Decentralized Failure-Tolerant Optimization of Electric Vehicle Charging

Ignacio Aravena, *Member, IEEE*, Steve J. Chapin, and Colin Ponce

***Abstract*—We present a decentralized failure-tolerant algorithm for optimizing electric vehicle (EV) charging, using charging stations as computing agents. The algorithm is based on the alternating direction method of multipliers (ADMM) and it has the following features: (*i*) It handles coupling constraints for capacity, peak demand, and ancillary services. (*ii*) It does not require a central agent collecting information and performing coordination (e.g., an aggregator), instead all agents exchange information and computations are carried out in a fully decentralized fashion. (*iii*) It can withstand the failure of any number of computing agents, as long as the remaining computing agents are in a connected communications network. We construct this algorithm by reformulating the optimal EV charging problem in a decomposable form, amenable to ADMM, and then developing efficient decentralized solution methods for the subproblems dealing with coupling constraints. We conduct numerical experiments on industry-scale synthetic EV charging datasets, with up to 1 152 charging stations, using a high-performance computing cluster. The experiments demonstrate that the proposed algorithm can solve the optimal EV charging problem fast enough to permit the integration of EV charging with real-time electricity markets, even in the presence of failures.**

***Index Terms*—Electric vehicle charging, ADMM, decentralized algorithm, fail-proof algorithm.**

I. Introduction

THE ANTICIPATED massive adoption of electric vehicles (EVs) will bring some of the biggest challenges and transformational opportunities for power systems operation. It will inevitably mean that current load patterns, for which transmission and distribution systems have been designed, will drastically change, requiring either new infrastructure or new operating paradigms to maintain the reliability of the power grid. For example, the average family, driving light-duty passenger EVs, would see their electricity usage increase by almost 50%.[1] These changes in load are intensified by the progressive integration of intermittent resources in the generation mix.

EVs' characteristics distinguishing them from traditional loads. EVs can store energy in their batteries, bringing a form of distributed storage capacity to the power grid, and their instantaneous demand is very flexible, only requiring that net energy stored (the battery's State of Charge (SOC)) reaches a required level by a given time limit. This flexibility, if harnessed, could help maintain and improve the reliability of the grid, even in the presence of intermittent resources. However, fully harnessing EVs' flexibility is not straightforward because of their uncertain individual nature and the large number of them, which complicates EVs' participation in electricity markets and real-time operations as a managed resource.

Managing EV charging involves, roughly, three different stages [3]:

1) *Forward scheduling*, where EV charging needs are forecast days to hours in advance and bid in forward electricity markets (day-ahead, intraday markets), allowing for the entire system to be scheduled while accounting for EV demand and flexibility.
2) *Near-real-time scheduling*, where EV charging needs are forecast a couple of minutes in advance in order to couple EV charge with 5-minute system dispatch.
3) *Real-time control*, where EV charge actually happens, following the patterns set in near-real-time scheduling as close as possible, correcting for deviations and responding to instructions from the system operator to deliver ancillary services.

These stages have different computational needs and resources, e.g., forward scheduling can be performed by an aggregator or other central entity because there is enough time to collect information and recover from any errors, but near-real-time scheduling and real-time control must be handled in a local fashion because collecting information from millions of devices and performing computations at scale, reliably, with deadlines only seconds or minutes away is not realistic. Furthermore, in reality, imperfect communications infrastructure, disconnections, and other unexpected events increase the challenges faced by near-real-time scheduling and real-time control of EV charging.

Manuscript received November 17, 2020; revised March 21, 2021; accepted May 5, 2021. Date of publication May 14, 2021; date of current version August 23, 2021. This work was supported in part by the U.S. Department of Energy (DOE) Vehicle Technologies Office under Agreement 34828, and in part by the DOE by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. Paper no. TSG-01732-2020. *(Corresponding author: Ignacio Aravena.)*

The authors are with the Lawrence Livermore National Laboratory, Livermore, CA 94550 USA (e-mail: aravenasolis1@llnl.gov; chapin8@llnl.gov; ponce11@llnl.gov).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TSG.2021.3080583.

Digital Object Identifier 10.1109/TSG.2021.3080583

[1]Based on 54 daily vehicle miles/household [1], a 4 mi/kWh energy usage in a typical modern EV, and an average household electricity consumption of 30 kWh [2].

In this paper, we focus on the near-real-time scheduling stage. We present a new algorithm for recurringly solving the optimal EV charging problem in a fully decentralized fashion, thereby tackling the two aforementioned challenges in harnessing EV charging flexibility at this stage: scaling to real-world fleet and demand sizes, and tolerance to real-world failures.

A. Literature Review

A significant amount of research in recent years has been devoted to the problem of coordinating EV chargers in a scalable fashion, from the perspectives of both scheduling and real-time control. Across these studies, the terms *distributed* and *decentralized* are often used interchangeably to characterize parallel algorithms. Following [4], we use these terms to refer to two different parallel algorithm classes:

Distributed algorithms have multiple *workers* communicate with a *central coordinator* to compute common quantities.

Decentralized algorithms have no *central coordinator*, and all computations are performed collectively by multiple *workers*.

Most existing approaches for EV charge coordination fall under the first category. In the following, we review a subset of them, representative of the most salient ideas in the area. Wen *et al.* [5] uses ADMM [6] to relax a mixed-integer optimal EV charging problem, where EVs communicate their intended consumption to an aggregator, which updates averages and dual multipliers, and broadcasts them back to the EVs. Joo and Ilić [7] proposes a distributed algorithm based on Lagrange relaxation to solve the more general optimal demand response problem. Gan *et al.* [8] develops a specialized trust-region distributed approach for finding the optimal valley-filling EV charging schedule. Vayá *et al.* [9] proposes a distributed ADMM approach for scheduling EV charging under uncertainty considering only physical EV constraints. Le Floch *et al.* [10] applies Nesterov's smoothing [11] to the optimal EV charging problem and uses gradient ascent and incremental stochastic gradient methods, implemented in a distributed fashion, to solve the dual problem. Ghavami *et al.* [12] proposes a distributed primal-dual subgradient method to find the optimal valley-filling EV charging schedule while considering thermal capacity limits of distribution lines within a transportation model. Grammatico [13] proposes a distributed game theoretic approach for EV charging control where a central controller gathers aggregate information of agents and broadcasts a pricing signal to all agents of each iteration. Wang *et al.* [14] presents a distributed implementation of the A* algorithm to optimally schedule renewable generation and EV charging, using a transportation model for the power grid. These approaches, as well as similar ones found in the literature, fail our scaling and resilience requirements because of the communications/computation bottleneck imposed by the central coordinator and the single point of failure (or attack) it introduces.

A handful of authors have proposed decentralized methods for optimal EV charging or smart grid optimization under different assumptions and diverse underlying ideas. Ma *et al.* [15] proposes a decentralized algorithm for scheduling EV charging based on non-cooperative games. The paper assumes the existence of an energy-only market, that prices are the same for all vehicles, and that these prices follow a prespecified function of the margin between demand and installed capacity. Adika and Wang [16] follows a similar approach, using a supplementary pricing strategy to influence the charging/discharging of batteries more directly. Rahbari-Asr and Chow [17] develops a failure-proof decentralized heuristic based on the first-order optimality conditions of a single-period optimal EV charging model, where communication between computation nodes is handled by resilient consensus algorithms. Zhang *et al.* [18] proposes a distributed Frank-Wolfe decomposition method for optimal valley-filling EV charging and a decentralized ADMM method for optimizing EV charging considering linearized and unbalanced distribution network constraints. The latter approach, which shares many aspects with the proximal message passing approach of Kraning *et al.* [19] for decentralized power grid scheduling, becomes decentralized as a consequence of the graph structure of power grid constraints, and therefore it cannot directly accomodate features that couple agents outside the power flow equations, such as ancillary services. Münsing *et al.* [20] uses the same underlying algorithm as [9] to optimize the operation of a microgrid, and makes the method decentralized by carrying out the dual variable update step, which was originally done by an aggregator, through a *blockchain* where each local update is recorded. Li *et al.* [21] proposes a decentralized approach based on non-cooperative games for optimal EV charging with capacity restrictions for all distribution branches in tree-shaped feeders. Atallah *et al.* [22] formulates a mixed-integer optimal EV charging problem, where EVs can be assigned to neighboring stations instead of being fixed at a given station, and propose a decentralized heuristic based on non-cooperative games for finding high-quality solutions of the problem in actual operation. Qin *et al.* [23] studies the coordination of economic dispatch (without transmission) and demand response using a decentralized ADMM algorithm where the dual variable update is performed through a consensus protocol. Yang *et al.* [24] proposes a decentralized approach based on non-cooperative games for optimal storage scheduling which is complemented with peer-to-peer contracts stored in a *blockchain*, creating a fully decentralized physical and financial scheduling platform. Wan *et al.* [25] studies retail EV charging modeled as a non-cooperative game between EVs behind the same meter, for which the authors propose a distributed equilibrium seeking algorithm based on ADMM where the best response is computed in a decentralized manner.

Another significant aspect of the EV charging coordination problem is its integration with existing and envisioned electricity markets. In this regard, all the work reviewed so far considers exclusively energy-only markets, whereas optimal EV charge scheduling also has the potential to significantly contribute to ancillary services, particularly to regulation and spinning reserves [26]. With the aim of exploiting this potential, Lin *et al.* [27] proposes a formulation for optimal

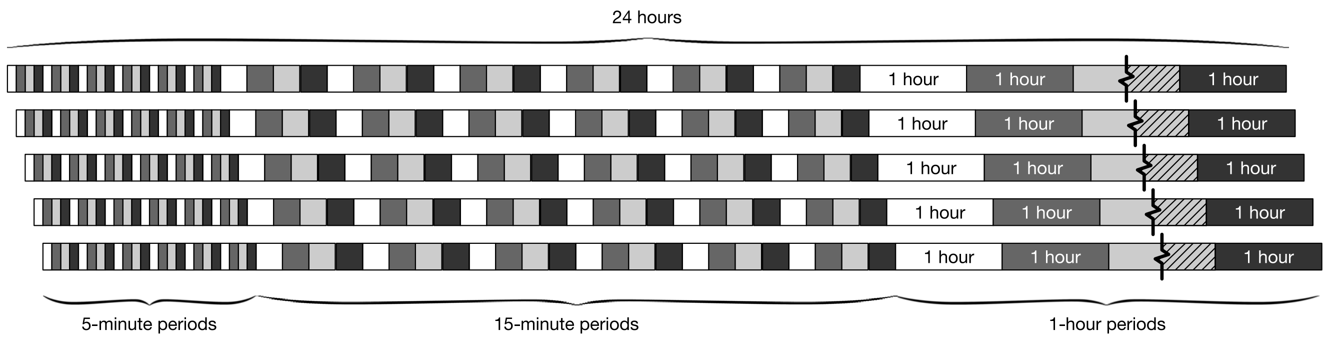

Fig. 1. Rolling horizon for EV charge schedule computation every 5 minutes with increasing interval sizes. A computed schedule looks 24 hours into the future, comprising 24 5-minute periods (2 hours), 24 15-minute periods (6 hours), and 16 1-hour periods. Scheduling decisions pertaining only to the first 5 minutes are implemented, whereas future intervals are only modeled to prevent short-sighted decisions for the first 5 minutes. In stochastic programming terms this corresponds to a *deterministic look-ahead policy* [32] for the optimal EV charging problem.

EV charging with frequency regulation as a vehicle-to-grid service. The paper takes advantage of the zero-energy nature of frequency regulation to obtain a compact formulation and proposes a distributed algorithm, similar to that of [8], for solving the resulting optimization problem in practice. Juul *et al.* [28] studies how to integrate EV storage while charging into CAISO's ancillary services markets and propose a convex model for optimal tracking of ancillary services, activation signals while respecting EV charging constraints. Peng and Hao [29] proposes a decentralized control algorithm for active power compensation provided by EVs on distribution grids based on a task-swap mechanism among charging stations, where tasks are discrete active power compensation portions. The algorithm is experimentally shown to mitigate sudden (significant) load changes at the distribution system level.

Finally, it is worthwhile to mention that decentralized approaches based on ADMM and other methods are very common in large-scale optimization, particularly in *consensus* and *sharing* problems, among others [6]. For these classes of problems, decentralization is the consequence of either local coupling constraints (relating only pairs of neighboring agents) or simple global coupling constraints leading to trivial global updates such as averaging, which can be handled via decentralized reduction operations. Recent literature on the topic has focused on creating robust variants of the ADMM method. Li *et al.* [30] proposes a decentralized robust ADMM method for consensus problems, where misbehaving agents are detected based on statistical thresholds and temporally replaced in the computation with estimates from the misbehaving agents' neighbors. Münsing and Moura [31] analyzes different types of cyberattacks on distributed and decentralized ADMM methods, and proposes to use Hessian approximations–constructed from iterates–for detecting noise injection attacks via convexity verification. It is expected that future research in distributed and decentralized EV charging methods will incorporate these, and other, cybersecurity considerations, as they are a requisite for storage and EV management over public communication networks, such as the Internet.

B. Contributions

Our main contribution is the design of a fully-decentralized algorithm for solving the optimal EV charge scheduling problem that arises in near-real-time scheduling of EV charge. The algorithm design takes into consideration local constraints at each charging station or hub, capacity constraints of the power grid, peak-demand charges and provision of ancillary services to the grid. It is meant to be used within a rolling horizon scheme, such as the one presented in Fig. 1, where 5-minute schedules are implemented by the real-time control and forecasts for future intervals are updated at each run, locally at each station or hub.

Though designed for our formulation of the near-real-time optimal EV charge scheduling problem, our algorithm is applicable to many other convex optimization problems with global and groupal coupling constraints. Specifically, our algorithm can solve problems of the form

$$\begin{aligned}\min_{x_i \in \mathcal{X}_i} \quad & \sum_{i=1}^{I} f_i(\boldsymbol{x}_i) + \sum_{k=1}^{K} \max_{j \in \mathcal{J}(k)} \left(\sum_{i \in \mathcal{I}^g(k)} g_{i,j}(\boldsymbol{x}_i) \right) \\ \text{s.t.} \quad & \sum_{i \in \mathcal{I}^h(l)} h_{i,l}(\boldsymbol{x}_i) \le B_l \quad l = 1, \ldots, L,\end{aligned}$$

where $\boldsymbol{x}_i$ is the variable block associated with agent i; $\mathcal{X}_i,\ i = 1, \ldots, I$ is a closed convex set; f_i, $g_{i,j}$ and $h_{i,l}$ are convex functions; $\mathcal{J}(k),\ k = 1, \ldots, K$ is a partition of $\{1, \ldots, I\}$; and $\mathcal{I}^g(k) \subseteq \{1, \ldots, I\},\ k = 1, \ldots, K$ and $\mathcal{I}^h(l) \subseteq \{1, \ldots, I\},\ l = 1, \ldots, L$. This problem structure arises in several variants of optimal EV charging, such as all the formulations in the references of the previous section, as well as other problems in smart grids, particularly those where ancillary services are considered [4], [20], [27].

The key features of the proposed algorithm are the following. (*i*) It avoids inefficiencies introduced when using a blockchain for the aggregation step [20], large expansions of the constraint space (e.g., reformulating the constraints over a graph [30]), or re-formulating the problem as a consensus or exchange problem. The approach does not rely on the graph structure of the power flow constraints [18] to perform decentralized updates, making it able to handle ancillary services and enabling its applicability outside the power systems field. (*ii*) It is failure-tolerant, in the sense that if agent i fails, or it is removed from the system due to security/cybersecurity considerations, the algorithm will find the solution to the problem without agent i, as long as the underlying communication network between continuing agents remains connected. (*iii*) Our algorithm is based on ADMM and it is mathematically equivalent, in terms of iterations, to the distributed approach of [9]. Therefore it finds the exact solution of the optimal EV charging problem, as opposed to heuristics [22] or game-theoretic approaches converging only in the limit with infinitely many agents [15], [16].

In terms of formulation of the optimal EV charging problem, we advance the state-of-the-art in two respects: (*i*) we model peak power procurement by charging stations (common in distribution pricing) and (*ii*) we model spinning reserves, which are a nonzero-energy type of services, requiring the consideration of energy recovery in the case of activation so as to meet the target SOC of EVs.

C. Notation and Paper Organization

Unless indicated otherwise, we use lowercase for optimization variables, uppercase for parameters and calligraphic symbols for sets. We denote vectors using boldface and use partial indexing, e.g., for $\boldsymbol{x} \in \mathbb{R}^{N\times M}$, with elements $x_{i,j}$, we denote $\boldsymbol{x}_i = (x_{i,1}, \dots, x_{i,M})$.

The rest of the paper is organized as follows. Section II presents a stylized version of our optimal EV charging problem, focusing mostly on its coupling constraints. Section III presents a distributed ADMM approach for solving the optimal EV charging problem. Section IV derives our decentralized failure-tolerant ADMM approach. Section V presents numerical results of our decentralized algorithm on synthetic instances of the optimal EV charging problem. Finally, Section VI concludes this paper and suggests directions for future research.

II. Optimal EV Charging Problem

We model the optimal EV charging problem from the perspective of charging station owners or operators, that is, we minimize the total electricity bill of all charging stations. We consider that each EV arrives at a certain (predefined) charging station, and it indicates how long it will be plugged-in and what should be its SOC by the time it unplugs. We consider only grid-to-vehicle power injections (V1G) for simplicity of our formulation. However, our algorithm does not make any assumptions based on V1G technology and it could already incorporate vehicle-to-grid injections (V2G).

Fig. 2 presents a schematic of the charging infrastructure considered in this work. Each *charging station* $s \in \mathcal{S}$ is behind a certain *meter* $m(s) \in \mathcal{M}$ (the point of connection with the distribution network), which in turn is connected to a certain *distribution feeder* $f \in \mathcal{F} : s \in \mathcal{S}(f)$. Charging station s buys electricity from the distribution network at the energy price

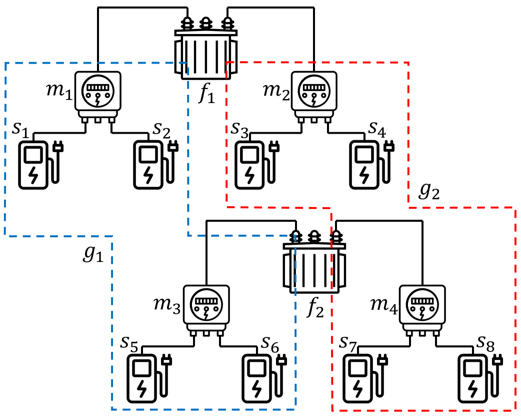

Fig. 2. Example topology of the charging infrastructure considered in this work.[3] Here $\mathcal{S}(m_1) = \{s_1, s_2\}$, $\mathcal{S}(f_1) = \{s_1, s_2, s_3, s_4\}$, $\mathcal{S}(g_1) = \{s_1, s_2, s_5, s_6\}$ (reserve group g_1 delimited with by dashed blue line), and so forth. Notably, stations in different feeders can be part of the same reserve group, extending coupling constraints beyond the limits of individual feeders.

at its corresponding meter, $\Pi^{\text{energy}}_{m(s)}$. The peak power drawn from the grid at meter m (by all stations in $\mathcal{S}(m)$) over the horizon $\mathcal{T} = \{1, \dots, T\}$ is priced at $\Pi^{\text{peak}}_m > 0$ and billed to the charging stations in $\mathcal{S}(m)$. At the same time, each charging station s participates in a *reserve group* $g \in \mathcal{G} : s \in \mathcal{S}(g)$ which provides aggregated ancillary services to the grid. We consider that charging stations can fail at any point, in the following sense:

Definition 1: Failure. We understand failure as the disconnection of a component from the system. In the case of charging stations, we consider failed charging stations as if they were powered off, that is, they do not interact with other charging stations nor charge electric vehicles.

This failure mode, while fairly simple, is able to represent a wide range of possible malfunctions in the real world. For example, feeder and meter outages can be represented by multiple failing stations. Any type of malfunction that can be detected by a real-time control algorithm (e.g., adversarial manipulation) and causes the control algorithm to put the station offline, is also captured. That said, we do not deal with detection and handling of agent malfunctions in the present work, and we use *failure* as defined in Def. 1 throughout the rest of the paper.

For ancillary services, we focus only on spinning reserve, which is sold to the system by each reserve group at $\Pi^{\text{spin}}_{g,t}$ for intervals in $\{T^{\text{spin}} + 1, \dots, T\}$, where T^{spin} is the number of intervals between the last spinning reserve market clearing and real-time operation. For intervals in $\mathcal{T}^{\text{spin}} = \{1, \dots, T^{\text{spin}}\}$, the amount of spinning reserve to be provided has already been determined in previous runs of the scheduling process, and a penalty of $\Pi^{\text{fail}} > 0$ is billed to charging stations per unit of reserves they fail to provide. Spinning reserves can be activated or not at each interval t of the horizon $\mathcal{T}$ and, when activated, charging operations in later intervals must be adjusted in order to meet the target SOC of each EV at

[3]Station, meter and transformer illustrations sourced from *Noun Project*, https://thenounproject.com/.

the time it unplugs. We take a worst-case approach, ensuring that no matter the reserve activation signal, there exists, always, a feasible charging trajectory to meet the target SOC. We refer the reader to Section EC.1 of the electronic companion [33] for a detailed formulation of constraints internal to charging stations, encompassing our EV models providing spinning reserves. In the following we focus on the coupling constraints between charging stations, which is where our core contributions lie.

Let $p^{\mathbb{E}}_{s,t}, p^{\max}_{s,t}$ be the expected and maximum power draw, respectively, of station s at interval t, over all possible realizations of activation of spinning reserves; θ_m be the expected peak demand at meter m; $\phi_{g,t}$ be promised but not delivered spinning reserve by group g at interval $t \in \mathcal{T}^{\text{spin}}$; and $r^{\text{spin}}_{s,t}$ be the spinning reserve provided by station s at interval t. Additionally, let $\boldsymbol{u} \in \mathbb{R}^{|\mathcal{S}|\times T}$, $\boldsymbol{v} \in \mathbb{R}^{|\mathcal{S}|\times T}$ and $\boldsymbol{w} \in \mathbb{R}^{|\mathcal{S}|\times T^{\text{spin}}}$ be clone variables of $\boldsymbol{p}^{\max}$, $\boldsymbol{p}^{\mathbb{E}}$ and $\boldsymbol{r}^{\text{spin}}$. Then, with all the previous considerations, the optimal EV charging problem can be formulated as (1) – (8).

$$\min_{\substack{p,r\\u,v,w\\\theta,\phi}} \sum_{s\in\mathcal{S}}\left(\sum_{t\in\mathcal{T}}\Pi^{\text{energy}}_{m(s),t}\,p^{\mathbb{E}}_{s,t} - \sum_{t=T^{\text{spin}}+1}^{T}\Pi^{\text{spin}}_{g(s),t}\,r^{\text{spin}}_{s,t}\right) + \sum_{m\in\mathcal{M}}\Pi^{\text{peak}}_{m}\,\theta_m + \sum_{g\in\mathcal{G}}\sum_{t=1}^{T^{\text{spin}}}\Pi^{\text{fail}}\,\phi_{g,t} \tag{1}$$

$$\text{s.t.} \sum_{s\in\mathcal{S}(f)} u_{s,t} \le P_{f,t} \quad \forall f\in\mathcal{F}, t\in\mathcal{T} \tag{2}$$

$$\sum_{s\in\mathcal{S}(m)} v_{s,t} \le \theta_m \quad \forall m\in\mathcal{M}, t\in\mathcal{T} \tag{3}$$

$$\sum_{s\in\mathcal{S}(g)} w_{s,t} \ge R^{\text{spin}}_{g,t} - \phi_{g,t} \quad \forall g\in\mathcal{G}, t\in\mathcal{T}^{\text{spin}} \tag{4}$$

$$\boldsymbol{p}^{\max}_s = \boldsymbol{u}_s \quad \forall s\in\mathcal{S} \quad (\boldsymbol{\mu}_s) \tag{5}$$

$$\boldsymbol{p}^{\mathbb{E}}_s = \boldsymbol{v}_s \quad \forall s\in\mathcal{S} \quad (\boldsymbol{\nu}_s) \tag{6}$$

$$\boldsymbol{r}^{\text{spin}}_{s,\mathcal{T}^{\text{spin}}} = \boldsymbol{w}_s \quad \forall s\in\mathcal{S} \quad (\boldsymbol{\xi}_s) \tag{7}$$

$$(\boldsymbol{p}^{\max}_s, \boldsymbol{p}^{\mathbb{E}}_s, \boldsymbol{r}^{\text{spin}}_s) \in \mathcal{D}_s \qquad \forall s\in\mathcal{S} \tag{8}$$

The objective function (1) corresponds to the total cost of operating the charging stations, including energy charges, peak demand charges, spinning reserve penalties and revenues for spinning reserve provision. Constraint (2) ensures that total load from charging stations $s \in \mathcal{S}(f)$ does not surpass the net capacity (capacity minus inflexible demand) $P_{f,t}$ of the distribution feeder f, at all intervals $t \in \mathcal{T}$. Constraint (3) computes the expected peak demand at each meter m using an epigraph formulation for the max operator. Constraint (4) ensures that the spinning reserves promised by each reserve group are either provided or the corresponding penalty is charged to the group. Constraints (5) – (7), with dual variables $\boldsymbol{\mu} \in \mathbb{R}^{|\mathcal{S}|\times T}$, $\boldsymbol{\nu} \in \mathbb{R}^{|\mathcal{S}|\times T}$ and $\boldsymbol{\xi} \in \mathbb{R}^{|\mathcal{S}|\times T^{\text{spin}}}$, ensure that $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$ are clones of $\boldsymbol{p}^{\max}, \boldsymbol{p}^{\mathbb{E}}, \boldsymbol{r}^{\text{spin}}$. Constraint (8) enforces the internal constraints of charging stations, represented here by the polyhedral set $\mathcal{D}_s$ (defined formally in Section EC.1 of [33]), on power draw and reserves.

III. Distributed Decomposition Algorithm

We can solve problem (1) – (8) using ADMM by iteratively solving (*i*) *station subproblems*

$$\begin{aligned}
&\left(\boldsymbol{p}^{\max,k+1}_s,\ \boldsymbol{p}^{\mathbb{E},k+1}_s, \boldsymbol{r}^{\text{spin},k+1}_s\right)\\
&= \arg\min_{p,r} \sum_{t\in\mathcal{T}}\Pi^{\text{energy}}_{m(s),t}\,p^{\mathbb{E}}_{s,t} - \sum_{t=T^{\text{spin}}+1}^{T}\Pi^{\text{spin}}_{g(s),t}\,r^{\text{spin}}_{s,t}\\
&\quad + \left\langle \boldsymbol{\mu}^k_s - \rho\boldsymbol{u}^k_s, \boldsymbol{p}^{\max}_s\right\rangle + \frac{\rho}{2}\left\|\boldsymbol{p}^{\max}_s\right\|_2^2\\
&\quad + \left\langle \boldsymbol{\nu}^k_s - \rho\boldsymbol{v}^k_s, \boldsymbol{p}^{\mathbb{E}}_s\right\rangle + \frac{\rho}{2}\left\|\boldsymbol{p}^{\mathbb{E}}_s\right\|_2^2\\
&\quad \left\langle \boldsymbol{\xi}^k_s - \rho\boldsymbol{w}^k_s, \boldsymbol{r}^{\text{spin}}_{s,\mathcal{T}^{\text{spin}}}\right\rangle + \frac{\rho}{2}\left\|\boldsymbol{r}^{\text{spin}}_{s,\mathcal{T}^{\text{spin}}}\right\|_2^2\\
&\quad \text{s.t.}\ \left(\boldsymbol{p}^{\max}_s, \boldsymbol{p}^{\mathbb{E}}_s, \boldsymbol{r}^{\text{spin}}_s\right) \in \mathcal{D}_s
\end{aligned} \tag{9}$$

for all stations $s \in \mathcal{S}$; (*ii*) *capacity subproblems*

$$\begin{aligned}
\boldsymbol{u}^{k+1}_{\mathcal{S}(f),t} &= \arg\min_{u} \sum_{s\in\mathcal{S}(f)}\left(\left(-\mu^k_{s,t} - \rho\, p^{\max,k+1}_{s,t}\right)u_{s,t} + \frac{\rho}{2}u^2_{s,t}\right)\\
&\quad \text{s.t.} \sum_{s\in\mathcal{S}(f)} u_{s,t} \le P_{f,t} \quad (\alpha_{f,t})
\end{aligned} \tag{10}$$

for all feeders $f \in \mathcal{F}$ and intervals $t \in \mathcal{T}$; (*iii*) *peak demand subproblems*

$$\begin{aligned}
\left(\boldsymbol{v}^{k+1}_{\mathcal{S}(m)}, \cdot\right) &= \arg\min_{v,\theta}\ \Pi^{\text{peak}}_m\theta_m\\
&\quad + \sum_{s\in\mathcal{S}(m)}\left(\left\langle -\boldsymbol{\nu}^k_s - \rho\,\boldsymbol{p}^{\mathbb{E},k+1}_s, \boldsymbol{v}_s\right\rangle + \frac{\rho}{2}\|\boldsymbol{v}_s\|_2^2\right)\\
&\quad \text{s.t.} \sum_{s\in\mathcal{S}(m)} v_{s,t} \le \theta_m \quad \forall t\in\mathcal{T}\left(\beta_{m,t}\right)
\end{aligned} \tag{11}$$

for all meters $m \in \mathcal{M}$; and (*iv*) *ancillary services subproblems*

$$\begin{aligned}
\left(\boldsymbol{w}^{k+1}_{\mathcal{S}(g),t}, \cdot\right) &= \arg\min_{w,\phi}\ \Pi^{\text{fail}}\phi_{g,t}\\
&\quad + \sum_{s\in\mathcal{S}(g)}\left(\left(-\xi^k_{s,t} - \rho\, r^{\text{spin},k+1}_{s,t}\right)u_{s,t} + \frac{\rho}{2}w^2_{s,t}\right)\\
&\quad \text{s.t.} \sum_{s\in\mathcal{S}(g)} w_{s,t} \ge R^{\text{spin}}_{g,t} - \phi_{g,t}\left(\gamma_{g,t}\right)
\end{aligned} \tag{12}$$

for all groups $g \in \mathcal{G}$, intervals $t \in \mathcal{T}^{\text{spin}}$, as indicated in the distributed implementation of Fig. 3. This algorithm corresponds to the generic ADMM algorithm described in [6] with variable blocks $\boldsymbol{x}^k = [(\boldsymbol{p}^{\max,k})^{\mathsf{T}}\,(\boldsymbol{p}^{\mathbb{E},k})^{\mathsf{T}}\,(\boldsymbol{r}^{\text{spin},k}_{\mathcal{T}^{\text{spin}}})^{\mathsf{T}}]^{\mathsf{T}}$, $\boldsymbol{z}^k = [(\boldsymbol{u}^k)^{\mathsf{T}}\,(\boldsymbol{v}^k)^{\mathsf{T}}\,(\boldsymbol{w}^k)^{\mathsf{T}}]^{\mathsf{T}}$, and $\boldsymbol{y}^k = [(\boldsymbol{\mu}^k)^{\mathsf{T}}\,(\boldsymbol{\nu}^k)^{\mathsf{T}}\,(\boldsymbol{\xi}^k)^{\mathsf{T}}]^{\mathsf{T}}$, and with primal and dual residuals:

$$\begin{aligned}
\Delta^{\text{primal}} &= \|\boldsymbol{x}^{k+1} - \boldsymbol{z}^{k+1}\|_2\\
\Delta^{\text{dual}} &= \rho\cdot\|\boldsymbol{z}^{k+1} - \boldsymbol{z}^k\|_2.
\end{aligned} \tag{13}$$

The need for a central computation agent, the *aggregator*, (*i*) creates a communication bottleneck (every *station* $s \in \mathcal{S}$ needs to exchange information with the *aggregator* at each iteration; steps 3–6 of the *aggregator* pseudocode,

```
Aggregator:
 1: Set $\boldsymbol{u}^0 = \boldsymbol{v}^0 = \boldsymbol{\mu}^0 = \boldsymbol{\nu}^0 = \boldsymbol{0}$, and $\boldsymbol{w}^0 = \boldsymbol{\xi}^0 = \boldsymbol{0}$.
 2: for $k = 0, 1, \ldots, K$ do
 3:     for $s \in \mathcal{S}$ do
 4:         Send $\boldsymbol{u}_s^k, \boldsymbol{v}_s^k, \boldsymbol{w}_s^k, \boldsymbol{\mu}_s^k, \boldsymbol{\nu}_s^k, \boldsymbol{\xi}_s^k$ to Station $s$.
 5:     for $s \in \mathcal{S}$ do
 6:         Receive $\boldsymbol{p}_s^{\max,k+1}, \boldsymbol{p}_s^{\mathbb{E},k+1}, \boldsymbol{r}_{s,\mathcal{T}^{\text{spin}}}^{\text{spin},k+1}$ from Station $s$.
 7:     for $f \in \mathcal{F}, t \in \mathcal{T}$ do                    ▷ z-update
 8:         Solve (10) to compute $\boldsymbol{u}_{\mathcal{S}(f),t}^{k+1}$.
 9:     for $m \in \mathcal{M}$ do
10:         Solve (11) to compute $\boldsymbol{v}_{\mathcal{S}(m)}^{k+1}$.
11:     for $g \in \mathcal{G}, t \in \mathcal{T}$ do
12:         Solve (12) to compute $\boldsymbol{w}_{\mathcal{S}(g),t}^{k+1}$.
13:     if $\Delta^{\text{primal}} \le \epsilon^{\text{primal}}$ and $\Delta^{\text{dual}} \le \epsilon^{\text{dual}}$ then
14:         Terminate all processes.
15:     for $s \in \mathcal{S}$ do                                   ▷ y-update
16:         Let $\boldsymbol{\mu}_s^{k+1} = \boldsymbol{\mu}_s^k + \rho(\boldsymbol{p}_s^{\max,k+1} - \boldsymbol{u}_s^{k+1})$.
17:         Let $\boldsymbol{\nu}_s^{k+1} = \boldsymbol{\nu}_s^k + \rho(\boldsymbol{p}_s^{\mathbb{E},k+1} - \boldsymbol{v}_s^{k+1})$.
18:         Let $\boldsymbol{\xi}_s^{k+1} = \boldsymbol{\xi}_s^k + \rho(\boldsymbol{r}_s^{\text{spin},k+1} - \boldsymbol{w}_s^{k+1})$.

Station $s \in \mathcal{S}$:
 1: for $k = 0, 1, \ldots, K$ do
 2:     Receive $\boldsymbol{u}_s^k, \boldsymbol{v}_s^k, \boldsymbol{w}_s^k, \boldsymbol{\mu}_s^k, \boldsymbol{\nu}_s^k, \boldsymbol{\xi}_s^k$ from Aggregator.
 3:     Solve (9) to compute $\boldsymbol{p}_s^{\max,k+1}, \boldsymbol{p}_s^{\mathbb{E},k+1}, \boldsymbol{r}_{s,\mathcal{T}^{\text{spin}}}^{\text{spin},k+1}$. ▷ $x$-update
 4:     Send $\boldsymbol{p}_s^{\max,k+1}, \boldsymbol{p}_s^{\mathbb{E},k+1}, \boldsymbol{r}_{s,\mathcal{T}^{\text{spin}}}^{\text{spin},k+1}$ to Aggregator.
```

Fig. 3. Distributed ADMM algorithm for problem (1) – (8), derived from [6], [9]. The algorithm uses a *master-worker* design, with the *Aggregator* corresponding to the *master* and each *Station* corresponding to a *worker*. Right-aligned comments demark the start of the major ADMM steps (*x-update*, *z-update*, or *y-update*) [6] in the computations carried out by the algorithm.

and steps 2 and 4 of the *station* pseudocode in Fig. 3), which poses scalability challenges for the approach, as well as (*ii*) introduces a single point of failure (i.e., the *aggregator*) for the entire computation process. Single points of failures are undesirable in any system (e.g., preventing them is the reason behind the N-1 criterion used to operate high-voltage power grids) because they limit the reliability of the system to their own, in terms of withstanding random errors, and they incentivise potential adversaries to find ways to attack these critical components in order to make the entire system fail. Given these limitations, in the next section, we develop a new method in which the functions of the *aggregator* are shared among the *stations*, so that the *aggregator* is no longer necessary, removing both the communication bottleneck and single point of failure from the EV charge scheduling process.

IV. Decentralized Failure-Tolerant Decomposition Algorithm

Our approach to decentralize the algorithm in Fig. 3 can be summarized as follows. We derive specialized solution methods for updating the clone variables (*z-update*, steps 7–12 for the *aggregator*) that depend only on aggregate quantities and local computations at the *stations*. These aggregates are sums of local quantities at each *station* and they are computed using decentralized (failure-tolerant) reduction operations. We use the same reduction operations to compute primal and dual

```
1: procedure allreduce_r($x_i$, $\otimes$, $p$)
2:     $\hat{x}_i \leftarrow x_i$.
3:     for $k = 1, \ldots, p$ do
4:         Send $\hat{x}_i$ to agents $(i+\ell)\%p$ for $\ell = 1, ..., r$.
5:         $\ell^* \leftarrow \min\{\ell \in \{1, ..., r\} | \text{received } \hat{x}_{i-\ell}\}$
6:         $\hat{x}_i \leftarrow \hat{x}_{i-\ell^*} \otimes x_i$.
```

Fig. 4. Decentralized (resilient) allreduce implementation. Procedure takes as input the local data x_i, the operation to perform the reduce with $\otimes$, and the agents over which the reduction is to be performed p. The procedure works by building a ring in the communications graph, where each agent sends the same information to $r \in \mathbb{N}$ ($r > 0$) neighbors, introducing redundancy in the computation, in case an agent or communications channel fails during the allreduce procedure.

residuals (steps 13–14 for the *aggregator*) in a decentralized fashion and perform dual updates (*y-update*, steps 15–18 for the *aggregator*) locally at each *station*. In this setting, the failure tolerance of our algorithm comes, in part, as a by-product of our use of reduction operations, as a failed station no longer contributes to the aggregates for updating clone variables, effectively disappearing from the optimization problem being solved.

In the following, first, we briefly describe the properties and implementation of decentralized reduction operations, then, we present our decentralized update methods for clone variables, and conclude with a full description of our decentralized scheduling algorithm for optimal EV charging.

A. Allreduce Operations

An *allreduce* operation is a common communication collective in which every computing agent i in a group holds some data x_i, and the group collectively calculates

$$\bar{x} = \bigotimes_i x_i,$$

where $\otimes$ is any operation that is both commutative and associative, and all agents receive the answer. In our case, we use $\otimes = +$, as we need to sum decentralized pieces of data.

Allreduce has been well-studied in the past in the context of computations on reliable computer clusters, and a number of very efficient algorithms exist (see, e.g., [34]). Furthermore, many of these algorithms are decentralized; in the context of occasional failures, one of these methods combined with a simple check-and-retry procedure is likely ideal.

In unreliable environments, where greater resilience is required, it is advisable to use a resilient allreduce method. Previous work has investigated gossip-based resilient allreduce algorithms (see, e.g., [35]), which can be effective if low accuracy is acceptable. We present a resilient allreduce design in Fig. 4 that only introduces a bounded amount of noise when failures occur, and is exact otherwise, that is, it is computationally less expensive than the algorithm of [35] at the cost of larger errors in the case of failures, which are acceptable for our application.

With the algorithm of Fig. 4, the system as a whole is resilient to up to $r - 1$ crash failures. Should a failure occur during the allreduce, some noise will be introduced, as not all agents will receive the contribution from the agent that failed. However, this is not a problem, as our iterative optimization method will eliminate that noise in the next iteration. For scalability, this procedure can be used on subsets of agents with all subsets organized in an exponenetially growing tree, in order to achieve logarithmic scaling of communications costs.

B. Decentralized Solution of Subproblems for Updating Clone Variables z

We focus now on devising methods for solving the capacity subproblem (10), the peak demand subproblem (11) and the ancillary services subproblem (12), which can be implemented in a decentralized fashion using allreduce. Proofs for the propositions in this subsection are presented in Section EC.2 of the electronic companion [33].

1) Decentralized Solution of Capacity Subproblem: Consider a particular feeder $f \in \mathcal{F}$ and interval $t \in \mathcal{T}$. The capacity subproblem (10) associated with (f, t) can be interpreted as a biased projection of each individual station's maximum power draw $p_{s,t}^{\max,k+1}, s \in \mathcal{S}(f)$ onto the set admitted by the capacity of feeder $P_{f,t}$. Intuitively, if the stations' maximum power draw $\sum_{s\in\mathcal{S}(f)} p_{s,t}^{\max,k+1}$ exceeds the feeder's capacity $P_{f,t}$, then we should reduce the maximum draw of each station until the capacity limit is respected, which should manifest in the clone variables $u_{s,t}^{k+1}, s \in \mathcal{S}(f)$ since they always respect the capacity limit. This intuition is formalized in Proposition 1.

Proposition 1: Let $\tilde{u}_{s,t} = p_{s,t}^{\max,k+1} + (1/\rho)\mu_{s,t}^k$ for all $s \in \mathcal{S}(f)$, $\chi_{f,t} = \sum_{s\in\mathcal{S}(f)} \tilde{u}_{s,t}$, and $\alpha_{f,t}^{\dagger} = (\rho/|\mathcal{S}(f)|)(\chi_{f,t} - P_{f,t})_+$. Then, $u_{s,t}^{k+1} = \tilde{u}_{s,t} - (1/\rho)\alpha_{f,t}^{\dagger}$ for all $s \in \mathcal{S}(f)$.

In words, this proposition tells us that the biased maximum power draw of each station is $\tilde{u}_{s,t}$. If the total biased maximum draw $\chi_{f,t}$ is smaller than the capacity $P_{f,t}$, then the clone variables take the value of the biased maximum power draw $\tilde{u}_{s,t}$. Otherwise, the clone variables will correspond to $\tilde{u}_{s,t}$ reduced by an adjustment factor that is equal for all stations under the feeder.

Note that the computation of $\tilde{u}_{s,t}$ is local at each station, so the computation of $\chi_{f,t}$ can be done via allreduce over the stations in $\mathcal{S}(f)$, and the computation of $u_{s,t}^{k+1}$ is local to each station (after $\chi_{f,t}$ has been computed and shared). Therefore, Proposition 1 presents a decentralized approach for computing $\boldsymbol{u}^{k+1}$ at each iteration.

2) Decentralized Solution of Peak Demand Subproblem: Consider a particular meter $m \in \mathcal{M}$. The peak demand subproblem (11) aims at finding a slightly modified version of the expected power draw of each station $\boldsymbol{p}_s^{\mathbb{E},k+1}, s \in \mathcal{S}(m)$, that would decrease the peak demand penalization $\Pi_m^{\text{peak}}\theta_m$ of the meter. Intuition, in this case, indicates that the result of solving this problem should result in a flattened version of the total expected draw $\sum_{s\in\mathcal{S}(m)} \boldsymbol{p}_s^{\mathbb{E},k+1}$, where intervals with the larger power draws will be decreased the most. This intuition can be verified using Proposition 2.

Proposition 2: Let $\tilde{\boldsymbol{v}}_s = \boldsymbol{p}_s^{\mathbb{E},k+1} + (1/\rho)\boldsymbol{v}_s^k$ for all $s \in \mathcal{S}(m)$, $\boldsymbol{\psi}_m = \sum_{s\in\mathcal{S}(m)} \tilde{\boldsymbol{v}}_s$, and

$$\begin{aligned}\left(\boldsymbol{\beta}_m^{\dagger}, \cdot\right) = \ & \underset{\beta,\theta}{\arg\min} \ \theta_m \\ & \text{s.t.} \ \theta_m \mathbf{1}_{T\times 1} \geq \boldsymbol{\psi}_m - \frac{|\mathcal{S}(m)|}{\rho}\boldsymbol{\beta}_m \\ & \sum_{t\in\mathcal{T}} \beta_{m,t} = \Pi_m^{\text{peak}}, \ \boldsymbol{\beta}_m \geq 0. \end{aligned} \tag{14}$$

Then, $\boldsymbol{v}_s^{k+1} = \tilde{\boldsymbol{v}}_s - (1/\rho)\boldsymbol{\beta}_m^{\dagger}$ for all $s \in \mathcal{S}(m)$.

Proposition 2 indicates that the biased expected peak demand of each station corresponds to $\tilde{\boldsymbol{v}}_s$, and it presents a linear program (LP) to compute the decrease vector $\boldsymbol{\beta}_m$ (which is guaranteed to be nonnegative). The proposition also presents a decentralized approach for computing $\boldsymbol{v}^{k+1}$: computing $\tilde{\boldsymbol{v}}_s$ locally, computing $\boldsymbol{\psi}_m$ via allreduce over the stations in $\mathcal{S}(m)$, and solving (14) and computing $\boldsymbol{v}_s^{k+1}$ locally at each station. While this decentralized approach requires the repeated solution of problem (14) in all stations at each iteration, we note that this particular LP can be solved in super linear time, as indicated in Proposition 3. We provide an algorithm achieving such performance in the proof of the proposition (see Section EC.2 of [33]).

Proposition 3: Problem (14) can be solved in, at most, $\mathcal{O}(T \log T)$ operations.

3) Decentralized Solution of Ancillary Services Subproblem: Consider a particular reserve group $g \in \mathcal{G}$ and interval $t \in \mathcal{T}$. The ancillary services subproblem (12) associated with (g, t) can be interpreted as a biased projection of individual station's reserve provision $r_{s,t}^{\text{spin},k+1}, s \in \mathcal{S}(g)$ onto the set admitted by the promised reserves of the group $R_{g,t}^{\text{spin}}$. In contrast to the capacity subproblem (10), the projection in this case is *soft*, in the sense that we are only projecting onto the set up to the point where it becomes more expensive than paying the penalty Π^{fail}. Proposition 4 presents an analogous result to that of Proposition 1 for the capacity subproblem, allowing us to solve the ancillary services subproblem at each iteration in a decentralized fashion.

Proposition 4: Let $\tilde{w}_{s,t} = r_{s,t}^{\text{spin},k+1} + (1/\rho)\xi_{s,t}^k$ for all $s \in \mathcal{S}(g)$, $\omega_{g,t} = \sum_{s\in\mathcal{S}(g)} \tilde{w}_{s,t}$, and $\gamma_{g,t}^{\dagger} = \min\{(\rho/|\mathcal{S}(g)|)(R_{g,t}^{\text{spin}} - \omega_{g,t})_+, \Pi^{\text{fail}}\}$. Then, $w_{s,t}^{k+1} = \tilde{w}_{s,t} + (1/\rho)\gamma_{g,t}^{\dagger}$ for all $s \in \mathcal{S}(g)$.

C. Decentralized Algorithm

Using the results of the previous subsection we propose the decentralized algorithm of Fig. 5 to solve problem (1) – (8). Note that, as we have not modified the ADMM algorithm itself, but rather found a manner to carry out each step in a decentralized fashion, the algorithm in Fig. 5 is guaranteed to converge to an optimal solution, as stated in the following lemma.

Lemma 1: In the absence of failures, the decentralized ADMM algorithm (Fig. 5) converges to an optimal solution of problem (1) – (8).

As anticipated, our algorithm does not require an aggregator collecting station information, and yet it is equivalent in terms of iterates to the algorithm of Fig. 3. Furthermore,

```
Station $s \in \mathcal{S}$:
 1: Let $p_s^{\max,0} = p_s^{\mathbb{E},0} = u_s^0 = v_s^0 = \mu_s^0 = \nu_s^0 = \mathbf{0}$, and $r_{s,\mathcal{T}^{\text{spin}}}^{\text{spin},0} = w_s^0 = \xi_s^0 = \mathbf{0}$.
 2: for $k = 0, 1, \dots, K$ do
 3:     Solve (9) to compute $p_s^{\max,k+1}, p_s^{\mathbb{E},k+1}, r_{s,\mathcal{T}^{\text{spin}}}^{\text{spin},k+1}$.   ▷ x-update
 4:     Let $\tilde{u}_s = p_s^{\max,k+1} + (1/\rho)\,\mu_s^k$.   ▷ z-update
 5:     allreduce($[\tilde{u}_s^\mathsf{T}, 1], +, s \in \mathcal{S}(f)) \rightarrow [\chi_f^\mathsf{T}, S_f]$.
 6:     Let $u_{s,t}^{k+1} = \tilde{u}_{s,t} - (1/S_f)(\chi_{f,t} - P_{f,t})_+ \;\forall t \in \mathcal{T}$.
 7:     Let $\tilde{v}_s = p_s^{\mathbb{E},k+1} + (1/\rho)\,\nu_s^k$.
 8:     allreduce($[\tilde{v}_s^\mathsf{T}, 1], +, s \in \mathcal{S}(m)) \rightarrow [\psi_m^\mathsf{T}, S_m]$.
 9:     Solve (14) to compute $\beta_m^\dagger$.
10:     Let $v_s^{k+1} = \tilde{v}_s - (1/\rho)\beta_m^\dagger$.
11:     Let $\tilde{w}_s = r_{s,\mathcal{T}^{\text{spin}}\}}^{\text{spin},k+1} + (1/\rho)\xi_s^k$.
12:     allreduce($[\tilde{w}_s^\mathsf{T}, 1], +, s \in \mathcal{S}(g)) \rightarrow [\omega_g^\mathsf{T}, S_g]$.
13:     Let $w_{s,t}^{k+1} = \tilde{w}_{s,t} + \min\left\{\frac{(R_{g,t}^{\text{spin}} - \omega_{g,t})_+}{S_g}, \frac{\Pi^{\text{fail}}}{\rho}\right\} \forall t \in \mathcal{T}^{\text{spin}}$.
14:     Let $D_s^{\text{primal}} = \|p_s^{\max,k+1} - u_s^{k+1}\|_2^2 + \|p_s^{\mathbb{E},k+1} - w_s^{k+1}\|_2^2 + \|r_{\mathcal{T}^{\text{spin}}}^{\text{spin},k+1} - w_s^{k+1}\|_2^2$ and $D_s^{\text{dual}} = \|u^{k+1} - u^k\|_2^2 + \|v^{k+1} - v^k\|_2^2 + \|w^{k+1} - w^k\|_2^2$.
15:     allreduce($[D_s^{\text{primal}}, D_s^{\text{dual}}, 1], +, s \in \mathcal{S}) \rightarrow [D^{\text{primal}}, D^{\text{dual}}, S]$.
16:     if $\sqrt{D^{\text{primal}}\,|\mathcal{S}|/S} \le \epsilon^{\text{primal}}$ and $\sqrt{D^{\text{dual}}\,|\mathcal{S}|/S} \le \epsilon^{\text{dual}}$ then
17:         Terminate.
18:     Let $\mu_s^{k+1} = \mu_s^k + \rho\big(p_s^{\max,k+1} - u_s^{k+1}\big)$.   ▷ y-update
19:     Let $\nu_s^{k+1} = \nu_s^k + \rho\big(p_s^{\mathbb{E},k+1} - v_s^{k+1}\big)$.
20:     Let $\xi_s^{k+1} = \xi_s^k + \rho\big(r_s^{\text{spin},k+1} - w_s^{k+1}\big)$.
```

Fig. 5. Decentralized ADMM algorithm for problem (1) – (8). The implementation of the algorithm uses variables S_f, S_m, S_g, and S to maintain up-to-date values of $|\mathcal{S}(f)|$, $|\mathcal{S}(m)|$, $|\mathcal{S}(g)|$, and $|\mathcal{S}|$ which can vary in the presence of failures. These variations in the number of stations also require us to rescale tolerances whenever stations fail, in order to maintain the relative accuracy of the solution with respect to problem size. Right-aligned comments demark the start of the major ADMM steps (*x-update*, *z-update*, or *y-update*) [6] in the computations carried out by the algorithm.

our algorithm does not have any communication bottleneck, leading to better scalability in practical applications since `allreduce` operations (steps 5, 8, 12, and 15 in Fig. 3) can be implemented over exponentially-growing communication trees (communication costs scale with the log of the number of stations).

The proposed algorithm, additionally, can tolerate any finite number of failures as long as the underlying communications network remains connected so that reductions can be performed, as shown in the following lemma.

Lemma 2: Let $\mathcal{S}^{\text{fail}} \subset \mathcal{S}$ be a set of stations that fail during the execution of the decentralized ADMM algorithm (Fig. 5). Assume that the communications graph between the remaining $\mathcal{S} \setminus \mathcal{S}^{\text{fail}}$ stations is strongly connected. Then, the decentralized ADMM algorithm will converge to an optimal solution of problem (1) – (8) where stations in $\mathcal{S}^{\text{fail}}$ are not considered.

As a final remark, note that both the distributed algorithm of Fig. 3 and the decentralized algorithm of Fig. 5 can tolerate a finite number of modifications to the constraint sets of charging stations $\mathcal{D}_s$ during execution and still converge to the optimal solution of the problem post-modifications. These events include EVs leaving/arriving mid-computation, a feature very likely to observe in real implementations, but whose consequences should be ultimately handled by real-time control.

V. Numerical Experiments

We implement the algorithm of Fig. 5 in C++, using Ipopt [36] compiled with HSL [37] for solving the capacity subproblem (9) and MPI [38] for handling communications in our tests. We conduct experiments on realistic-size synthetic instances using the Quartz cluster (2 × Intel Xeon E5-2695, 36 cores, 128 GB RAM per node), hosted at the Lawrence Livermore National Laboratory. In the following, we describe our process for generating instances, present experimental results on the performance of the proposed algorithm, and present experimental results on the algorithm's performance in the presence of failures.

A. Synthetic Data Generation

We generate synthetic instances with a symmetric topology: within every instance, each distribution feeder has the same number of meters under it, and each meter has the same number of charging stations under it. Every station has 4 EV plugs, with a maximum capacity of 7kW per plug, and a total capacity of 18kW. All instances have a 24-hour horizon, divided into 24 5-minute intervals (the next two hours), 24 15-minute intervals (the following 6 hours), and then 16 1-hour intervals to complete the 24-hour period. The remaining parameters of each instance are then randomly generated. Stations are randomly assigned to reserve groups. EVs arrive randomly at each station, each with a random SOC between 15% and 45% of their total storage capacity, remain charging between 20 minutes and 8 hours, and must leave with a SOC randomly sampled between the arrival SOC and the storage capacity (capped at the maximum possible charge given by the plug capacity and the time that it remains plugged in). EVs' technical characteristics are uniformly sampled between 5kW and 7kW for input power, between 40kWh and 70kWh for storage capacity, and between 90% and 100% for charging efficiency. Prices for energy and reserve products are randomly generated within realistic bounds. Using these parameters, we create instances containing 36, 72, 144, 288, 576 and 1152 charging stations, 10 instances per station count, for a total of 60 distinct instances.

B. Performance Analysis

We solve all instances described in the previous sections with the following settings. Each charging station is assigned to a single core of LLNL's HPC cluster Quartz, emulating a practical deployment where each station is a computational unit. We set $\rho = 5$, $\epsilon^{\text{primal}} = \sqrt{|\mathcal{S}| \times (2T + T^{\text{spin}})} \times 50\text{mW}$ and $\epsilon^{\text{dual}} = \rho \times \epsilon^{\text{primal}}$.

Fig. 6 shows solution times for all our synthetic instances. It can be observed that solution times remain within the hard constraints imposed by real-time electricity market operations (5 minutes), even for instances with a large number of charging stations. The tendency for increasing solution times is explained, mainly, by two factors. (*i*) Longer solution times of certain charging station subproblems (9) due to the near singularity of the linear system to be solved at each interior point iteration (within Ipopt). This required us to

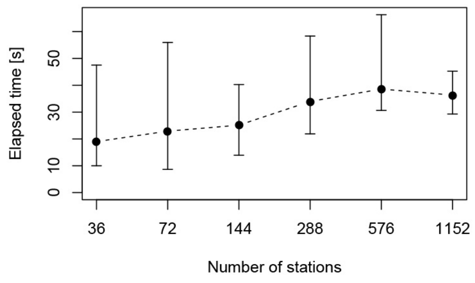

Fig. 6. Whiskers plot for solution times (wall clock) of our algorithm on all synthetic instances. Dots indicates median times over 10 instances for every charging station count.

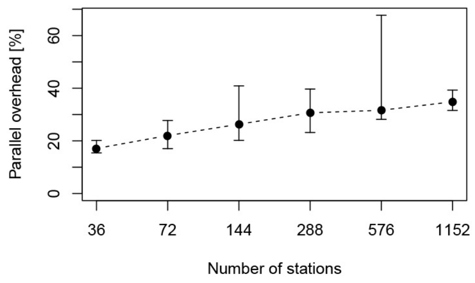

Fig. 7. Whiskers plot for parallel overhead of our algorithm on all synthetic instances. Dots indicates median overheads over 10 instances for every charging station count.

hot-swap linear solvers (ma57 to ma27) to solve the subproblem and continue with the decentralized algorithm. These near singularity situations are more likely to occur with a larger number of subproblems to be solved at each step, hence the increasing time required to solve station subproblems as the number of charging stations increases. (*ii*) The algorithm has a logarithmically-increasing parallel overhead, as can be seen in Fig. 7, because of the use of communication structures similar to binary trees for allreduce operations.

The non-increasing number of iterations for convergence along with the major contributors to solution time increase indicate that—with a more robust solution approach for solving the charging station subproblem (9), such as using a specialized QP solver which could be built based on the OOQP package [39]—the total solution time of our algorithm should scale logarithmically with the number of charging stations, which would make our approach practical for deployment in real-world charging infrastructure.

C. Failure Tolerance

We re-solve all 288-station instances while simulating stations going offline at different rates. We use the same solution parameters as in the previous section, except that we recompute the primal and dual tolerances after each failure, so that the optimal EV charging problem without the failed charging station is solved to the same relative accuracy as the original problem.

Fig. 8 presents the scaled primal and dual residuals for a representative subset of instances, all other present similar qualitative behaviour. Failures manifest as sharp increases in both primal and dual residuals, which our algorithm quickly recovers from within the next couple of iterations after the failure. As a result we observe little to no increase in iteration count to solution, for failure rates of up to 20%. These results are also very encouraging, as they demonstrate that our algorithm can effectively withstand a large number of failures, taking advantage of the algorithm's partial solution before the failure to warm start the solution to the modified problem, indicating that our algorithm can work even under partial outages of the power grid.

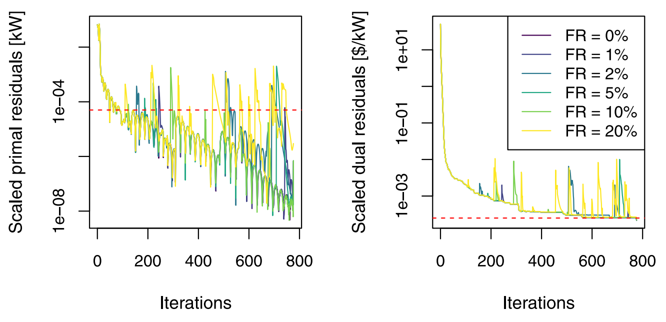

Fig. 8. Primal and dual residuals for synthetic a instance with 288 stations, while simulating random failures with different failure rates (FR), corresponding to the probability that each station will fail at some iteration count $k \in \{1, \dots, K\}$. For failing stations, the iteration count at which they fail is sampled uniformly at random from $\{1, \dots, K\}$. Residuals scaled by $\sqrt{|\mathcal{S}|/S}$, as indicated in Fig. 5, step 16. Tolerances indicated with a dashed red line.

VI. Conclusion

We have presented a fully decentralized algorithm for solving the optimal EV charging problem which can withstand any number of failures of charging stations. While we focus on the optimal EV charging problem, the ideas we presented can be applied to a larger class of problems, which may benefit from a decentralized approach to solving them.

We have numerically shown that the algorithm scales to industry-scale fleet sizes, respecting time limits for integration with real-time electricity markets, even with a high number of failures. These encouraging results can be further improved with more careful parameter setting (different ρ for different relaxed constraints), relaxed dual tolerances, and more robust methods for solving each charging station subproblem.

Further research will investigate methods for failure/corruption detection, efficient privacy-preserving extensions for optimizing systems with potentially many charging station owners, efficient formulations for V2G technologies, and asynchronous decentralized extensions for faster convergence and communication delay/error tolerance.

Appendix

Nomenclature

Sets and Indexes:

$\mathcal{T}$	ordered set of time intervals in the horizon
$\mathcal{T}^{\text{spin}}$	ordered set of time intervals for which spinning reserves are already committed

$\mathcal{S}$	charging stations
$\mathcal{M}$	power meters
$\mathcal{F}$	distribution feeders
$\mathcal{G}$	reserve groups
$\mathcal{S}(m)$	charging stations under meter m
$\mathcal{S}(f)$	charging stations under feeder f
$\mathcal{S}(g)$	charging stations participating in reserve group g
$m(s)$	meter where station s is connected
$g(s)$	reserve group where station s participates
$\mathcal{D}_s$	feasible operating set of charging station s.
Parameters:	
$\Pi^{\text{energy}}_{m,t}$	energy price at meter m, interval t
$\Pi^{\text{spin}}_{g,t}$	spinning reserve price for group g, interval t
Π^{peak}_{m}	peak power price at meter m
Π^{fail}	penalty for failing to provide committed spinning reserve capacity
$P_{f,t}$	net capacity for feeder f, interval t
$R^{\text{spin}}_{g,t}$	spinning reserve commitment for group g, interval t
ρ	ADMM regularization parameter.
Variables:	
$p^{\mathbb{E}}_{s,t}$	expected power draw for station s at interval t
$p^{\max}_{s,t}$	maximum power draw for station s at interval t
$r^{\text{spin}}_{s,t}$	spinning reserve provided by station s at interval t
θ_m	expected peak demand at meter m
$\phi_{g,t}$	committed but not provided spinning reserve by group g at interval t
$u_{s,t}, v_{s,t}, w_{s,t}$	clones of $p^{\mathbb{E}}_{s,t}$, $p^{\max}_{s,t}$, and $r^{\text{spin}}_{s,t}$, respectively
$\mu_{s,t}, \nu_{s,t}, \xi_{s,t}$	dual variables for replication constraints (5)–(7).
Others:	
$p^{\mathbb{E},k}_{s,t}, p^{\max,k}_{s,t}, r^{\text{spin},k}_{s,t}$	*x-block* (primal variables) k-th iterates of ADMM method applied to problem (1)–(8)
$u^k_{s,t}, v^k_{s,t}, w^k_{s,t}$	*z-block* (clone variables) k-th iterates of ADMM method applied to problem (1)–(8)
$\mu^k_{s,t}, \nu^k_{s,t}, w^k_{s,t}$	*y-block* (dual variables) k-th iterates of ADMM method applied to problem (1)–(8)
$\epsilon^{\text{primal}}, \epsilon^{\text{dual}}$	primal and dual termination tolerances of ADMM.

Acknowledgment

The authors would like to thank Dr. Jonathan Donadee (Geli) and Dr. Amelia Musselman (LLNL) for formulating the initial version of the EV charging model, presented in Section II and Section EC.1 of [33], on which the current work is based.

References

[1] USDoT, *Summary of Travel Trends: 2017 National Household Travel Survey*, U.S. Dept. Transp. and Federal Highway Admin., Washington, DC, USA, Jul. 2018.

[2] USEIA. (Oct. 2019). *Frequently Asked Questions. U.S. Energy Information Administration*. [Online]. Available: https://www.eia.gov/tools/faqs/faq.php?id=97&t=3

[3] *Innovation Landscape Brief: Electric-Vehicle Smart Charging*, Int. Renew. Energy Agency (IRENA), Abu Dhabi, UAE, 2019.

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

[5] C.-K. Wen, J.-C. Chen, J.-H. Teng, and P. Ting, "Decentralized plug-in electric vehicle charging selection algorithm in power systems," *IEEE Trans. Smart Grid*, vol. 3, no. 4, pp. 1779–1789, Dec. 2012.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[7] J.-Y. Joo and M. D. Ilić, "Multi-layered optimization of demand resources using lagrange dual decomposition," *IEEE Trans. Smart Grid*, vol. 4, no. 4, pp. 2081–2088, Dec. 2013.

[8] L. Gan, U. Topcu, and S. H. Low, "Optimal decentralized protocol for electric vehicle charging," *IEEE Trans. Power Syst.*, vol. 28, no. 2, pp. 940–951, May 2013.

[9] M. G. Vayá, G. Andersson, and S. Boyd, "Decentralized control of plug-in electric vehicles under driving uncertainty," in *Proc. IEEE PES Innovat. Smart Grid Technol. Eur.*, Istanbul, Turkey, 2014, pp. 1–6.

[10] C. Le Floch, F. Belletti, S. Saxena, A. M. Bayen, and S. Moura, "Distributed optimal charging of electric vehicles for demand response and load shaping," in *Proc. 54th IEEE Conf. Decis. Control (CDC)*, Osaka, Japan, 2015, pp. 6570–6576.

[11] Y. Nesterov, "Smooth minimization of non-smooth functions," *Math. Program.*, vol. 103, no. 1, pp. 127–152, 2005.

[12] A. Ghavami, K. Kar, and A. Gupta, "Decentralized charging of plug-in electric vehicles with distribution feeder overload control," *IEEE Trans. Autom. Control*, vol. 61, no. 11, pp. 3527–3532, Nov. 2016.

[13] S. Grammatico, "Exponentially convergent decentralized charging control for large populations of plug-in electric vehicles," in *Proc. IEEE 55th Conf. Decis. Control (CDC)*, Las Vegas, NV, USA, 2016, pp. 5775–5780.

[14] L. Wang, S. Sharkh, and A. Chipperfield, "Optimal decentralized coordination of electric vehicles and renewable generators in a distribution network using A* search," *Int. J. Elect. Power Energy Syst.*, vol. 98, pp. 474–487, Jun. 2018.

[15] Z. Ma, D. S. Callaway, and I. A. Hiskens, "Decentralized charging control of large populations of plug-in electric vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 1, pp. 67–78, Jan. 2013.

[16] C. O. Adika and L. Wang, "Non-cooperative decentralized charging of homogeneous households' batteries in a smart grid," *IEEE Trans. Smart Grid*, vol. 5, no. 4, pp. 1855–1863, Jul. 2014.

[17] N. Rahbari-Asr and M.-Y. Chow, "Cooperative distributed demand management for community charging of PHEV/PEVs based on KKT conditions and consensus networks," *IEEE Trans. Ind. Informat.*, vol. 10, no. 3, pp. 1907–1916, Aug. 2014.

[18] L. Zhang, V. Kekatos, and G. B. Giannakis, "Scalable electric vehicle charging protocols," *IEEE Trans. Power Syst.*, vol. 32, no. 2, pp. 1451–1462, Mar. 2017.

[19] M. Kraning, E. Chu, J. Lavaei, and S. Boyd, "Dynamic network energy management via proximal message passing," *Found. Trends Optim.*, vol. 1, no. 2, pp. 73–126, Jan. 2014.

[20] E. Münsing, J. Mather, and S. Moura, "Blockchains for decentralized optimization of energy resources in microgrid networks," in *Proc. IEEE Conf. Control Technol. Appl. (CCTA)*, Maui, HI, USA, 2017, pp. 2164–2171.

[21] J. Li, C. Li, Y. Xu, Z. Y. Dong, K. P. Wong, and T. Huang, "Noncooperative game-based distributed charging control for plug-in electric vehicles in distribution networks," *IEEE Trans. Ind. Informat.*, vol. 14, no. 1, pp. 301–310, Jan. 2018.

[22] R. F. Atallah, C. M. Assi, W. Fawaz, M. H. K. Tushar, and M. J. Khabbaz, "Optimal supercharge scheduling of electric vehicles: Centralized versus decentralized methods," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 7896–7909, Sep. 2018.

[23] J. Qin, Y. Wan, X. Yu, F. Li, and C. Li, "Consensus-based distributed coordination between economic dispatch and demand response," *IEEE Trans. Smart Grid*, vol. 10, no. 4, pp. 3709–3719, Jul. 2019.

[24] X. Yang, G. Wang, H. He, J. Lu, and Y. Zhang, "Automated demand response framework in ELNS: Decentralized scheduling and smart contract," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 50, no. 1, pp. 58–72, Jan. 2020.

[25] Y. Wan, J. Qin, F. Li, X. Yu, and Y. Kang, "Game theoretic-based distributed charging strategy for PEVS in a smart charging station," *IEEE Trans. Smart Grid*, vol. 12, no. 1, pp. 538–547, Jan. 2021.

[26] P. Denholm, J. Eichman, T. Markel, and O. Ma, "Summary of market opportunities for electric vehicles and dispatchable load in electrolyzers," Nat. Renew. Energy Lab. (NREL), Washington, DC, USA, Rep. NREL/TP-6A20-64172, May 2015.

[27] J. Lin, K.-C. Leung, and V. O. K. Li, "Optimal scheduling with vehicle-to-grid regulation service," *IEEE Internet Things J.*, vol. 1, no. 6, pp. 556–569, Dec. 2014.

[28] F. Juul, M. Negrete-Pincetic, J. MacDonald, and D. Callaway, "Real-time scheduling of electric vehicles for ancillary services," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Denver, CO, USA, 2015, pp. 1–5.

[29] Z. Peng and L. Hao, "Decentralized coordination of electric vehicle charging stations for active power compensation," in *Proc. IEEE 86th Veh. Technol. Conf. (VTC-Fall)*, Toronto, ON, Canada, 2017, pp. 1–5.

[30] Q. Li, B. Kailkhura, R. Goldhahn, P. Ray, and P. K. Varshney, "Robust decentralized learning using ADMM with unreliable agents," 2017. [Online]. Available: arXiv:1710.05241.

[31] E. Munsing and S. Moura, "Cybersecurity in distributed and fully-decentralized optimization: Distortions, noise injection, and ADMM," 2018. [Online]. Available: https://arxiv.org/abs/1805.11194

[32] W. B. Powell, "A unified framework for optimization under uncertainty," in *Optimization Challenges in Complex, Networked and Risky Systems*. Hanover, MD, USA: INFORMS, Oct. 2016, ch. 3, pp. 45–83. [Online]. Available: https://pubsonline.informs.org/doi/abs/10.1287/educ.2016.0149

[33] I. Aravena, S. Chapin, and C. Ponce, *Electronic Companion to: Decentralized Failure-Tolerant Optimization of Electric Vehicle Charging*, IEEE Xplore, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9431209/media

[34] R. Thakur, R. Rabenseifner, and W. Gropp, "Optimization of collective communication operations in MPICH," *Int. J. High Perform. Comput. Appl.*, vol. 19, no. 1, pp. 49–66, 2005.

[35] M. Casas, W. N. Gansterer, and E. Wimmer, "Resilient gossip-inspired all-reduce algorithms for high-performance computing: Potential, limitations, and open questions," *Int. J. High Perform. Comput. Appl.*, vol. 33, no. 2, pp. 366–383, 2018.

[36] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, no. 1, pp. 25–57, 2006.

[37] HSL. (2015). *A Collection of Fortran Codes for Large Scale Scientific Computation*. [Online]. Available: http://www.hsl.rl.ac.uk/

[38] Message Passing Interface Forum. (Jun. 2015). *MPI: A Message-Passing Interface Standard Version 3.1*. [Online]. Available: https://www.mpi-forum.org/docs/mpi-3.1/mpi31-report.pdf

[39] E. M. Gertz and S. J. Wright, "Object-oriented software for quadratic programming," *ACM Trans. Math. Softw.*, vol. 29, no. 1, pp. 58–81, Mar. 2003. [Online]. Available: https://doi.org/10.1145/641876.641880

[40] R Core Team, *R: A Language and Environment for Statistical Computing*, R Found. Stat. Comput., Vienna, Austria, 2020. [Online]. Available: https://www.R-project.org/

Ignacio Aravena (Member, IEEE) received the B.Sc. and M.Sc. degrees in electrical engineering from Universidad Tecnica Federico Santa Maria (UTFSM), Valparaíso, Chile, and the Ph.D. degree in applied mathematics from the Université catholique de Louvain, Ottignies-Louvain-la-Neuve, Belgium. He is currently an Operations Research Engineer with Lawrence Livermore National Laboratory, Livermore, CA, USA. He has served as a part-time Lecturer in operations research with the University of California, Berkeley, and in electrical engineering with UTFSM, as well as a Consultant with ENEL and Powel AS. His research interests include energy optimization, power system resilience, stochastic and robust programming, and distributed optimization.

Steve J. Chapin received the B.S. degree in dual majors of mathematics and computer science from Heidelberg University (née Heidelberg College) and the M.S. and Ph.D. degrees in computer sciences from Purdue University. He is currently a Lead Cybersecurity Researcher with Lawrence Livermore National Laboratory, Livermore, CA, USA. Prior to joining LLNL, he held faculty positions with Syracuse University, the University of Virginia, and Kent State University. His research interests include computer systems, computer security, and formal methods.

Colin Ponce received the B.S.E. degree in computer science from Princetion University, Princeton, NJ, USA, in 2010, and the Ph.D. degree from Cornell University, Ithaca, NY, USA, in 2016. He is currently an Applied Mathematician with Lawrence Livermore National Laboratory, Livermore, CA, USA, where he was previously a Postdoctoral Fellow. His research interests include resilient decentralized computing, network science, power systems analysis, multilevel methods, and machine learning.