An Asynchronous, Decentralized Solution Framework for the Large Scale Unit Commitment Problem

Paritosh Ramanan*†,Murat Yildirim‡, Edmond Chow* and Nagi Gebraeel†

***Abstract*—With increased reliance on cyber infrastructure, large scale power networks face new challenges owing to computational scalability. In this paper we focus on developing an asynchronous decentralized solution framework for the Unit Commitment(UC) problem for large scale power networks. We exploit the inherent asynchrony in a region based decomposition arising out of imbalance in regional subproblems to boost computational efficiency. A two phase algorithm is proposed that relies on the convex relaxation and privacy preserving valid inequalities in order to deliver algorithmic improvements. Our algorithm employs a novel *interleaved binary* mechanism that locally switches from the convex subproblem to its binary counterpart based on consistent local convergent behavior. We develop a high performance computing (HPC) oriented software framework that uses Message Passing Interface (MPI) to drive our benchmark studies. Our simulations performed on the IEEE 3012 bus case are benchmarked against the centralized and a state of the art synchronous decentralized method. The results demonstrate that the asynchronous method improves computational efficiency by a significant amount and provides a competitive solution quality rivaling the benchmark methods.**

***Index Terms*—Asynchronous decentralized optimization, unit commitment, privacy preserving algorithm.**

NOMENCLATURE

Sets:

$\mathcal{R}$	The set of all regions
$\mathcal{N}_r, G_r, \mathcal{U}_r, \mathcal{V}_r, \mathcal{I}_r$	Neighboring regions, generators, boundary, foreign and internal buses of region r
$\mathcal{B}_r$	$\mathcal{U}_r \cup \mathcal{V}_r$, Boundary, foreign buses of r
$\mathcal{N}_r^b$	Neighboring regions connected to bus $b \in \mathcal{U}_r$
$G_r^b, \mathcal{U}_r^b, \mathcal{V}_r^b, \mathcal{I}_r^b$	Generators, boundary, foreign and internal buses connected to bus $b \in \mathcal{U}_r \cup \mathcal{I}_r$
$\mathcal{B}_r^b$	$\mathcal{U}_r^b \cup \mathcal{V}_r^b \cup \mathcal{I}_r^b$, Neighboring buses of bus b
T	Operational planning horizon

Decision Variables (at $t \in T$):

y_t^g	The electricity dispatch of generator g
$x_t^g \in \{0,1\}$	The commitment decision variable of g
θ_t^b	The phase angle at bus b
$\tilde{\theta}_t^{b,r'}$	The phase angle of bus b where $b \in \mathcal{U}_{r'}$ and $r' \in \mathcal{N}_r$
f_t^{uv}	Power flow from bus u to v such that $u \in \mathcal{U}_r$ and $v \in \mathcal{V}_r^u$
π_{Ut}^g, π_{Dt}^g	The up and down variable of generator g
$p_{r,t}$	Production difference at region r
λ_t^b	The Lagrangian multiplier with respect to phase angles of bus b where $b \in \mathcal{U}_r \bigcup \mathcal{V}_r$
ϕ_t^{uv}	The Lagrangian multiplier with respect to flow from bus u to bus v where $u \in \mathcal{U}_r$ and $v \in \mathcal{V}_r$ for any region r
η_t	The Lagrangian multiplier with respect to production difference of region $r \in \mathcal{R}$

Constants:

d^g, c^g, S_U^g, S_D^g	The dispatch cost, commitment cost start-up cost shut-down cost of generator g
P_{min}^g, P_{max}^g	Minimum and maximum capacity of g
M_U^g, M_D^g, R^g	Minimum up time, down time and ramp-up ramp-down constant for g
δ_t^b	The demand at bus b at $t \in T$
F_{max}^{uv}	Maximum capacity of line connecting buses u and v such that $u \in \mathcal{U}_r$ and $v \in \mathcal{V}_r^u$
$\rho_\theta, \rho_f, \rho_p$	Penalty parameter for phase angles, flows and production difference
Γ^{uv}	Phase angle conversion for line uv

I. INTRODUCTION

Unit Commitment (UC) in power networks is a well-studied optimization problem that determines the optimal power generation schedule for a fleet of networked generators. Any UC solution framework can be broadly divided into two parts, the data component and the UC problem component. While the problem component consists of the solution methodology and the optimization problem formulation, the data component consists of infrastructural information pertaining to the network topology, transmission lines, buses and generators. UC is a computationally challenging problem due to its scale, discrete nature and the vast amount of data that is essential to obtain a solution. Usually, UC is solved in a centralized manner at a control center where the problem component must be co-located with the data component. A distributed UC solution relies mainly on parallel two-stage techniques in order to obtain faster solution times. Such techniques exploit

*School of Computational Science and Engineering, Georgia Institute of technology, Atlanta, GA, USA 30332
‡College of Engineering, Wayne State University, Detroit, MI, USA 48202
†H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of technology, Atlanta, GA, USA 30332.
paritoshpr@gatech.edu,murat@wayne.edu,echow@cc.gatech.edu, nagi@isye.gatech.edu

the scope for parallelism among the two stages to provide computational speedup [1], [2], [3], [4]. However they fail to remove the constraint of co-locating the data and the problem components. The UC problem is a stepping stone towards more complex planning problems. Emerging applications of UC involving data-driven operations planning will not be amenable to a centralized model of computation. This will be particularly important as UC problem faces new challenges in data acquisition and interpretations in the areas of integration of renewables, incorporation of maintenance, transmission line switching and prevention of cascading failures. In such problems, co-locating data and problem components may prove to be infeasible.

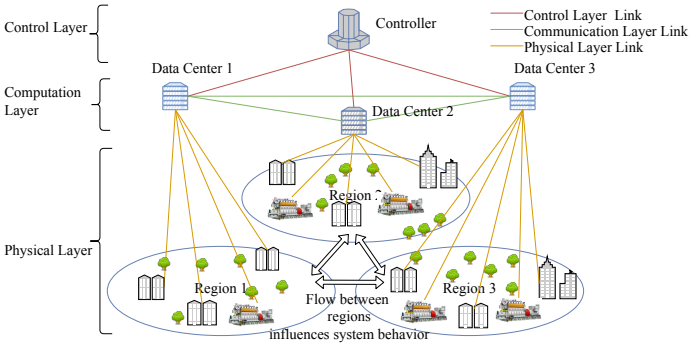

Fig. 1: A power network schematic for a decentralized framework

Unlike the distributed method, a decentralized solution framework is obtained by decomposing the global UC problem component into smaller local subproblems held by multiple computing agents. Each agent only holds a slice of the global data pertaining to its own local subproblem. Therefore, only the local subproblem and the corresponding network data needs to be co-located. The decentralized UC problem considered in this paper utilizes a region based decomposition strategy for a large scale power network. Every region is assigned to a unique computing agent denoted by a data center as shown in Figure 1. The region based decentralized framework is laterally divided into three parts: physical, computation and control layers. The system level data comprising of network flow estimates is gleaned from the physical network layer by data centers located in the computation layer. These data centers, scattered across a wide geographic area are responsible only for their own local subproblems. At each region, local computation at the data centers yields new estimates of network flow variables that govern transmission lines. The updated estimates pertaining to shared transmission lines between two regions is communicated by the corresponding agents. The newly received network flow estimate is used as input to the next local computation step. The global progress of such an iterative scheme involving local computation and communication is tracked by the control layer. Eventually, the decentralized approach leads to balance of network flow estimates among the regions thereby signifying a solution to the global UC problem.

A decentralized approach to the UC problem has numerous advantages. First, it improves computational efficiency since a region based decomposition of the global problem yields much smaller local subproblems that can be solved in parallel. Second, since each region operates independently, a decentralized method is highly suited for a geographically distributed computational architecture. Third, sensitive operational data can be held privately by each region, thus, limiting data sharing to specific parameters that do not violate privacy. Lastly, it has been found that a decentralized agent-to-agent communication is more efficient in terms of communication latencies [5], [6]. A distributed agent-to-master communication model introduces a single point of failure with the master bearing the heavy burden of processing information sent by all the workers. Since the master can only process information from one agent at a time, the other agents must wait till the master has had an opportunity to process and respond to their corresponding message resulting in poor scalability with respect to increasing problem sizes. Since communication is more expensive than computation [7] idle time can be eliminated or significantly reduced by allowing agents to send messages to each other instead of waiting on one master to respond As a result decentralized communication improves computational performance.

However, one of the disadvantages of current decentralized methods is the need to fully *synchronize* computation among all agents. In synchronous settings, all regions perform their local computational step, and wait for the other regions to finish before proceeding to the communication step. In a large scale decentralized power network, the expectation of a fully synchronous system can be highly misplaced [8]. Local subproblems might vary in their computational complexity owing to different problem sizes, which often leads to computational imbalance. The computing hardware employed by the various regions might be different, leading to an imbalance at the hardware level itself. In reality, a region's UC problem is localized to a data center within close geographical proximity to regional power assets. A practical solution to a large scale decentralized UC problem with a sizable number of regions would involve multiple data centers scattered over a vast geographical area thereby incurring significant communication costs. Therefore, in a real world implementation, a synchronous approach might suffer from significant idle time incurred due to local heterogeneity and increased communication costs leading to poor computational efficiency and slower progress.

An asynchronous approach has the potential to provide reliable, fast and robust decisions for power system optimization problems. In an asynchronous setting, all the regions perform their local updates based on the latest available information from their peers. Therefore, unlike the synchronous approach, the computational bottleneck arising from slower regions is eliminated, which minimizes idle time and improves computational efficiency. It naturally follows that an asynchronous method would also be resilient to the computational imbalance of local problems due to heterogeneous hardware and communication latencies. Further, an asynchronous decentralized method is regarded more favorably with respect to the mitigation of cyber-attacks since regional computations progress independently and the global objective remains unchanged.

In this paper, we focus on developing a novel *asynchronous*

decentralized computational framework for solving large scale UC problems. The main contributions of our paper can be summarized as follows:

- We develop a two phase asynchronous decentralized algorithm for solving the decentralized UC problem asynchronously. The algorithm iteratively solves the convex relaxation in the first phase. In the second phase, the binary constraints on the decision variables are imposed. We strengthen our two phase approach with privacy preserving valid inequalities that lead to sound solution quality and robust computational performance.
- We propose a novel *interleaved binary* mechanism that allows regions to advance to the binary phase after having exhibited consistent local convex convergence behavior irrespective of global convex convergence. This also leads to a significant improvement in computational efficiency.
- We propose and implement the concept of a controller to facilitate two-way message exchanges at discrete global clock ticks among neighboring regions which satisfies a crucial convergence assumption [9].
- We develop a custom-made software framework based on the asynchronous reformulation that is fine tuned specifically for the UC problem. We present simulations with respect to the 75, 100 and 120 region scenarios of the IEEE 3012 bus system on a high performance computing environment using MPI semantics.

A typical application of our method would be for a large scale ISO or vertically integrated power company. In such a scenario, participating regions could solve the global problem in a decentralized manner without revealing their infrastructural data while having superior computational performance with respect to centralized methods. Our algorithm is mainly for transmission level operators. However, our methodology is generic and could potentially be applied for coordinating transmission and distribution coordinated system operations as well.

The rest of the paper is divided as follows. Section II we discuss various other approaches explored in literature with respect to decentralized, asynchronous methods. In Section III, we present the region based decentralized decomposition of the UC problem. We present the asynchronous decentralized algorithm in Section IV based on [9] and develop a privacy preserving valid inequality that delivers algorithmic improvements. We discuss the implementation aspects of our algorithm and introduce the concept of a controller to successfully orchestrate two-way message exchanges as required by our algorithm. We present results from a robustness and benchmarking study in Section V and compare the asynchronous method against the centralized and synchronous variants. We conclude our paper in Section VI.

II. Related Works

Augmented Lagrangian techniques like the Alternating Direction Method of Multipliers (ADMM) [10] have been useful for solving decentralized constrained optimization problems with good convergence properties [11]. In the asynchronous optimization literature, the main focus has been on unconstrained optimization [12], [13], [14]. Recently, there has been a growing literature on asynchronous constrained models to address a more general class of optimization problems. Chang [15] proposes an asynchronous ADMM oriented solution for constrained optimization problems with time-varying networks and also under communication errors, whereas Eckstein [16] proposes an asynchronous ADMM like method for multi-block decomposable problems suited for an HPC environment with shared memory capabilities and all-to-all connectivity among compute nodes. In contrast, Wei and Ozdaglar [9] propose an asynchronous ADMM algorithm for distributed constrained optimization that makes no assumptions about compute capabilities of the hardware or about the communication links present in the network. Further the authors show the convergence of their algorithm for distributed constrained optimization problems making their algorithm highly suited for a geographically distributed set of compute agents communicating over potentially high delay inducing links.

Asynchronous master-slave type distributed optimization techniques have recently gained popularity within the power systems domain. Zhang and Kwok [17] propose an ADMM implementation that takes advantage of partial progress being made by slaves with respect to the master, resulting in higher computational efficiency. Within the domain of power systems a similar idea is explored by Aravena and Papavasilou[1] in terms of a distributed asynchronous two stage stochastic UC model where the dual iterations and the feasibility recovery between the master and the slaves occurs asynchronously. Papavasilou *et al.*[4] propose a HPC solution framework for solving the stochastic UC problem with dual decomposition. Similar work has been done in the asynchronous domain by Kim *et al.* [2], [3] exploiting the asynchrony arising out of load imbalance between the various slaves and the master in two-stage stochastic problems specifically for the security constrained UC and the stochastic UC problems. A limitation of these methods over the decentralized approaches, is the requirement of a master node. While offering much potential in terms of computational efficiency, the presence of a master problem with infrastructural data pertaining to the entire network can prove to be impractical in a real world setting owing to reasons mentioned in Section I.

A hallmark of a decentralized solution is the absence of a master problem holding global data. In this domain, the work done by Feizollahi *et al.* [18] provides a synchronous decentralized fix-and-release approach for large scale UC problems. However, in their framework, the UC fix-and-release pertaining to the MIP may not be ideal for a geographically distributed computing environment. A direct asynchronous extension of the synchronous decentralized computational framework for UC presented in [18] has been explored in [19], [20]. In a similar context, Guo *et al.* [21] provide an asynchronous decentralized method for non-convex problems in power systems, which has a similar computational outline as the previous two works but as pointed out in [19], such an approach might suffer from poor solution quality when it comes to MIP problems such as asynchronous decentralized UC. While these methods show great promise in computational efficiency, the solution quality arising from such types of asynchronous decentralized UC frameworks has been shown

to be poor at times.

III. Decentralized Unit-Commitment

In this section, we present an enhanced decentralized UC problem formulation that is derived from the asynchronous ADMM framework for constrained optimization problems proposed by Wei and Ozdaglar [9]. This approach targets a multi-agent decentralized solution to the constrained optimization problem. Each agent exchanges its local estimate of the consensus variable with a neighbor after each local solve. A message exchange between two neighbors is triggered by the local clock tick associated with that edge. In doing so iteratively, all the agents converge to a solution for the global optimization problem. In our current formulation, we rely on two-way exchange of messages by a pair of neighboring regions at each global tick unlike our previous work [19] that used a broadcast based method. In addition to the consensus quantities themselves, we also exchange their respective Lagrangian information in order to adhere to the convergence conditions set forth in [9].

Our decentralized formulation for the UC problem is geared towards improving solution quality in an asynchronous setting to bolster its applicability in a real world, geographically distributed computational environment. In asynchronous computational conditions, heuristics proposed in the literature with a synchronous approach in mind (i.e. fix-and-release) might be impractical. Further, the solution mechanism illustrated in [19] makes it evident that decentralized asynchronous methods offer good computational potential but leave a lot of room for improvement in terms of the solution quality owing to oscillations from the binary components of the UC problem.

A region based decomposition partitions the set of buses such that each bus is uniquely owned by only one region. It follows that every bus in a region can be categorized either as a boundary or an internal bus. Boundary buses of a region are ones which have a transmission line connecting them to at least one bus belonging to a neighboring region. Further, buses owned by a neighboring region lying on the other end of a transmission line from a boundary bus are termed as foreign buses. On the other hand, all buses owned by a region which have no transmission lines connecting to foreign buses are termed as internal buses. Transmission lines are identified using a unique identifier representing the buses at either end.

We present the decentralized UC objective function in Problem (1), the model and the constraints are listed in (2).

$$\begin{aligned}\mathcal{L}_r(\bar{\boldsymbol{\theta}},\bar{\boldsymbol{F}},\boldsymbol{\lambda},\boldsymbol{\phi}) = &\sum_{t\in T}\sum_{g\in G_r} d^g y_t^g + c^g x_t^g \\ &+ \sum_{t\in T}\sum_{g\in G_r} S_U^g \pi_{Ut}^g + S_D^g \pi_{Dt}^g \\ &+ \sum_{t\in T}\sum_{b\in\mathcal{B}_r}\Big[\lambda_t^b|\theta_t^b-\bar{\theta}_t^b|+\frac{\rho_\theta}{2}(\theta_t^b-\bar{\theta}_t^b)^2\Big] \\ &+ \sum_{t\in T}\sum_{u\in\mathcal{U}_r}\sum_{v\in\mathcal{V}_r^u}\Big[\phi_t^{uv}|f_t^{uv}-\bar{f}_t^{uv}| \\ &+ \frac{\rho_f}{2}(f_t^{uv}-\bar{f}_t^{uv})^2+\frac{\rho_f}{2}(f_t^{uv}-\tilde{f}_t^{uv})^2\Big]\end{aligned} \tag{1}$$

$$\min_{\boldsymbol{\theta},\boldsymbol{f},\boldsymbol{x},\boldsymbol{y}} \ \mathcal{L}_r(\bar{\boldsymbol{\theta}},\bar{\boldsymbol{F}},\boldsymbol{\lambda},\boldsymbol{\phi}) \tag{2a}$$

$$\text{s.t. } P_{min}^g x_t^g \le y_t^g \le P_{max}^g x_t^g, \quad \forall t\in T, \forall g\in G_r \tag{2b}$$

$$-\pi_{Dt}^g \le x_t^g - x_{t-1}^g \le \pi_{Ut}^g, \ \forall t\in[2,T], \forall g\in G_r \tag{2c}$$

$$-R^g \le y_t^g - y_{t-1}^g \le R^g, \ \forall t\in[2,T], \forall g\in G_r \tag{2d}$$

$$\Gamma^{uv}(\theta_t^u-\theta_t^v) = f_t^{uv}, \ \forall u\in\mathcal{U}_r, \forall v\in\mathcal{V}_r^u, \forall t\in T \tag{2e}$$

$$-F_{max}^{uv} \le \Gamma^{uv}(\theta_t^u-\theta_t^v) \le F_{max}^{uv}, \ \forall u\in\mathcal{U}_r\cup\mathcal{I}_r, \ \forall v\in\mathcal{B}_r^u, \forall t\in T \tag{2f}$$

$$\sum_{\forall g\in G_r^u} y_t^g - \delta_t^u = \sum_{\forall v\in\mathcal{B}_r^u}[\Gamma^{uv}(\theta_t^u-\theta_t^v)], \ \forall u\in\mathcal{U}_r\cup\mathcal{I}_r, \ \forall t\in T \tag{2g}$$

$$\sum_{\forall i\in U_t}\pi_{Ui}^g \le x_t^g \le 1-\sum_{\forall i\in D_t}\pi_{Di}^g, \ \forall t\in T, \forall g\in G_r$$
$$U_t=[t-M_U^g+1,t], D_t=[t-M_D^g+1,t] \tag{2h}$$

Constraint (2b) ensures production at each generator being bounded by its minimum and maximum capacity. Constraints (2c) and (2h) enforce minimum up and down time for each generator. Constraint (2d) ensures generators adhere to their respective ramping limitations. Equation (2e) ensures that flow across inter-regional links is a function of the respective phase angles. Constraint (2f) enforces transmission line capacity constraints. Equation (2g) ensures that at each bus the demand is met by either power generated by attached generators, or with the flow into the region. Equations (2e)-(2g) enforce global network flow constraints.

We estimate two important quantities, *intermediate flows* $\bar{F}_t^{uv}$ and *intermediate phase angles* $\bar{\theta}_t^b$ using Equation (3) based on values received from neighboring region $r'\in\mathcal{N}_r$. In each update, region r' sends flow dual values $\tilde{\phi}_t^{uv,r'}$, $\forall u\in\mathcal{U}_r, \forall v\in\mathcal{V}_r^u\cap\mathcal{U}_{r'}, \forall t\in T$ and phase angle and phase dual estimates $\tilde{\theta}_t^{b,r'}, \tilde{\lambda}_t^{b,r'} \forall b\in\mathcal{B}_r\cap\mathcal{B}_{r'}$.

$$\hat{\lambda}_t^b = \frac{-1}{2}(\lambda_t^b+\tilde{\lambda}_t^{b,r'})+\frac{\rho_\theta}{2}(\theta_t^b-\tilde{\theta}_t^{b,r'}) \tag{3a}$$

$$\bar{\theta}_t^b = \frac{1}{\rho_\theta}(\hat{\lambda}_t^b+\lambda_t^b)+\tilde{\theta}_t^{b,r'}, \quad \lambda_t^b=\hat{\lambda}_t^b \tag{3b}$$

$$\hat{\phi}_t^{uv} = \frac{-1}{2}(\phi_t^{uv}+\tilde{\phi}_t^{uv,r'})+\frac{\rho_f}{2}(f_t^{uv}-\tilde{f}_t^{uv,r'}) \tag{3c}$$

$$\bar{f}_t^{uv} = \frac{1}{\rho_f}(\hat{\phi}_t^{uv}+\phi_t^{uv})+\tilde{f}_t^{uv,r'}, \quad \phi_t^{uv}=\hat{\phi}_t^{uv} \tag{3d}$$

In order to avoid redundant and expensive communication, the flow estimates $\tilde{f}_t^{uv}$ of the neighbor region are computed based on the phase angles sent by the neighbor.

IV. Asynchronous Solution Methodology

In this section, we seek to design a solution methodology for Problem 2 that performs well in asynchronous conditions and maintains operational privacy. We augment the existing formulation with a redundant valid inequality based on production and demand to boost computational performance in an asynchronous system. The two-way message exchange necessitated by local clock tick as mentioned in [9] and explained in Section III is tedious to implement and may cause significant computational overhead. Therefore, we introduce the concept of a controller that matches two neighboring regions on the completion of their respective local computation step. We design and develop the controller mechanism to also track asynchronous global progress of the regions while preserving asynchronous convergence conditions. Finally, we present our two phase asynchronous decentralized UC algorithm that solves the local convex relaxations of Problem 2 in the first phase before imposing binary constraints in the second phase. In order to improve computational speedup, our algorithm incorporates a novel interleaved binary mechanism that lets regions advance to their local binary problems based on consistent local convergence of their convex relaxations.

A. A privacy preserving valid inequality

In a decentralized UC solution, global production has a direct bearing on the binary commitment variables. Theoretically, network flow constraints ought to be sufficient conditions for the UC problem solution to balance global production and demand. However, in a decentralized environment, network flow constraints are enforced using Lagrangian decomposition between regions. Decomposed network flow constraints drive the optimal assignment of binary decision variables which ultimately culminates in global convergence. Owing to volatility arising out of heavy latency induced message passing in an asynchronous system, the network flow constraints alone might not be strong enough to meet this balance. The decentralized formulation must therefore be further secured by a globally redundant constraint based on production and demand. These constraints must also retain the privacy preserving nature of decentralized methods.

We consider a balance of global production and demand denoted by $\sum_{\forall r\in\mathcal{R}}\sum_{\forall b\in\mathcal{U}_r\bigcup\mathcal{I}_r}\delta_t^b = \sum_{\forall r\in\mathcal{R}}\sum_{\forall g\in G_r} y_t^g$. In order to decentralize the production-demand balance constraint, we establish an asynchronous friendly mechanism in order to enforce it globally. We respectively compute the local production difference $\psi_{r,t}$, the inverse of the average production residual cost $s_{r,t}$ and its multiplier $\mu_{r,t}$ for every region r and for every time period in the planning horizon as follows.

$$\psi_{r,t} = \sum_{\forall b\in\mathcal{U}_r\bigcup\mathcal{I}_r}\delta_t^b - \sum_{\forall g\in G_r} y_t^g \tag{4a}$$

$$s_{r,t} = \frac{\sum_{\forall g\in G_r}(P_{max}^g - y_t^g)}{\sum_{\forall g\in G_r} d^g(P_{max}^g - y_t^g)} \tag{4b}$$

$$\mu_{r,t} = \frac{s_{r,t}}{\sum_{\forall r\in\mathcal{R}} s_{r,t}} \tag{4c}$$

Then, the local production target is then computed as follows.

$$\bar{p}_{r,t} = \sum_{\forall g\in G_r} y_t^g + \mu_{r,t}\sum_{\forall r\in\mathcal{R}}\psi_{r,t} \tag{5}$$

Therefore, Problem (1) is further augmented by Lagrangian penalty terms pertaining to production difference as described in Problem (6).

$$\min_{\boldsymbol{\theta},\boldsymbol{f},\boldsymbol{x},\boldsymbol{y}} \quad \mathcal{L}_r(\bar{\boldsymbol{\theta}},\bar{\boldsymbol{F}},\boldsymbol{\lambda},\boldsymbol{\phi}) + \sum_{t\in T}\left[\eta_t|p_{r,t}-\bar{p}_{r,t}| + \frac{\rho_p}{2}(p_{r,t}-\bar{p}_{r,t})^2\right] \tag{6a}$$

$$\text{s.t.} \quad \text{(2b)} - \text{(2h)}$$

$$\sum_{\forall g\in G_r} y_t^g = p_{r,t}, \quad \forall t\in T \tag{6b}$$

We further add Equation (6b) to the existing constraint set where we try to provide a production target for Problem (6). This is achieved by performing a global weighted average of the production difference arising out of every region. Intuitively, each region is assigned a customized production target as given by $\bar{p}_{r,t}$ further strengthening convergence. It is important to note that computation of $\bar{p}_{r,t}$ is highly suited for an asynchronous model owing to a reduced volatility in values due to the multiplier $\mu_{r,t}$.

B. Compute Architecture

One of the most important conditions imposed by the asynchronous algorithm in [9] is the presence of a global clock that drives two-way exchange of messages among agents [9]. At each global clock tick a pair of neighboring agents are triggered and exchange local information with each other leading to a two-way message exchange paradigm referred to as a doubly stochastic system.

The requirement of double stochasticity can prove to be a limitation for a variety of reasons. From a practical standpoint, especially in a geographically distributed computational setup, the implementation of a global clock is tedious and leads to a heavier computational burden with lesser accuracy [22]. From a computational perspective, techniques relying on a doubly stochastic system also suffer from issues related to potential bottlenecks in case of compute node failure [23]. If not given designed in the right manner, doubly stochastic systems can undermine the benefits of a decentralized method.

In order to solve a decentralized asynchronous constrained optimization problem, it is necessary to comply with the convergence conditions proposed in [9] and simultaneously address the issues arising out of a doubly stochastic algorithm. Therefore, in this paper we propose the concept of an additional computational agent called a controller. The controller for a doubly stochastic asynchronous decentralized scheme has the following roles:

- facilitating exchange of messages between neighboring regions on each clock tick.
- helping detect global convergence of the algorithm for an asynchronous method.
- computing an estimate of a global sum asynchronously.

Algorithm 1 Controller Logic

initialize $\tilde{\boldsymbol{\psi}}^r, \tilde{\boldsymbol{s}}^r, \tilde{\xi}^r, \tilde{\kappa}^r \leftarrow 0, \forall r \in \mathcal{N}$
while GC = false **do**
  recv $\{\tilde{\boldsymbol{\psi}}^{r_1}, \tilde{\boldsymbol{s}}^{r_1}, \tilde{\xi}^{r_1}, \tilde{\kappa}^{r_1}\}$ from some region r_1
  if $\tilde{\xi}^{r_2} = 1$, such that $\exists r_2 \in \mathcal{N}_{r_1}$ **then**
    send to r_1 $\{\sum_{r=1}^{|\mathcal{R}|} \tilde{\boldsymbol{\psi}}^r, \sum_{r=1}^{|\mathcal{R}|} \tilde{\boldsymbol{s}}^r, \sum_{r=1}^{|\mathcal{R}|} \tilde{\xi}^r, r_2\}$
    send to r_2 $\{\sum_{r=1}^{|\mathcal{R}|} \tilde{\boldsymbol{\psi}}^r, \sum_{r=1}^{|\mathcal{R}|} \tilde{\boldsymbol{s}}^r, \sum_{r=1}^{|\mathcal{R}|} \tilde{\xi}^r, r_1\}$
  end if
  if $\tilde{\xi}^r, \tilde{\kappa}^r = 1, \forall r \in \mathcal{R}$, then GC $\leftarrow$ true
end while

Algorithm 1 introduces the logic behind the asynchronous controller. The controller maintains a running list across the planning horizon of the global production difference vector $\tilde{\boldsymbol{\psi}}^r$, the global inverse average residual cost vector $\tilde{\boldsymbol{s}}^r$, the local convergence values $\tilde{\xi}^r$ and the phase $\tilde{\kappa}^r$ of each region. As soon as a region finishes its local computation, it sends its updated local production residual as well as the inverse average residual cost, its local convergence value and its phase to the controller. The controller updates its running estimate of these values and tries to match the aforementioned region with any of its neighbors that has also completed its local computation thereby preserving the concept of a global clock. If a match is found, the latest global running estimates are communicated to the matched pair. If no other neighboring region is active, the region simply waits until it hears back from the controller. GC denotes global convergence which occurs when all regions have converged with respect to the binary phase. Any private infrastructural data of the regions remain opaque to the controller, since production difference and the multiplier values shield any private information local to the regions.

C. Asynchronous Decentralized UC Algorithm

The convex relaxation of the UC problem plays a key role in obtaining good solution quality [18], [19]. It is also clear that UC problems in general have a very good convex relaxation[24]. Therefore, in this paper, we focus on improving the overall solution quality and computational performance of the asynchronous decentralized UC problem by first strengthening the performance of the convex relaxation with the help of the framework proposed in [9].

We divide the problem into two phases pertaining to the convex phase followed by the binary phase. In each phase, the Lagrangian values are also exchanged in addition to the phase angles and flow information, as dictated by [9], in order to obtain good solution quality. The solution from the convex phase is then used as a starting point for solving the binary phase.

The Interleaved Binary Asynchronous Decentralized Unit Commitment (IBAD-UC) solution methodology is presented in Algorithm 2. At every iteration k, each region solves its own local subproblem and generates the commitment decisions $\boldsymbol{x}_k$, dispatch decisions $\boldsymbol{y}_k$, local phase angle values $\boldsymbol{\theta}_k$, flow values $\boldsymbol{f}_k$. The region communicates the updated local production difference values and the inverse average residual cost to the controller and waits for a response. A response from the controller provides an estimate of the global production and the multiplier vector $\sum_{r=1}^{|\mathcal{R}|} \tilde{\boldsymbol{\psi}}^r, \sum_{r=1}^{|\mathcal{R}|} \tilde{\boldsymbol{s}}^r$ which are used in the next iteration of the local subproblem. The controller also identifies the neighbor with which the region must perform a two-way message exchange. This step initiates an exchange of information denoted by the tuple $\Delta_{rr'}^{ADUC}$ between region r and its neighbor r', where,

$$\Delta_{rr'}^{ADUC} = \left\{ \begin{array}{l} \{\boldsymbol{\Theta}_r, \boldsymbol{\Lambda}_r\} = \{\{\boldsymbol{\theta}^b, \boldsymbol{\lambda}^b\} | \forall b \in \mathcal{B}_r \cap \mathcal{B}_{r'}\} \\ \boldsymbol{\Phi}_r = \{\boldsymbol{\phi}^{uv} | \forall u \in \mathcal{U}_r, \forall v \in \mathcal{V}_r^u \cap \mathcal{U}_{r'}\} \end{array} \right\}$$

This tuple consists of the newly generated primal values as well as the Lagrangian information. After observing consistent local convex convergence behavior indicated by the Interleaved Binary (IB) constant ζ, the local subproblem of the region switches from the convex relaxation to its binary counterpart.

Figure 2 illustrates the flowchart corresponding to Algorithm 2. LC refers to local convergence value (ξ). The convex relaxation phase is denoted by CR ($\kappa = 0$), whereas the imposition of binary constraints on commitment variables is represented by the MIP phase ($\kappa = 1$). Local convergence occurs when the primal and dual variables with respect to the phase angles are close to some predetermined limit denoted by α and β respectively. Global convergence occurs when all

Algorithm 2 Interleaved Binary Asynchronous Decentralized UC (ADUC) Algorithm

for $r = 1, 2, 3 \ldots |\mathcal{R}|$ **do**
  Initialize $\bar{\boldsymbol{\theta}}_0, \bar{\boldsymbol{f}}_0, \bar{\boldsymbol{F}}_0, \boldsymbol{\lambda}_0, \boldsymbol{\phi}_0, \boldsymbol{x}_0, \boldsymbol{y}_0, k \leftarrow 0$
  $\kappa \leftarrow 0$, set starting phase to convex
  while GC = false **do**
    if $(||\boldsymbol{\theta}_k - \tilde{\boldsymbol{\theta}}_k|| < \alpha)$ and $(||\tilde{\boldsymbol{\theta}}_k - \tilde{\boldsymbol{\theta}}_{k-1}|| < \beta)$ **then**
      set $\xi_k \leftarrow 1$
      if $\xi_i = 1,\ \forall i \in [k - \zeta, k]$, then set $\kappa \leftarrow 1$
    end if
    $\boldsymbol{\theta}_{k+1}, \boldsymbol{f}_{k+1}, \boldsymbol{x}_{k+1}, \boldsymbol{y}_{k+1}$ calculated by Problem (6)
    calculate $\boldsymbol{\psi}, \boldsymbol{s}, \boldsymbol{\mu}$ using Equation (4)
    send $\{\boldsymbol{\psi}, \boldsymbol{s}, \xi_k, \kappa\}$ to the controller
    recv $\{\sum_{r=1}^{|\mathcal{R}|} \tilde{\boldsymbol{\psi}}^r, \sum_{r=1}^{|\mathcal{R}|} \tilde{\boldsymbol{s}}^r, \sum_{r=1}^{|\mathcal{R}|} \tilde{\xi}^r, r'\}$ from controller
    if $\sum_{r=1}^{|\mathcal{R}|} \tilde{\xi}^r = |\mathcal{R}|$ **then**
      if $\kappa = 1$, set GC $\leftarrow$ true, otherwise $\kappa \leftarrow 1$
    end if
    compute $\bar{\boldsymbol{p}}_r$ using Equation (5)
    send tuple $\Delta_{rr'}^{ADUC} = \{\boldsymbol{\Theta}, \boldsymbol{\Lambda}, \boldsymbol{\Phi}\}$ to r'
    recv tuple $\Delta_{r'r}^{ADUC} = \{\tilde{\boldsymbol{\Theta}}_{r'}, \tilde{\boldsymbol{\Lambda}}_{r'}, \tilde{\boldsymbol{\Phi}}_{r'}\}$ from r'
    compute $\tilde{\boldsymbol{f}}_{r'}$ based on $\tilde{\boldsymbol{\Theta}}_{r'}$
    update $\bar{\boldsymbol{\theta}}_{k+1}, \boldsymbol{\lambda}_{k+1}, \bar{\boldsymbol{f}}_{k+1}, \boldsymbol{\phi}_{k+1}$, using Equation (3)
    $\eta_r^t = \eta_r^t + \rho_p(p_{r,t} - \bar{p}_t), \quad \forall t \in T$
    $k \leftarrow k + 1$
  end while
end for

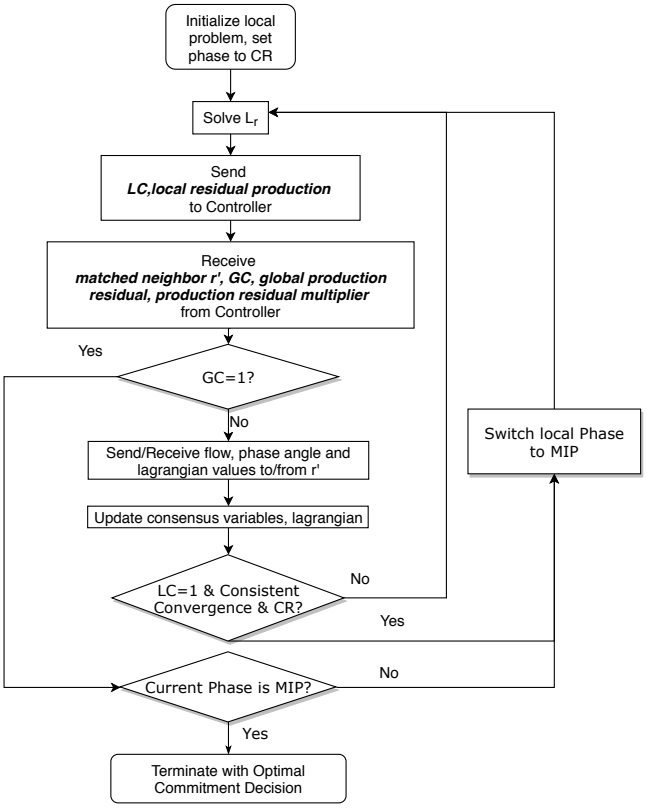

Fig. 2: Flowchart representing Algorithm 2.

regions have locally converged with respect to the MIP phase.

V. Experimental Results

We perform benchmarking studies comparing the IBAD-UC algorithm with its synchronous counterpart to demonstrate its superior computational efficiency and speed. We consider the centralized solution method in which the entire large scale UC problem is solved without region based decomposition. We benchmark the IBAD-UC algorithm with the centralized method to demonstrate comparable solution quality with a significant reduction in solution times.

In order to demonstrate robustness of our proposed solution methodology, we show that the IBAD-UC algorithm yields consistently good quality results with respect to the 75, 100 and 120 region decompositions of the IEEE 3012 bus case. The region decompositions each consist of approximately 40, 30 and 25 buses per region on an average respectively, depicting a valid real world scenario. We consider 150 generators in the 3012 bus case that have a non trivial production capacity.

A. Experiment Setup

We develop a distributed, parallel software framework that uses the MPI to orchestrate Algorithm 2 on a high performance compute cluster. Within MPI, we rely on Remote Memory Access (RMA) paradigm for asynchronous communication. RMA windows allow remote processes to read and write their latest values. Since each region occupies one process, regions communicate with their neighbors with much simpler semantics offered by RMA.

We perform our experiments on a HPC cluster comprised of Intel Xeon compute nodes with 20 cores per node with a clock rate of 2.80GHz. Each region and the controller are assigned to one core. The *mpi4py* [25] framework was used to interface with MPI to conduct our HPC simulations. *Gurobi 6.5* was used to solve Problem (2) locally on each core with multi threading turned off on each region to prevent oversubscription of computational resources. We used IEEE 3012 bus case data from *MATPOWER* [26] for our experiments. Our experimental results pertain to the 24 hour planning horizon spanning an entire day collected from over 400 experiments conducted with the IBAD-UC algorithm. We benchmark our results against the decentralized synchronous solution methodology presented in [18] augmented with the production difference valid inequality with the multiplier being constant $\mu_{r,t} = \frac{1}{|\mathcal{R}|}$.

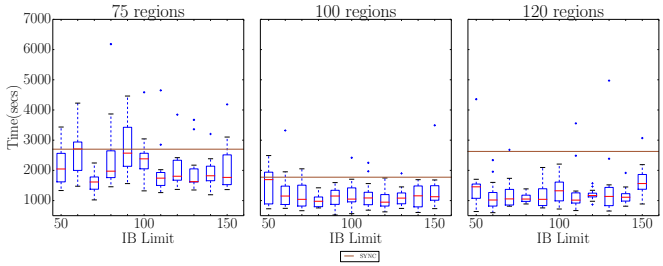

Fig. 3: Total time taken and the effect of IB limit ζ

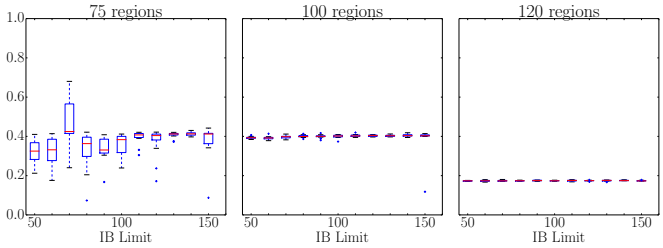

Fig. 4: Effect of IB limit ζ on asynchronous degree

B. Total solve time and effect of IB Limit

Figure 3 presents box plots for the time taken for convergence by the IBAD-UC algorithm against the time taken by its synchronous counterpart. Results depict the effects of variation in the IB Limit parameter represented by ζ. We observe a large variation in the time taken for convergence in the 75 region case with varying ζ, whereas, for 100 and 120 region cases, the variation consistently decreases, with 120 region case being the fastest. Figure 3 tells us that the 75 region decomposition likely has a relatively greater degree of imbalance in the problem sizes compared to the 100 and 120 region cases. It is also interesting to note that performance trends with respect to ζ oscillate between high and low variations in convergence time successively as ζ is increased, although this trend becomes much more subtle as we move from the 75 to the 120 region case. Overall, it can be seen that despite a variation in performance with respect to ζ, the IBAD-UC algorithm outperforms synchronous in all three cases.

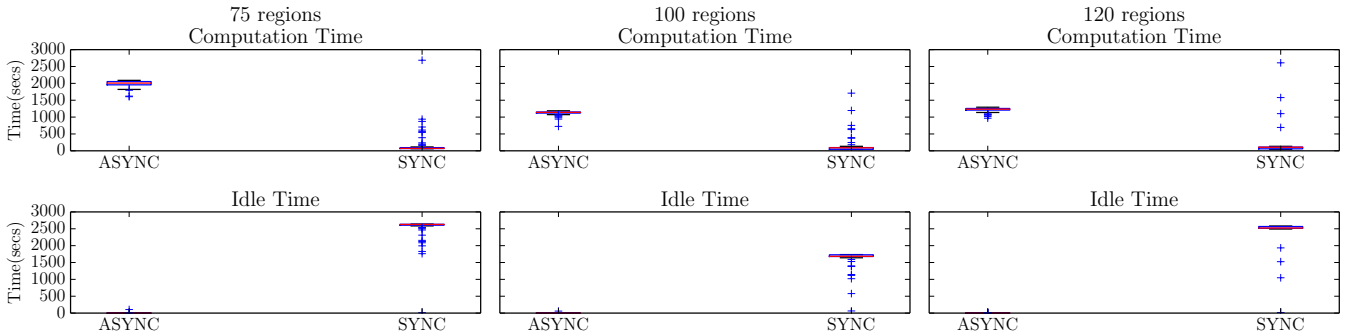

Fig. 5: Variation in Computation and Idle time for 75,100 and 120 regions

C. Asynchronous Degree

The asynchronous degree is the ratio of the minimum and the maximum number of updates performed by any region within each case. In Figure 4 we present box plots of the asynchronous degree based on ζ as a means to measure how asynchronous the system is. We can see that the asynchronous degree shows significant variation with a lower ζ value and stabilizes as we increase ζ. For the 100 and 120 region case presented in Figure there is little variation in the asynchronous degree as ζ is changed. This observation indicates that the 75 region case is relatively imbalanced leading to variation in asynchronous degree which is exacerbated at lower ζ values indicating premature advancement into the binary phase leading to higher volatility in solution times. Figure 4 thereby corroborates Figure 3 since higher variation in asynchronous degree might lead to higher variations in solution times as well.

D. Average Times

Table I presents the average computation, communication and idle times incurred by the asynchronous and the synchronous methods for the 75, 100 and 120 region cases. Figure 5 presents the variation in the mean computation and idle times for every region incurred by IBAD-UC alongside those for the synchronous method.

From the tables, we observe that the average synchronous computation time in general is much lower than the asynchronous while also incurring a smaller percentage share of the total as well. While the asynchronous method spends relatively less time idling, the synchronous method suffers from greater amount of idle time, both in terms of average and the percentage times. In addition, the communication times and the respective percentages do not depict much variation between asynchronous and synchronous methods. The consistency observed in terms of computation, communication and idle times by various region decompositions show robustness in computational performance by the IBAD-UC algorithm.

Figure 5 provides deeper insight into the trends presented in Table I by presenting *regional variations* in computation and idle times. We observe that there is wide variation in computation and idle times for regions in the synchronous method. For the synchronous method, the highest computation time incurred by a region is also very similar in value to the highest idle time incurred by any region for all cases. However, in the asynchronous method, the lower and upper bounds on computation times are much tighter and idle times are negligible. This behavior in computation and idle time is observed uniformly across all the region decompositions.

The data presented in Figure 5 and Table I indicates that the higher computation time for a few regions form the main bottlenecks for global progress which are readily circumvented by asynchronous methods. Meanwhile, global computational progress in the synchronous method is held up by the slowest region which simultaneously incurs very high idling times on the fastest region. Despite strongly asynchronous systems, more frequent asynchronous updates are able to successfully drive the problem towards the global solution much faster leading to superior computational efficiency.

E. Solution Quality

We solve the centralized problem with a 0.1% MIPGAP, the lower bound of which is used to compute the worst case benchmarking for the IBAD-UC algorithm solution quality. We denote γ to be the total optimal objective value comprised of operations and commitment components. γ_{async}, γ_c represent the optimal objective for the asynchronous method and the centralized method respectively. We present the centralized results in Table II where γ_c and $\lfloor\gamma_c\rfloor$ represent the associated objective and lower bound. We use this to calculate a conservative solution quality in terms of the optimality gap($\frac{(\gamma_{async}-\lfloor\gamma_c\rfloor)*100}{\lfloor\gamma_c\rfloor}$) in order to provide the worst case optimality gap for our algorithm. We compute the mean optimality gap among multiple runs of the asynchronous method for 75, 100 and 120 region cases.

Table II shows that, the asynchronous solution quality is highly consistent among the regions. The asynchronous method on an average is able to consistently solve the decentralized UC problem with less than 2% optimality gap. Drawing insights from Figure 4 and Figure 3, it can be argued that despite a highly asynchronous system, the variation in solution times as well as the solution quality are relatively small. Further, the optimal objective costs are close to that of the centralized solution indicating a robust solution with higher computational efficiency.

TABLE I: Time (secs) 75 Regions

	75 Regions				100 Regions				120 Regions			
	Asynchronous		Synchronous		Asynchronous		Synchronous		Asynchronous		Synchronous	
Time	Mean	% of Total	Mean	% of Total	Mean	% of Total	Mean	% of Total	Mean	% of Total	Mean	% of Total
Comp	1989.25	92.76	171.47	6.35	1128.81	94.28	130.61	7.36	1222.36	94.11	137.03	5.21
Comm	1.05	0.05	0.87	0.03	0.76	0.06	0.65	0.04	1.14	0.09	0.87	0.03
Idle	153.62	7.16	2528.97	93.61	67.42	5.63	1642.65	92.58	74.87	5.76	2489.31	94.73
Total	2144.43	-	2701.43	-	1197.32	-	1774.23	-	1298.80	-	2627.67	-

TABLE II: Centralized Solution

Total Objective (γ_c)	108316.6
Lower Bound ($\lfloor\gamma_c\rfloor$)	108218.5
Time (secs)	19139

TABLE III: Solution Quality

Regions	Mean Gap (%)	Std Dev.
75	1.891	0.881
100	1.305	0.285
120	1.611	0.122

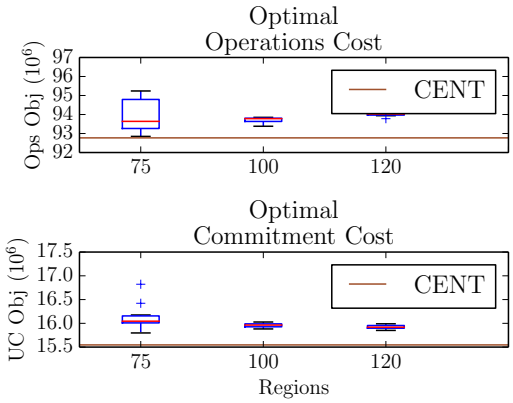

Fig. 6: Objective costs

F. Objective Costs

Figure 6 shows the objective cost comparison of IBAD-UC relative to the centralized solution with respect to the 75,100 and 120 region cases. We can see that the IBAD-UC yields commitment and operation decisions which are close enough to the centralized method. We can also observe the stability of the solutions within each region case owing to the valid inequalities enforced by IBAD-UC.

VI. Conclusion

In this paper we present a novel asynchronous decentralized solution methodology for solving the UC problem for large scale power systems. Unlike other asynchronous reformulations proposed in the past that leverage a master-slave hierarchical computational model, our IBAD-UC algorithm is decentralized in nature and intended for a real-time geographically distributed heterogeneous computing environment. Our decentralized problem formulation is constructed with a strong emphasis on the privacy of region level infrastructure data and incorporates a redundant privacy preserving valid inequality. Leveraging the valid inequality the proposed asynchronous method is able to offer considerable algorithmic improvements with respect to stability and robustness of the solution. We propose a controller mechanism that implements two-way message exchanges between regions at discrete global clock ticks without a significant computational burden.

We present HPC simulation studies based on a custom-made software framework developed by us to show the superior computational performance of the asynchronous method, along with stable solution quality. We also benchmark the asynchronous convergence characteristics with respect to the synchronous method and analyze the solution quality against that of the centralized method. Our experiments show that asynchronous methods offer a viable, robust and computationally efficient alternative to the state of the art synchronous decentralized methods.

References

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

[2] V. M. Z. Kibaek Kim, Cosmin G Petra, "An asynchronous bundle-trust-region method for dual decomposition of stochastic mixed-integer programming." http://www.mcs.anl.gov/ kibaekkim/AsyncDD-preprint.pdf.

[3] V. M. Z. Kibaek Kim, Mihai Anitescu, "An asynchronous decomposition algorithm for security constrained unit commitment under contingency events." http://www.mcs.anl.gov/ kibaekkim/PSCC-KimAnitescuZavala.pdf.

[4] A. Papavasiliou, S. S. Oren, and B. Rountree, "Applying high performance computing to transmission-constrained stochastic unit commitment for renewable energy integration," *IEEE Transactions on Power Systems*, vol. 30, pp. 1109–1120, May 2015.

[5] M. Assran and M. Rabbat, "An empirical comparison of multi-agent optimization algorithms," in *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 573–577, Nov 2017.

[6] A. Aytekin, *Asynchronous Algorithms for Large-Scale Optimization: Analysis and Implementation*. PhD thesis, KTH Royal Institute of Technology, 2017.

[7] A. Nedi, A. Olshevsky, and M. G. Rabbat, "Network topology and communication-computation tradeoffs in decentralized optimization," *Proceedings of the IEEE*, vol. 106, pp. 953–976, May 2018.

[8] O. Babaoglu, R. Davoli, L. A. Giachini, and M. G. Baker, "Relacs: A communications infrastructure for constructing reliable applications in large-scale distributed systems," in *Proceedings of the Twenty-Eighth Annual Hawaii International Conference on System Sciences*, vol. 2, pp. 612–621, Jan 1995.

[9] E. Wei and A. Ozdaglar, "On the o (1= k) convergence of asynchronous distributed alternating direction method of multipliers," in *Global conference on signal and information processing (GlobalSIP), 2013 IEEE*, pp. 551–554, IEEE, 2013.

[10] 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, pp. 1–122, Jan. 2011.

[11] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the admm in decentralized consensus optimization," *IEEE Transactions on Signal Processing*, vol. 62, pp. 1750–1761, April 2014.

[12] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE transactions on automatic control*, vol. 31, no. 9, pp. 803–812, 1986.

[13] J. Liu, S. J. Wright, C. Ré, V. Bittorf, and S. Sridhar, "An asynchronous parallel stochastic coordinate descent algorithm," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 285–322, 2015.

[14] M. G. Rabbat and K. I. Tsianos, "Asynchronous decentralized optimization in heterogeneous systems," in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pp. 1125–1130, IEEE, 2014.

[15] T.-H. Chang, "A proximal dual consensus admm method for multi-agent constrained optimization," *IEEE Transactions on Signal Processing*, vol. 64, no. 14, pp. 3719–3734, 2016.

[16] J. Eckstein, "A simplified form of block-iterative operator splitting and an asynchronous algorithm resembling the multi-block alternating direction method of multipliers," *Journal of Optimization Theory and Applications*, vol. 173, no. 1, pp. 155–182, 2017.
[17] R. Zhang and J. T. Kwok, "Asynchronous distributed admm for consensus optimization," in *Proceedings of the 31st International Conference on International Conference on Machine Learning*, vol. 32 of *ICML'14*, pp. 1701–1709, 2014.
[18] 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.
[19] P. Ramanan, M. Yildirim, E. Chow, and N. Gebraeel, "Asynchronous decentralized framework for unit commitment in power systems," *Procedia Computer Science*, vol. 108, pp. 665–674, 2017.
[20] Y. Wang, L. Wu, and J. Li, "A fully distributed asynchronous approach for multi-area coordinated network-constrained unit commitment," *Optimization and Engineering*, pp. 1–34, 2018.
[21] J. Guo, G. Hug, and O. Tonguz, "Asynchronous admm for distributed non-convex optimization in power systems," *arXiv preprint arXiv:1710.08938*, 2017.
[22] H. Kopetz and W. Ochsenreiter, "Clock synchronization in distributed real-time systems," *IEEE Transactions on Computers*, vol. 100, no. 8, pp. 933–940, 1987.
[23] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Consensus-based distributed optimization: Practical issues and applications in large-scale machine learning," in *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, pp. 1543–1550, IEEE, 2012.
[24] J. Ostrowski, M. F. Anjos, and A. Vannelli, "Tight mixed integer linear programming formulations for the unit commitment problem," *IEEE Transactions on Power Systems*, vol. 27, no. 1, pp. 39–46, 2012.
[25] L. Dalcin, R. Paz, M. Storti, and J. D., "Mpi for python: Performance improvements and mpi-2 extensions," *Journal of Parallel and Distributed Computing*, vol. 68, no. 5, pp. 655 – 662, 2008.
[26] R. D. Zimmerman, C. E. Murillo-Sanchez, and R. J. Thomas, "Matpower: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Transactions on Power Systems*, vol. 26, pp. 12–19, Feb 2011.

Paritosh Ramanan Paritosh Ramanan is a 4th year PhD Candidate in Computational Science and Engineering with the School of Industrial and Systems Engineering at Georgia Institute of Technology in Atlanta, Georgia. Prior to his PhD he earned a Masters in Computer Science from Georgia State University in Atlanta, Georgia in 2015 and obtained his Bachelors in Information Systems from Birla Institute of Technology and Science (BITS) Pilani, Goa Campus in 2013. His research focuses on developing decentralized algorithms for improved computational performance of large scale optimization problems through the use of parallel and distributed computing paradigms.

Murat Yildirim Dr. Murat Yildirim is an Assistant Professor in the Department of Industrial and Systems Engineering at Wayne State University. Prior to joining Wayne State, he worked as a postdoctoral fellow at the Georgia Institute of Technology (2016-2018), and obtained a Ph.D. degree in Industrial Engineering, and B.Sc. degrees in Electrical and Industrial Engineering from the same institution. Dr. Yildirim's research interest lies in advancing the integration of mathematical programming and data analytics in large scale energy systems. Specifically, he focuses on the modeling and the computational challenges arising from the integration of real-time sensor inferences into large-scale mixed integer programs (MIPs) used for optimizing and controlling networked systems.

Edmond Chow Edmond Chow is an Associate Professor in the School of Computational Science and Engineering at Georgia Institute of Technology. He previously held positions at D. E. Shaw Research and Lawrence Livermore National Laboratory. His research is in developing numerical methods specialized for high-performance computers, including asynchronous iterative methods, and applying these methods to solve large-scale scientific computing problems. Dr. Chow was awarded the 2009 ACM Gordon Bell prize and the 2002 U.S. Presidential Early Career Award for Scientists and Engineers (PECASE). He serves as Associate Editor for ACM Transactions on Mathematical Software and previously served as Associate Editor for SIAM Journal on Scientific Computing.

Nagi Gebraeel Dr. Nagi Gebraeel is the Georgia Power Early Career Professor and Professor in the Stewart School of Industrial and Systems Engineering at Georgia Tech. His research interests lie at the intersection of industrial predictive analytics and decision optimization models for large scale power generation applications. Dr. Gebraeel serves as an associate director at Georgia Tech's Strategic Energy Institute and the director of the Analytics and Prognostics Systems laboratory at Georgia Tech's Manufacturing Institute. Dr. Gebraeel was the former president of the Institute of Industrial Engineers (IIE) Quality and Reliability Engineering Division, and is currently a member of the Institute for Operations Research and the Management Sciences (INFORMS).