right)}{\hat{f}_{w,t}\left(\tilde{\xi}_t\right)}\right) \hat{f}_{w,t}\left(\tilde{\xi}_t\right) \mathrm{d}\tilde{\xi}_t \tag{40}$$

where $\phi(\cdot)$ is a convex function on $\mathbb{R}^+$. The summation of generation minus reserve down should be less than or equal to demand. The reserve down is required for situations in which the true realization of wind generation is larger than its predicted value.

$$\inf_{\mathbb{P}\in\mathcal{D}} \mathbb{P}\left\{\sum_u p_{u,t} - r^d_{u,t} + \sum_s p_{dc,s,t} - p_{ch,s,t} + \sum_i p_{sh,i,t} + \tilde{P}_{w,t} \le PD_t\right\} \ge 1-\alpha \qquad \forall t \tag{41}$$

where the inf operator represents PDF with the largest distance from the true PDF within a tolerance d. The nonparametric reserve up chance constraints is required when the true realization of wind generation is less than its expected value.

$$\inf_{\mathbb{P}\in\mathcal{D}_{w,t}} \mathbb{P}\left\{\sum_u p_{u,t} + r^{up}_{u,t} + \sum_s p_{dc,s,t} - p_{ch,s,t} + \sum_i p_{sh,i,t} + \tilde{P}_{w,t} \ge PD_t\right\} \ge 1-\alpha \qquad \forall t \tag{42}$$

We replace the risk level α with a reduced nonnegative risk level α'_+ with respect to the divergence function $\phi(\cdot)$ and the divergence tolerance d. Then, as proven in [76], the predicted PDF for wind generation, $\widehat{\mathbb{P}}$, can be used to reformulate (41) and (42) as parametric chance constraint using $\widehat{\mathbb{P}}$ instead of $\inf_{\mathbb{P}\in\mathcal{D}}$. This procedure is described below.

- Impose no assumption on the PDF of wind generation at each time interval, and estimate the unknown PDF ($\widehat{\mathbb{P}}$) of random parameters from historical data using adaptive kernel density estimator (AKDE) [78].

- Form a histogram set of real data, from historical wind generation at time t. Determine pointwise errors between the histogram and $\widehat{\mathbb{P}}$, and calculate the distance d from the square of errors as [75]:

$$d = SE_{1-\alpha}^2 \tag{43}$$

- Choose an appropriate divergence function $\phi(\cdot)$, such as the χ divergence of order two that is suitable for small risk levels, and solve a univariate optimization problem to find α' for each time interval t as [76]:

$$\alpha' = \alpha - \frac{\sqrt{d^2 - 4d(\alpha - \alpha^2)} - (1 - 2\alpha)d}{2d + 2} \tag{44}$$

- Avoid negative risk levels as:

$$\alpha'^{+} = \max(0, \alpha') \tag{45}$$

- In (41) and (42), replace $\inf_{\mathbb{P}\in\mathcal{D}}$ by $\widehat{\mathbb{P}}$ and α by α'_{+}.

We now rewrite reserve down constraint (41) as follows:

$$\widehat{\mathbb{P}}\left\{\tilde{P}_{w,t} \le PD_t - \left(\sum_u p_{u,t} - r_{u,t}^{d} + \sum_s p_{dc,s,t} - p_{ch,s,t} + \sum_i p_{sh,i,t}\right)\right\} \ge 1 - \alpha_t'^{+} \quad \forall t \tag{46}$$

Knowing that in the probability theory $\mathbb{P}\{Y \le y\} = 1 - \mathbb{P}\{Y > y\}$, we reorganize the reserve up (45) as:

$$\widehat{\mathbb{P}}\left\{\tilde{P}_{w,t} \le PD_t - \left(\sum_u p_{u,t} + r_{u,t}^{up} + \sum_s p_{dc,s,t} - p_{ch,s,t} + \sum_i p_{sh,i,t}\right)\right\} \le \alpha_t'^{+} \quad \forall t \tag{47}$$

The left sides of (46) and (47) are the estimated cumulative distribution function (CDF) of wind generation $\tilde{P}_{w,t}$ at time t. By taking the CDF inverse, (46) and (47) are expressed as:

$$PD_t - \left(\sum_u p_{u,t} + r_{u,t}^{up} + \sum_s p_{dc,s,t} - p_{ch,s,t} + \sum_i p_{sh,i,t}\right) \le \widehat{CDF}^{-1}_{\tilde{P}_{w,t}}(\alpha_t'^{+}) \quad \forall t \tag{48}$$

$$PD_t - \left(\sum_u p_{u,t} - r^d_{u,t} + \sum_s p_{dc,s,t} - p_{ch,s,t} + \sum_i p_{sh,i,t} \right) \geq \widehat{CDF}^{-1}{}_{\tilde{P}_{w,t}}(1 - \alpha'^{+}_t) \quad \forall t \quad (49)$$

The left-hand sides of (48) and (49) are variables and the right-hand sides are constant values. These linear constraints, which are linear equivalents of (41) and (42), ensure that the reserve up and down constraints are satisfied for all wind power distribution functions in the confidence set.

The reduced risk level makes chance constraints more conservative, which is needed to account for wind power PDFs estimation errors. The level of conservativeness is related to the reduced risk level that itself depends on errors of estimated PDFs. If more data or better estimation approaches are available to obtain more accurate wind power PDFs, the reduced risk level becomes closer to the user's predetermined risk level and the level of conservativeness of constraints goes down.

3.5. Modeling Subhorizons Temporal Interdependencies

3.5.1. Recovering Centralized SCED from Subproblems

Consider two consecutive subhorizons m and $m+1$ that cover intervals $\{T_{m-1}+1, \dots, T_m\}$ and $\{T_m+1, \dots, T_{m+1}\}$, respectively. Formulating SP_m and SP_{m+1} as presented in the previous section models intertemporal constraints inside each subhorizon, but not constraints for the transition between two consecutive subhorizons. Thus, the constraint sets of subproblems (denoted by *CSP*) are mutually exclusive and their intersection is empty.

$$CSP_m \cap CSP_{m+1} = \emptyset \quad (50)$$

This is not correct from the centralized SCED's perspective. Although the overall objective function of the centralized SCED can be recovered by combining all subproblems' local objective functions, its constraint set cannot be reconstructed since in the centralized SECD we have:

$$CSP_{T_m} \cap CSP_{T_{m+1}} \neq \emptyset \quad (51)$$

That means collecting all elements (i.e., objective terms and constraints sets) of subproblems by

taking their union does not construct the centralized problem.

$$CP \neq SP_1 \cup SP_2 \ldots \cup SP_m \cup SP_{m+1} \cup \ldots \cup SP_n \qquad (52)$$

Hence, solving these independent subproblems provide a solution that may be infeasible or suboptimal from the perspective of the whole scheduling horizon T. To solve this challenge, as shown in Fig. 2, a set of intertemporal consistency constraints (ICCs) must be added to the problem to model transition between subproblems. Now, the centralized SCED can be reconstructed by collecting all elements of SP_1 to SP_n and ICCs.

$$CP = SP_1 \cup ICC_{1,2} \cup SP_2 \cup ICC_{2,3} \cup \ldots \cup ICC_{n-1,n} \cup SP_n \qquad (53)$$

Although these ICCs are required for accurate SCED decomposition over the time horizon, the two following challenges must be addressed.

1. How to model ICCs knowing that there are several intertemporal constraints corresponding to system requirements and equipment models.
2. ICCs are obstacles for solving subproblems in a distributed manner independently as $CSP_{T_m} \cap CSP_{T_{m+1}} = ICC_{T_m,T_{m+1}}$.

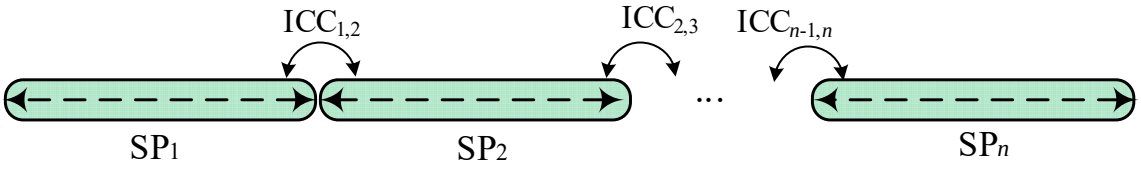


Figure 3.2. Intertemporal Consistency constraints (ICCs) for modeling transition between SPs.

3.5.2. Modeling Interdependencies with Overlapping Intervals

Intertemporal constraints of SCED include generators' ramp limits, (10)-(11), and the state of charge of energy storage, (16) and (34). Modeling these constraints as ICCs in the context of temporal decomposition becomes more challenging taking into account possible corrective actions after the occurrence of a contingency, reserve up/down requirements, wind uncertainties, and demand response. A naïve approach is to start solving the first subproblem and fix the state of the

last interval of a subproblem in the first interval of its next subproblem. SP_1 is solved first and the variable values at the last interval are sent to SP_2. Ramping and storage constraints in SP_2 are formulated using these values as a fixed initial state. This procedure is carried out until SP_n is solved. This process is neither sequential nor iterative. Information exchange flow is from a subproblem to its next subproblems. No feedback is sent from a subproblem to its previous one. Although this makes the solution feasible, the first few subproblems force their desired states to other subproblems as hard constraints that lead to a suboptimal solution for the overall SCED problem.

To ensure both feasibility and optimality of the solution obtained by the temporal decomposition strategy, we propose the concept of *overlapping time intervals* to facilitate modeling ICCs between subproblems. We add dummy intervals to the ending of all subproblems, except for the last SP_n. The dummy time interval are copies of the first intervals of all subproblems, except for SP_1. Let us name subproblems with dummy time intervals as SP^+. Consider Fig. 3 that shows a horizon that is decomposed into three consecutive subproblems SP^+_{m-1}, SP^+_m and SP^+_{m+1}. The intra intervals of SP^+_{m-1}, SP^+_m and SP^+_{m+1} are $\{1, \dots, T_{m-1}\}$, $\{T_{m-1}+2, \dots, T_m\}$, $\{T_m+2, \dots, T_{m+1}\}$, respectively. The overlapping time intervals, indicated by red in Fig. 3, are $T_{m-1}+1$ between SP^+_{m-1}, SP^+_m, and T_m+1 between SP^+_m and SP^+_{m+1}. Since each overlapping time interval is considered in two neighboring subhorizons, internal intertemporal constraints of SP^+_{m-1}, SP^+_m and SP^+_{m+1} will be satisfied locally with respect to constraints of overlapping intervals. Now, subproblems are rewritten as follows:

$$SP^+_{m-1}: \qquad \min\ (2)$$

$$s.t.\,(3)-(38),(48)\&(49)$$

$$\forall t \in \{1,2,\dots,T_{m-1}\} \cup \{T_{m-1}+1\}$$

$$SP_m^+: \qquad \min \ (2)$$

$$s.t. (3) - (38), (48)\&(49)$$

$$\forall t \in \{T_{m-1} + 1\} \cup \{T_{m-1} + 2, \dots, T_m\} \cup \{T_m + 1\}$$

$$SP_{m+1}^+: \qquad \min \ (2)$$

$$s.t. (3) - (38), (48)\&(49)$$

$$\forall t \in \{T_m + 1\} \cup \{T_m + 2, \dots, T_{m+1}\}$$

Variables and constraints of an overlapping time interval appear in two neighboring subproblems. By collecting all elements of subproblems, the intersection of consecutive subproblems, which includes variables, objective terms, and constraints at overlapping intervals, is counted twice. The centralized problem can be recovered by taking the union of SP_{m-1}^+, SP_m^+, and SP_{m+1}^+ and subtracting the intersection of each two consecutive subproblems from it as follows:

$$CP = (SP_{m-1}^+ \cup SP_m^+ \cup SP_{m+1}^+) - (SP_{m-1}^+ \cap SP_m^+) \ - (SP_m^+ \cap SP_{m+1}^+) \qquad (54)$$

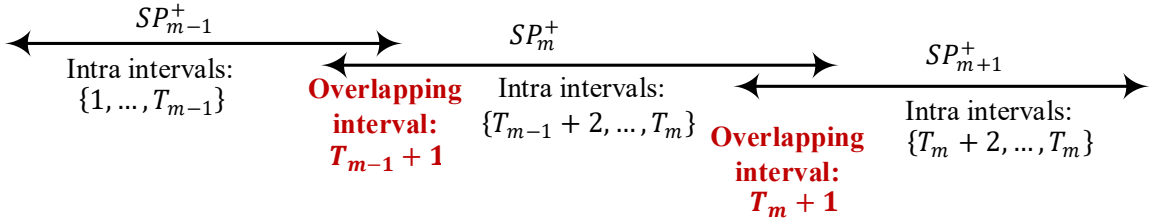


Figure 3.3. Three consecutive subhorizons with overlapping time intervals.

3.5.3. Converting Overlapping Interval into Shared Variables

Control variables at overlapping intervals $T_{m-1} + 1$ and $T_m + 1$ are shared between SP_{m-1}^+ and SP_m^+, SP_m^+ and SP_{m+1}^+, respectively. Each pair of shared variables must be the same to ensure consensus between subproblems and satisfying intertemporal constraints. Control variables and

we use as shared variables are two sets: pre- and post-contingency variables. These sets change depending on the SCED model, whether it is preventive or corrective. Pre-contingency shared variables are power generated by thermal units, reserve up and down of thermal units, storage state of charge, and storage charge and discharge powers. Since corrective actions are considered in the formulated SCED, we model generators' power output and reserve, storage state of charge, and storage charge and discharge powers pertaining to each contingency as post-contingency shared variables. The set of shared variables between SP_{m-1}^{+} and SP_{m}^{+} are

$$\Xi_{m-1,m} = \left\{p_{u,t}, r_{u,t}^{up}, r_{u,t}^{d}, E_{s,t}, p_{ch,s,t}, p_{dc,s,t}\right\} \tag{55}$$

$$\cup \left\{p_{u,t}^{c}, r_{u,t}^{up,c}, r_{u,t}^{d,c}, E_{s,t}^{c}, p_{ch,s,t}^{c}, p_{dc,s,t}^{c}\right\} \quad \forall t = T_{m-1} + 1$$

Shared variables between SP_{m}^{+} and SP_{m+1}^{+}, $\Xi_{m,m_{+1}}$, are the same as (59) but for $t = T_m + 1$.

3.5.4. Consistency Constraints

The shared variables between two consecutive SPs are duplicated to assign a set to each SP. For instance, $\Xi_{m-1,m}$ is duplicated to create Ξ_{m-1} and Ξ_{m}. A set of consistency constraints is formulated and enforced in each SP to ensure that each pair of shared variables reaches the same value. The intertemporal consistency constraint between SP_{m-1}^{+} and SP_{m}^{+} are

$$ICC_{m-1,m} : \Xi_{m-1} - \Xi_{m} = 0 \tag{56}$$

and they are as follows for SP_{m}^{+} and SP_{m+1}^{+}

$$ICC_{m,m+1} : \Xi_{m'} - \Xi_{m+1} = 0 \tag{57}$$

Ξ_{m} and $\Xi_{m'}$ denote the shared variables of subproblem m with its previous and next neighbors, respectively.

3.5.5. Storage Initial State of Charge

In the conventional centralized SCED, the initial state of charge of storage is given. However, in the proposed temporal decomposition, the initial state of charge of storage is known only for

the first SP, not others. This is an obstacle for formulating (16) at the first time interval of subproblems. One possible solution is to solve the first SP, pass the state of charge of storage obtained in its last time interval to the next SP, and use it as the initial state of charge of storage. This, in turn, causes two problems: 1) the subproblems cannot be solved in parallel that increases the solution time, and 2) each SP forces its willingness for the state of charge of storage on its next subproblem that makes the resulted solution suboptimal. To solve this challenge and allow a parallel solution of SPs while satisfying the solution optimality and feasibility, we introduce a new shared variable between each two consecutive SPs. Consider SP_{m-1}^{+} and SP_{m}^{+}. The initial state of charge of storage in SP_{m}^{+} needed for the overlapping interval $T_{m-1}+1$ is the state of charge storage at interval T_{m-1}. We introduce $E_{s,0}$ that is a control variable SP_{m}^{+} allowing formulating (16) for this subproblem's first interval $T_{m-1}+1$. $E_{s,0}$ and its corresponding copy in SP_{m-1}^{+}, which is $E_{s,T_{m-1}}$, are share variables between these two subproblems. $E_{s,T_{m-1}}$ and $E_{s,0}$ are added to Ξ_{m-1} and Ξ_{m}, respectively, and the following intertemporal consistency constraint is included in these two SCED subproblems:

$$E_{s,T_{m-1}} - E_{s,0} = 0 \tag{58}$$

3.6. Coordination Strategy

Although enforcing (50) and (51) satisfies consistency between SPs, they are obstacles for solving SPs in parallel and make their solution suboptimal. We relax these constraints in SPs' objective functions using Augmented Lagrangian Relaxation and apply auxiliary problem principle (APP) to make mismatches between shared variables zero based on the concept of penalizing violations of consistency constraints. APP is an iterative method that is suitable for parallel computing [11]. APP is a message passing-based iterative method that is suitable for parallel computing [11].

3.6.1.Initialization Strategy: APP, like most distributed optimization algorithms, is sensitive to the choice of initial values. We suggest an initialization strategy to enhance the performance of the coordination algorithm. The intertemporal coupling between subproblems are neglected, the overlapping time intervals are ignored, and subproblems are solved independently. The obtained values for variables corresponding to the overlapping time intervals are used for initialization. This enhances the performance of the coordination algorithm significantly as the obtained values for shared variables are, usually, near their optimal values, even if correlations between subproblems are ignored.

3.6.2. APP Implementation: In the context of APP, the SCED subproblem SP_m^+ at iteration k is modified as follows:

$$min \sum_t \sum_u a_u {p_{u,t}}^2 + b_u p_{u,t} + o_{u,t} + b_{ru} r_{u,t}^{up} + \sum_i b_{sh,i,t}\, p_{sh,i,t} \tag{59}$$

$$+\frac{\rho}{2}\|\Xi_m^k - \Xi_m^{*k-1}\|^2 + \gamma \Xi_m^{k\,\dagger}\left(\Xi_m^{*k-1} - \Xi_{m-1}^{*k-1}\right) + \lambda^{(k-1)\,\dagger}\Xi_m^k$$

$$+\frac{\rho}{2}\left\|\Xi_{m'}^k - \Xi_{m'}^{*k-1}\right\|^2 + \gamma \Xi_{m'}^{k\,\dagger}\left(\Xi_{m'}^{*k-1} - \Xi_{m+1}^{*k-1}\right) + \lambda'^{(k-1)\,\dagger}\Xi_{m'}^k,$$

$$s.t. \qquad (3)-(38), (48)\&\ (49)$$

$$\forall t = \{T_{m-1}+1, \dots, T_m+1\}$$

where † is the transpose operator. The asterisk (*) refers to a variable whose value is determined by one of the neighboring subproblems and is kept fixed in subproblem m. SP_{m-1}^+ and SP_{m+1}^+ are formulated analogously with different time subhorizons. Another difference is in the last term of the penalty function that must be $-\lambda^{(k-1)\,\dagger}\Xi_m^k$ for SP_{m-1}^+ and $-\lambda'^{(k-1)\,\dagger}\Xi_{m'}^k$ for SP_{m+1}^+. After each iteration, multipliers λ and λ' are updated as:

$$\lambda^k = \lambda^{k-1} + \omega\left(\Xi_{m-1}^{*k} - \Xi_m^{*k}\right) \tag{60}$$

$$\lambda'^{k} = \lambda'^{k-1} + \omega\left(\Xi_{m+1}^{*k} - \Xi_{m'}^{*k}\right) \quad (61)$$

where ω is a suitable constant (step size). Note that the value of the Lagrange multiplier λ in each iteration corresponds to the cost of maintaining the consistency constraints.

3.6.1. Discussion on Convergence

APP is proven to converge if subproblems are convex or the global optimal solution of each subproblem at each iteration is obtained [11]. The objective function and constraints of the considered SCED problem are convex, except for (17)-(19) and (35)-(37) in which four sets of binary variables are defined for modeling storage charge/discharge status. These three constraints do not add much complexity to the model. Solvers, such as CPLEX and Gurobi, are well advanced and can provide a very high-quality solution for each subproblem. Therefore, at each iteration of APP, almost the global solution of each subproblem is obtained by the solver. Thus, after each iteration of APP, differences between shared variables decrease and the algorithm converges to the optimal point of the whole problem. It is also possible to convexify storage constraints by adding two small positive cost terms for storage charging and discharging (e.g., storage operation & maintenance costs) in subproblems objective functions and dropping binary variables [79]. This technique prevents simultaneous storage charge and discharge while making subproblems convex and ensuring APP convergence.

3.6.2. Guidelines on Number of Subproblems

Increasing the number of subhorizons for a given scheduling horizon results in smaller subproblems that are potentially less computationally expensive. This reduces the solution time of each iteration. However, if we increase the number of subhorizons over a certain limit, the number of shared variables will increase that increases the required number of iterations for the distribution algorithm to converge. Hence, increasing the number of subhorizons is not necessarily efficient

for time saving.

Our observations show that the load pattern has a significant impact on the optimal number of subhorizons. Decomposing the considered horizon from intervals with a low rate of change of load as compared to their neighboring intervals reduces the required number of iterations by the coordination algorithm to converge. Since the pattern of load is predictable in periodic, if an operator finds a good number of subhorizons for a load pattern, this information can be used for similar load patterns. Another factor that should be considered to take advantage of parallel computing is to make subproblems with similar size and computational complexity. This reduces the idle time of computing processors if synchronous coordination strategy, such as APP, is used. The strength of computing processors is another factor that should be considered. Making subhorizons smaller beyond a level may not lead to significant time saving as a computing processor could be strong enough to solve problems with X or 2X sizes within roughly the same time.

In a nutshell, we suggest the following steps to decompose a considered scheduling horizon. 1) Estimate processors' computation time by reducing the size of the SCED problem. 2) Determine the size (assume it is X) beyond which the time reduction is not significant (steps 1 and 2 should be performed once, not once per load pattern). 3) Divide the size (assume it is NX) of the SCED problem into X to determine the number subhorizons (N). 4) Determine the lowest rate of change of load between consecutive intervals. 5) Decompose the scheduling horizon into N subhorizon from intervals with the lowest rate of change of load so that the size of subhorizons is similar.

3.7. Case Study

The proposed algorithm is applied to solve a week-ahead SCED problem for a six-bus system, the IEEE 24-bus system, and a 472-bus system. System and equipment data are given in [103].

Simulations are carried out on Matlab using YALMIP [70] as modeling software and Gurobi on a 3.7 GHz PC with 16GB of RAM. We have used one computing processor and have solved subproblems on this processor sequentially. To mimic parallel computing, when all subproblems are solved at each iteration of APP, the longest solution time is assumed as the runtime of that iteration. Upon convergence of the distributed algorithm, the runtimes of all iterations are summed up to determine the overall solution time.

3.7.1. Six-Bus Test System

This case serves as a tutorial for the proposed algorithm. The system includes six buses, three generating units, seven lines, three load points, one storage device, and a wind farm.

Validation of Nonparametric Chance Constraints for Reserve Procurement: The confidence level α is set to 0.95 for all intervals. To study the effectiveness of the proposed data-driven nonparametric chance constraints modeling with a reference point, we consider that the wind power follows a Gaussian PDF and generate 100 samples for each interval. The wind generation mean values over the scheduling horizon are given in [36, 80] and the standard deviation is assumed to be 20%. Assuming that we do not know that wind generation follows Gaussian distribution, AKDE is applied to find nonparametric wind generation PDFs from the dataset. The point-wise error is calculated, and the adjusted risk levels of chance constraints are computed. For time interval 15, for instance, the squared pointwise error is 0.01 and according to (43), d =1e-04. Plugging d in (44) yields $\alpha_{15}'^{+} = 0.048$. The resulted $CDF^{-1}{}_{\tilde{P}_{w,t}}(1-\alpha'^{+})$ and $CDF^{-1}{}_{\tilde{P}_{w,t}}(\alpha'^{+})$ are 24.3376 and 23.5679, respectively. These values are used to formulate chance constraints and the centralized SCED is solved. The operation cost is \$391,225. As we assumed Gaussian PDFs for wind power, we can formulate classical parametric chance constraints and obtain the benchmark operation costs if complete information of wind power distributions is

known. Table 3.1 shows that the relative error between the obtained costs is 3e-05.

We have also assumed that wind generation follows Gamma distribution. The same procedure as of that for Gaussian distribution is implemented, and the results are depicted in Table 3.2. The relative error is 3e-5. The small relative errors reported in Tables 3.1 and 3.2 show that the proposed nonparametric approach can be adopted to formulate a data-driven SCED problem if only historical data is available and no information is known about the type of wind generation PDFs.

Table 3.1. Comparison between Parametric and Nonparametric Chance Constraints using Gaussian Distribution as Benchmark

Method	SCED cost ($)	Relative error
Parametric (benchmark)	$391,213	-
Nonparametric	$391,225	3e-5

Table 3.2. Comparison between Parametric and Nonparametric Chance Constraints using Gamma Distribution

Method	SCED cost ($)	Relative error
Parametric (benchmark)	399,054	-
Nonparametric	399,068	3e-5

Sensitivity to Confidence Level: We have used three different values for α to study the performance of nonparametric models under different risk levels. It is assumed that wind power follows Gaussian distribution and 100 samples are generated for each interval. Table 3.3 shows operation costs obtained by parametric and nonparametric models. For all three cases, the relative error between the nonparametric model and benchmark results is acceptable, which shows that the proposed nonparametric chance-constrained programming works well under different risk levels.

Table 3.3. Operation Costs Under Different Risk Levels

α	Parametric (benchmark)	Nonparametric	Relative error
0.1	\$391,116	\$391,124	2e-5
0.05	\$391,213	\$391,225	3e-5
0.01	\$391,496	\$391,514	5e-5

Distributed SCED Analysis: The operation horizon is decomposed into seven same sized subhorizons, each including 24 intervals. Line outage contingencies are considered. Adding the overlapping intervals results in a SCED subproblem with 25 time intervals for each subhorizon. There are 26 shared variables between every two consecutive SCED subproblems. The suggested initialization strategy is applied and its results are used to initialize APP. The distributed algorithm converges after ten iterations plus one initialization step. Fig. 3.4 shows the differences between shared variables over the course of iterations. Mismatches go to zero as more iterations are carried out. The majority of pairs of shared variables reach the consensus after six iterations. However, mismatches of Ξ_6 and Ξ_7 become less than the stopping threshold after ten iterations. As an example, Fig. 3.5 shows $p^c_{dc,1,25}$, storage discharge power after contingency one, from the perspective of SP_1 and $p^c_{dc,1,1}$ obtained by SP_2. These two variables reach the same value after ten iterations.

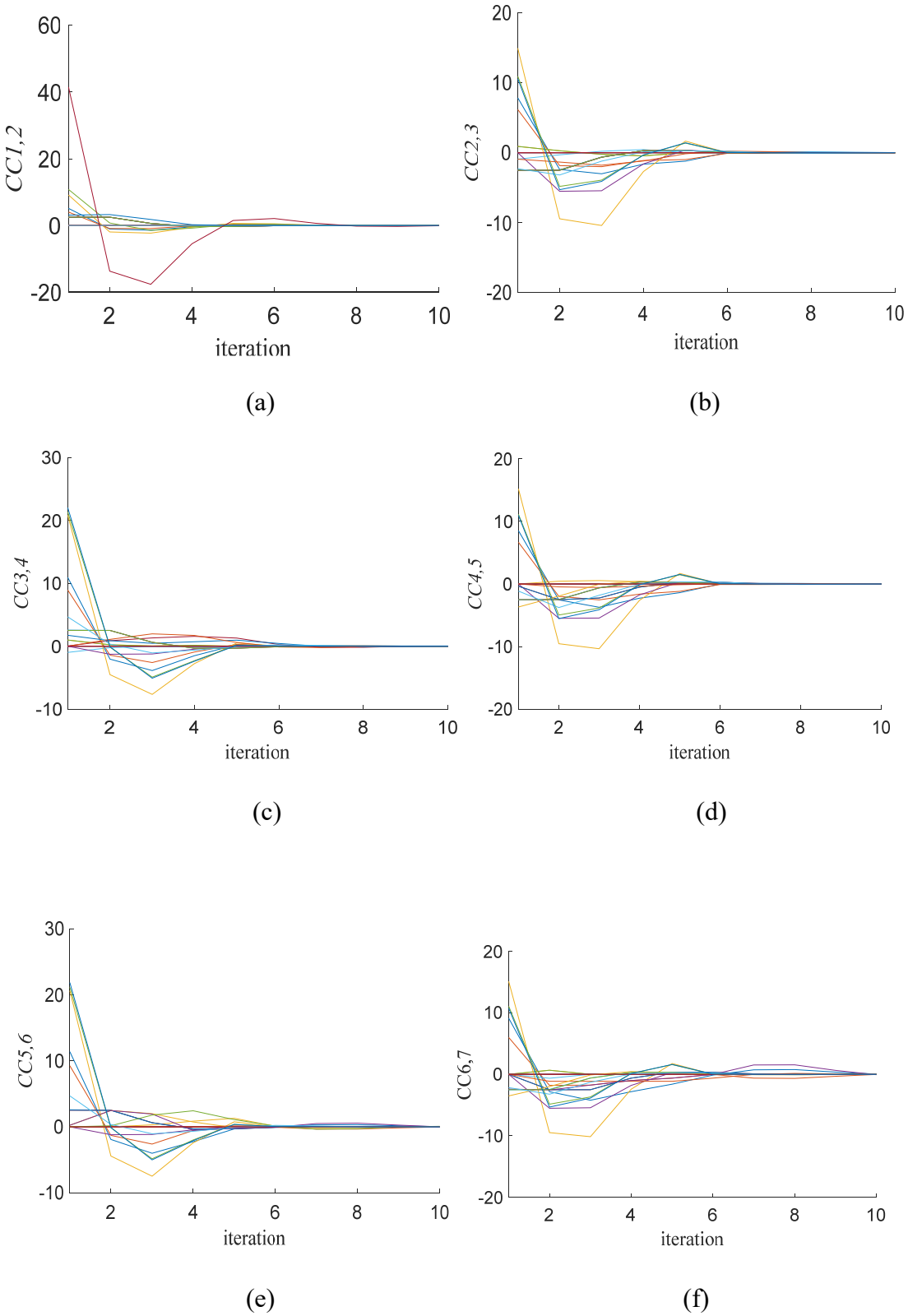


Figure 3.4. Six-bus system: mismatches of consistency constraints between (a) SP_1 and SP_2, b) SP_2 and SP_3, c) SP_3 and SP_4, d) SP_4 and SP_5, e) SP_5 and SP_6, and f) SP_6 and SP_7.

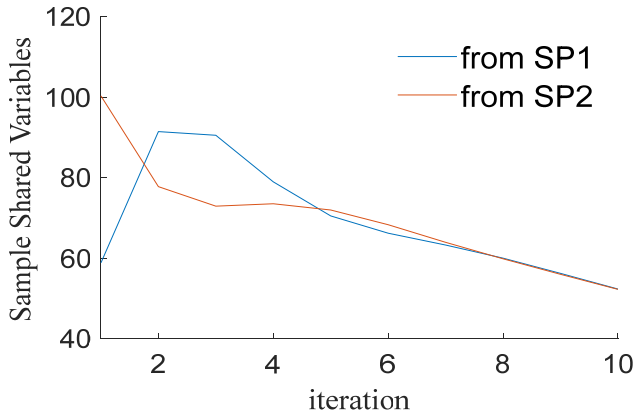


Figure 3.5. Storage discharge power (MW) after contingency one.

Comparison with Centralized SCED: We have compared the values of decision variables obtained by the distributed and centralized SCED approaches. For instance, Figs. 6 (a) and (b) show generated powers by units one and two in the first subhorizon, and Figs. 6 (c) and (d) depict the storage discharge power and energy level in subhorizon seven. The optimal values obtained by the two approaches are almost the same. In addition, a convergence index is defined to measure the relative error between the operation costs determined by the distributed SCED (f^d) and the centralized SCED (f^{cp}), which is considered as benchmark results. The closer the convergence measure becomes to zero, the more precise solution is obtained.

$$rel = \frac{|f^{cp} - f^d|}{f^{cp}} \tag{62}$$

The convergence measure rel is depicted in Fig. 7. This index is 9e-05 upon convergence, which shows the accuracy of the proposed distributed SCED algorithm.

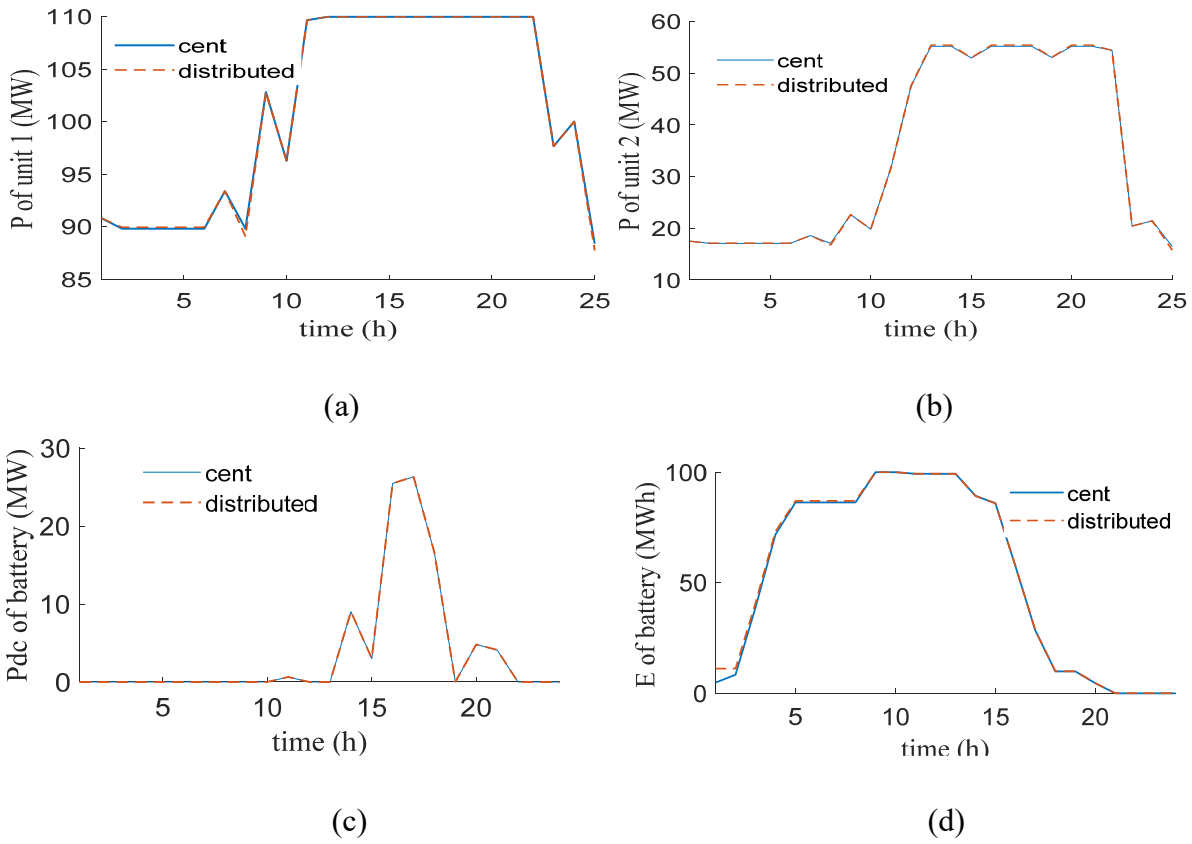


Figure 3.6. The optimal values of a) $p_{1,t}$, b) $p_{2,t}$ in subhorizon one, and c) $p_{dc,t}$, and d) E_t subhorizon seven.

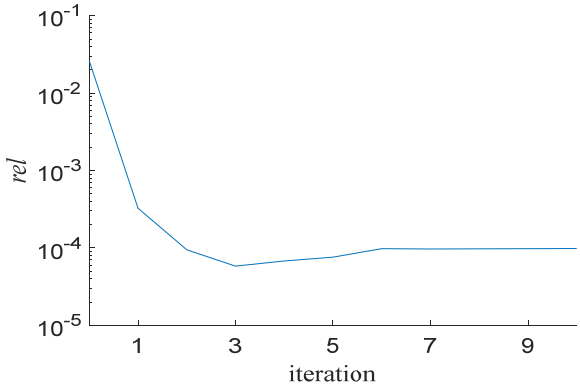


Figure 3.7. The relative error for the six-bus system.

Storage Initial State of Charge Modeling: Ignoring the initial state of charge consistency constraint (58), the distributed SCED is solved. The obtained solution is infeasible. For instance, the state of charge of storage at the last interval of SP_3, determined by SP_3, is 19 MWh. However, the initial state of charge of storage at the first interval of SP_4, determined by SP_4, is 100 MWh.

Since these two values refer to the same physical variables, the obtained solution is infeasible. Incorporating (58) makes the solution feasible from the perspective of the whole scheduling horizon.

3.7.2. IEEE 24-Bus System

We have studied the impact of increasing the number of subhorizons on the overall solution time using the IEEE 24-bus system. We have increased the number of subproblems up to 168. Fig. 8 shows the solution time versus the number of subproblems. Increasing the number of subproblems up to 24 results in the overall solution time reduction (although some jumps are observed, the general pattern is decreasing). While increasing the number of subproblems from 24 to 84 does not have a considerable impact on the solution time, increasing the number of subproblems beyond 84 increases the solution time. Such a curve with this trend is reported in parallel computing literature.

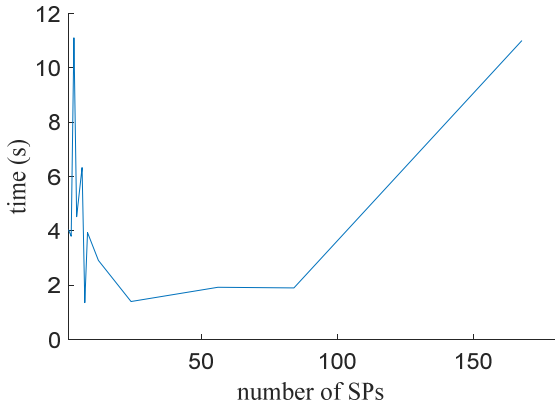


Figure 3.8. Overall solution time versus number of subproblems.

3.7.3. 472-Bus Test System

This system, which is created by connecting four IEEE 118-bus systems, has 216 generators, 784 lines, 364 load points, one storage device, and a wind farm. The confidence levels of chance constraints are set to 0.95. The size of the centralized problem is relatively large. As illustrated in

Table 3.4, the solver takes 654 seconds to solve the centralized SCED and return the optimal operation cost of $41,281,398.

We partition the overall time horizon into seven subhorizons and implement the proposed algorithm. We have selected three values for step size ω for updating Lagrange multipliers. Increasing the value of ω speeds up the solution procedure. However, it affects the relative error. In general, the larger ω is, the faster the algorithm would converge and the larger the rel index would be. While $\omega = 1$ yields the least solution time, 312 seconds, $\omega = 0.05$ results in the least rel index, almost zero. Comparing the solution time and rel, we select $\omega = 0.05$. The distributed algorithm converges after 11 iterations within 396 seconds. The proposed distributed SCED algorithm is 40% faster than the centralized SCED, while the operation costs obtained by both algorithms are almost the same.

Table 3.4. Operation Costs Under Different APP Multiplier ω

Algorithm	ω	Cost ($)	rel	Time (s)	Time saving
Centralized	-	41,281,398	-	654	-
Distributed	1	41,283,289	4e-5	312	53%
	0.2	41,281,975	1e-5	346	47%
	0.05	41,281,401	2e-8	396	40%

3.7.4. Hybrid Geographical and Temporal Decomposition

In this case, we demonstrate that temporal decomposition can be combined with geographical decomposition to enhance the overall performance of distributed SCED. A 944-bus test system, whose information is given in [31, 81], is used. The considered scheduling horizon includes 168 intervals. For the sake of explanation and simplicity, we have ignored energy storage, wind generation, load shedding, and contingency. As shown in Table 3.5, the centralized approach takes

46 seconds to solve the problem, and the total cost is $88,749,812. The system is decomposed into two regions, and geographical decomposition is applied [82]. The distributed SCED converges after 31 seconds with $rel \approx 0$. The proposed temporal decmposition is applied. Each subhorizon includes 24 intervals. The algorithm takes 25 seconds to converge to $rel \approx 0$.

We have combined geographical and temporal decomposition strategies to solve SCED in a distributed manner. This hybrid approach has two loops, an inner loop and an outer loop. In the outer loop, subproblems obtained by geographical decomposition are solved, and the corresponding Lagrange multipliers are updated iteratively using APP. Whereas in the inner loop, each subproblem of geographical decomposition is further decomposed over the time horizon, and APP is applied to coordinate subproblems. To have a fair comparison, similar to solely geographical decomposition and solely temporal decomposition, we have considered two zones and seven subhorizons. The hybrid decomposition strategy outperforms both geographical and temporal decompositions and converges within ten seconds with $rel \approx 0$. This shows that the temporal decomposition can be combined with geographical decomposition to enhance the performance of the distributed SCED.

Table 3.5. Comparison between Different Approaches to Solve SCED for the 994-bus System

Method	Time (sec)	rel
Centralized (benchmark)	46	-
Geographical decomposition	31	≈ 0
Temporal decomposition	25	≈ 0
Hybrid geographical + Temporal	10	≈ 0

3.8. Conclusion

A temporal decomposition approach is proposed to decompose the SCED problem over the

scheduling horizon with the goal of reducing the computational burden of the optimization problem. Two data-driven nonparametric chance constraints are formulated for reserve up and down requirements considering wind generation uncertainty. These constraints are replaced by their equivalent parametric constraints to make SCED solvable by standard solvers. Generators' ramp limits and the state of charge of storage for the transition between subhorizons are modeled by introducing overlapping intervals and 12 sets of shared variables. It is discussed that the initial state of charge of storage at the beginning of each subproblem must be the same as the state of charge of storage at the interval before the overlapping interval; otherwise, the solution will be infeasible. An initialization strategy is suggested to enhance the performance of the distributed coordination algorithm.

A tutorial is presented based on a small system, the impact of number of subhorizons on the algorithm is studied using the IEEE 24-bus system, and the results for a 472-bus system show that the SCED solution time is reduced by a factor of 1.65 as compared to that of the centralized SCED. As the size of the optimization problem increases, the effectiveness of the proposed method is more considerable. In addition, it is illustrated that data-driven nonparametric chance constraints provide a solution close to the benchmark results obtained using the complete information of wind power generation probability distributions.

A direction of research is to combine reserve up (or down) requirements at all intervals in a probabilistic constraint and develop a nonparametric joint chance constraint model. Another possible research direction is to develop methods to find an optimal time decomposition strategy and to improve the convergence performance of the coordination algorithm by using momentum and second-order derivative information for Lagrange multipliers updating.

CHAPTER 4
TEMPORAL DECOMPOSITION AND COORDINATION FOR SECURITY- CONSTRAINED UNIT COMMITMENT

4.1. Introduction

In the previous chapters, the problem was a convex optimization and no binary or integer variables were included in the problem. However, this chapter deals with complexity that arises when integer variables and constraints are included in the distributed optimization problem. The main challenge is to model minimum on/off time constraints between subproblems. In fact, the novelty of this chapter is twofold. The scheduling horizon is decomposed into multiple subhorizons. The concept of coupling intervals is introduced, and a set of nonlinear constraints and logical expressions along with their equivalent linear models are formulated to handle intertemporal ramp constraints and minimum on/off times between consecutive subhorizons. An accelerated analytical target cascading (A-ATC) algorithm is developed to coordinate SCUC subproblems and find the optimal solution for the whole operation horizon in a distributed manner. An initialization strategy is presented to enhance the convergence performance of A-ATC. The proposed algorithm is tested on four test systems.

4.2. Symbols

A. Indices and Sets:

t	Index for time intervals.
u	Index for units.
c	Index for contingencies.
k	Index for iterations.
t_o	Index for coupling time intervals.
ρ	Penalty factor.
T_u^{on}, T_u^{off}	Minimum on/off time of unit u.
s	Index for subhorizxon (subproblem) s.
s_-	Subhorizon (subproblem) before subhorizon s.
s_+	Subhorizon (subproblem) after subhorizon s.

T_u^{on}, T_u^{off}	Minimum on/off time of unit u.
f(·)	Generation cost function.

B. Variables:

$x_{s\_}$	Set of variables in time intervals belonging to subhorizon $s\_$.
$\chi_{s\_,s}^k$	Target shared variables between subhorizons $s\_$ and s at iteration k.
$r_{s\_,s}^k$	Responce shared variables between subhorizons $s\_$ and s calculated by subproblem $s\_$ at iteration k.
p_{ut}	Power generated by unit u at interval t.
$p_{ut_o(s,s\_)}$	Power generated by unit u at coupling time interval t_o between subhorizons $s\_$ and s determined by subproblem s and sent to subproblem $s\_$.
I_{ut}	On/off status of unit u at interval t.
$I_{ut_o(s,s\_)}$	On/off status of unit u at coupling time interval t_o between subhorizons $s\_$ and s determined by subproblem s and sent to subproblem $s\_$.
λ	Vector of Lagrange multipliers.
y_{ut}	Startup indicator of unit u at time t.
z_{ut}	Shutdown indicator of unit u at time t.

C. Auxiliary Counting Variables and Functions:

$\mathcal{F}_{us}^{sd}$	First shutdown time of SP_s.
$\mathcal{F}_{us}^{su}$	First startup time of SP_s.
h_{us-}^{on}	An auxiliary variable to calculate $h_{u(s\_,s)}^{on}$.
h_{us-}^{off}	An auxiliary variable to calculate $h_{u(s\_,s)}^{off}$.
$h_{u(s\_,s)}^{on}$	Number of hours that $SP_{s\_}$ asks SP_s to keep unit u on.
$h_{u(s,s\_)}^{on}$	Number of hours that SP_s asks $SP_{s\_}$ to keep unit u on.$h_{u(s\_,s)}^{off}$
$h_{u(s,s\_)}^{off}$	Number of hours that SP_s asks $SP_{s\_}$ to keep unit u off.
H_u^{on}	An auxiliary variable to linearize max of Φ_{ut}^{on}.
ϕ_{us}^{on}	A vector whose tth element is nonzero if the status of unit u has changed in time interval t from off to on.
ϕ_{us}^{off}	A vector whose tth element is nonzero if the status of unit u has changed in time interval t from on to off.

4.3. Contributions

The main contributions of this chapter are summarized as follows:

- A novel temporal decomposition strategy is proposed to decompose the SCUC problem into multiple time-interdependent SCUC subproblems.

- Modeling approaches are developed to handle generating units' ramp and minimum on/off restrictions for the transition between subhorizons.
- Based on the concept of Nesterov momentum for gradient descent methods, an *accelerated* ATC is presented to coordinate SCUC subproblems distributedly.
- An initialization strategy is presented to enhance the performance of the proposed distributed SCUC algorithm.

4.4. Proposed Time Decomposition Strategy

The main philosophy of the proposed algorithm is based on Augmented Lagrangian relaxation [7, 66, 83, 84]. To take advantage of distributed computing, we divide the scheduling horizon into several smaller, consecutive subhorizons. An SCUC subproblem is formulated for each subhorizon. SCUC subproblems are formulated exactly the same as the centralized SCUC for the whole scheduling horizon, with no need for modifying the intertemporal and non-intertemporal constraints. Intertemporal interdependencies between every two consecutive subproblems are converted into a set of shared variables. Consistency constraints are formed to ensure that each pair of shared variables reaches the same values from the perspective of the two subproblems. The concept of augmented Lagrangian relaxation is used to penalize the consistency constraints into local objective functions and make mismatches between each pair of the shared variables zero iteratively.

4.4.1. SCUC Formulations and Notations

For brevity, we show compact forms of SCUC subproblems. We have adopted the SCUC formulation presented in Chapters 3 and 4 of [5] except for modeling startup and shutdown indices and minimum on/off time constraints that are adopted from [85]. s_-, s, and s_+ refer to three consecutive subhorizons. We use subscript (s_-, s) to distinguish coupling variables for modeling

transition between subproblems $s\_$ and s. A coupling variable indicated by subscript $(\cdot,\cdot)$ is a decision variable determined by the left side subproblem. For instance, $m_{(s\_,s)}$ are coupling variables m between subproblems $s\_$ and s that are among decision variables in subproblem $s\_$.

4.4.2. Decomposing SCUC

Without loss of generality, we derive equations for two consecutive subhorizons $s\_$ and s and decompose the SCUC problem into SCUC subproblems (1) and (2). Assume that the scheduling time horizon is T. We assign intervals one to $n\_$ to the SCUC subproblem (SP) of subhorizon $s\_$ (named $SP_{s\_}$).

$$x_{s\_} = \text{argmin} \sum_{t=1}^{n\_} \sum_{u} f(p_{ut}, I_{ut}) \qquad (1a)$$

$$s.t. \qquad h_{s\_}(x_{s\_}) = 0 \quad \& \quad g_{s\_}(x_{s\_}) \leq 0 \qquad (1b)$$

$$\text{with } t = \{1, \dots, n\_\}$$

where $x_{s\_}$ refer to the sets of variables of $SP_{s\_}$ (corresponding to intervals one to $n\_$). $h_{s\_}$ includes power balance constraints under normal and contingency conditions at intervals $\{1, \dots, n\_\}$ that belong to $SP_{s\_}$, and $g_{s\_}$ refers to generations upper and lower bounds, line flow limits, units ramping up/down restrictions, minimum on/off time restrictions under normal and contingency conditions, and expressions coupling power generated by units before and after contingencies. We also assign intervals $n\_ + 1$ to T to SP_s.

$$x_s = \text{argmin} \sum_{t=n\_+1}^{n} \sum_{u} f(p_{ut}, I_{ut}) \qquad (2a)$$

$$s.t. \qquad h_s(x_s) = 0 \quad \& \quad g_s(x_s) \leq 0 \qquad (2b)$$

$$\text{with } t = \{n\_ + 1, \dots, n\} z$$

x_s refer to the sets of variables of SP_s (corresponding to intervals $n\_ + 1$ to T). Subproblems (1)

and (2) are smaller than the original centralized SCUC and hence are computationally less expensive. However, the solution to (1) and (2) may be physically infeasible as interdependencies, i.e., temporal constraints, between subhorizons s_- and s are ignored. We propose the concept of coupling time intervals and present several constraints and logical expressions to model temporal constraints between subproblems.

4.4.3. Modeling Ramp Limits with Coupling Time Intervals

Generation ramp up and down constraints pertaining to power generated in the last interval of subhorizon s_- ($t = n_-$) and the first interval of subhorizon s ($t = n_- + 1$) interconnect decisions made in these subhorizons. Decomposing the centralized SCUC into (1) and (2), as shown in Fig. 1a, does not take into account these ramp constraints at boundary intervals. To enable modeling the boundary ramp constraints, we represent the whole operation horizon as depicted in Fig. 2a. Without loss of generality and to better explain decomposition with a middle SP, we divide the scheduling horizon it into three subhorizons s_-, s, and s_+.

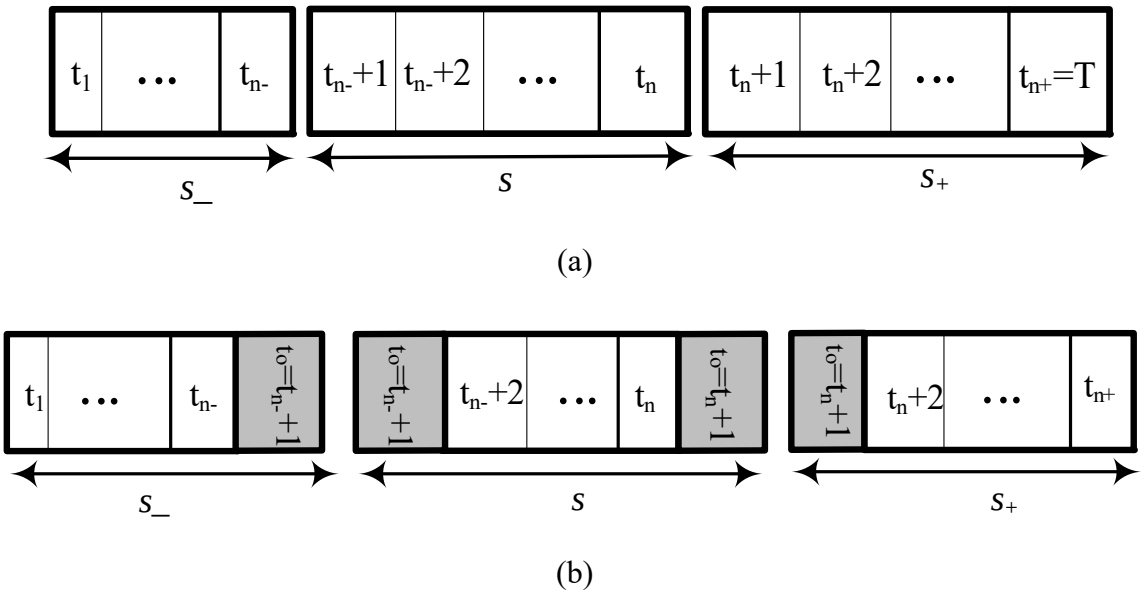


Figure 4.1. a) Three consecutive subhorizons and b) subhorizons decomposition with coupling intervals.

As shown in Fig. 1b, the first interval of each subhorizon, except for subhorizon one, is introduced as a coupling time interval, indicated by t_o, between two consecutive subhorizons. We duplicated each coupling interval and assign each copy to a subhorizon to separate the subhorizons. Each copy contains variables and constraints of the corresponding coupling interval. The SP (1) for subhorizon $s\_$ is now reformulated as:

$$x_{s\_} = \text{argmin} \sum_{t=1}^{n\_+1} \sum_{u} f(p_{ut}, I_{ut}) \qquad (3a)$$

$$s.t. \qquad h_{s\_}(x_{s\_}) = 0 \;\; \& \;\; g_{s\_}(x_{s\_}) \leq 0 \qquad (3b)$$

$$\text{with } t = \{1, \dots, n\_, \underbrace{n\_ + 1}_{\text{coupling interval}} \}$$

where $x_{s\_}$ contains variables corresponding to intervals one to $n\_$ as well as interval $n\_ + 1$ that is interval one of subhorizon s. The SP (2) for subhorizon s is reformulated by (4). Since subhorizon s has two neighboring subhorizons, it contains two coupling intervals, i.e., $t = n\_ + 1$ between SP_s and $SP_{s\_}$ and $t = n + 1$ between SP_s and SP_{s_+}.

$$x_s = \text{argmin} \sum_{t=n\_+1}^{n} \sum_{u} f(p_{ut}, I_{ut}) \qquad (4a)$$

$$s.t. \qquad h_s(x_s) = 0 \quad \& \quad g_s(x_s) \leq 0 \qquad (4b)$$

$$\text{with } t = \{ \underbrace{n\_ + 1}_{\text{coupling interval}}, n\_ + 2, \dots, n, \underbrace{n + 1}_{\text{coupling interval}} \}$$

This makes $SP_{s\_}$ and SP_s non-separable as power produced by units at coupling intervals $n\_ + 1$ appear in both subproblems. Similarly, $p_{u,n+1}$ makes SP_s and SP_{s_+} non-separable. For brevity of notations, we name the coupling intervals $n\_ + 1$ and $n + 1$ as t_o. To allow the separate solution of subproblems, we duplicate these coupling variables to create new sets of coupling variables.

For example, $p_{ut_o(s_-,s)}$ is a coupling variable between SP_{s_-} and SP_s that is handled by SP_{s_-}, and $p_{ut_o(s,s_-)}$ is a coupling variable between SP_{s_-} and SP_s being handled by SP_s. Introducing these variables makes SP_{s_-}, SP_s, and SP_{s_+} separable. To make the SCUC solutions feasible from the perspective of the whole operating horizon, we convert each pair of coupling variables into a consistency constraint as

$$CC_{s_-,s}: p_{ut_o(s_-,s)} - p_{ut_o(s,s_-)} = 0 \qquad \forall u, t_o = n_- + 1 \qquad (5)$$

$$CC_{s,s_+}: p_{ut_o(s,s_+)} - p_{ut_o(s_+,s)} = 0 \qquad \forall u, t_o = n + 1 \qquad (6)$$

and enforce (5) in SP_{s_-} and SP_s and (6) in SP_s and SP_{s_+}.

4.4.4. Minimum On/Off Time Limits

The concept of coupling intervals is introduced for modeling ramping between subproblems; however, they cannot solve challenges for decomposing minimum on/off time constraints between subproblems. These constraints must be decomposed and modeled appropriately to ensure feasibility of the proposed time decomposition strategy's solution. Modeling boundary minimum on/off time constraints is more complex than that of ramping as the number of intervals that connect neighboring subhorizons depends on minimum on/off time that is different for different units. That is, two consecutive subhorizons are coupled through several ending/beginning coupling intervals at the boundary of subhorizons.

The main idea for solving this challenge is to count the number of on/off times at boundary intervals between consecutive subproblems and try to reach a consistency between these numbers iteratively by penalizing their mismatches in local objective functions. As depicted in Fig. 2, we introduce two sets of new variables that count the number of on/off times at boundary intervals in each SP. These variables are shared between subproblems to measure inconsistencies between

minimum on/off time constraints of an SP with its neighbors. Note that these variables are only counters and have no impact on the form of minimum on/off time constraints. To formulate each SP, the minimum on/off time constraints for all time intervals, including coupling intervals, are formulated using models developed in [85], with no modifications. For $SP_{s\_}$, for instance, we have:

1. $h^{on}_{u(s\_,s)}$: Remaining minimum on time of unit u in $SP_{s\_}$ that should be satisfied in SP_s.
2. $h^{off}_{u(s\_,s)}$: Remaining minimum off time of unit u in $SP_{s\_}$ that should be satisfied in SP_s.

We define several functions to model these counting shared variables.

Minimum on time counter for $SP_{s\_}$: Using startup indicator y [85], in $SP_{s\_}$, we define a vector $\Phi^{on}_{us\_}$ whose tth element is equal to the time interval index t if the status of unit u has changed from off to on in that interval, otherwise zero.

$$\Phi^{on}_{uts\_} = t \times y_{ut} \qquad \forall u, \forall t \qquad (7)$$

The last interval in which an off to on status change occurs for unit u is equal to the maximum value of the vector $\Phi^{on}_{us\_}$. Hence, the number of intervals that unit u could have been on in final intervals of $SP_{s\_}$, i.e., $h^{on}_{us\_}$, is:

$$h^{on}_{us\_} = n\_ - \max\left(\Phi^{on}_{us\_}\right) + 1 \qquad \forall u \qquad (8)$$

For instance, for a subhorizon of 24 intervals, if the last interval in which unit u is turned on at interval 22, $\max\left(\Phi^{on}_{us\_}\right) = 22$ and expression (8) returns $h^{on}_{us\_} = 24 - 22 + 1 = 3$.

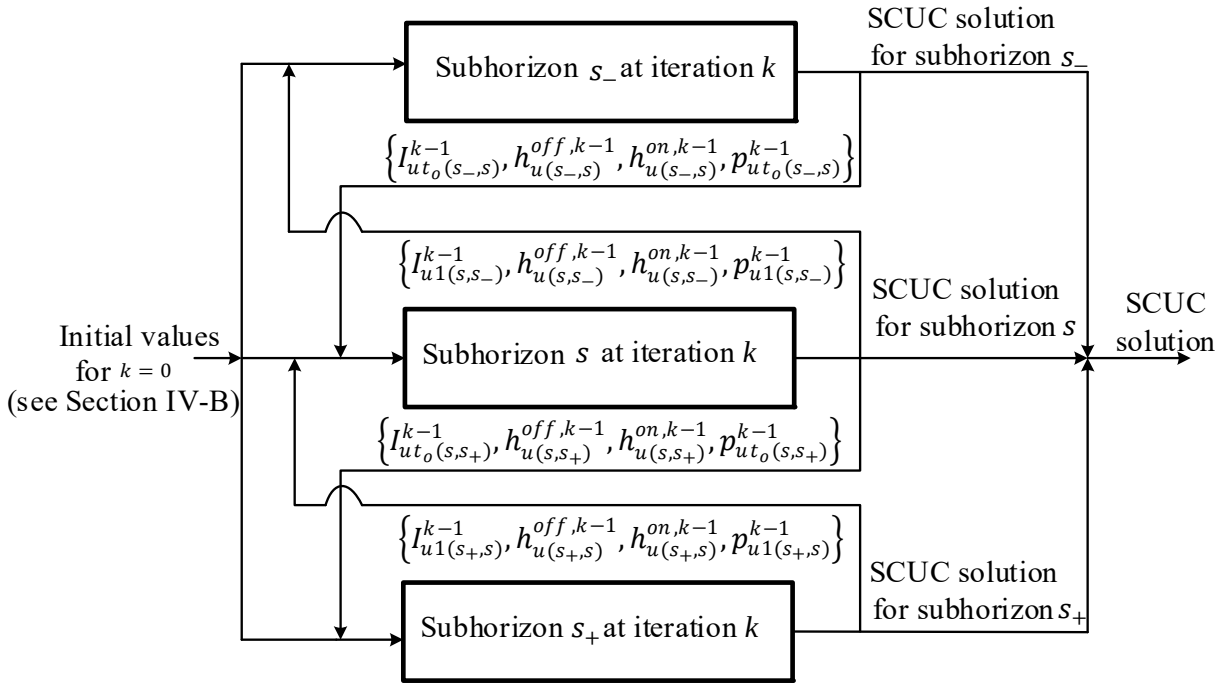


Figure 4.2. Block diagram of information sharing for coordinating subproblems.

Therefore, the number of intervals that $SP_{s\_}$ wants SP_s to keep unit u on is:

$$h^{on}_{u(s\_,s)} = \max\left(0, T^{on}_u - h^{on}_{us\_}\right) \qquad \forall u \qquad (9)$$

If $h^{on}_{us\_}$ is smaller than the minimum on time, $T^{on}_u - h^{on}_{us\_}$ is positive and $SP_{s\_}$ wants SP_s to keep unit u on for at least $T^{on}_u - h^{on}_{us\_}$ intervals at the beginning of its subhorizon; otherwise $SP_{s\_}$ sends zero to SP_s. For instance, for a subhorizon of 24 intervals, if $h^{on}_{us\_} = 3$ and $T^{on}_u = 5$, then $h^{on}_{u(s\_,s)} = 2$; and if $h^{on}_{us\_} = 3$ and $T^{on}_u = 2$, then $h^{on}_{u(s\_,s)} = 0$. Also, if $T^{on}_u = 2$, unit u is turned on at hour 20 and off at hour 23, and was kept off for the rest of hours in $SP_{s\_}$, (7) produces 20, (8) gives 5, and (9) returns 0.

Minimum on time counter for SP_s: For SP_s, using shutdown indicator z [85], we define a vector Φ^{off}_{us} whose tth element is equal to the time interval index t if the status of unit u has changed from on to off in that interval, otherwise zero.

$$\Phi^{off}_{uts} = t \times z_{ut} \qquad \forall u, \forall t \qquad (10)$$

The first shutdown time of SP_s is then the minimum nonzero element of Φ_{us}^{off}.

$$\mathcal{F}_{us}^{sd} = \text{minnonzero}\left(\Phi_{us}^{off}\right) \qquad \forall u \qquad (11)$$

Three possible situations may happen:

1. $h_{u(s\_,s)}^{on}$ is less than minimum on time, and it is also more beneficial for SP_s to keep unit u on for at least $h_{u(s\_,s)}^{on}$ hours. In this situation, coupling variables between the two subproblems are the same.
2. $h_{u(s\_,s)}^{on}$ is less than minimum on time but SP_s prefers to keep unit u on for less than $h_{u(s\_,s)}^{on}$ hours. In this situation, coupling variables are different and subproblems must come to a trade-off about the number of on hours.
3. $h_{u(s\_,s)}^{on}$ is larger than or equal to minimum on time. Therefore, $SP_{s\_}$ does not send any request to SP_s.

To cover these possibilities, we determine $h_{u(s,s\_)}^{on}$ that must take the minimum of $h_{u(s\_,s)}^{on}$, which is given from $SP_{s\_}$, and the number of on intervals at the beginning of subhorizon s.

$$h_{u(s,s\_)}^{on} = \min\left(h_{u(s\_,s)}^{on}, \mathcal{F}_{us}^{sd} - 1, M \times I_{ut_o(s,s\_)}\right) \quad \forall u \quad (12)$$

If unit u is on at the first interval, $\mathcal{F}_{us}^{sd} - 1$ returns the number of hours that is optimal for SP_s to keep unit u on. The number of initial on hours of SP_s is counted by the first hour of this subproblem at which the first shutdown happens minus one (i.e., $\mathcal{F}_{us}^{sd} - 1$). In addition, the term $M \times I_{ut_o(s,s\_)}$, in which a big-M is used, models if unit u is off at the first interval.

Minimum off time counter: An analogous strategy as of that for minimum on time is used to model minimum off time.

In $SP_{s\_}$:

$$h_{us\_}^{off} = n\_ - \max\left(\Phi_{us\_}^{off}\right) + 1 \qquad \forall u \qquad (13)$$

$$h_{u(s_-,s)}^{off} = \max\left(0, T_u^{off} - h_{us_-}^{off}\right) \qquad \forall u \qquad (14)$$

In SP_s:

$$\mathcal{F}_{us}^{su} = \text{minnonzero}(\Phi_{us}^{on}) \qquad \forall u \qquad (15)$$

$$h_{u(s,s_-)}^{off} = \min\left(h_{u(s_-,s)}^{off}, \mathcal{F}_{us}^{su} - 1, M \times \left(1 - I_{ut_o(s,s_-)}\right)\right) \forall u (16)$$

Similar expressions can be used to model the counting variables between SP_s and SP_{s_+} and any two consecutive subproblems. The combination of (7)-(16) provides accurate counting variables for all possible on/off situations that might happen for the transition between subproblems.

Illustrative example: Consider the boundary intervals shown in Fig. 3. Assume that $T_u^{on} = 3$, $T_u^{off} = 1$, and the length of each SP is 24. We have $h_{u1}^{on} = 24 - 23 + 1 = 2$, $h_{u(1,2)}^{on} = \max(0, 3 - 2) = 1$, $\mathcal{F}_{u2}^{sd} = 6$, $h_{u(2,1)}^{on} = \min(1, 6 - 1, M \times 0) = 0$, $h_{u1}^{off} = 24 - 21 + 1 = 4$, $h_{u(1,2)}^{off} = \max(0, 1 - 4) = 0$, $\mathcal{F}_{u1}^{su} = 3$, and $h_{u(2,1)}^{off} = \min(0, 3 - 1, M \times 1) = 0$. Note that this example is not feasible. A detailed tutorial for minimum on/off modeling is provided in Section V.A.

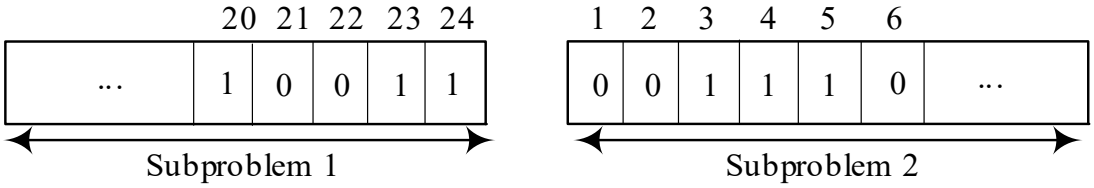


Figure 4.3. An illustrative example for on/off status at boundary intervals.

Consistency constraints: $h_{u(s_-,s)}^{on}$ and $h_{u(s,s_-)}^{on}$ must reach the same value to have consistency between the on-counting variables in SP_{s_-} and SP_s. $h_{u(s_-,s)}^{off}$ and $h_{u(s,s_-)}^{off}$ must also reach the same values for consistency between the off-counting variables. Otherwise, the obtained solution is not feasible from the perspective of the whole scheduling horizon. To ensure consistency of results

from the perspective of $SP_{s\_}$ and SP_s, we formulate three sets of consistency constraints and impose them to each subproblem.

$$CC_{s\_,s}: \begin{cases} I_{ut_o(s\_,s)} - I_{ut_o(s,s\_)} = 0 & \forall u, t_o = n\_ + 1 \\ h^{on}_{u(s\_,s)} - h^{on}_{u(s,s\_)} = 0 & \forall u \\ h^{off}_{u(s\_,s)} - h^{off}_{u(s,s\_)} = 0 & \forall u \end{cases} \quad (17)$$

The consistency constraints between SP_s and SP_{s_+} are:

$$CC_{s,s_+}: \begin{cases} I_{ut_o(s,s_+)} - I_{ut_o(s_+,s)} = 0 & \forall u, t_o = n + 1 \\ h^{on}_{u(s,s_+)} - h^{on}_{u(s_+,s)} = 0 & \forall u \\ h^{off}_{u(s,s_+)} - h^{off}_{u(s_+,s)} = 0 & \forall u \end{cases} \quad (18)$$

Each set is considered as variables in the left SP indicated in the subscript $(\cdot,\cdot)$. For instance, $h^{on}_{u(s\_,s)}$ is a variable in $SP_{s\_}$ for modeling transition from $SP_{s\_}$ to SP_s. Constraints (17) are enforced in $SP_{s\_}$ and SP_s, and (18) are added in SP_s and SP_{s_+}.

4.4.5. Summary of SCUC Subproblems Formulation

In summary, the SCUC subproblem for each subhorizon s is formulated by adding (5)-(18) to the optimization problem (4). Although (7)-(16) model interdependencies between SP_s and its previous subproblem $SP_{s\_}$, a set of similar expressions are required to model interdependencies between SP_s and its next subproblem SP_{s_+}. For modeling minimum on/off time constraints, we use models presented in [85] where h^{on}_u and h^{off}_u are known parameter from initial conditions. However, in our developed models, h^{on}_u (indicated by $h^{on}_{u(s,s\_)}$ and $h^{on}_{u(s,s_+)}$) and h^{off}_u (indicated by $h^{off}_{u(s,s\_)}$ and $h^{off}_{u(s,s_+)}$) are shared decision variables that are byproducts of SCUC decision variables. In addition to regular SCUC input parameters (e.g., network topology and capacity of units), SP_s receives $p_{ut_o(s\_,s)}$, $I_{ut_o(s\_,s)}$, $h^{on}_{u(s\_,s)}$, and $h^{off}_{u(s\_,s)}$ from $SP_{s\_}$ and $p_{ut_o(s_+,s)}$, $I_{ut_o(s_+,s)}$, $h^{on}_{u(s_+,s)}$, $h^{off}_{u(s_+,s)}$ from SP_{s_+} as inputs, and determines Ξ_s (see (4)), $p_{ut_o(s,s\_)}$, $I_{ut_o(s,s\_)}$, $p_{ut_o(s,s_+)}$, $I_{ut_o(s,s_+)}$,

$h^{on}_{u(s,s_-)}$, $h^{off}_{u(s,s_-)}$, $h^{on}_{u(s,s_+)}$, $h^{off}_{u(s,s_+)}$ as outputs.

4.4.6. Discussion on Number of Subhorizons

Increasing the number of subhorizons increases the number of shared variables and consequently the required number of iterations for the distribution algorithm to converge. Hence, although smaller subproblems take less time to be solved, increasing the number of subhorizons is not necessarily efficient for time-saving. There should be a tradeoff between the number of subhorizons and the number of iterations. Another factor that should be considered is the power of computational facilities. Although a computing machine takes less time to solve smaller optimization problems, further size reduction beyond a certain level may not yield significant time-saving.

Determining the optimal number of subhorizons is challenging and a valuable research question. A possible approach is to solve a given SCUC problem considering different numbers of subhorizons and select the one that yields the best time-saving. We suggest to determine the length of subhorizons in a way to be larger than the largest minimum on/off times of generating units. Another suggestion is to have subhorizons with a similar size to take advantage of parallel computing with the least idle times of CPUs. Since the behavior of the problem may change depending on the system loading condition, different number of subhorizon may be more appropriate for different load pattern. The yearly load can be classified into different groups (e.g., a group for each season of a year), and a suitable number of subhorizons can be found for each group using a trial and error approach. We are working on this research question as future work.

4.5. Converting Model into MIP

Closed-form linear models minnonzero, min, and max functions in (10)-(18) are needed to formulate the problem in the standard MIP format.

4.5.1. Minnonzero Functions

We model the minnonzero function in (11) using a Big-M value and replace (13) by:

$$\mathcal{F}_{us}^{sd} = \min\left((1 - z_u) \times M + \Phi_{us}^{off}\right) \quad \forall u \quad (19)$$

where z_u is a vector containing the shutdown indicator for each interval t. This constraint assigns the first time index of SP_s in which a shutdown happens (i.e., $z_{ut} = 1$) to $\mathcal{F}_{us}^{sd}$. To model the minnonzero function for minimum off time constraint (15), we replace (15) by (20) to determine the first startup time.

$$\mathcal{F}_{us}^{su} = \min\left((1 - y_u) \times M + \Phi_{us}^{on}\right) \quad \forall u \quad (20)$$

4.5.2. Min Function

We follow the concepts presented in [86] for linearizing min functions. In SP_s, instead of equation (11), we use:

$$\mathcal{F}_{us}^{sd} \leq \left((1 - z_u) \times M + \Phi_{u}^{off}\right) \quad \forall u \quad (21)$$

The right-hand-side of (21) is a vector and its left-hand-side is a variable. Therefore, $\mathcal{F}_{uN}^{sd}$ is smaller than the minimum value of the right-hand-side vector. We also replace (12) as follows:

$$h_{u(s,s_-)}^{on} \leq h_{u(s_-,s)}^{on} \quad \forall u \quad (22)$$

$$h_{u(s,s_-)}^{on} \leq \mathcal{F}_{us}^{sd} - 1 \quad \forall u \quad (23)$$

$$h_{u(s,s_-)}^{on} \leq M \times I_{ut_o(s,s_-)} \quad \forall u, t_o = n_- + 1 \quad (24)$$

As explained in Section IV, we formulate augmented Lagrangian penalty functions, whose goal is to vanish mismatches between coupling variables, in a way for $h_{u(s,s_-)}^{on}$ to appear with negative terms in the objective function of SP_s. Since SCUC is a minimization problem, (22)-(24) along with including $h_{u(s,s_-)}^{on}$ in the objective function result in $h_{u(s,s_-)}^{on}$ taking the minimum of $h_{u(s_-,s)}^{on}$, $M \times I_{ut_o(s,s_-)}$, and $\mathcal{F}_{us}^{sd} - 1$. Other min functions can be converted analogously.

4.5.3. Max Function

We follow the concepts presented in [86] for linearizing max functions. For calculating $h_{us\_}^{on}$ in (8), we introduce an auxiliary variable H_u^{on} for each unit u to formulate (25)-(26) and use (27)-(28) instead of (9) for calculating $h_{u(s\_,s)}^{on}$.

$$H_u^{on} \geq \Phi_{ut}^{on} \qquad \forall u, \forall t \qquad (25)$$

$$h_{us\_}^{on} = n\_ - H_u^{on} + 1 \qquad \forall u \qquad (26)$$

$$h_{u(s\_,s)}^{on} \geq 0 \qquad \forall u \qquad (27)$$

$$h_{u(s\_,s)}^{on} \geq \left(T_u^{on} - h_{us\_}^{on}\right) \qquad \forall u \qquad (28)$$

As explained in Section IV, $h_{u(s\_,s)}^{on}$ appears with positive terms in the objective function of $SP_{s\_}$ with penalizing consistency constraints (17) and (18). As explained in [86], this penalization strategy along with (25)-(28) results in $h_{u(s\_,s)}^{on}$ taking the maximum of $\left(0, T_u^{on} - h_{us\_}^{on}\right)$. The minimization problem tries to reduce $h_{u(s\_,s)}^{on}$, and (25)-(28) enforce $h_{u(s\_,s)}^{on}$ to be larger than zero and $T_u^{on} - h_{us\_}^{on}$. In addition, $H_u^{on} \geq \Phi_{ut}^{on}$ means that H_u^{on} is larger than or equal to the largest element of vector Φ_u^{on}. Since the objective function tends to minimize and the penalty term $h_{u(s\_,s)}^{on}$ pertaining to H_u^{on} appears with positive terms in the objective function, H_u^{on} sticks to the smallest possible value that is the maximum of vector Φ_u^{on}. Other max functions can be modeled in a similar manner.

4.6. Coordination Algorithm

We propose an accelerated analytical target cascading (A-ATC) algorithm to coordinate SCUC subproblems. An acceleration technique and an initialization strategy are proposed to enhance the convergence performance of ATC.

4.6.1. Normal Parallel ATC

Consider $SP_{s\_}$ and SP_s formulated in Section II.E. For brevity of equations, we use a vector format to represent shared variables, e.g., $r_{s\_,s} = \{p_{ut_o s\_}, I_{ut_o s\_}, h^{on}_{u(s\_,s)}, h^{off}_{u(s\_,s)}\}$ for $SP_{s\_}$. We use the parallel ATC, which is based on augmented Lagrangian relaxation, to coordinate subproblems [53]. SCUC subproblems are in level two, and a coordinator is in level one. The actual shared variables r (called responses in the context of ATC) are duplicated to create a set of auxiliary variables χ (called targets) for the coordinator and separate subproblems to make them solvable in parallel [53]. Differences between r and χ are consistency constraints between the coordinator and subproblems. Violations of consistency constraints are penalized in objective functions (3a) and (4a). The relaxed SCUC subproblem for subhorizon $s\_$ at iteration k is written as:

$$\left(x^k_{s\_}, r^k_{s\_,s}\right) = \text{argmin} \sum_{t=1}^{n\_+1} \sum_{u} f(p_{ut}, I_{ut}) \qquad (29)$$

$$-\lambda^{k\dagger}_{s\_,s}\left(\chi^{k-1}_{s\_,s} - r_{s\_,s}\right) + \rho\left\|\chi^{k-1}_{s\_,s} - r_{s\_,s}\right\|^2$$

$$s.t. \qquad (3b), (7)-(16)$$

where $x^k_{s\_}$ and $r^k_{s\_,s}$ are decision variables while $\chi^{k-1}_{s\_,s}$ are known target variables received from the coordinator. λ and ρ are Lagrange and penalty multipliers. Superscript † is the transpose operator. Analogously, SP_s is formulated as:

$$\left(x^k_s, r^k_{s,s\_}, r^k_{s,s_+}\right) = \text{argmin} \sum_{t=1}^{n+1} \sum_{u} f(p_{ut}, I_{ut}) \qquad (30)$$

$$-\lambda^{k\dagger}_{s\_,s}\left(\chi^{k-1}_{s,s\_} - r_{s,s\_}\right) + \rho\left\|\chi^{k-1}_{s,s\_} - r_{s,s\_}\right\|^2$$

$$+\lambda^{k\dagger}_{s,s_+}\left(\chi^{k-1}_{s,s_+} - r_{s,s_+}\right) + \rho\left\|\chi^{k-1}_{s,s_+} - r_{s,s_+}\right\|^2$$

$$s.t. \qquad (4b), \quad \underbrace{(7)-(16)}_{\text{for transition from } s\_ \text{ to } s} \quad \& \quad \underbrace{(7)-(16)}_{\text{for transition from } s \text{ to } s_+}$$

The coordinator receives r^k and solves the following problem to determine target variables χ^k.

$$\chi^k = \text{argmin } \lambda^\dagger(\chi - r^k) + \rho\|\chi - r^k\|^2 \quad (31)$$

For penalty terms with a positive sign for λ, Lagrange multipliers are updated as follows:

$$\lambda^k = \lambda^{k-1} + 2\rho^2(\chi^k - r^k) \quad (32)$$

and for penalty terms with a negative sign for λ, we have:

$$\lambda^k = \lambda^{k-1} + 2\rho^2(r^k - \chi^k) \quad (33)$$

This formulation can be generalized for multiple subproblems. λ and ρ need to be initialized in acceptable ranges. If λ and ρ are selected too large, the convergence speed may increase; however, it increases the chance of losing optimality [66]. Although selecting small λ and ρ enhances solution accuracy, the convergence speed degrades [66]. Knowledge about suitable ranges for λ and ρ will be gained by solving the problem multiple times and using historical information obtained from past implementations of the distributed SCUC.

4.6.2. Proposed Accelerated ATC

The convergence performance of ATC might degrade when more iterations are carried out and the solution becomes close to the optimal point, or when the optimal point is in a shallow area (or a ravine). Near the optimal point, the term $\chi^k - r^k$ might become small that leads to updated multipliers in iteration $k+1$ that are almost the same as them in iteration k, i.e., $\lambda^{k+1} \approx \lambda^k$. This slows the convergence speed.

We present an accelerated ATC based on the technique that first proposed by Nesterov to accelerate gradient descent methods [67, 68]. The proposed accelerated ATC utilizes a prediction type acceleration step. The concept of momentum is used to prevent the algorithm from deceleration while more iterations are carried out. After each iteration k, the cumulated gradient of previous iterations (i.e., momentum) is calculated as the predicted direction in the next iteration,

and a big jump is made in that direction. Then, the gradient is measured, and a correction is made to avoid moving forward too fast.

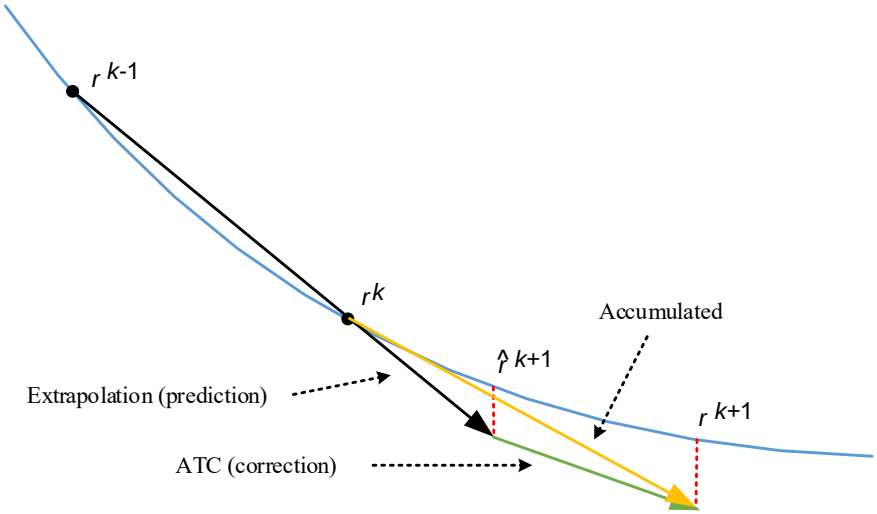


Figure 4.4. Prediction and correction procedure.

Accelerated ATC (A-ATC): At iteration k, we predict that to what point ATC directs coupling variables and Lagrange multipliers at iteration $k+1$. To do so, we add the effects of momentum in ATC. Consider coupling variable r at two consecutive iterations, i.e., r^{k-1} and r^k, as shown in Fig. 4. To predict r^{k+1}, we connect a line between points $(k-1, r^{k-1})$ and (k, r^k). The line equation is:

$$y = (r^k - r^{k-1})x + kr^{k-1} - (k-1)r^k \qquad (34)$$

By replacing $x = k+1$, we predict the next point as:

$$y = (r^k - r^{k-1})(k+1) + kr^{k-1} - (k-1)r^k = 2r^k - r^{k-1} \qquad (35)$$

We reorganize (35) and name $y = \hat{r}^{k+1}$ that is the predicted r at iteration $k+1$.

$$\hat{r}^{k+1} = r^k + (r^k - r^{k-1}) \qquad (36)$$

The first term is the current r and the second term is the line slope. We add a coefficient $\eta \in [0,1]$ to the slope to have a better prediction. In the first few iterations of ATC, large oscillations

may be observed, and the jump should not be large, otherwise, it may cause a large error on $\hat{r}^{k+1}$. Thus, in the first few iterations $\eta \ll 1$ to prevent large error and it becomes gradually larger to enhance the speed. We use the following expressions to increase η gradually until $\eta \cong 1$ [67, 68].

$$\alpha_0 = 1 \quad \& \quad \alpha_{k+1} = (1 + \sqrt{1 + 4\alpha_k^2})/2 \tag{37}$$

$$\eta = \frac{\alpha_k - 1}{\alpha_{k+1}} \tag{38}$$

Since α_k has an increasing trend, η becomes closer to one over the course of iterations. This leads to a larger ATC step size, especially when r approaches the optimal point. Near the optimal point, the slope becomes smaller and does not vary considerably. By this prediction-correction procedure, the convergence performance of ATC is enhanced.

A-ATC for SCUC: The penalty term in the objective function (29) of $SP_{s\_}$ at iteration k is modified as:

$$-\hat{\lambda}_{s\_,s}^{k\dagger}\left(\hat{\chi}_{s\_,s}^{k-1} - r_{s\_,s}\right) + \rho\left\|\hat{\chi}_{s\_,s}^{k-1} - r_{s\_,s}\right\|^2 \tag{39}$$

where $\chi_{s\_,s}$ and $\lambda_{s\_,s}$ are replaced by $\hat{\chi}_{s\_,s}$ and $\hat{\lambda}_{s\_,s}$. The two penalty terms in the objective function (30) of SP_s at iteration k are modified by replacing $\chi_{s,s\_}$, χ_{s,s_+}, $\lambda_{s\_,s}$, and $\lambda_{s,s\_}$ with $\hat{\chi}_{s,s\_}$, $\hat{\chi}_{s,s_+}$, $\hat{\lambda}_{s\_,s}$, and $\hat{\lambda}_{s,s_+}$. The new parameters shown by ($\hat{\cdot}$) are predictions of coordinator variables and Lagrange multipliers. The predicted values used at iteration $k+1$ are calculated based on the actual shared variables and penalty multipliers obtained at iterations k and $k-1$ as follows:

$$\alpha_{k+1} = \left(1 + \sqrt{1 + 4\alpha_k^2}\right)/2 \tag{40}$$

$$\hat{\chi}^{k+1} = \chi^k + \frac{\alpha_k - 1}{\alpha_{k+1}}\left(\chi^k - \chi^{(k-1)}\right) \tag{41}$$

$$\hat{\lambda}^{k+1} = \lambda^k + \frac{\alpha_k - 1}{\alpha_{k+1}}(\lambda^k - \lambda^{k-1}) \tag{42}$$

Convergence analysis of A-ATC is provided in Appendix I. Implementing A-ATC is similar to the classical ATC with a few more simple steps. The proposed A-ATC can be used instead of the classical ATC to solve various multi-agent and distributed optimization problems, such as distributed SCUC and optimal power flow

Initialization Strategy: ATC, similar to most distributed algorithms, is sensitive to initialization. To choose a good-enough starting point, we take advantage of power system characteristics and propose a method to initialize coupling variables. We ignore the coupling intervals and intertemporal connectivity between subhorizons. That means the coupling variables between subproblems are disregarded and subproblems are considered to be completely independent. The SCUC subproblems are solved in parallel. Although the achieved results might not be feasible as the connectivity between subproblems is ignored, they are close to the optimal solution as all other SCUC constraints at most intervals are respected. Hence, this solution can be used as a good starting point for initializing the distributed algorithm. Our simulation results illustrate that this initialization strategy works well for all studied cases.

The pseudocode for the A-ATC-based distributed SCUC is given in the following Algorithm.

Algorithm Pseudocode for the proposed A-ATC-based distributed SCUC

1: Decompose the considered horizon into multiple subhorizons
2: Ignore the coupling intervals and shared variables
3: Solve SCUC subproblems in parallel
4: Use the obtained results to initialize $\hat{\chi}^0$
5: Initialize multipliers λ^0, ρ in their acceptable ranges and set $k = 0$
6: **while** $|\chi^k - r^k| > \varepsilon$, $k = k + 1$ **do**
7: Solve SCUC subproblems in parallel and determine r^k (outputs of SCUC subproblems and inputs for the coordinator)
8: Solve the coordinator problem to determine χ^k (outputs of the coordinator and inputs

for SCUC subproblems)

9: Exchange r^k and χ^k between the coordinator and subproblems

10: Update λ^k by (32) or (33)

11: Calculate α_{k+1} by (40)

12: Update $\hat{\chi}^{k+1}$ and $\hat{\lambda}^{k+1}$ by (41) and (42) (inputs for the coordinator and SCUC subproblems)

13: **end while**

4.7. Case Study

In this chapter The proposed algorithm is implemented on a 3-bus system, the IEEE 24-bus system, the IEEE 118-bus system, and a 472-bus system. All systems information is given in [103]. The considered scheduling problem includes 72 intervals, each representing an hour. For most cases, we have observed that decomposing the consider scheduling horizon into three subhorizons, each with 24 intervals, will lead to good time-saving. Having three subhorizons also shows the situation of modeling intertemporal constraints for a middle subproblem. Hence, without loss of generality, we have divided the scheduling horizon into three subhorizons. In addition, we study the impact of number of subhorizons on solution performance using the IEEE 24-bus system. To initialize Lagrange multipliers and penalty parameters multipliers, we have run multiple cases with λ^0 and ρ in the range of 0.01 to 100. For all cases, setting $\lambda^0 = \rho = 1$ for continuous shared variables and $\rho = 3$ for integer shared variables provides good results. All simulations are carried out using YALMIP toolbox [70] and Gurobi solver on a PC with 16GB of RAM. The reported times for the distributed SCUC include the initialization time and time that A-ATC takes to converge. The following convergence index is used to measure the relative distance between operation costs determined by the distributed SCUC, f^d, and benchmark results obtained by the centralized SCUC, f^*:

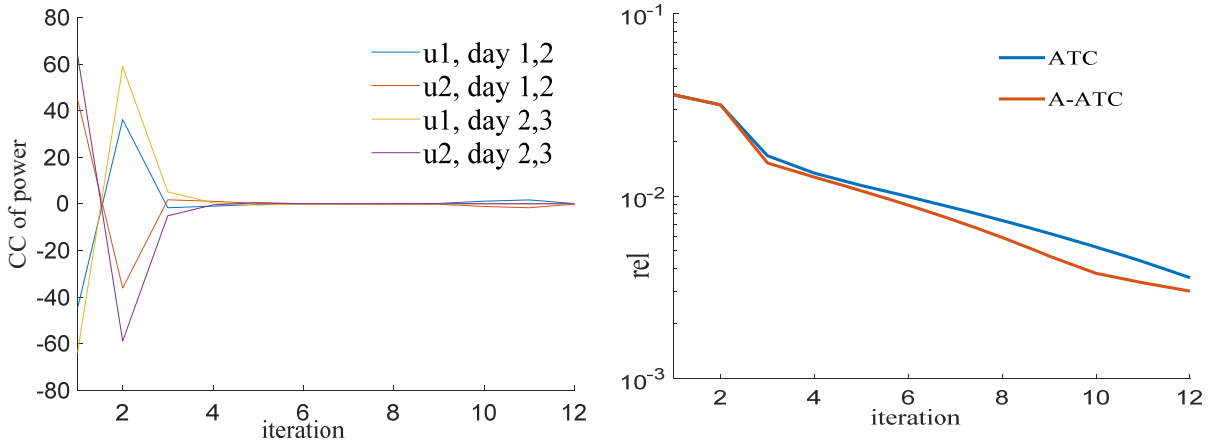
$$rel = \frac{|f^* - f^d|}{f^*} \qquad (43)$$

4.7.1. 3-Bus System

This small case serves as a tutorial for the proposed algorithm. Two load patterns are considered to analyze all three possible situations explained in Section 4.2.

Case1: The load pattern one is considered. Intervals 1-25, 25-49, and 49-72 belong, respectively, to subhorizons one, two, and three. A-ATC, without applying the suggested initialization strategy, converges after 12 iterations. As shown in Fig. 4.5a, consistency constraints (5) and (6) are roughly zero upon convergence that means power generated by units in coupling intervals are the same from the perspective of consecutive subproblems. Table 4.1 shows units' on/off status at the boundary intervals of SP_1 and SP_2 after the first iteration. The same results are obtained with and without considering on/off time consistencies since minimum on/off times (i.e., 5 hours) are already satisfied. Consider the on/off status of unit two. SP_1 sends $h^{on}_{2(1,2)} = 3$ to ask SP_2 to keep unit two on for three more hours. From the perspective of SP_2, this is satisfied as according to (12) $h^{on}_{2(2,1)} = 3$. This is situation 1 in Section 4.2 in which shared variables corresponding to on/off status of the units are equal from the first iteration and consistency constraints (17) are satisfied. Table 4.2 shows the units' on/off status at the boundary intervals between SP_2 and SP_3 after the first iteration. SP_2 sends $h^{on}_{2(2,3)} = 0$ to SP_3 for unit two. On the other hand, according to (12), $h^{on}_{2(3,2)} = \min\{0,2\} = 0$, which means unit two in SP_3 stays on only at the first two intervals. This is situation 3 in Section 4.2 in which consistency constraints (18) are satisfied at the first iteration. However, the power generation in coupling intervals 25 and 49 must be equal to satisfy consistency constraints (5) and (6) for generation ramp constraints. All consistency constraints are satisfied upon convergence of the algorithm after 12 iterations.

Figure 4.5b shows the rel index, and Table 4.3 compares cost obtained by the distributed SCUC and the benchmark cost determined by the classical centralized SCUC. Upon convergence, the cost of distributed SCUC is $668,713, and the



(a) (b)

Figure 4.5. a) The difference between shared variables in coupling intervals and b) the rel index for case 1of 3-bus system.

Table 4.1. Units On/Off Status in Boundary Intervals Between Days One and Two After The First Iteration (Case 1)

	Subhorizon one						Subhorizon two					
Hour	20	21	22	23	24	25	25	26	27	28	29	30
Unit 1	1	1	1	1	1	1	1	1	1	1	1	1
Unit 2	0	0	0	1	1	1	1	1	1	1	1	1

Table 4.2. Units On/Off Status in Boundary Intervals Between Days Two and Three After The First Iteration (Case 1)

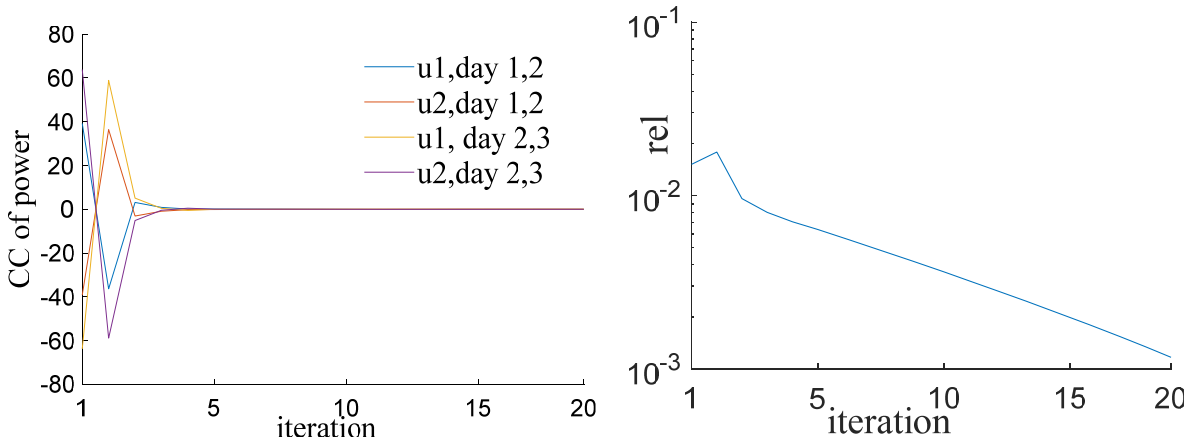
	Subhorizon two						Subhorizon three					
Hour	44	45	46	47	48	49	49	50	51	52	53	54
Unit 1	1	1	1	1	1	1	1	1	1	1	1	1
Unit 2	1	1	1	1	1	1	1	1	1	0	0	0

rel index is 0.003. For the sake of comparison, we have simulated ATC and A-ATC. As shown in Fig. 5b, rel obtained by A-ATC is always less than that obtained by ATC. This means that after a specific number of iterations, A-ATC provides a more accurate solution than the normal ATC.

To enhance the distributed SCUC performance, we apply the suggested initialization strategy and re-run A-ATC. The algorithm converges after three iterations. The units' on/off status and power dispatches are the same as those obtained by the centralized SCUC. Operation costs obtained by A-ATC is \$666,709 that is equal to the cost determined by the classical centralized SCUC (i.e., $rel \approx 0$).

Case 2: Load pattern two is considered. Table 4.4 shows the units' on/off status after the first iteration. According to (14), for unit two, SP_1 sends $h^{off}_{2(1,2)} = 3$ to SP_2. But, according to (16), from the view of SP_2, it is optimal to have $h^{off}_{2(1,2)} = 2$ (situation 2 in Section II.B). Therefore, if subproblems are solved separately, $h^{off}_{2(1,2)} \neq h^{off}_{2(2,1)}$ and SCUC results are infeasible.

With the stopping criterion of 0.01 MW, A-ATC, without applying the suggested initialization strategy, converges after 20 iterations. Figure 4.6 shows the difference between shared variables and rel over iterations. Table 4.5 depicts the units' on/off status at the boundary intervals upon convergence. The unit commitment results and power generations are similar to those obtained by the classical centralized SCUC.



(a) (b)

Figure 4.6. a) The difference between shared variables in coupling intervals and b) the rel index for case 2 of the 3-bus system.

Table 4.3. Results For 3-Bus System (Cases 1) w/wo initialization Strategy

Algorithm	Cost ($)	Iteration	rel
Centralized	666,709	-	-
Distributed A-ATC	668,713	12	0.003
Distributed A-ATC+ initialization	666,709	2+1	≈0

Table 4.4. Units On/Off Status in Boundary Intervals Between Days One and Two after the first iteration (Case 2)

	Subhorizon one						Subhorizon two					
Hour	20	21	22	23	24	25	25	26	27	28	29	30
Unit 1	1	1	1	1	1	1	1	1	1	1	1	1
Unit 2	1	1	1	0	0	0	0	0	1	1	1	1

Table 4.5. Units On/Off Status in Boundary Intervals Between Days One and Two Upon convergence (Case 2)

	Subhorizon one						Subhorizon two					
Hour	20	21	22	23	24	25	25	26	27	28	29	30
Unit 1	1	1	1	1	1	1	1	1	1	1	1	1
Unit 2	1	1	1	1	1	1	1	1	1	1	1	1

Table 4.6. Results For the 3-Bus System (Case 2) w/wo Initialization Strategy

Algorithm	Cost ($)	Iteration	rel
Centralized	817,755	-	-
Distributed A-ATC	818,710	20	0.001
Distributed A-ATC+ initialization	817,755	10+1	≈0

Table 4.7. Results of Different Algorithms for 24-Bus System

Algorithm	Iteration	rel	Time (s)
Centralized NCUC	-	-	17
NCUC with A-ATC + initialization	2+1	≈0	2
Centralized SCUC	-	-	135
SCUC with A-ATC + initialization	3+1	≈ 3e-8	12

Table 4.8. Power of units in Coupling Intervals (24-Bus System)

Unit No.	Hour 25, subhorizon 1	Hour 25, subhorizon 2	Hour 49, subhorizon 2	Hour 49, subhorizon 3
1	184.5	184.5	177.7	177.7
3	153.7	153.7	148.1	148.1
5	184.5	184.5	177.7	177.7
7	162.1	162.1	156.4	156.4
9	204.5	204.5	197.7	197.7

Table 4.9. Results for the IEEE 118-Bus System

Algorithm	Iteration	rel	Time (s)
Centralized	-	-	3818
ATC + initialization	4+1	≈e-5	216
A-ATC + initialization	4+1	≈e-6	216

A-ATC is applied with the suggested initialization strategy. As shown in Table 4.6, operation cost is $817,755 using distributed SCUC and the classical centralized SCUC. The suggested initialization strategy reduces the number of iterations from 20 to 11 and makes $rel \approx 0$.

4.7.2. IEEE 24-Bus System

Two cases are studied: network constrained UC (NCUC) with no contingency and SCUC with ten plausible contingencies. The initialization strategy and A-ATC are implemented for both cases. The distributed NCUC converges after three iterations within two seconds. As depicted in Table 4.7, operation costs determined by the distributed and centralized NCUC algorithms are $3,774,232. The distributed algorithm decreases the solution time by 88%. Table 4.8 shows power

generated by several sample units in coupling intervals. The rel becomes zero upon convergence.

For SCUC, ten scenarios for transmission line outage are considered at each interval. This increases the size of the problem. The distributed SCUC converges to the same cost as of that for the centralized SCUC (i.e., $3,833,162) while being 91% faster. This implies that when the size of the problem increases, the effectiveness of the distributed SCUC algorithm is more prominent.

Number of Subhorizons: We have studied the impact of number of subhorizons on the solution time. We have decomposed the scheduling horizon into different number of subhorizons and run the distributed SCUC. Figure 4.7 shows the solution time versus the number of subhorizons for the 24-bus system. It is observed that increasing the number of subhorizons reduces the solution time; however, decomposition the scheduling horizons beyond a certain number, which is nine for this case, increases the solution time. Such a trend is widely observed in parallel computing approaches. Breaking the horizon into two and three subhorizons leads to the best time-saving. Increasing the number of subhorizons beyond three does not result in significant time-saving.

We have also studied the impact of load on the optimal number of subhorizons. Several low-load and high-load scenarios are analyzed. Although we have observed that the load changes the curve pattern shown in Fig. 4.7, one pattern, similar to Fig. 4.7, is obtained for all load conditions between 70%~100% of the baseload (Fig. 7 is plotted for the baseload) and another pattern for load conditions larger than the baseload. Hence, we conclude that one can categorize the load into low, medium, and high (e.g., seasonal load), decompose the problem into several subhorizons for each loading condition, and select the best number of subhorizons for each loading condition. This is an efficient approach as a user can use the same number of subhorizon for many similar load scenarios.

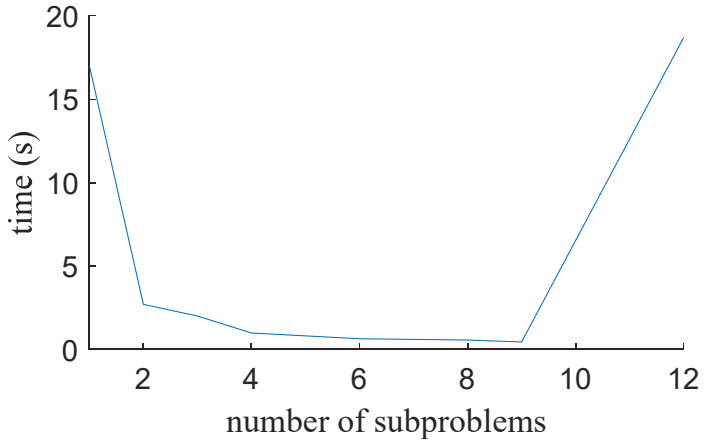


Figure 4.7. Time versus number of subhorizons for the IEEE 24-bus system.

Table 4.10. Results for the 472-Bus System

Algorithm	Iteration	Cost ($)	Time (hour)
Centralized	-	17,866,016	111
A-ATC + initialization	6+1	17,768,289	14

4.7.3. IEEE 118-Bus System

As shown in Table 4.10, the centralized approach provides a cost of $4,928,242 after 3818 seconds. The distributed algorithm converges to a cost of $4,928,268 after five iterations within 216 seconds that is 94% faster than the centralized approach. We also compare ATC and A-ATC. A-ATC provides a smaller rel index upon convergence.

To have a better comparison, we stop the centralized approach when a cost of $4,928,268 (the same as A-ATC) is obtained. While A-ATC converges after 216 seconds, the centralized approach takes 2406 seconds. This means that the centralized SCUC takes 91% more computation time to converge to the same solution as the distributed algorithm.

4.7.4. A 472-Bus System

The superiority of the proposed algorithm as compared to the centralized SCUC is more

considerable for larger cases. We combine four IEEE 118-bus systems to build a 472-bus system. Table 4.10 shows the results. We stop the centralized approach after 111 hours, and the achieved cost is $17,866,016. However, the proposed distributed algorithm converges after seven iterations to a cost of $17,768,289 within 14 hours.

4.8. Conclusion

A strategy is proposed to decompose SCUC over the scheduling time horizon, create several subproblems each for a subhorizon, and model temporal interdependencies between subproblems. The proposed strategy is called a temporal decomposition. The concept of coupling intervals is introduced to model ramping limitation of generating units between subproblems. In addition, several counting auxiliary variables are determined to coordinate minimum on/off times for transition between consecutive subproblems. An accelerated ATC algorithm with an initialization strategy is proposed to coordinate subproblems. The simulation results show that the proposed algorithm obtains the same SCUC results (i.e., binary variables for units' on/off status and power dispatch) as the traditional centralized SCUC while reducing the computation time considerably. For instance, for the IEEE 118-bus system, the solution time decreases by 94%. We have observed that as the size of the problem increases, the distributed algorithm shows better performance than the centralized SCUC. The results show the privilege of A-ATC to ATC and effectiveness of the suggested initialization strategy to enhance the convergence performance. For the future work, we will focus on developing methods to find the best time intervals for decomposing the considered scheduling horizon and the optimal number of subhorizons.

CHAPTER 5
MULTI-CLASS LEARNING-BASED TEMPORAL DECOMPOSITION AND DISTRIBUTED OPTIMIZATION

5.1. Introduction

As shown in Chapters 2 to 4, temporal decomposition is a potential approach to relieve the computation costs of power system multi-interval scheduling problems. However, the main existing challenge is how to decompose the scheduling horizon to gain the most time saving from distributed computing. In this chapter, we present a machine learning-based temporal decomposition strategy to decompose a considered scheduling horizon optimally. the mian focus is on SCED problem presented in Chapter 2. We have found that, the load profile, which is known before solving SCED, has a significant on the optimal number of subhorizons. We have used load profiles as inputs to the learner whose goal is to assign a temporal decomposition class to each load profile. Possible decomposition classes are divisors of the considered scheduling horizon. Thus, the proposed learning procedure is a multi-class classification. We have selected Extreme Gradient Boosting that is a tree-based classification learner. Simulation results using real-world load profiles show the promising performance of the proposed algorithm.

5.2. Contribution

The main contributions of this chapter are as follows:

- To study the effect of temporal partitioning on the solver time and accuracy of results.
- To study parameters that have a significant effect on the optimal number of subproblems.
- To develop a methodology to find the best number of subproblems in a temporal decomposition optmization problem.

5.3. Important Factors and Motivating Examples

Reducing the length of subhorizons and the number of iterations of the coordination algorithm

reduces the overall solution time distributed optimization. The less the number of variables and constraints of a subproblem is, the less the computational time would be. However, because of increasing the number of shared variables, the number of iterations of the distributed algorithm needed to coordinate subproblems might increase, and hence, the total solution time increases. Moreover, decomposition the problem from intervals with many active intertemporal consistency ramping up and down constraints results in a larger gap between shared variables in the first few iterations. Larger differences between shared variables may increase the required number of iterations of distributed optimization to converge. The scheduling horizon must be decomposed carefully to obtain the best solution time.

In this section, we illustrate a few examples to show the necessity of optimal temporal decomposition and the factors that affect it. The considered scheduling problem is a week-ahead economic dispatch. The objective function is to minimize the generation cost subject to generating unit constraints, power balance equalities, preventing action constraints, line flow limits, voltage angles limitations, and units ramping up/down constraints. Simulations are carried out in MATLAB using YALMIP [70] as modeling software and Gurobi solver on a 3.7 GHz personal computer with 16GB of RAM.

5.3.1. Motivating Examples

A 3-Bus System with Smooth Load Profile: The maximum load change in two consecutive time intervals is less than the ramping limitations of generating units. Therefore, intertemporal consistency constraints connecting subhorizons, i.e., ramp up and down constraints, are not active. This makes subproblems loosely connected and yields small differences between shared variables, i.e., power generated by units at overlapping time intervals. As shown in Fig. 5.1, by increasing the number of subproblems, the size of each subproblem will be smaller, and the overall solution

time decreases. However, since the system is small, decomposing the problem beyond 20 subhorizons does not lead to significant time-saving. At the same time, it needs more computational resources and also may slightly increase the solution error as compared to the centralized method. Hence, we suggested not decomposing the scheduling horizon beyond 20 subhorizons.

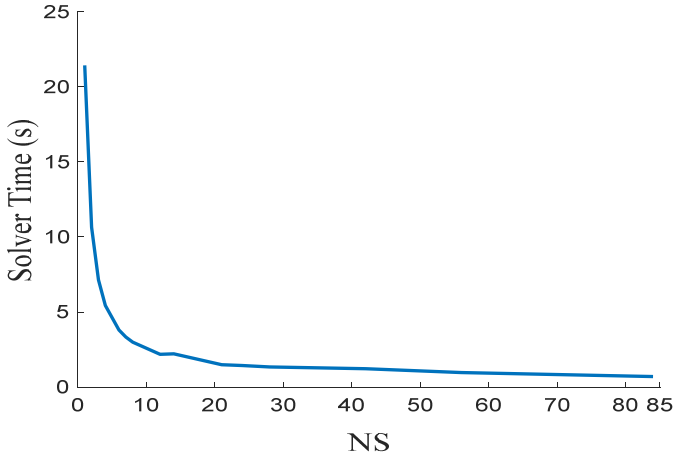
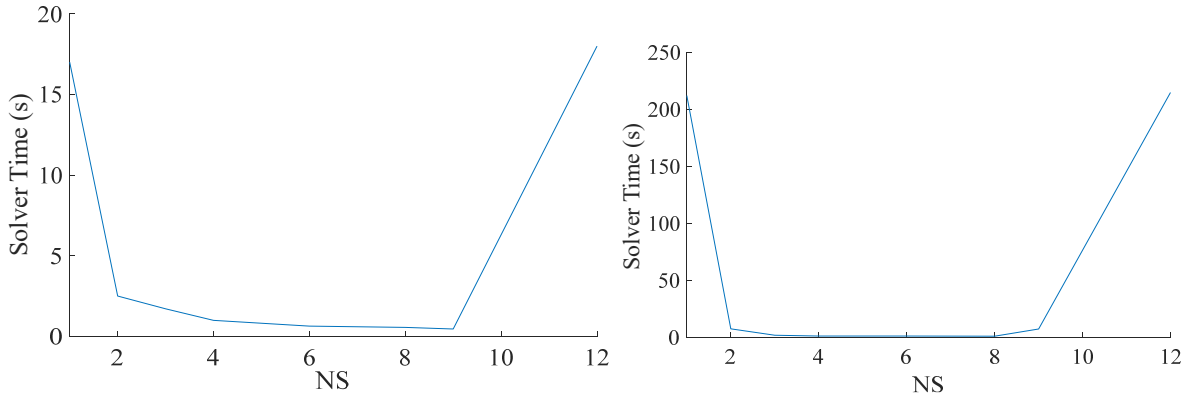


Figure 5.1. Overall solution time versus number of subhorizons (NS) for the 3-bus test system with a flat load profile.

IEEE 24-Bus System with Variable Load Profile: Figure 2.a shows the solution time versus the number of subhorizons for a given load pattern one. Increasing the number of subhorizons reduces the solution time; however, increasing the number of subhorizons beyond nine increases the solution time. This is because of having many active intertemporal consistency constraints and large differences in the desired values for shared variables from the perspective of neighboring subhorizons. Hence, the number of iterations of the distributed algorithm grows that increases the overall solution time. We have reduced the load by 5% and redraw the curve, as shown in Fig. 5.2 b. This load decrease does not change the pattern of the curve.

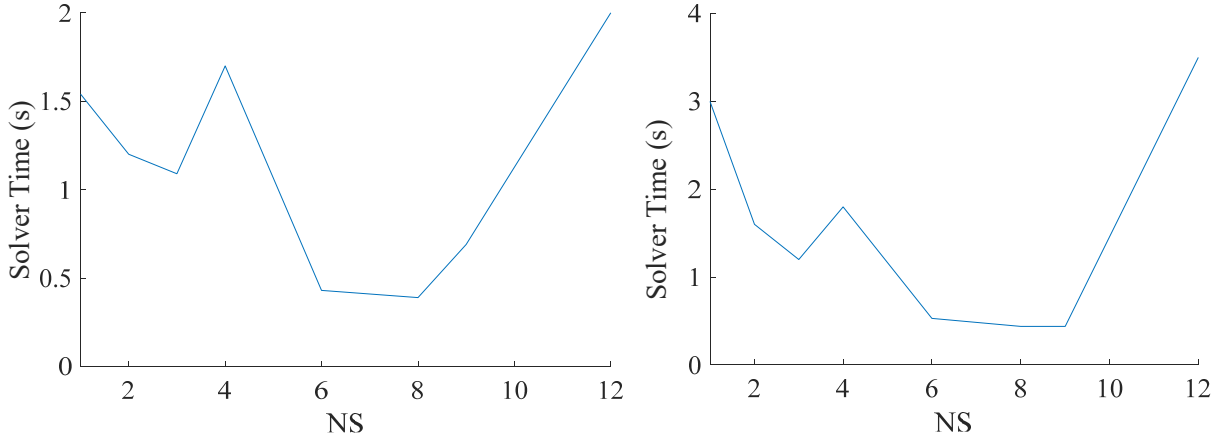
We have tested another load pattern two. The solution time does not follow a curve similar to Fig. 5.1a and rather has monotonic behavior. This is because of the sophisticated behavior of

intertemporal consistency ramp up/down constraints. These constraints connect intervals $\{1, \dots, T\}$ and will be active depending on the load pattern and system characteristics. This results in an unknown pattern in the desired values of shared variables from the perspective of neighboring subhorizons and hence a non-monotonic behavior in the solution time pattern. The load is reduced by 5%, and the curve is plotted in Fig. 5.3.b, which is similar to Fig. 5.3a. Comparing Figs. 5.2 and 5.3 shows that the load profile pattern has a more significant impact on the solution time curve rather than a small load increase or decrease.



(a) (b)

Figure 5.2. Overall solution time versus number of subhorizons (NS) for the IEEE 24-bus system with a) load pattern one and b) load pattern one with 5% decrease.



(a) (b)

Figure 5.3. Overall solution time versus number of subhorizons (NS) for the IEEE 24-bus system with a) load pattern two and b) load pattern two with 5% decrease.

5.3.2. Important Factors

Factors that affect the overall solution time versus the number of subhorizons include 1) system characteristics, 2) generators characteristics, 3) the number of active generation ramping up/down constraints for the transition between subhorizons, and 4) the desired values of shared variables from the perspective of neighboring subhorizons. For a given system with a set of generating units, the third and fourth factors are to be analyzed to obtain the optimal temporal decomposition. However, they are unknown before solving the economic dispatch problem. The load profile, which is known before solving economic dispatch, plays a critical role in the status of intertemporal constraints and the values of variables. Hence, the load profile can be used to analyze the number of subhorizons versus the solution time and find the optimal temporal decomposition scheme.

5.4. Proposed Learning-Based Methodology

Given that the load profile is known to system operators before solving economic dispatch, we propose a learning-based algorithm for temporal partitioning. The goal of this learner is to project the optimal number of subhorizons to the demand profile pattern. As illustrated in Fig. 5.4, the input to the learner, which is a multi-class classifier, is the load profile over the considered scheduling horizon, and its output is the best time partitioning scheme.

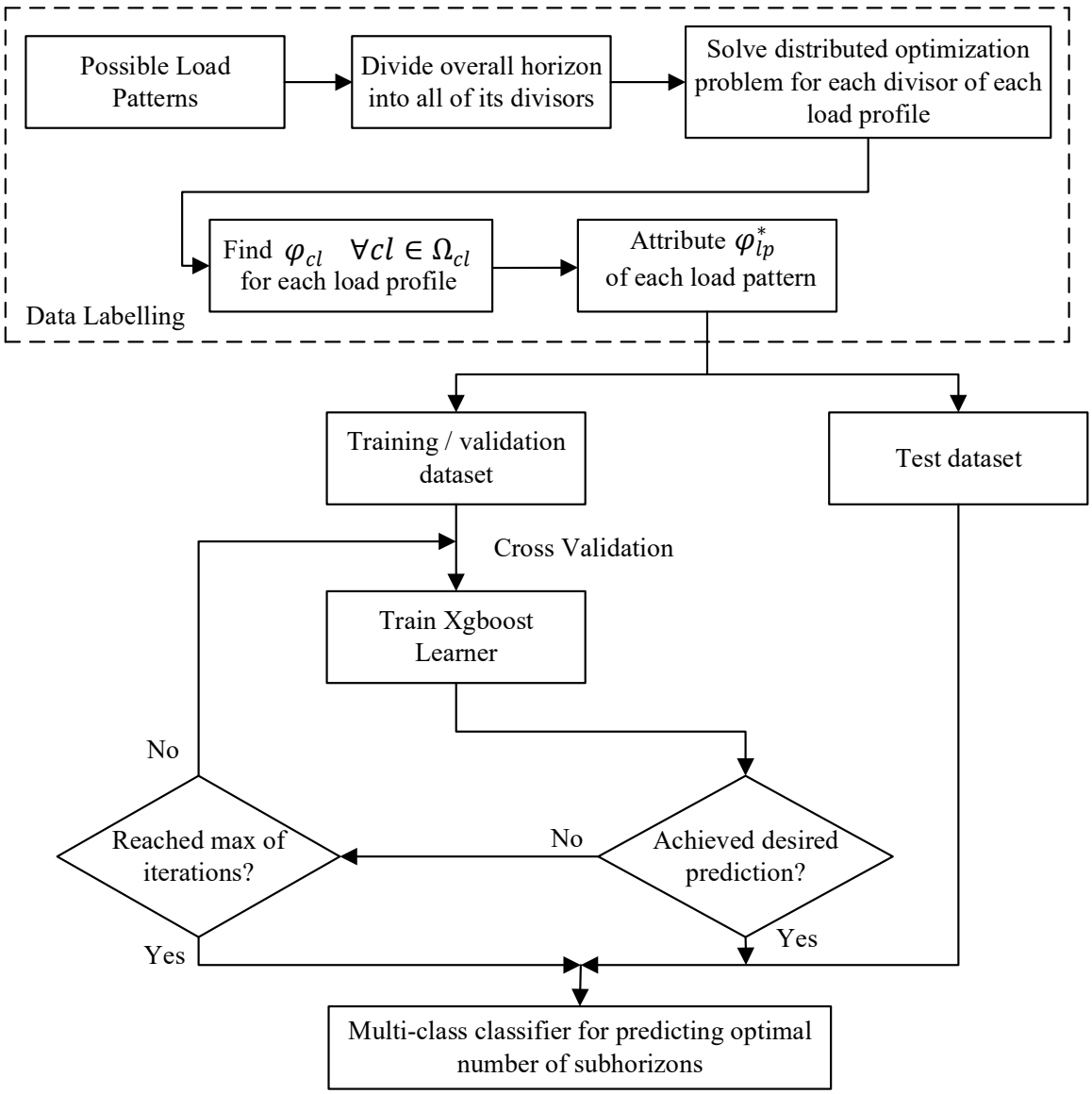


Figure 5.4. Flowchart of the proposed learning-based temporal decomposition algorithm.

5.4.1. Offline Data Labelling

Historical load profile patterns and predicted load patterns for the considered scheduling horizon are collected as the possible load patterns. Subproblems should have roughly the same size to gain the most advantage of parallel computing with minimum idle time. For each load pattern lp, all divisors of the considered scheduling horizon are determined as the possible decomposition classes (denoted by Ω_{cl}) with subhorizons with equal length. For a scheduling horizon with 72 intervals,

for instance, $\Omega_{cl} = \{1,2,3,4,6,12,18,24,36,72\}$. The coordination strategy is applied to solve SCED in a distributed manner for each decomposition class cl. An error-time index is created by combining the solution time and the relative error to determine the best class for each load pattern.

$$\varphi_{cl} = \omega_1 \times rel_{cl} + \omega_2 \times CPU_{time,cl} \qquad \forall cl \in \Omega_{cl} \quad (1)$$

where the rel index is the relative error between the optimal costs obtained by centralized (f^*) and distributed (f_{cl}^d) approaches.

$$rel_{cl} = \left| \frac{f^* - f_{cl}^d}{f^*} \right| \qquad (2)$$

Weighting factors ω_1 and ω_2 assign priority to rel_{cl} and $CPU_{time,cl}$ indices. A system operator can determine the weight values based on its preference for solution time and accuracy. After solving distributed SCED for each cl of the load pattern lp, the decomposition class with the smallest error-time index

$$\varphi_{lp}^* = \min\{\varphi_{cl} \quad \forall cl \in \Omega_{cl}\} \qquad (3)$$

is determined (denoted by cl_{lp}^*), and the load pattern is assigned to this class. As depicted in Fig. 5, this offline procedure, whose pseudocode is shown in Data Labelling Algorithm, assigns the optimal number of subhorizons as the class label for each load pattern.

Data Labelling Algorithm Pseudocode for multi-class classification of load patterns and data preparation

1: Read historical load patterns for the considered scheduling horizon
2: Determine weighting factors ω_1 and ω_2
3: Do for all possible load patterns
4: **for** $cl \in \Omega_{cl}$
5: Decompose the considered horizon into cl equal subhorizons

6: **while** $|CC| > \varepsilon, k = k + 1$ **do**

7: Solve SCED subproblems in parallel and determine optimal values of $p^{*kN_-}_{ut_oN_-\to N}$, $p^{*kN}_{ut_oN_-\to N}$, $p^{*kN}_{ut_oN\to N_+}$, and $p^{*kN_+}_{ut_oN\to N_+}$

8: Exchange $p^{*kN_-}_{ut_oN_-\to N}$, $p^{*kN}_{ut_oN_-\to N}$, $p^{*kN}_{ut_oN\to N_+}$, and $p^{*kN_+}_{ut_oN\to N_+}$ between SCED subproblems

9: Update λ^k by (12) and (13) in chapter 2

10: **end while** (see Algorithm I in Chapter 2 for more details)

11: Record solution time $CPU_{time,cl}$ and calculate rel_{cl}

12: Calculate $\varphi_{cl} = \omega_1 \times rel_{cl} + \omega_2 \times CPU_{time,cl}$

13: **end for**

14: Determine φ^*_{lp} for each load pattern

15: Assign the load pattern to cl^*_{lp}

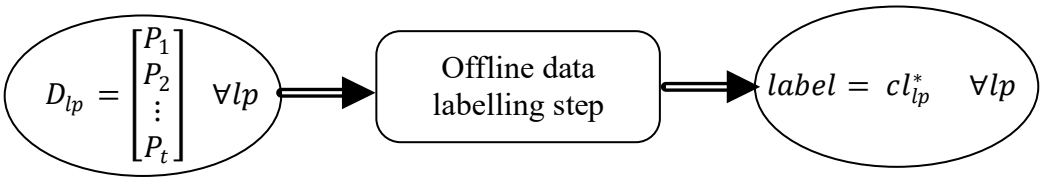


Figure 5.5. Offline data labelling procedure.

5.4.2. Multi-Class Learning for Temporal Decomposition

For the temporal decomposition purpose, we need a learner to project each load profile lp to its optimal decomposition class cl^*_{lp}. The input to this learner is a time-varying demand vector, and the output is an integer, which is limited amongst the divisors of the considered scheduling horizon. This learning procedure can be structured by a multi-class classification learner.

Selecting Learner: Various algorithms exist to form a multi-class classifier, among which tree-based algorithms are widely used as their results can be interpreted properly. Decision tree, random forest, and Extreme Gradient Boosting (XGBoost) are among the most popular and efficient tree-based algorithms [87]. Classification trees are made of nodes, which separates the data based on

some impurity criteria, and leaves that determine classes. In the decision tree, each node is selected based on a characteristic that provides the best split with the least impurity [87]. To select a certain characteristic to split a node, the information gain by splitting on that node is calculated and the split with the largest gain is made [88]. In the random forest, a random number of characteristics is selected at each step, and different decision trees are made based on those characteristics. A class receiving the most votes from the decision trees is determined as the final class. Figure 5.6 shows a forest with three decision trees. Black rectangles refer to leaves, and white rectangles denote nodes. In random forest, trees are independent of each other; however, in XGBoost, trees are made based on regression and the predicted value is updated after each regression tree is made until a suitable prediction is made. XGBoost is a combination of gradient boosting, regularization, unique regression trees, approximate greedy algorithm, weighted quantile sketch, sparsity-aware split finding, parallel learning, cache-award access, and blocks for out-of-core computation. [89-94] As compared to other gradient boosting methods, XGBoost has two additional features to prevent over-fitting. The weights of a new tree can be scaled down by a given constant to reduce the impact of a single tree on the final score and provide an opportunity for the next trees to improve the model. Furthermore, a column sampling that builds each tree using only a column-wise sample from the training dataset. XGBoost also performs better than other tree boosting methods. This is mainly because it supports an approximate split finding, which improves the process of building trees and scales well with the number of CPU cores [95]. Thus, we have selected XGBoost as it, in general, provides better performance as compared to not only the decision tree and random forest but also many other multi-class calcification methods [96-99].

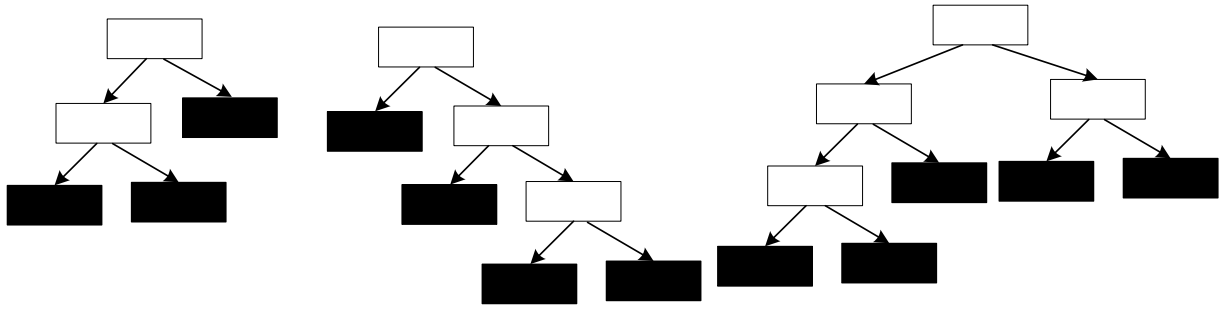


Figure 5.6. A forest of three independent decision trees.

XGBoost Model for Load Profile Classification: Assuming M and K denote, respectively, the number of rounds of boosting and the number of trees. A total number of $M \times K$ decision trees are generated. The XGBoost algorithm pre-sorts attributes and greedily finds the split point with the largest information gain [100]. Assume a dataset $\mathcal{D} = \{(P_{lp}, \varphi^*_{lp}): lp = 1, \ldots, N_s\}$ where P_{lp} is the demand vector lp over the scheduling horizon whose corresponding optimal decomposition class, selected based on (3), is φ^*_{lp}. We define $\widehat{\varphi^*}$ as predicted classes by the learner.

$$\widehat{\varphi^*_{lp}} = \psi(P_{lp}) = \sum_{k=1}^{K} f_k(P_{lp}) \qquad (4)$$

$f_k(\cdot)$ is a regression tree, and $f_k(P_{lp})$ represents the score given by the kth tree to lpth observation. If the regression tree f_k for $k = 1, \ldots, K$ is achieved, expression (4) will provide the predicted temporal decomposition class. Thus, the goal of training the learner is to find the optimal regression trees (denoted by $f^*_k(P_{lp})\ \forall k$) that minimize the following regularized objective function.

$$L(\psi) = \sum_{lp} l(\varphi^*_{lp}, \widehat{\varphi^*_{lp}}) + \sum_{k} \Omega(f_k) \qquad (5)$$

where $l(\cdot)$ is the loss function quantifying how good is the prediction. XGBoost uses this loss function to build trees by minimizing $L(\psi)$. The penalty function Ω is added to prevent over-fitting.

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \|f_k\|^2 \qquad (6)$$

where γ and λ control penalty for the number of terminal nodes or leaves (T). Parameter γ encourages pruning trees. The second term in (6) is a regularization term. This penalty function, which simplifies models produced by the learner, is one of the most important features of XGBoost in comparison to other tree boosting methods.

An iterative method is used to minimize the objective function. At iteration j, XGBoost builds a tree by finding the value of f_j that minimizes the following objective function:

$$L^j = \sum_{lp=1}^{N_s} l(\varphi_{lp}^*, \hat{\varphi}_{lp}^{*(j-1)} + f_j(P_{lp})) + \Omega(f_j) \qquad (7)$$

Using the second-order Taylor expansion, we can simplify this function and derive a formula for loss reduction when solving for the optimal value after the tree split from a given node.

$$L^j = \sum_{lp=1}^{N_s} l(\varphi_{lp}^*, \hat{\varphi}_{lp}^{*(j-1)}) + g_{lp} f_j(P_{lp}) + \frac{1}{2} h_{lp\,j} f_j^2(P_{lp}) + \Omega(f_j) \qquad (8)$$

After getting derivatives and finding the root and several simplifications, the desired output value becomes as follows:

$$f_k^*(P_{lp}) = -\frac{1}{2}\sum_{k=1}^{K} \frac{\left(\sum_{lp\in LP} g_{lp}\right)^2}{\sum_{lp\in LP} h_{lp} + \lambda} + \gamma T \qquad (9)$$

Expression (10) is used to measure the information gain of candidate splits.

$$L_{split} = \frac{1}{2}\left[\frac{(\sum_{lp\in\Omega_{lp}^{L}} g_{lp})^2}{\sum_{lp\in\Omega_{lp}^{L}} h_{lp}+\lambda} + \frac{(\sum_{lp\in\Omega_{lp}^{R}} g_{lp})^2}{\sum_{lp\in\Omega_{lp}^{R}} h_{lp}+\lambda} - \frac{(\sum_{lp\in\Omega_{lp}} g_{lp})^2}{\sum_{lp\in\Omega_{lp}} h_{lp}+\lambda}\right] - \gamma \qquad (10)$$

where Ω_{lp} is the set of load profiles in the current node, and the sets of observations available in the left and right nodes after the split are denoted by Ω_{lp}^{L} and Ω_{lp}^{R}, respectively. The functions g_{lp} and h_{lp} are defined as follows:

$$g_{lp} = \partial_{\hat{\varphi}_{lp}^{*(j-1)}} l\left(\varphi_{lp}^{*}, \hat{\varphi}_{lp}^{*(j-1)}\right) \qquad (11)$$

$$h_{lp} = \partial^2_{\hat{\varphi}_{lp}^{*(j-1)}} l\left(\varphi_{lp}^{*}, \hat{\varphi}_{lp}^{*(j-1)}\right) \qquad (12)$$

Equations (11) and (12) are used for finding the best split at any given node, which is the split with the largest information gain.

Following these steps, the training goal, which was to make $f_k^*(P_{lp})\ \forall k$, is achieve. A user can put these optimal regression trees in (4) to predict the temporal decomposition class. The trained XGBoost can be used as shown in Fig. 5.7. The learner reads the load profile and decomposes the scheduling horizons into the optimal number of subhorizons (SH).

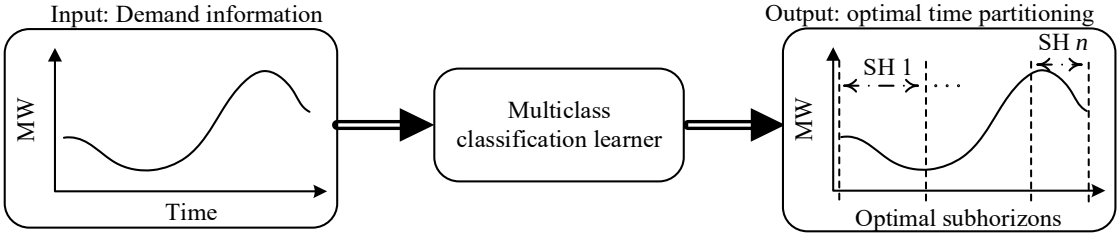


Figure 5.7. Utilizing procedure of training XGBoost learner.

5.5. Numerical Results

The proposed algorithm is tested on the IEEE 118-bus test system. Simulations are carried out on a personal computer with a 3.70 GHz Intel(R) Xeon(R) CPU and 16 GB of RAM. The Yalmip

toolbox in Matlab and Gurobi are used to solve optimization problems, and Python learning toolboxes XGBoost and sklearn are used to train multi-class classification learners [90, 101].

5.5.1. Dataset Preparation

The considered scheduling problem is a week-ahead economic dispatch. We have extracted 6669 weekly load patterns from PJM to study realistic demand patterns [104]. These load profiles are normalized according to the IEEE 118-bus system baseload. Figure 5.8 shows some samples of load profiles. All divisors of 168, which is the length of scheduling horizon, are considered as the possible classes that a load profile can belong to. For each load profile and each divisor, the distributed algorithm is implemented. For each load profile, the best divisor that provides φ_{lp}^{*} is assigned to the load profile as its decomposition class. This offline procedure provides train and test datasets.

After classifying the training data, we have observed that all load profiles belong to only [67 8, 12] classes of divisors of 168. No load profile is assigned to other divisors. Thus, we keep these five classes and drop others as they have no useful information in the learning process. We define the decomposition classes as $cl_1 = 2$, $cl_2 = 4$, $cl_3 = 7$, $cl_4 = 8$, and $cl_5 = 12$. $cl_2 = 4$, for instance, means that the decomposition class two includes load profiles with the optimal number of subhorizons equals to 4.

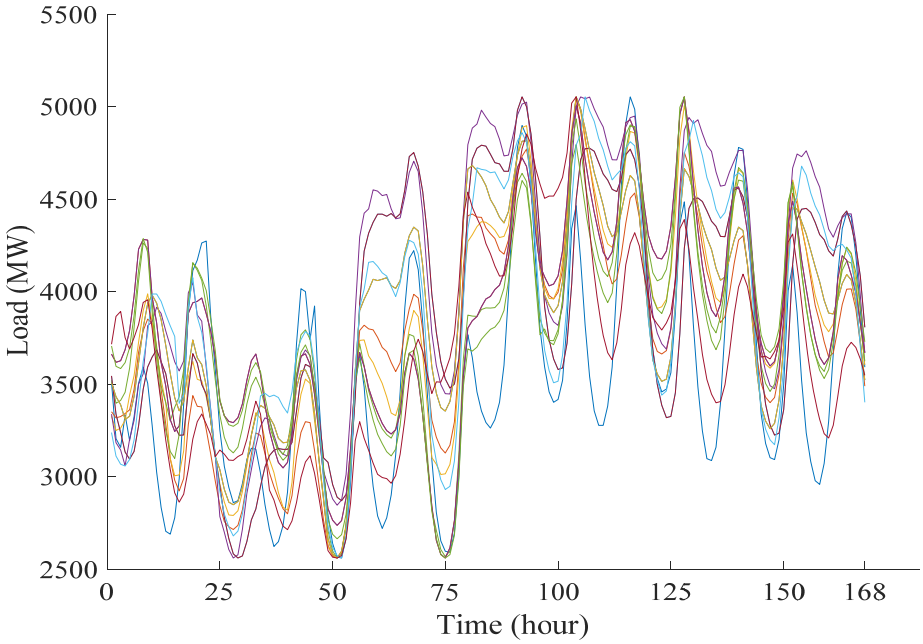


Figure 5.8. Some samples of PJM load profiles being normalized and used for the IEEE 118-bus system [104].

5.5.2. Temporal Decomposition Classifier Training

80% of the dataset is used for training. The softmax objective is used for XGBoost. Softmax is used to normalize the probability distribution of predicted output classes in a way that the sum of all probabilities becomes one. The maximum tree depth D, the number of trees K, and regularization parameters such as learning rate L, γ, and λ need to be tuned to train XGBoost. A combination of ten-fold and grid search techniques is applied to tune these hyper-parameters [89]. Grid search, which is an exhaustive search, determines the optimal combination of hyper-parameters values. Parameters γ and λ are set to one while tuning other hyper-parameters. We have set the parameter D, which controls the sequential process of growing trees, in a rage of $D \in [1, 2, \dots, 6]$. Please note that the default is not to exceed the depth of trees more than 6. The role of L is to avoid overfitting by decreasing the contribution of each successive tree. Parameter L should be in the range of $0 < L < 1$. We have tested various values as $L \in$

$[0.01, 0.02, 0.03, 0.05, 0.1, 0.5]$. The accuracy of the learner increases by increasing T; however, it may cause overfitting [89]. We have tested various values for this parameter as $K \in [1, 2, \ldots, 10]$. The search learning rate values and the varying step size have been determined based on sensitivity analysis and preliminary investigation with different values. Table 5.1 shows the learner's accuracy for different parameters. We have found the best combination for hyper-parameters as $L = 0.5$, $D = 6$, and $K = 6$.

Table 5.1. Accuracy of Parameter Tuning

Accuracy	D	K	L	Rounds
0.566	1	4	0.1	1
0.849	2	4	0.1	1
0.934	3	4	0.1	1
0.983	4	4	0.1	1
0.980	5	4	0.1	1
0.980	6	4	0.1	1
0.982	4	1	0.1	1
0.982	4	10	0.1	1
0.982	4	4	0.01	1
0.982	4	4	1	1
0.985	4	4	0.1	2
0.991	4	4	0.1	3
0.991	4	4	0.1	4
0.992	6	6	0.5	5

5.5.3. Evaluating Temporal Decomposition Classifier

20% of the dataset is used for testing the multi-class classification learner. The following four primary indices are introduced for each decomposition class to interpret predicted results and analyze the accuracy of the proposed learning-based temporal decomposition method.

- *True positives* (TP): a load profile lp is predicted to belong to a decomposition class cl and its actual class is cl.
- *True negatives* (TN): a load profile lp is predicted to not belong to a decomposition class cl and its actual class is not cl.
- *False positives* (FP): a load profile lp is predicted to belong to a decomposition class cl, but its actual class is not cl.
- *False negatives* (FN): a load profile lp is predicted to not belong to a decomposition class cl, but and its actual class is cl.

First, we use classification precision and recall metrics to analyze the quality of the classification leaner [102].

Precision is defined as the fraction of true positives out of total instances predicted as positives.

$$Precision_{cl} = \frac{TP_{cl}}{TP_{cl} + FP_{cl}} \tag{13}$$

Recall is defined as the fraction of instances belonging to positive classes that were predicted as positives.

$$Recall_{cl} = \frac{TP_{cl}}{TP_{cl} + FN_{cl}} \tag{14}$$

The confusion matrix is represented in Fig. 5.9. Green blocks show the number of load profiles that belong to a decomposition class and are predicted correctly. Orang blocks depict the number of load profiles that are misclassified. As an example, 668 load profiles in the test dataset belong

to cl_1. 664 (99.4%) of them are predicted correctly, and only four (0.6%) load profiles are misclassified.

	Actual cl_1 2 SHs	Actual cl_2 4 SHs	Actual cl_3 7 SHs	Actual cl_4 8 SHs	Actual cl_5 12 SHs	Total predict	$Precision_{cl}$
Predicted cl_1	664	0	0	2	3	669	99.253%
Predicted cl_2	3	199	0	0	0	202	98.515%
Predicted cl_3	0	0	145	0	0	145	100%
Predicted cl_4	1	2	0	194	0	197	98.477%
Predicted cl_5	0	0	0	0	121	121	100%
Total actual	668	201	145	196	124	1334	
$Recall_{cl}$	99.40%	99.01%	100%	98.98%	97.5%		

Figure 5.9. Confusion matric (SH: subhorizon).

Then, we have calculated the overall classification accuracy, precision, recall, and F_1 score metrics for all classes combined. The overall TP and TN are the summations of all correctly classified load patterns, regardless of their classes. The overall FP and FN refer to the overall incorrectly predicted load patterns. In fact, the accuracy is proportion of the correct predictions over all predictions. In addition, the average of precision and recall of all classes is considered as overall precision and recall:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (15)$$

The F_1 score is the harmonic mean of precision and recall, defined as:

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall} \qquad (16)$$

The values of the four metrics are depicted in Table 5.2. The overall accuracy is more than 99%. The confusion matric and overall indices prove the promising performance of the proposed learning-based temporal decomposition algorithm.

Table 5.2. Evaluation of Overall Performance of Learner

Index	Value
Accuracy	0.992
F_1 score	0.991
precision	0.990
recall	0.992

5.5.4. Distributed Optimization Results

We have randomly selected a load profile shown in Fig. 5.10. The operation cost obtained by centralized economic dispatch is \$9,050,971. The relative error for all decomposition classes is less than $2e-6$. We choose $\omega_1 = 1e6$ and $\omega_2 = 1$. The classification leaner is carried out. Table 5.3 illustrates the relative error, solution time, and φ_{cl} for each decomposition class. The solver time versus the number of subhorizons is plotted in Fig. 5.11. The error-time index for this given load profile is $\varphi_{lp}^{*} = 0.7327$. Thus, the best strategy is to decompose the considered scheduling horizon into eight subproblems. The predicted decomposition class by the learner is also cl_4.

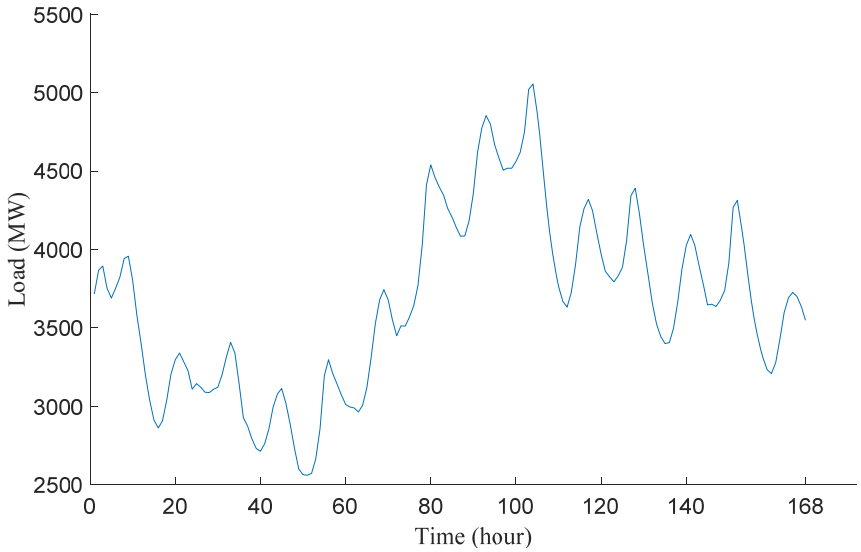


Figure 5.10. A load pattern used to test the algorithm [104].

Table 5.3. Relative Error, solver time, and Error-Time Indices for Decomposition Classes Corresponding to Load Profile in Fig. 5.10

Number of SPs	rel	Solver time (s)	φ_{cl}
1	-	1.4363	1.4363
2	2e-11	1.5626	1.5626
3	2e-09	1.5647	1.5649
4	2e-11	1.1307	1.1307
6	2e-09	0.9173	0.9175
7	2e-08	0.8118	0.8138
8	3e-11	0.7327	0.7327
12	3e-08	1.2197	1.2497
24	6e-07	0.994	1.594
28	2e-06	0.9599	2.9599
42	2e-07	2.6136	2.8136
56	1e-07	3.1102	3.2102
84	5e-07	4.4298	4.5298

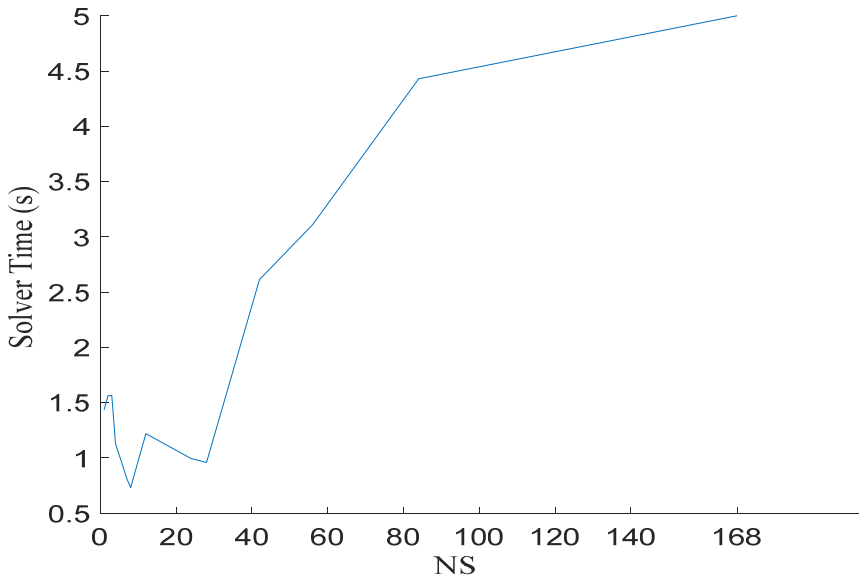


Figure 5.11. Solver time versus number of subhorizons for the sample load profile in Fig. 5.10.

5.6. Conclusion

In this chapter, we have focused on how to decompose the overall time horizon and to find the optimal number of subhorizons. We have observed that load values and profile patterns have a significant impact on the optimal number of subhorizons. The load is predictable and is known before solving the system scheduling problem. Possible load patterns have been decomposed into a various number of subhorizons, and distributed optimization has been solved. Each load profile has been labeled with its corresponding best decomposition class that results in the best time-saving and relative error. We have trained a multi-class classification learner, which is based on XGBoost, whose goal is to read the load data as the input and project it to the corresponding optimal decomposition class.

The simulation studies on the IEEE 118-bus system using read-word load patterns show that the proposed algorithm can efficiently and quickly find the optimal number of subproblems. More than 99% of the cases are predicted correctly, and using the optimal number of subproblems reduces the solution time more significantly as compared to a naïve decomposition. Another

observation is that for a given system with a set of plausible load patterns, no load profile is assigned to many of possible decomposition classes. Hence, they can be dropped from the training dataset.

CHAPTER 6
CONCLUDING REMARKS AND FUTURE WORK

6.1. Concluding Remarks

In this dissertation, several models, methods, and algorithms have been developed for temporal decomposition and distributed optimization for power systems multi-interval problems, with the main focus on economic dispatch and unit commitment.

A time partitioning strategy is proposed in Chapter 2 to divide the ramp-constrained SCED problem. In this chapter, we have introduced the concept of overlapping time intervals to model interdependencies, which originate from intertemporal constraints. For SCED, intertemporal constraints are ramping limitations of generating units. An accelerated APP and an initialization strategy are proposed to enhance the convergence performance of the distributed SCED algorithm. The simulation results show that the proposed algorithm reduces the computation time of SCED. The results show that the larger the size of the optimization problem is, the more time-saving is obtained using the proposed distributed SCED compared to the centralized method. For example, The proposed algorithm reduces the computation time of a simple SCED for a day-ahead 4720-bus system by 74%.

In Chapter 3, the algorithm presented in Chapter 2 is extended to decompose a more realistic, sophisticated SCED problem. The studied network is a smart grid that includes energy storage, thermal generators, renewables sources, and considers demand response and springing reserve up and down requirements. Data-driven nonparametric chance constraints are formulated for reserve up and down to model the uncertainty of wind generation. These constraints are replaced by their equivalent parametric constraints to make SCED solvable by standard solvers. Consistency constraints include generators' ramp limits and the state of charge of storage, and the problem

requires a larger set of shared variables. It is discussed that the initial state of charge of storage at the beginning of each subproblem should also be shared with the state of charge of storage at one interval before the overlapping interval. A tutorial is presented based on a small 6-bus system, and the results for a 472-bus system show that the SCED solution time is reduced by 53% as compared to that for the centralized SCED and a more optimal result is obtained. As the size of the optimization problem increases, the time saving is more considerable. In addition, it is illustrated that data-driven nonparametric chance constraints provide an accurate prediction for probability distributions of wind in each time interval.

In Chapter 4, the temporal decomposition strategy is extended to a SCUC problem. The SCUC problem includes binary variables for on/off status of generating units, and consistency constraints include minimum on/off status of generating units as well as ramping. The complexity is to model minimum on/off time constraints since the number of overlapping time intervals is not known before solving the SUC problem. To solve this challenge, several counting auxiliary variables are defined to coordinate consecutive subproblems. An accelerated ATC algorithm with an initialization strategy is proposed to coordinate subproblems. The simulation results show that the proposed algorithm obtains the same SCUC results as the traditional centralized SCUC while reducing the computation time considerably. For example, for the IEEE 118-bus system, the solution time decreases by 94%. The results also show the privilege of A-ATC over ATC to improve convergence performance.

In Chapter 5, we focus on how to decompose the overall time horizon and to find the optimal number of subhorizons for the SCED problem. The possibility of having equal length subhorizons is studied to reduce idle time. We observed that load patterns have a considerable impact on selecting optimal partitions. The load is known before solving the SCED problem. Possible load

patterns have been decomposed into the possible number of subhorizons, and distributed optimization is solved. Each load profile has been labeled with its decomposition class that is selected based on the most time-saving and the smallest relative error. We trained a multi-class classification learner, which is based on XGBoost, whose goal is to read the load data as the input and map it to the optimal decomposition class. The simulation studies on the IEEE 118-bus system using real-word load patterns show that the proposed algorithm can efficiently and quickly find the optimal number of subproblems. More than 99% of the cases are predicted correctly. Finding the optimal number of subproblems reduces the solution time more significantly as compared to a naïve decomposition.

6.2. Future Work

Following suggestions can be considered as extensions of the proposed methods:

- Temporal decomposition should be developed for a more realistic, sophisticated SCUC model taking into account renewable uncertainties, demand response, and storage systems requirements.
- The proposed learning-based optimal decomposition can be extended to time decomposition for SCUC, which not only contains ramping limits but also includes minimum on/off time constraints of generating units
- The possibility of having unequal-sized subproblems can be studied, and asynchronous distributed algorithms can be applied to coordinate subproblems and avoid a considerable CPU idle time.
- A direction of research is to combine reserve up (or down) requirements at all intervals in a probabilistic constraint and develop a nonparametric joint chance constraint model.
- Second-order derivative information can be used to enhance the convergence

performance of ATC and APP coordination strategies.

- Temporal decomposition can be combined with other decompositions strategies, such as geographical decomposition, to create more complex, efficient hybrid decomposition strategies.
- Data-driven joint chance constraints can be developed for modeling reserve up/down requirements considering temporal correlations between reserve and power produced by suppliers.

APPENDIX. PROOF FOR CONVERGENCE OF AATC

In this Appendix, we discuss the convergence of A-ATC. Consider the following problem:

$$\min F_1(r_1) + F_2(r_2) \qquad \text{(a1)}$$

$$s.t. \quad r_1 = r_2$$

Assume that F_1 and F_2 are strongly convex. In the normal parallel ATC, a coordinator exists to coordinate the two subproblems. Subproblems one and two send updated r_1 and r_2 to the coordinator and receive χ. Consider the coordinator's optimization problem as $G(\chi)$ and $H(r) = F_1(r_1) + F_2(r_2)$. We restate the problem (a1) as:

$$\min H(r) + G(\chi) \qquad \text{(a2)}$$

$$s.t. \quad \chi = r$$

Then:

$$r^+ = \text{argmin}\, H(r) + \lambda(\chi - r) + \rho^2(\chi - r)^2 \qquad \text{(a3)}$$

$$\chi^+ = \text{argmin}\, G(\chi) + \lambda(\chi^+ - r) + \rho^2(\chi^+ - r)^2 \qquad \text{(a4)}$$

$$\lambda^+ = \lambda + 2\rho^2\,(\chi^+ - r^+) \qquad \text{(a5)}$$

where $\chi^+ = \chi^{k+1}$ and $r^+ = r^{k+1}$. We write the dual conjugate function of (a2) as:

$$\max D(\lambda) = -H^*(-\lambda) - G^*(\lambda) \qquad \text{(a6)}$$

Introducing $\lambda^{1/2} = \lambda + 2\rho^2(\chi^+ - r)$ and $\lambda^+ = \lambda + 2\rho^2(\chi^+ - r^+)$, and using optimality conditions, we calculate r^+ and χ^+ as follows:

$$\partial H(r^+) = -\lambda^{1/2}, r^+ = \nabla H^*\left(-\lambda^{\frac{1}{2}}\right) \qquad \text{(a7)}$$

$$\partial G(\chi^+) = +\lambda^+, \chi^+ = \nabla G^*(\lambda^+) \qquad \text{(a8)}$$

To calculate the convergence rate of the parallel ATC, the relation between $(\lambda^*) - D(\lambda^+)$ and λ should be determined. According to (a6):

$$D(\lambda^*) - D(\lambda^+) = -H^*(-\lambda^*) - G^*(\lambda^*) + H^*(-\lambda^+) + G^*(\lambda^+) \quad \text{(a9)}$$

We rewrite the term $H^*(\lambda^*) - H^*(-\lambda^+)$ in (a9) as follows:

$$H^*(\lambda^*) - H^*(-\lambda^+) = H^*(\lambda^*) - H^*\left(-\lambda^{\frac{1}{2}}\right) + H^*\left(-\lambda^{\frac{1}{2}}\right) - H^*(-\lambda^+) \quad \text{(a10)}$$

After some calculations and using convex properties:

$$H^*(\lambda^*) - H^*(-\lambda^+) \geq (\lambda^* - \lambda^+)\left(\nabla H^*\left(-\lambda^{\frac{1}{2}}\right)\right) - \frac{1}{2\rho^2}(\|\lambda^+ - \lambda\|) \quad \text{(a11)}$$

Similarly, for the term $G^*(\lambda^*) - G^*(-\lambda^+)$ in (a9):

$$G^*(\lambda^*) - G^*(-\lambda^+) \geq (\lambda^* - \lambda^+)\left(\nabla G^*(\lambda^+)\right) \quad \text{(a12)}$$

By incorporating (a10) and (a11) into (a9), we then have:

$$D(\lambda^+) - D(\lambda^*) \geq \frac{1}{\rho}(\lambda^* - \lambda)(\lambda - \lambda^+) + \frac{1}{2\rho}\|\lambda^+ - \lambda\|^2 \quad \text{(a13)}$$

To convert (a3)- (a5) to the accelerated ATC, we replace $\lambda = \hat{\lambda}^+$ and $\chi = \hat{\chi}^+$.

$$D(\lambda^+) - D(\lambda^*) \geq \frac{1}{\rho}\left(\hat{\lambda} - \lambda^*\right)\left(\lambda^+ - \hat{\lambda}^+\right) + \frac{1}{2\rho}\left\|\lambda^+ - \hat{\lambda}^+\right\|^2 \text{(a14)}$$

Using the telescopic summation and after some simplifications:

$$D(\lambda^*) - D(\lambda^+) \leq \frac{2\left\|\hat{\lambda}_1 - \lambda^*\right\|}{\rho^2(k+2)^2} \quad \text{(a15)}$$

This proves the convergence of A-ATC whose convergence rate is $O(1/k^2)$.

REFERENCES

[1] R. Rios-Zalapa, X. Wang, J. Wan, and K. Cheung, "Robust dispatch to manage uncertainty in real time electricity markets," in *Innovative Smart Grid Technologies (ISGT), 2010*, 2010: IEEE, pp. 1-5.

[2] C.-N. Yu, A. I. Cohen, and B. Danai, "Multi-interval optimization for real-time power system scheduling in the ontario electricity market," in *Power Engineering Society General Meeting, 2005. IEEE*, 2005: IEEE, pp. 2717-2723.

[3] J. Zhu, "Security-Constrained Economic Dispatch," *Optimization of Power System Operation,* pp. 141-210, 2009.

[4] A. J. Wood and B. F. Wollenberg, *Power generation, operation, and control*. John Wiley & Sons, 2012.

[5] M. F. Anjos and A. J. Conejo, "Unit commitment in electric energy systems," *Foundations and Trends® in Electric Energy Systems,* vol. 1, no. 4, pp. 220-310, 2017.

[6] M. H. Amini *et al.*, "Decomposition Methods for Distributed Optimal Power Flow: Panorama and Case Studies of the DC Model," in *Classical and Recent Aspects of Power System Optimization*: Elsevier, 2018, pp. 137-155.

[7] A. Kargarian *et al.*, "Toward distributed/decentralized DC optimal power flow implementation in future electric power systems," *IEEE Transactions on Smart Grid,* vol. 9, no. 4, pp. 2574-2594, 2018.

[8] M. B. Cain, R. P. O'neill, and A. Castillo, "History of optimal power flow and formulations," *Federal Energy Regulatory Commission,* pp. 1-36, 2012.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning,* vol. 3, no. 1, pp. 1-122, 2011.

[10] A. J. Conejo, F. J. Nogales, and F. J. Prieto, "A decomposition procedure based on approximate Newton directions," *Mathematical programming,* vol. 93, no. 3, pp. 495-515, 2002.

[11] G. Cohen, "Auxiliary problem principle and decomposition of optimization problems," *Journal of optimization Theory and Applications,* vol. 32, no. 3, pp. 277-305, 1980.

[12] A. R. Malekpour and A. Pahwa, "Stochastic networked microgrid energy management with correlated wind generators," *IEEE Transactions on Power Systems,* vol. 32, no. 5, pp. 3681-3693, 2017.

[13] H. M. Kim, N. F. Michelena, P. Y. Papalambros, and T. Jiang, "Target cascading in optimal system design," *Journal of mechanical design,* vol. 125, no. 3, pp. 474-480, 2003.

[14] D. P. Palomar and M. Chiang, "A tutorial on decomposition methods for network utility maximization," *IEEE Journal on Selected Areas in Communications,* vol. 24, no. 8, pp. 1439-1451, 2006.

[15] D. K. Molzahn *et al.*, "A survey of distributed optimization and control algorithms for electric power systems," *IEEE Transactions on Smart Grid,* vol. 8, no. 6, pp. 2941-2962, 2017.

[16] S. Kar, G. Hug, J. Mohammadi, and J. M. Moura, "Distributed State Estimation and Energy Management in Smart Grids: A Consensus ${+} $ Innovations Approach," *IEEE Journal of selected topics in signal processing,* vol. 8, no. 6, pp. 1022-1038, 2014.

[17] J.-H. Liu and C.-C. Chu, "Iterative distributed algorithms for real-time available transfer capability assessment of multiarea power systems," *IEEE Transactions on Smart Grid,* vol. 6, no. 5, pp. 2569-2578, 2015.

[18] R. Baldick, B. H. Kim, C. Chase, and Y. Luo, "A fast distributed implementation of optimal power flow," *IEEE Transactions on Power Systems,* vol. 14, no. 3, pp. 858-864, 1999.

[19] D. Hur, J.-K. Park, and B. H. Kim, "Evaluation of convergence rate in the auxiliary problem principle for distributed optimal power flow," *IEE Proceedings-Generation, Transmission and Distribution,* vol. 149, no. 5, pp. 525-532, 2002.

[20] J. Hu, M. Z. Chen, J. Cao, and J. M. Guerrero, "Coordinated active power dispatch for a microgrid via distributed lambda iteration," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems,* vol. 7, no. 2, pp. 250-261, 2017.

[21] A. Cherukuri and J. Cortés, "Distributed generator coordination for initialization and anytime optimization in economic dispatch," *IEEE Transactions on Control of Network Systems,* vol. 2, no. 3, pp. 226-237, 2015.

[22] M. J. Feizollahi, M. Costley, S. Ahmed, and S. Grijalva, "Large-scale decentralized unit commitment," *International Journal of Electrical Power & Energy Systems,* vol. 73, pp. 97-106, 2015.

[23] F. Guo, C. Wen, J. Mao, J. Chen, and Y.-D. Song, "Hierarchical Decentralized Optimization Architecture for Economic Dispatch: A New Approach for Large-Scale Power System," *IEEE Transactions on Industrial Informatics,* vol. 14, no. 2, pp. 523-534, 2018.

[24] C. Lin, W. Wu, X. Chen, and W. Zheng, "Decentralized dynamic economic dispatch for integrated transmission and active distribution networks using multi-parametric programming," *IEEE Transactions on Smart Grid,* vol. 9, no. 5, pp. 4983-4993, 2018.

[25] M. H. Amini, R. Jaddivada, S. Mishra, and O. Karabasoglu, "Distributed security constrained economic dispatch," in *Innovative Smart Grid Technologies-Asia (ISGT ASIA), 2015 IEEE*, 2015: IEEE, pp. 1-6.

[26] T. Xu, W. Wu, H. Sun, and L. Wang, "Fully distributed multi-area dynamic economic dispatch method with second-order convergence for active distribution networks," *IET Generation, Transmission & Distribution,* vol. 11, no. 16, pp. 3955-3965, 2017.

[27] N. DeForest, J. S. MacDonald, and D. R. Black, "Day ahead optimization of an electric vehicle fleet providing ancillary services in the Los Angeles Air Force Base vehicle-to-grid demonstration," *Applied Energy,* vol. 210, pp. 987-1001, 2018.

[28] H. Xing, Y. Mou, M. Fu, and Z. Lin, "Distributed bisection method for economic power dispatch in smart grid," *IEEE Transactions on power systems,* vol. 30, no. 6, pp. 3024-3035, 2015.

[29] A. Asrari, S. Lotfifard, and M. Ansari, "Reconfiguration of smart distribution systems with time varying loads using parallel computing," *IEEE Transactions on Smart Grid,* vol. 7, no. 6, pp. 2713-2723, 2016.

[30] V. Loia and A. Vaccaro, "Decentralized economic dispatch in smart grids by self-organizing dynamic agents," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* vol. 44, no. 4, pp. 397-408, 2013.

[31] Y. Zhang and G. B. Giannakis, "Efficient decentralized economic dispatch for microgrids with wind power integration," in *2014 Sixth Annual IEEE Green Technologies Conference*, 2014: IEEE, pp. 7-12.

[32] C. Lin, W. Wu, X. Chen, and W. Zheng, "Decentralized dynamic economic dispatch for integrated transmission and active distribution networks using multi-parametric programming," *IEEE Transactions on Smart Grid,* vol. 9, no. 5, pp. 4983-4993, 2017.

[33] Q. Li, D. W. Gao, H. Zhang, Z. Wu, and F.-y. Wang, "Consensus-based distributed economic dispatch control method in power systems," *IEEE Transactions on Smart Grid,* vol. 10, no. 1, pp. 941-954, 2017.

[34] W. T. Elsayed and E. F. El-Saadany, "A fully decentralized approach for solving the economic dispatch problem," *IEEE Transactions on power systems,* vol. 30, no. 4, pp. 2179-2189, 2014.

[35] F. Guo, C. Wen, J. Mao, and Y.-D. Song, "Distributed economic dispatch for smart grids with random wind power," *IEEE Transactions on Smart Grid,* vol. 7, no. 3, pp. 1572-1583, 2015.

[36] S. Yang, S. Tan, and J.-X. Xu, "Consensus based approach for economic dispatch problem in a smart grid," *IEEE Transactions on Power Systems,* vol. 28, no. 4, pp. 4416-4426, 2013.

[37] I. U. Nutkani, P. C. Loh, P. Wang, and F. Blaabjerg, "Decentralized economic dispatch scheme with online power reserve for microgrids," *IEEE Transactions on Smart Grid,* vol. 8, no. 1, pp. 139-148, 2015.

[38] Y. Zhang, Y. Sun, X. Wu, D. Sidorov, and D. Panasetsky, "Economic Dispatch in Smart Grid Based on Fully Distributed Consensus Algorithm with Time Delay," in *2018 37th Chinese Control Conference (CCC)*, 2018: IEEE, pp. 2442-2446.

[39] A. Charnes and W. W. Cooper, "Chance-constrained programming," *Management science,* vol. 6, no. 1, pp. 73-79, 1959.

[40] D. E. Olivares, J. D. Lara, C. A. Cañizares, and M. Kazerani, "Stochastic-predictive energy management system for isolated microgrids," *IEEE Transactions on Smart Grid,* vol. 6, no. 6, pp. 2681-2693, 2015.

[41] Y. Zhang, N. Gatsis, and G. B. Giannakis, "Robust energy management for microgrids with high-penetration renewables," *IEEE Transactions on Sustainable Energy,* vol. 4, no. 4, pp. 944-953, 2013.

[42] W. Wei, F. Liu, and S. Mei, "Energy pricing and dispatch for smart grid retailers under demand response and market price uncertainty," *IEEE transactions on smart grid,* vol. 6, no. 3, pp. 1364-1374, 2014.

[43] A. Shapiro and A. Philpott, "A tutorial on stochastic programming," *Manuscript. Available at www2. isye. gatech. edu/ashapiro/publications. html,* vol. 17, 2007.

[44] A. J. Conejo, M. Carrión, and J. M. Morales, *Decision making under uncertainty in electricity markets*. Springer, 2010.

[45] A. Lorca and X. A. Sun, "Adaptive robust optimization with dynamic uncertainty sets for multi-period economic dispatch under significant wind," *IEEE Transactions on Power Systems,* vol. 30, no. 4, pp. 1702-1713, 2014.

[46] J. Yu, Z. Li, Y. Guo, and H. Sun, "Decentralized Chance-Constrained Economic Dispatch for Integrated Transmission-District Energy Systems," *IEEE Transactions on Smart Grid,* 2019.

[47] O. Ciftci, M. Mehrtash, and A. K. Marvasti, "Data-Driven Nonparametric Chance-Constrained Optimization for Microgrid Energy Management," *IEEE Transactions on Industrial Informatics,* 2019.

[48] J.-P. Watson and D. L. Woodruff, "Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems," *Computational Management Science,* vol. 8, no. 4, pp. 355-370, 2011.

[49] G. Cong, C. Meyers, D. Rajan, and T. Parriani, "Parallel strategies for solving large unit commitment problems in the California ISO planning model," in *2015 IEEE International Parallel and Distributed Processing Symposium*, 2015: IEEE, pp. 710-719.

[50] I. Aravena and A. Papavasiliou, "A distributed asynchronous algorithm for the two-stage stochastic unit commitment problem," in *2015 IEEE Power & Energy Society General Meeting*, 2015: IEEE, pp. 1-5.

[51] M. El-Sharkh, M. Tanrioven, A. Rahman, and M. Alam, "A study of cost-optimized operation of a grid-parallel PEM fuel cell power plant," *IEEE Transactions on Power Systems,* vol. 21, no. 3, pp. 1104-1114, 2006.

[52] A. Ahmadi-Khatir, A. J. Conejo, and R. Cherkaoui, "Multi-area energy and reserve dispatch under wind uncertainty and equipment failures," *IEEE Transactions on Power Systems,* vol. 28, no. 4, pp. 4373-4383, 2013.

[53] A. Kargarian, Y. Fu, and Z. Li, "Distributed security-constrained unit commitment for large-scale power systems," *IEEE Transactions on Power Systems,* vol. 30, no. 4, pp. 1925-1936, 2015.

[54] A. Kargarian, M. Mehrtash, and B. Falahati, "Decentralized Implementation of Unit Commitment with Analytical Target Cascading: A Parallel Approach," *IEEE Transactions on Power Systems,* vol. 33, no. 4, pp. 3981-3993, 2018.

[55] F. Safdarian, O. Ciftci, and A. Kargarian, "A time decomposition and coordination strategy for power system multi-interval operation," in *2018 IEEE Power & Energy Society General Meeting (PESGM)*, 2018: IEEE, pp. 1-5.

[56] E. Alpaydin, *Introduction to machine learning*. MIT press, 2020.

[57] F. Hasan, A. Kargarian, and A. Mohammadi, "A Survey on Applications of Machine Learning for Optimal Power Flow," in *2020 IEEE Texas Power and Energy Conference (TPEC)*, 2020: IEEE, pp. 1-6.

[58] F. Hasan and A. Kargarian, "Combined Learning and Analytical Model Based Early Warning Algorithm for Real-Time Congestion Management," in *2020 IEEE Texas Power and Energy Conference (TPEC)*, 2020: IEEE, pp. 1-6.

[59] D. Deka and S. Misra, "Learning for DC-OPF: Classifying active sets using neural nets," in *2019 IEEE Milan PowerTech*, 2019: IEEE, pp. 1-6.

[60] T. Yalcinoz and O. Köksoy, "A multiobjective optimization method to environmental economic dispatch," *International Journal of Electrical Power & Energy Systems,* vol. 29, no. 1, pp. 42-50, 2007.

[61] W. L. Snyder, H. D. Powell, and J. C. Rayburn, "Dynamic programming approach to unit commitment," *IEEE Transactions on Power Systems,* vol. 2, no. 2, pp. 339-348, 1987.

[62] V. Dieu and W. Ongsakul, "Enhanced augmented Lagrangian Hopfield network for unit commitment," *IEE proceedings-generation, transmission and distribution,* vol. 153, no. 6, pp. 624-632, 2006.

[63] S. H. Low, "Convex relaxation of optimal power flow—Part II: Exactness," *IEEE Transactions on Control of Network Systems,* vol. 1, no. 2, pp. 177-189, 2014.

[64] H. Wu, M. Shahidehpour, Z. Li, and W. Tian, "Chance-constrained day-ahead scheduling in stochastic power system operation," *IEEE Transactions on Power Systems,* vol. 29, no. 4, pp. 1583-1591, 2014.

[65] A. Kargarian *et al.*, "Toward distributed/decentralized DC optimal power flow implementation in future electric power systems," *IEEE Transactions on Smart Grid,* vol. 9, no. 4, pp. 2574-2594, 2016.

[66] D. P. Bertsekas, "Nonlinear programming," *Journal of the Operational Research Society,* vol. 48, no. 3, pp. 334-334, 1997.

[67] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate O (1/k^ 2)," in *Dokl. akad. nauk Sssr*, 1983, vol. 269, pp. 543-547.

[68] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk, "Fast alternating direction optimization methods," *SIAM Journal on Imaging Sciences,* vol. 7, no. 3, pp. 1588-1623, 2014.

[69] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747,* 2016.

[70] J. Lofberg, "YALMIP: A toolbox for modeling and optimization in MATLAB," in *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*, 2004: IEEE, pp. 284-289.

[71] B. H. Kim and R. Baldick, "Coarse-grained distributed optimal power flow," *IEEE Transactions on Power Systems,* vol. 12, no. 2, pp. 932-939, 1997.

[72] A. J. Wood, B. F. Wollenberg, and G. B. Sheblé, *Power generation, operation, and control*. John Wiley & Sons, 2013.

[73] M. B. Cain, R. P. O'neill, and A. Castillo, "History of optimal power flow and formulations," *Federal Energy Regulatory Commission,* vol. 1, pp. 1-36, 2012.

[74] Y. Wen, C. Guo, H. Pandžić, and D. S. Kirschen, "Enhanced security-constrained unit commitment with emerging utility-scale energy storage," *IEEE Transactions on power Systems,* vol. 31, no. 1, pp. 652-662, 2015.

[75] B. A. Calfa, I. E. Grossmann, A. Agarwal, S. J. Bury, and J. M. Wassick, "Data-driven individual and joint chance-constrained optimization via kernel smoothing," *Computers & Chemical Engineering,* vol. 78, pp. 51-69, 2015.

[76] R. Jiang and Y. Guan, "Data-driven chance constrained stochastic program," *Mathematical Programming,* vol. 158, no. 1-2, pp. 291-327, 2016.

[77] L. Pardo, *Statistical inference based on divergence measures*. CRC press, 2005.

[78] P. Van Kerm, "Adaptive kernel density estimation," *The Stata Journal,* vol. 3, no. 2, pp. 148-156, 2003.

[79] P. Yang and A. Nehorai, "Joint optimization of hybrid energy storage and generation capacity with renewable energy," *IEEE Transactions on Smart Grid,* vol. 5, no. 4, pp. 1566-1574, 2014.

[80] J. Guo, G. Hug, and O. K. Tonguz, "Intelligent partitioning in distributed optimization of electric power systems," *IEEE Transactions on Smart Grid,* vol. 7, no. 3, pp. 1249-1258, 2016.

[81] M. Peng, L. Liu, and C. Jiang, "A review on the economic dispatch and risk management of the large-scale plug-in electric vehicles (PHEVs)-penetrated power systems," *Renewable and Sustainable Energy Reviews,* vol. 16, no. 3, pp. 1508-1515, 2012.

[82] A. Mohammadi, F. Safdarian, M. Mehrtash, and A. Kargarian, "A System of Systems Engineering Framework for Modern Power System," *Sustainable Interdependent Networks II: From Smart Power Grids to Intelligent Transportation Networks,* p. 217, 2019.

[83] S. DorMohammadi and M. Rais-Rohani, "Exponential penalty function formulation for multilevel optimization using the analytical target cascading framework," *Structural and Multidisciplinary Optimization,* vol. 47, no. 4, pp. 599-612, 2013.

[84] S. Tosserams, L. Etman, P. Papalambros, and J. Rooda, "An augmented Lagrangian relaxation for analytical target cascading using the alternating direction method of multipliers," *Structural and multidisciplinary optimization,* vol. 31, no. 3, pp. 176-189, 2006.

[85] Y. Sun, Z. Li, W. Tian, and M. Shahidehpour, "A Lagrangian Decomposition Approach to Energy Storage Transportation Scheduling in Power Systems," *IEEE Transactions on Power Systems,* vol. 31, no. 6, pp. 4348-4356, 2016.

[86] H. P. Williams, *Model building in mathematical programming*. John Wiley & Sons, 2013.

[87] A. Liaw and M. Wiener, "Classification and regression by randomForest," *R news,* vol. 2, no. 3, pp. 18-22, 2002.

[88] J. R. Quinlan, "Induction of decision trees," *Machine learning,* vol. 1, no. 1, pp. 81-106, 1986.

[89] S. Soleimani, S. R. Mousa, J. Codjoe, and M. Leitner, "A comprehensive railroad-highway grade crossing consolidation model: a machine learning approach," *Accident Analysis & Prevention,* vol. 128, pp. 65-77, 2019.

[90] T. Chen, T. He, M. Benesty, V. Khotilovich, and Y. Tang, "Xgboost: extreme gradient boosting," *R package version 0.4-2,* pp. 1-4, 2015.

[91] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785-794.

[92] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics,* pp. 1189-1232, 2001.

[93] H. Alaiz-Moreton, J. Aveleira-Mata, J. Ondicol-Garcia, A. L. Muñoz-Castañeda, I. García, and C. Benavides, "Multiclass classification procedure for detecting attacks on MQTT-IoT protocol," *Complexity,* vol. 2019, 2019.

[94] T. Chen, T. He, M. Benesty, and V. Khotilovich, "Package 'xgboost'," *R version 0.90,* 2019.

[95] K. Saeed, W. Homenda, and R. Chaki, *Computer Information Systems and Industrial Management: 16th IFIP TC8 International Conference, CISIM 2017, Bialystok, Poland, June 16-18, 2017, Proceedings*. Springer, 2017.

[96] M. Ahmadi, D. Ulyanov, S. Semenov, M. Trofimov, and G. Giacinto, "Novel feature extraction, selection and fusion for effective malware family classification," in *Proceedings of the sixth ACM conference on data and application security and privacy*, 2016, pp. 183-194.

[97] D. Barradas, N. Santos, and L. Rodrigues, "Effective detection of multimedia protocol tunneling using machine learning," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 169-185.

[98] Y. Meidan *et al.*, "ProfilIoT: a machine learning approach for IoT device identification based on network traffic analysis," in *Proceedings of the symposium on applied computing*, 2017, pp. 506-509.

[99] W. F. Mustika, H. Murfi, and Y. Widyaningsih, "Analysis Accuracy of XGBoost Model for Multiclass Classification-A Case Study of Applicant Level Risk Prediction for Life Insurance," in *2019 5th International Conference on Science in Information Technology (ICSITech)*, 2019: IEEE, pp. 71-77.

[100] B. Li, S. Fan, X. Gao, F. Zhai, K. Li, and Y. He, "Smart Meters Fault Prediction Technology Based on Cost-sensitive XGBoost Algorithm for Imbalanced Dat," in *2019 3rd International Conference on Robotics and Automation Sciences (ICRAS)*, 2019: IEEE, pp. 189-195.

[101] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of machine learning research,* vol. 12, no. Oct, pp. 2825-2830, 2011.

[102] J. R. Landis and G. G. Koch, "The measurement of observer agreement for categorical data," *biometrics,* pp. 159-174, 1977.

[103] https://sites.google.com/site/aminkargarian/test-system-data

[104] http://dataminer2.pjm.com/feed/load_frcstd_7_day.

VITA

Farnaz Safdarian received her B.Sc. degree in Electrical Engineering (Power) from Shahid Beheshti University, Tehran, Iran, in 2011, and her M.Sc. degree in Electrical Engineering (Power) from Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran, in 2014, with the first rank honor. She is pursuing his Ph.D. degree in Electrical Engineering (Power) with the Department of Electrical and Computer Engineering, Louisiana State University, LA, USA, with anticipated graduation in June 2020. She has more than eight years of experience in research and industry, involved in two national projects and published several papers and books as a part of her experiences. Her research interests include distributed and decentralized optimization, power systems operation and planning, machine learning and data science, energy management, electricity markets, smart grids, and microgrids.