Collaborative Distributed AC Optimal Power Flow: A Dual Decomposition Based Algorithm

Zheyuan Cheng, *Member*, *IEEE* and Mo-Yuen Chow, *Fellow*, *IEEE*

***Abstract*—We propose a dual decomposition based algorithm that solves the AC optimal power flow (ACOPF) problem in the radial distribution systems and microgrids in a collaborative and distributed manner. The proposed algorithm adopts the second-order cone program (SOCP) relaxed branch flow ACOPF model. In the proposed algorithm, bus-level agents collaboratively solve the global ACOPF problem by iteratively sharing partial variables with its 1-hop neighbors as well as carrying out local scalar computations that are derived using augmented Lagrangian and primal-dual subgradient methods. We also propose two distributed computing platforms, i. e., high-performance computing (HPC) based platform and hardware-in-the-loop (HIL) testbed, to validate and evaluate the proposed algorithm. The computation and communication performances of the proposed algorithm are quantified and analyzed on typical IEEE test systems. Experimental results indicate that the proposed algorithm can be executed on a fully distributed computing structure and yields accurate ACOPF solution. Besides, the proposed algorithm has a low communication overhead.**

***Index Terms*—Distributed convex optimization, distributed energy management system, optimal power flow, primal-dual decomposition.**

I. Introduction

THE AC optimal power flow (ACOPF) has been a classical topic in the power system operation and control since its introduction in 1962 [1]. The ACOPF tries to find a set of optimal operation setpoints that optimize a set of objectives, energy efficiency, power quality, security, etc., with certain network constraints. The ACOPF is typically formulated as a non-linear quadratically-constrained quadratic programming problem, due to its non-linear quadratic power flow equality constraints. Based on how the power flow equations are formulated, the ACOPF can be further categorized into bus injection model (BIM) and alternating direction method of multiplier (ADMM), which are mathematically equivalent. Due to the non-linear quadratic power flow constraints, the ACOPF is generally nonconvex and non-deterministic polynomial (NP)-hard [2]. A typical approximation is a linear program, also known as DC optimal power folw (DCOPF), in which the non-linear constraints are linearized. Thus, the linearized DCOPF is easy to solve. However, the DCOPF is typically used with strong assumptions, e. g., no reactive power, lossless line, fixed bus voltage, etc., which is applied to well-compensated networks, e. g., transmission networks, sub-transmission networks. The detailed formulation differences of DCOPF and ACOPF are well documented in [2]. An important alternative is convex relaxations [3], [4], i.e., second-order cone program (SOCP) relaxation and semidefinite program (SDP) relaxation. Both relaxations bound the optimal results of the original ACOPF problem. The SDP is tighter than the SOCP relaxation over generic networks. However, SOCP and SDP relaxations are found to have the same tightness over single-phase radial networks. Under sufficient conditions, the exact solution to the original ACOPF problem can be recovered [3].

ACOPF problems in the distribution systems and microgrids have caught significant research interests in recent decades for its wide applications in the operation and control of distributed energy resource (DER), e. g., in energy management system of DER [5], [6], energy market participation [7], EV charging management [8], service restoration [9]. Conventionally, these ACOPF problems are solved in a centralized framework. However, there have been increasing concerns about the scalability, reliability, resilience, and privacy of the centralized ACOPF. As DER becomes more and more modular, controllable and connected, the distributed ACOPF is often viewed as a proper tool as part of the optimal control algorithms of DER. Compared with the centralized ACOPF, the distributed ACOPF is more scalable, i. e., more robust to single-point-of-failure, more resilient to faults and attacks, and more private than centralized solution [10], [11].

The distributed ACOPF algorithm addressed in this paper is a distributed convex optimization. Such optimization problem is collectively solved by agents who collaborate with one another to minimize a global objective function that is a sum of some local objective functions. From the perspective of computing framework, we assume the distributed agents to possess independent local computation capability and peer-to-peer (P2P) communication capability. The preliminaries of distributed convex optimization theories and its applications in smart grids can be found in [3], [12], [13].

The mainstream distributed ACOPF algorithms can be further categorized into five classes [3]: ① dual decomposition [14], [15]; ② ADMM [16]-[18]; ③ auxiliary problem princi-

Manuscript received: June 23, 2020; accepted: December 29, 2020. Date of CrossCheck: December 29, 2020. Date of online publication: February 17, 2021.

This work was supported by the National Science Foundation (No. CNS-1505633).

Z. Cheng (corresponding author) and M. Chow are with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27606, USA (e-mail: zcheng3@ncsu.edu; chow@ncsu.edu).

DOI: 10.35833/MPCE.2020.000395

ple (APP) [19], [20]; ④ optimality condition decomposition (OCD) [21], [22]; ⑤ consensus + innovation [23]. Some typical distributed ACOPF algorithms are summarized in Table I.

TABLE I
TYPICAL DISTRIBUTED ACOPF ALGORITHMS

Reference	Algorithm	Initial configuration	Distributed agent type	Local computation type	Communication type	Convergence rate
[14]	Dual decomposition	Derive close-form local gradient descent equations	Bus-level agent	Substitutions and scalar calculations	One-hop communication with symmetrical information sharing	Linear
[15]	Dual decomposition	Problem decomposed into sub-problems	Bus-level agent	Local iterative minimizations	One-hop communication with asymmetrical information sharing	Linear
[16], [17]	ADMM	Problem decomposed into sub-problems	Region-level agent	Local iterative minimizations	Multi-hop communication with asymmetrical information sharing	Piecewise linear
[18]	ADMM	Problem decomposed into sub-problems and derive close-form solutions to sub-problems	Bus-level agent	Substitutions and scalar calculations	One-hop communication with symmetrical information sharing	Linear
[19], [20]	APP	Problem decomposed into sub-problems based on tie-lines	Region-level agent	Local iterative minimizations	Multi-hop communication with asymmetrical information sharing	Linear
[21], [22]	OCD	Problem decomposed into sub-problems based on areas and drive gradient descent equations	Region-level agent	Substitutions and scalar calculations	Multi-hop communication with asymmetrical information sharing	Linear
[23]	Consensus + innovation	Derive local gradient descent and consensus equations for shared variables estimation	Bus-level agent	Substitutions and scalar calculations	One-hop communication with symmetrical information sharing	Linear

From an initial configuration perspective, all the methods require a certain level of problem decomposition and derivation. Among the five classes, the close-form local gradient descent based dual decomposition algorithms, e.g., [14], typically require the least configuration efforts. With respect to agent modeling, depending on the applications and the adopted algorithms, there are typically two distributed agent types, i.e., bus-level and region-level agents. In term of local computation type, some close-form based algorithms, e. g., [14], [18], [23], require simple substitutions and scalar calculations, whereas some algorithms require local iterative minimization steps, e.g., [17], [20]-[22]. Regarding communication type, depending on the decomposition algorithms, some algorithms, e.g., [14], [18], [23], share symmetrical information among agents, whereas some algorithms, e.g., [15]-[17], [19]-[22], share asymmetrical information among agents. Additionally, some region-level distributed algorithms, e. g., [16], [17], [19]-[22], require multi-hop communications. In general, these five mainstream algorithms are analytically or numerically proven to possess linear convergence rates on the convexified ACOPF problems.

In recent years, many efforts are made to improve distributed ACOPF algorithms. Reference [24] proposes a component-level dual decomposition solver that can scale up to 10^4-bus test systems. Reference [25] improves the dual decomposition algorithm using the augmented Lagrangian and the block successive upper-bound minimization method. The proposed dual decomposition algorithm can handle non-convex constraints and discrete variables. Reference [26] applies an improved version of the ADMM called augmented Lagrangian alternating direction inexact Newton (ALADIN) to solve the distributed ACOPF. The proposed ALADIN reduces the number of iterations by at least one order of magnitude at the cost of the increased per-step communication effort. Reference [27] proposes an improved ADMM for the ACOPF problem in a hybrid AC-DC grid, and compares the performance with the ALADIN. The improved ADMM is found to be more scalable than ALADIN for large-scale distributed systems. Reference [28] proposes a new incremental-oriented ADMM algorithm by fusing the extended interior-point method and ADMM so that the improved ADMM can solve mixed-integer non-linear programming. Reference [29] proposes an OCD based three-stage distributed OPF algorithm that can handle discrete variables. On the whole, ADMM is popular and widely utilized. Some researchers are attempting to apply distributed ACOPF algorithms to large-scale distributed computing systems and to solve large-scale ACOPF problems. Also, they are extending the distributed ACOPF algorithms to solve mixed-integer programming problems, e.g., unit commitment.

On the basis of prior works and recent developments [14]-[29], we further identify three shortcomings that are yet to be fully addressed: ① many of the current decomposition-based distributed ACOPF algorithms require intensive preliminary clustering analysis and formulation efforts; ② many of the ADMM-based and OCD-based algorithms require considerable computing power on the edge unit to carry out the iterative optimization; ③ most ACOPF algorithms are validated and benchmarked on a single-thread environment, e. g., MATLAB on a single PC. The inter-agent communication is typically emulated via in-RAM variable assignments, and the corresponding computation time has limited significance on representing the actual performance on the distributed computing system.

To address aforementioned shortcomings, in this paper, a novel dual decomposition based distributed subgradient ACOPF algorithm is proposed and prototyped. The proposed algorithm can accurately solve convexified ACOPF problems in a distributed computing environment with modest communication overhead. Compared with surveyed existing distributed ACOPF algorithms in [14] - [29], the contributions of this paper can be summarized as follows.

1) A novel agent-level dual decomposition based distributed subgradient ACOPF algorithm is proposed, in which each agent computes simple close-form gradient update equations. The proposed algorithm does not require complex initial clustering or decomposition analysis.

2) The proposed algorithm is validated on actual distributed computing systems, i. e., high-performance computing (HPC) based platform and hardware-in-the-loop (HIL) testbed.

3) The computation and communication performances of the proposed algorithm are quantified and analyzed on both HPC based platform and HIL testbed.

The rest of the paper is structured as follows. Section II introduces the ACOPF problem formulation. Section III covers the preliminaries, i.e., augmented Lagrangian with primal-dual subgradient (PDS) method as well as the proposed algorithm. Section IV presents the proposed distributed computing structures for the validation and evaluation of the proposed algorithm. Section V provides the experimental results on various benchmark systems and the corresponding analysis. The conclusions are summarized in Section VI.

II. ACOPF Problem Formulation

The ACOPF problem formulation is established based on the following three assumptions: ① the physical system topology is radial; ② the slack bus voltage V_0 and angle θ is fixed and known, i.e., $V_0 = 1.1$ p.u. and $\theta = 0°$; ③ the SOCP relaxation, i. e., angle and conic relaxations [30], of branch flow model (BFM) ACOPF is convex and exact.

Typically, the optimization objective function $J(x)$ of the ACOPF is represented by a quadratic cost function that reflects the cost of generation. In this paper, we use the commonly used quadratic cost function in (1) to represent the generation cost at the slack bus and distributed generator (DG) buses.

$$J(x) = \sum_{i \in G} a x_i^2 + b x_i + c \tag{1}$$

where a, b, and c are the parameters of DG quadratic cost function; G is the generator set consisting of the slack and DG buses; and x_i is the optimization variable of bus i.

$J(x)$ is then minimized over primal variable x, while satisfying all the equality constraints $\boldsymbol{h}(x)$ and inequality constraints $\boldsymbol{g}(x)$ introduced by the BFM ACOPF. To formulate the ACOPF constraints, we define real and reactive power injections of the bus j as $p_j = p_j^g - p_j^c$ and $q_j = q_j^g - q_j^c$, respectively, where the superscripts g and c represent the generation and consumption, respectively. Then we model the physical system topology as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, in which the buses are denoted as vertices $\mathcal{V}$ while the lines are denoted as edges $\mathcal{E}$. Note that, in the convexified SOCP relaxation, squared variables are used to eliminate the angles, and all branch power flow equality constraints are relaxed to inequality constraints, i.e., (5). Based on the BFM notations in Fig. 1, p_0, q_0, p_i, q_i, p_m, q_m are the real and reactive power injections of buses 0, i, m, respectively; v_0, v_i, v_j, v_m, v_n are the squared bus voltage magnitudes of buses 0, i, j, m, n, respectively; P_{ij}, Q_{ij}, P_{jm}, Q_{jm} are the real and reactive power flows of the branch from buses i to j and from buses j to m, respectively; R_{ij}, X_{ij}, R_{jm}, X_{jm} are the resistances and inductances of the branch from buses i to j and buses j to m, respectively; and ℓ_{ij} and ℓ_{jm} are the squared branch current magnitudes from buses i to j and from buses j to m, respectively. The resulting equality and inequality constraints are presented in (2)-(9).

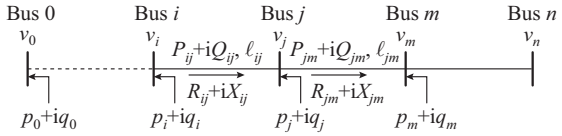

Fig. 1. BFM notation.

$$\sum_{k:j \to k} P_{jk} - \sum_{i:i \to j} \left(P_{ij} - R_{ij} \ell_{ij} \right) - p_j = 0 \quad \forall j \in \mathcal{V} \tag{2}$$

$$\sum_{k:j \to k} Q_{jk} - \sum_{i:i \to j} \left(Q_{ij} - X_{ij} \ell_{ij} \right) - q_j = 0 \quad \forall j \in \mathcal{V} \tag{3}$$

$$v_i - v_j - 2\left(R_{ij} P_{ij} + X_{ij} Q_{ij} \right) + \left(R_{ij}^2 + X_{ij}^2 \right) \ell_{ij} = 0 \quad \forall (i,j) \in \mathcal{E} \tag{4}$$

$$\frac{P_{ij}^2 + Q_{ij}^2}{v_i} - \ell_{ij} \le 0 \quad \forall (i,j) \in \mathcal{E} \tag{5}$$

$$\begin{cases} \underline{p}_j^g \le p_j^g \le \bar{p}_j^g & \forall j \in \mathcal{V} \\ \underline{q}_j^g \le q_j^g \le \bar{q}_j^g & \forall j \in \mathcal{V} \end{cases} \tag{6}$$

$$\begin{cases} \underline{p}_j^c \le p_j^c \le \bar{p}_j^c & \forall j \in \mathcal{V} \\ \underline{q}_j^c \le q_j^c \le \bar{q}_j^c & \forall j \in \mathcal{V} \end{cases} \tag{7}$$

$$\underline{v}_j \le \left| V_j \right|^2 \le \bar{v}_j \quad \forall j \in \mathcal{V} \tag{8}$$

$$\left| I_{ij} \right|^2 \le \bar{\ell}_{ij} \quad \forall (i,j) \in \mathcal{E} \tag{9}$$

where I_{ij} is the branch current from buses i to j; and V_j is the voltage of bus j; and $\overline{(\cdot)}$ and $\underline{(\cdot)}$ denote the upper and lower bounds, respectively.

Note that (2) and (3) are the real and reactive power balance constraints at each bus, respectively. Equation (4) is the Ohm's law constraint for each branch; (5) is the inequality constraint of relaxed branch power flow; (6) is the operation limit of the generator; (7) is the operation limits of demand response resources (p_j^c and q_j^c are constant when representing static loads); (8) is the operation limit of bus voltage, i. e., [0.9^2 1.1^2] per unit; and (9) is the line limit for each branch. With the constraints above, we denote the optimization search space $\mathbb{S}$ as:

$$\mathbb{S} =: \left\{ \boldsymbol{p}, \boldsymbol{q}, \boldsymbol{v}, \boldsymbol{\ell}, \boldsymbol{P}, \boldsymbol{Q} \,|\, \boldsymbol{p}, \boldsymbol{q}, \boldsymbol{v}, \boldsymbol{\ell} \text{ satisfy (6)-(9)} \right\} \tag{10}$$

Then, the equality constraints in (2) - (4) are denoted as $\boldsymbol{h}(x) = \boldsymbol{0}$, and the inequality constraint in (5) is denoted as $\boldsymbol{g}(x) \le \boldsymbol{0}$. Finally, the ACOPF problem is formulated as (11).

$$\begin{cases} \min\limits_{x\in\mathbb{S}} J(x) \\ \text{s.t. } \boldsymbol{h}(x)=\boldsymbol{0} \\ \quad \boldsymbol{g}(x)\le \boldsymbol{0} \end{cases} \tag{11}$$

III. Distributed ACOPF Algorithm

A. Augmented Lagrangian and PDS Method

Since the SOCP relaxation in (11) is convex, we can apply the dual decomposition method to find the optimal solution by reaching the saddle point of the Lagrangian $\mathcal{L}$. To improve the convergence property, instead of using the Lagrangian in the original dual decomposition method, we adopt the augmented Lagrangian and the method of Lagrange multiplier to formulate the optimization problem. To find the optimal solution to problem (11), we design the following two-step process. First, we augment the Lagrangian via multipliers and penalty terms:

$$\mathcal{L}=J(x)+\boldsymbol{\lambda}^{\mathrm{T}}\boldsymbol{h}(x)+\boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{g}(x)_{+}+\frac{\rho}{2}\left\|\boldsymbol{h}(x)\right\|_2^2+\frac{\rho}{2}\left\|\boldsymbol{g}(x)\right\|_2^2 \tag{12}$$

where $(\cdot)_+$ signifies the positive projection; ρ is the penalty factor of the augmented lagrangian; and $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ are the dual variables. Second, we apply the PDS method in [31] to find the saddle point of the augmented Lagrangian by separately and iteratively moving primal variable $\boldsymbol{x}$ towards negative gradient direction, whereas moving the dual variables $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ towards positive gradient direction, as defined in (13) - (15). Let's denote the step size as ξ. Moving from iterations k to $k+1$, $\forall x\in\mathbb{S}$, we can obtain:

$$\boldsymbol{x}^{k+1}=\boldsymbol{x}^{k}-\xi\left.\frac{\partial\mathcal{L}}{\partial\boldsymbol{x}}\right|_{\{\boldsymbol{x}^k,\boldsymbol{\lambda}^k,\boldsymbol{\mu}^k\}} \tag{13}$$

$$\boldsymbol{\lambda}^{k+1}=\boldsymbol{\lambda}^{k}+\xi\left.\frac{\partial\mathcal{L}}{\partial\boldsymbol{\lambda}}\right|_{\{\boldsymbol{x}^k,\boldsymbol{\lambda}^k,\boldsymbol{\mu}^k\}} \tag{14}$$

$$\boldsymbol{\mu}^{k+1}=\boldsymbol{\mu}^{k}+\xi\left.\frac{\partial\mathcal{L}}{\partial\boldsymbol{\mu}}\right|_{\{\boldsymbol{x}^k,\boldsymbol{\lambda}^k,\boldsymbol{\mu}^k\}} \tag{15}$$

B. Collaborative Distributed ACOPF Algorithm

This subsection introduces the proposed algorithm using the aforementioned dual decomposition concept with augmented Lagrangian and PDS methods. Each bus/node is modeled as an agent who possesses local computing power and a P2P communication interface with the adjacent agents, also known as 1-hop neighbors. This P2P communication network should be identical to the physical system topology. Additionally, we assume each agent has access to impedance parameters of the power lines that connect to itself and to the local generation cost function parameters. Because of the radial topology and BFM formulation, there are no global shared dual variables. All the dual variables are shared among 1-hop neighbors. Therefore, the ACOPF algorithm can be solved collectively by all the agents in a fully distributed way.

To demonstrate the proposed algorithm, we derive the agent update equations based on the BFM notation in Fig. 1. The first step is to take partial derivative of the augmented Lagrangian in (12) with respect to all the primal variable $\boldsymbol{x}$ and dual variables $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$. Then the partial derivatives are substituted into the PDS update equations in (13)-(15). Finally, we can get the individual variable update equations. Equations (16)-(25) present the derived PDS update questions. At every iteration, for all $x\in\mathbb{S}$, moving from iterations k to $k+1$, each agent $j\in\mathcal{V}$ updates its local primal variables as derived in (16)-(21).

$$p_j^{k+1}=p_j^k-\xi_1\left[b_j-\lambda_j^{p,k}+2a_jp_j^k+\rho\left(P_{ij}^k-P_{jm}^k+p_j^k-\ell_{ij}^kR_{ij}\right)\right] \tag{16}$$

$$q_j^{k+1}=q_j^k-\xi_2\left[\rho\left(Q_{ij}^k-Q_{jm}^k+q_j^k-\ell_{ij}^kX_{ij}\right)-\lambda_j^{q,k}\right] \tag{17}$$

$$\begin{aligned} v_j^{k+1}=v_j^k-\xi_3\Bigg\{&\lambda_j^{Q,k}-\lambda_i^{Q,k}+\rho\Bigg[2P_{ij}^kR_{ij}+2Q_{ij}^kX_{ij}-2P_{jm}^kR_{jm}- \\ &2Q_{jm}^kX_{jm}+2v_j^k-v_i^k-v_m^k-\ell_{ij}^k\left(R_{ij}^2+X_{ij}^2\right)+\ell_{jm}^k\left(R_{jm}^2+X_{jm}^2\right)- \\ &\frac{1}{\left(v_j^k\right)^3}\left(\left(P_{jm}^k\right)^2+\left(Q_{jm}^k\right)^2\right)\left(\left(P_{jm}^k\right)^2+\left(Q_{jm}^k\right)^2-\ell_{jm}^kv_j^k\right)\Bigg]- \\ &\frac{\mu_j^k}{\left(v_j^k\right)^2}\left(\left(P_{jm}^k\right)^2+\left(Q_{jm}^k\right)^2\right)\Bigg\} \end{aligned} \tag{18}$$

$$\begin{aligned} \ell_{jm}^{k+1}=\ell_{jm}^k-\xi_4\Bigg\{&\lambda_m^{p,k}R_{jm}+\lambda_m^{q,k}X_{jm}-\mu_j^k+\lambda_j^{Q,k}\left(R_{jm}^2+X_{jm}^2\right)+ \\ &\rho\Bigg[\ell_{jm}^k-\frac{1}{v_j^k}\left(\left(P_{jm}^k\right)^2+\left(Q_{jm}^k\right)^2\right)-R_{jm}\left(P_{jm}^k-P_{mn}^k+p_m^k-\right. \\ &\left.\ell_{jm}^kR_{jm}\right)-X_{jm}\left(Q_{jm}^k-Q_{mn}^k+q_m^k-\ell_{jm}^kX_{jm}\right)-\left(R_{jm}^2+X_{jm}^2\right)\left(v_m^k-\right. \\ &\left.v_j^k+2P_{jm}^kR_{jm}+2Q_{jm}^kX_{jm}-\ell_{jm}^k\left(R_{jm}^2+X_{jm}^2\right)\right)\Bigg]\Bigg\} \end{aligned} \tag{19}$$

$$\begin{aligned} P_{jm}^{k+1}=P_{jm}^k-\xi_5\Bigg\{&\lambda_j^{p,k}-\lambda_m^{p,k}-2\lambda_j^{Q,k}R_{jm}+\frac{2}{v_j^k}P_{jm}^k\mu_j^k+ \\ &\rho\Bigg[\left(P_{jm}^k-P_{mn}^k+p_m^k-\ell_{jm}^kR_{jm}\right)-\left(P_{ij}^n-P_{jk}^n+p_j^n-\ell_{ij}^nR_{ij}\right)+ \\ &2R_{jm}\left(v_m^k-v_j^k+2P_{jm}^kR_{jm}+2Q_{jm}^kX_{jm}-\ell_{jm}^k\left(R_{jm}^2+X_{jm}^2\right)\right)+ \\ &\frac{2}{\left(v_j^k\right)^2}P_{jm}^k\left(\left(P_{jm}^k\right)^2+\left(Q_{jm}^k\right)^2-\ell_{jm}^kv_j^k\right)\Bigg]\Bigg\} \end{aligned} \tag{20}$$

$$\begin{aligned} Q_{jm}^{k+1}=Q_{jm}^k-\xi_6\Bigg\{&\lambda_j^{q,k}-\lambda_m^{q,k}-2\lambda_j^{Q,k}X_{jm}+\frac{2}{v_j^k}Q_{jm}^k\mu_j^k+ \\ &\rho\Bigg[\left(Q_{jm}^k-Q_{mn}^k+q_m^k-\ell_{jm}^kX_{jm}\right)-\left(Q_{ij}^k-Q_{jm}^k+q_j^k-\ell_{ij}^kX_{ij}\right)+ \\ &2x_{jm}\left(v_m^k-v_j^k+2P_{jm}^kR_{jm}+2Q_{jm}^kX_{jm}-\ell_{jm}^k\left(R_{jm}^2+X_{jm}^2\right)\right)+ \\ &\frac{2}{\left(v_j^k\right)^2}Q_{jm}^k\left(\left(P_{jm}^k\right)^2+\left(Q_{jm}^k\right)^2-\ell_{jm}^kv_j^k\right)\Bigg]\Bigg\} \end{aligned} \tag{21}$$

Then, each agent updates its local dual variables as derived in (22)-(25).

$$\lambda_j^{p,k+1}=\lambda_j^{p,k}+\xi_7\left(P_{jm}^k-P_{ij}^k-p_j^k+\ell_{ij}^k R_{ij}\right) \tag{22}$$

$$\lambda_j^{q,k+1}=\lambda_j^{q,k}+\xi_8\left(Q_{jm}^k-Q_{ij}^k-q_j^k+\ell_{ij}^k X_{ij}\right) \tag{23}$$

$$\lambda_j^{\Omega,k+1}=\lambda_j^{\Omega,k}+\xi_9\left[v_j^n-v_m^k-2P_{jm}^k R_{jm}-2Q_{jm}^k X_{jm}+\ell_{jm}^k\left(R_{jm}^2+X_{jm}^2\right)\right] \tag{24}$$

$$\mu_j^{k+1}=\mu_j^k+\xi_{10}\left[\frac{1}{v_j^k}\left(\left(P_{jm}^k\right)^2+\left(Q_{jm}^k\right)^2\right)-\ell_{jm}^k\right] \tag{25}$$

where λ_j^p and λ_j^q are used for augmenting the real and reactive power balance equality constraints in (2) and (3) at node j, respectively; λ_j^Ω is used for augmenting the Ohm's law constraint in (4); and μ_j is used for augmenting the branch power flow inequality constraint in (5). Note that, according to Fig. 1, node i is the ancestor node of node j, and node k is the children node of node j.

To show that the above agent update equations can be carried out independently by each agent with 1-hop P2P communication, we first define the local variables of the agent. For any agent $j\in\mathcal{V}$, its local primal variables x_j and dual variables $\{\lambda_j,\mu_j\}$ are defined in (26)-(28).

$$x_j:=\left\{p_j,\ q_j,\ v_j,\ \ell_{jm},\ P_{jm},\ Q_{jm}\right\}\quad x_j\subset\boldsymbol{x} \tag{26}$$

$$\lambda_j:=\left\{\lambda_j^p,\ \lambda_j^q,\ \lambda_j^\Omega\right\}\quad \lambda_j\subset\boldsymbol{\lambda} \tag{27}$$

$$\mu_j:=\left\{\mu_j\right\}\quad \mu_j\subset\boldsymbol{\mu} \tag{28}$$

Then, based on the agent update equations derived in (16)-(25), we find that in addition to the agent local variables $\{x_j,\lambda_j,\mu_j\}$, additional variables from its ancestor node i, i.e., $\{v_i,\ell_{ij},P_{ij},Q_{ij},\lambda_i^\Omega\}$, and variables from its children node m, i.e., $\{p_m,q_m,v_m,P_{mn},Q_{mn},\lambda_m^p,\lambda_m^q\}$, are needed. Therefore, the optimal solution to the ACOPF problem in (11) can be collectively computed by the distributed agents in finite number of iterations, if every agent observes these three steps at every iteration $\forall j\in\mathcal{V}$.

Step 1: send partial local primal and dual variables $\mathcal{A}_j\{p_j,q_j,v_j,P_{jm},Q_{jm},\lambda_j^p,\lambda_j^q\}$ to its ancestor agent.

Step 2: send partial local primal and dual variables $\mathcal{C}_j\{v_j,\ell_{jm},P_{jm},Q_{jm},\lambda_j^\Omega\}$ to all of its children agents.

Step 3: update its local variables $\{x_j,\lambda_j,\mu_j\}$ using the received variables from its ancestor and children nodes.

The rigorous convergence proof and analysis of the distributed proposed ACOPF algorithm are presented in Supplementary material.

IV. Distributed Computing Structures

A. HPC-based Distributed Computing System

1) Computing Hardware

To emulate the distributed computing environment required by the proposed algorithm, i.e., individual processor, RAM, and communication interface, we leverage the vast amount of processing cores and RAM provided by the HPC system at North Carolina State University, USA, called Henry2 cluster. As presented in Fig. 2, when running the proposed algorithm, we emulate the on-site edge computing units using processors in Henry2 cluster, which is a Linux cluster with approximately 1000 computing nodes (with independent Intel Xeon processors, RAM, and communication interfaces) and approximately 10000 cores.

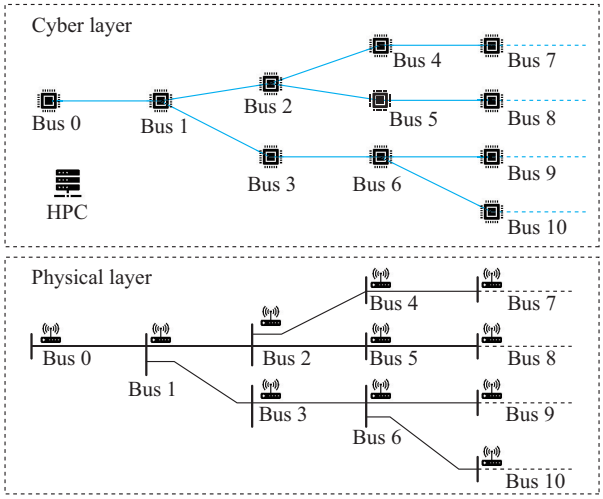

Fig. 2. HPC-based distributed computing system for distributed ACOPF.

2) Computing Software

As for the software environment, we use the MATLAB Parallel Server toolbox in MATLAB 2019a to create and operate the distributed computing cluster. The agent updating equations in (16) - (25) and 1-hop P2P communication are programmed using MATLAB language. When launching the experiment, we use "batch" commands in MATLAB to initiate the proposed algorithm on all distributed computing nodes so that these nodes can collectively solve the ACOPF problem.

3) Inter-processor Communication

The inter-processor communication is needed to emulate the P2P communication among field edge computing units. Message passing interface (MPI) is used by the inter-processor communication over a InfiniBand network. The variable value sharing between neighboring nodes is programmed in MATLAB language, and it is eventually handled by the corresponding MPI software package in the MATLAB Parallel Server toolbox.

4) Emulation Capability

Since each computing node in the Henry2 cluster has local processors, RAM, and communication interface, one can effectively use Henry2 cluster to emulate and validate distributed computing tasks with up to 1000 agents/nodes.

B. Raspberry Pi (RPI) Cluster HIL Testbed

Due to the fact that the P2P communication in HPC-based distributed computing system is handled by the MPI over a InfiniBand network, certain communication characteristics, e.g., delay, traffic pattern, overhead have limited significance representing the actual field edge computing unit of P2P communication. Therefore, to further investigate the communication performance of the proposed algorithm, we build

the HIL testbed in Fig. 3.

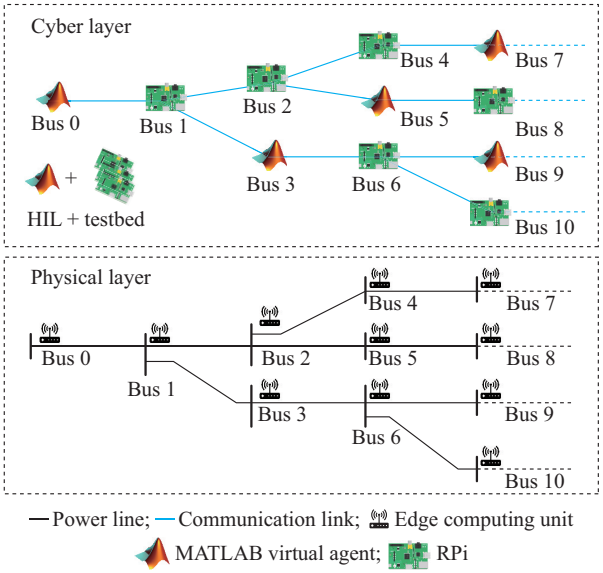

Fig. 3. HIL testbed for proposed algorithm.

1) Computing Hardware and Software

As presented in Fig. 3, we use actual edge unit RPi to perform distributed computation. The agent updating equations in (16)-(25) and 1-hop P2P communication are programmed using C++ that runs on the RPi in real time. In addition to the actual seven RPis used, we augment the rest of the system with virtual MATLAB computing agents. All of the virtual agents run concurrently on a single PC with MATLAB 2020a environment, Intel i7 processor, and 16 GB RAM. Overall, this HIL testbed is programmed to be an event-driven system with a sampling time of 1 ms, i.e., all agents will not perform current iteration variable updates until all the input variables from its 1-hop neighbors are received.

2) P2P Communication over WiFi

The P2P communications in the HIL testbed are classified into three types: ① RPi to RPi communication is implemented using user datagram protocol (UDP) over 5 GHz WiFi; ② RPi to MATLAB virtual agent communication is implemented using UDP over the Ethernet; ③ the communication among MATLAB virtual agents is implemented using the in-RAM variable assignment.

Figure 4 presents the P2P communication logistics between agent j and its 1-hop neighbors, where $\mathcal{A}$ and $\mathcal{C}$ are the data packets. The data packet $\mathcal{A}$ or $\mathcal{C}$ is sent once requested, and each agent will not request data packet unless it is ready to perform the corresponding iteration update. This on-demand UDP P2P communication logistics can avoid unnecessary UDP broadcasting, excessive re-sending, and UDP buffer of flooding receiver.

3) Emulation Capability

With actual edge unit RPi and UDP over WiFi used in the HIL testbed, one can investigate the communication performance of the proposed algorithm by quantifying and analyzing the communication performance of the distributed RPi agents.

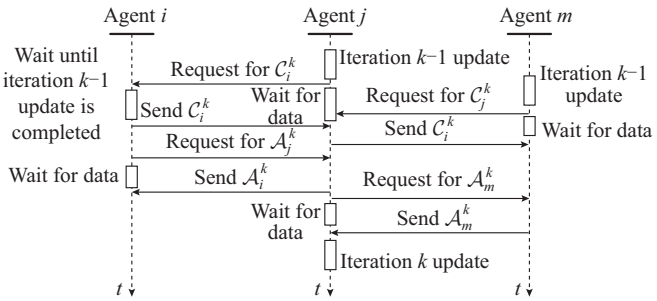

Fig. 4. Illustration of P2P communication logistics.

V. Experimental Results

In this section, we implement the proposed algorithm on both HPC-based platform and HIL testbed to benchmark its performance. The selected test systems are 11 kV 22-bus [32], 12.7 kV 69-bus [33], 11 kV 85-bus [34], and 12.5 kV 141-bus [35] radial distribution systems. The generation cost function parameters of the slack bus are $a=0.04$, $b=20$, and $c=0$. The references of ACOPF solution are obtained via three commonly used centralized optimization methods, i. e., interior point method (IPM), sequential quadratic programming (SQP) method, and active-set method (ASM). These centralized benchmark algorithms are implemented using the library provided by the MATLAB Optimization Toolbox.

A. Validation Using HPC-based Platform

The HPC-based platform presented in Section IV-A is used to validate the proposed algorithm. The proposed algorithm is used to solve the ACOPF problem on the four selected test systems, i.e., 22-bus, 69-bus, 85-bus, and 141-bus test systems. Figure 5 presents the results of the ACOPF problem (22-bus test system), where P^* and Q^* are the optimal real power and reactive power, respectively. Figure 5(a) depicts the convergence real and reactive power outputs of the slack bus. The algorithm converges at iteration 3098, whereas the total time used is summarized in Table II. Figure 5(b) shows the bus voltage at each bus, in which we observe that the voltage drops along the feeder. Figure 5(c) presents the incremental cost at each bus, in which we find that the lateral buses have higher incremental costs. This phenomenon is caused by an increase of line losses. Finally, the maximum violation progression of the constraint is rendered in Fig. 5(d). Note that the threshold for the maximum constraint violation is set to be 10^{-3} per unit.

Figure 6(a) presents the convergence of real and reactive power outputs of the slack bus in 69-bus test system. The algorithm converges at iteration 2386, whereas the total time used is summarized in Table II. Figure 6(b) shows the bus voltage at each bus, in which the inconsistencies in voltage drop at buses 28 and 66 are observed. This is because bus 28 is a branch head that connects to bus 3, and bus 66 is another branch head that connects to bus 11. Figure 6(c) presents the incremental cost at each bus. The discontinuities of the incremental cost increase are also caused by the branching effect. Finally, the maximum violation progression of the constraint is presented in Fig. 6(d).

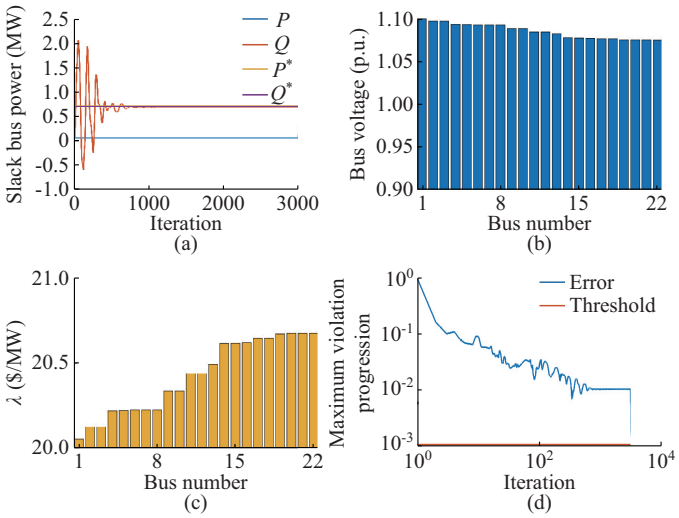

Fig. 5. Distributed ACOPF results (22-bus test system). (a) Convergence of slack bus. (b) Bus voltage. (c) Incremental cost at each bus. (d) Maximum violation progression of constraint.

TABLE II
PERFORMANCE STATISTICS OF PROPOSED ALGORITHM.

Test system	Iteration	CPU time (s)	Communication time (s)	Total time (s)	Receiving data rate (Mbit/s)	Sending data rate (Mbit/s)
22-bus	3098	1.53	189.30	190.83	1.41	0.56
69-bus	2386	1.22	411.94	413.16	0.50	0.20
85-bus	4842	2.50	1053.10	1055.60	0.40	0.16
141-bus	2690	1.52	968.58	970.10	0.24	0.10

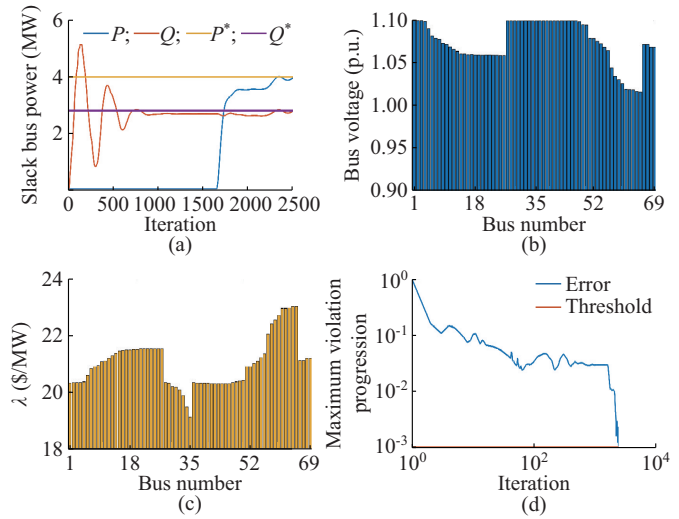

Fig. 6. Distributed ACOPF results (69-bus test system). (a) Convergence of slack bus. (b) Bus voltage. (c) Incremental cost at each bus. (d) Maximum violation progression of constraint.

Figure 7(a) presents the convergence of real and reactive power outputs of the slack bus in 85-bus test system. The algorithm converges at iteration 4842, whereas the total time used is summarized in Table II. Figure 7(b) shows the bus voltage at each bus. The discontinuities of the voltage drop are also caused by the branching effect. Figure 7(c) presents the incremental cost at each bus. The discontinuities of the incremental cost increase are caused by the branching effect. Finally, the maximum violation progression of the constraint is presented in Fig. 7(d).

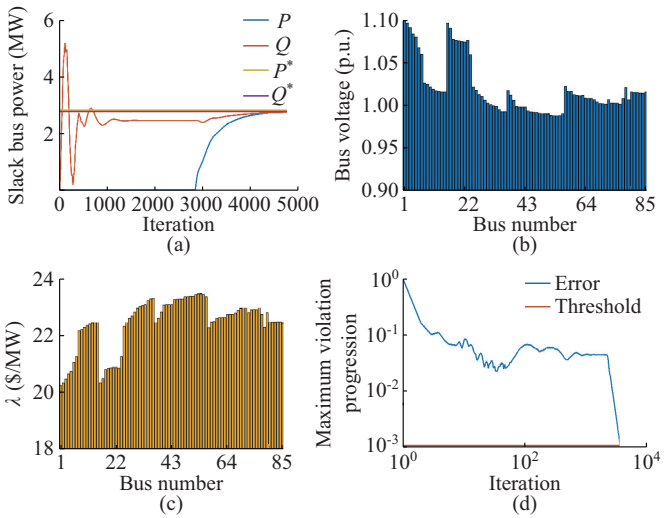

Fig. 7. Distributed ACOPF results (85-bus test system). (a) Convergence of slack bus. (b) Bus voltage. (c) Incremental cost at each bus. (d) Maximum violation progression of constraint.

Figure 8(a) presents the convergence of real and reactive power outputs of the slack bus in 141-bus test system. The algorithm converges at iteration 2690, whereas the total time used is summarized in Table II. Figure 8(b) shows the bus voltage at each bus. The discontinuities of the voltage drop are also caused by the branching effect. Figure 8(c) presents the incremental cost at each bus. The discontinuities of the incremental cost increase are caused by the branching effect. Finally, the maximum violation progression of the constraint is presented in Fig. 8(d).

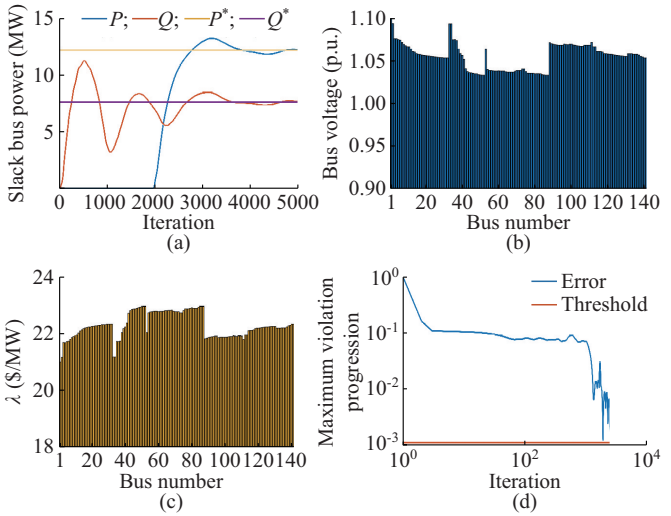

Fig. 8. Distributed ACOPF results (141-bus test system). (a) Convergence of slack bus. (b) Bus voltage. (c) Incremental cost at each bus. (d) Maximum violation progression of constraint.

To validate the accuracy of the proposed algorithm, we calculate the mean absolute error of the optimization vari-

ables $\{p, q, v, \ell, P, Q\}$ with respect to the three benchmark algorithms, i.e., IPM, SQP, and ASM. Accurate comparisons of the ACOPF solution between the reference algorithms and the proposed algorithm in the test systems are summarized in Table III. We found that the mean absolute error of the proposed algorithm is on the 10^{-4} scale in average. According to the results in Table III, the proposed algorithm appears to be consistent with the reference algorithms.

TABLE III
MEAN ABSOLUTE ERRORS OF ACOPF SOLUTIONS

Test system	Mean absolute error		
	Versus IPM	Versus SQP	Versus ASM
22-bus	9.96×10^{-4}	9.47×10^{-4}	9.90×10^{-4}
69-bus	9.92×10^{-4}	9.39×10^{-4}	9.72×10^{-4}
85-bus	1.01×10^{-3}	1.12×10^{-3}	1.00×10^{-3}
141-bus	9.88×10^{-4}	8.99×10^{-4}	9.91×10^{-4}
Average	9.97×10^{-4}	9.76×10^{-4}	9.88×10^{-4}

The algorithm performances, in terms of iterations, CPU time, communication time, total execution time, receiving data rate, and sending data rate, are summarized in Table III.

CPU time is the total amount of time when the processor is dedicated to performing algorithm updates, whereas communication time is the total amount of time when the processor is dedicated to communicating, e.g., waiting, sending, listening. The total execution time of the agent is a summation of CPU time and communication time. Based on these results, we first observe that there is no obvious correlation between iterations and system sizes, e.g., the required iteration grows as system size increases. This is due to the fact that these ACOPF problems are different, e.g., different loading conditions, constraints, topologies, etc. Thus, these ACOPF problems have different convexity levels. The proposed algorithm is configured to use fixed gradient step sizes for four ACOPF problems. Since the gradient step sizes are the same, it will take more time to converge in a less convex problem. This explains the reason why the 85-bus test system takes more iterations than the 141-bus test system. We can clearly observe that the processor spends the majority of the execution time (>99%) on communication. This is due to the inefficient communication handling among the MATLAB workers. Therefore, a more representative study is carried out using the HIL testbed and is presented in Section V-B. Besides, we find that the receiving and sending data rates drop as the system size grows. This is because the agent execution time per iteration increases as system size grows. As presented in Fig. 9(a), we can see that the total computation time, i.e., execution time, grows linearly as the system size grows. Additionally, the average agent receiving and sending data rates per iteration for all cases are presented in Fig. 9 (b). We find that both data rates stay the same. Since each agent only communicates with its 1-hop neighbor, the number of 1-hop neighbors does not grow as system size grows. The average 1-hop neighbors that each agent has in four cases are 1.91, 1.97, 1.98, and 1.99, respectively. The reason why this number is close to 2 is that the majority of the agents only have one ancestor node and one children node, and some of the "endpoint" agent does not have children nodes, which contributes to the great scalability in Fig. 9(b).

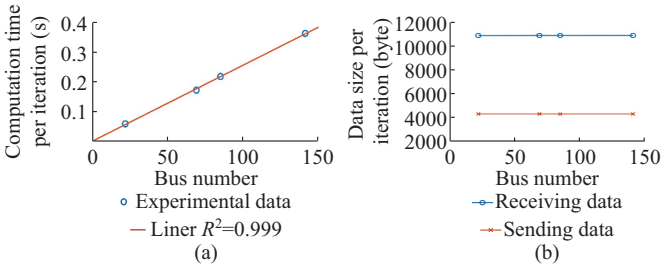

Fig. 9. Agent computation time and data size per iteration. (a) Agent computation time. (b) Data size of agent communication.

B. Algorithm Evaluation via HIL Testbed

The HIL testbed presented in Section IV-B is used to evaluate the communication performance of the proposed algorithm. The 141-bus test system is selected for the evaluation. Seven available RPis are randomly deployed among the 141 buses, i.e., buses 14, 22, 39, 77, 96, 123, and 138. To evaluate the communication characteristics of the algorithm, we use the Wireshark software to capture the network traffic. Figure 10 shows the example P2P communication between bus 39 RPi agent (192.168.1.103) and the bus 40 MATLAB agent (192.168.1.26). P2P communication logistics are implemented according to Fig. 4. Recall the data packets $\mathcal{A}$ and $\mathcal{C}$ defined in Section III-B, i.e., for any agent j. The P2P traffic ⓐ in Fig. 10 is when RPi agent requests data $\mathcal{A}$ from its ancestor MATLAB agent (2-B packet) and receives the data $\mathcal{A}$ (88-B packet). The P2P traffic ⓑ is when MATLAB agent requests data $\mathcal{C}$ from its children RPi agent (1-B packet) and receives the data $\mathcal{C}$ (72-B packet). Note that this request is responded at the fourth request attempt. In Fig. 10, L_{en} is the length in byte.

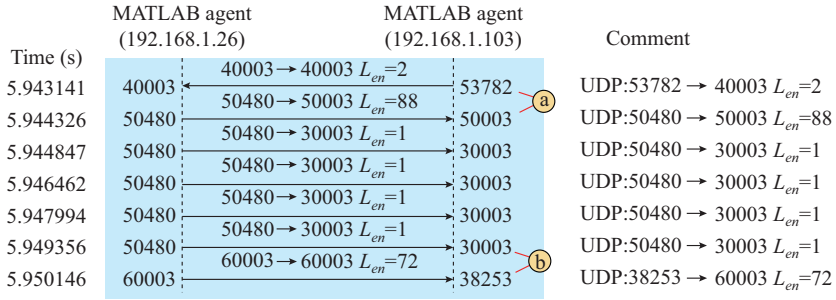

Fig. 10. Example captured P2P communication traffic.

Figure 11 presents the receiving and sending data rates of RPi agent during the algorithm execution. Both data rates are measured in terms of network bandwidth usage, i. e., Mbit/s. Note that both rates include the request and data traffic. The average receiving and sending data rates of the bus 39 RPi agent are 0.085 and 0.062 Mbit/s, respectively. Moreover, the average per RPi agent algorithm communication footprint is in terms of iteration (2690), CPU time (2.69 s), total execution time (187.03 s), receiving data rate (0.0849 Mbit/s) and sending data rate (0.0599 Mbit/s). Based on the results, we know that the ACOPF problem is solved in 187.03 s. Similar to the results obtained from the HPC-based platform, the CPU time only accounts for 1.44% of the total execution time. The average per agent receiving and sending data rates are 0.0849 Mbit/s and 0.0599 Mbit/s, respectively.

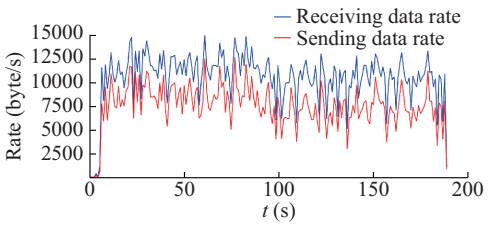

Fig. 11. Receiving and sending data rates of RPi agent.

VI. Conclusion

In summary, we focus on the SOCP relaxed ACOPF problems of the radial distribution systems and microgrids. The proposed ACOPF solver is developed based on the augmented Lagrangian and PDS method. In addition to the development of the algorithm, two distributed computing platforms, i.e., HPC based and RPi based HIL platforms, are developed to validate and benchmark the proposed algorithm. Upon the prototyping on two distributed computing platforms, the proposed algorithm is deployed to four typical radial IEEE distribution systems. Experimental results and comparative analysis indicate that: ① the solution of the proposed algorithm is accurate and consistent with mainstream ACOPF algorithms; ② the agent computation time of the proposed algorithm increases linearly as the system size grows; ③ the proposed algorithm has a low communication overhead, i.e., smaller than 0.1 Mbit/s.

References

[1] J. Carpentier, "Contribution to the economic dispatch problem," *Bulletin de la Societe Francoise des Electriciens*, vol. 3, no. 8, pp. 431-447, Aug. 1962.
[2] M. B. Cain, R. P. Oneill, A. Castillo *et al.*, "History of optimal power flow and formulations," *Federal Energy Regulatory Commission*, vol. 1, pp. 1-36, Jan. 2012.
[3] D. K. Molzahn, F. Dorfler, H. Sandberg *et al.*, "A survey of distributed optimization and control algorithms for electric power systems," *IEEE Transactions on Smart Grid*, vol. 8, no. 6, pp. 2941-2962, Nov. 2017.
[4] Y. Wang, S. Wang, and L. Wu, "Distributed optimization approaches for emerging power systems operation: a review," *Electric Power System Research*, vol. 144, pp. 127-135, Mar. 2017.
[5] M.-Y. Chow, Y. Zhang, and N. Rahbari-Asr, "Consensus-based distributed scheduling for cooperative operation of distributed energy resources & storage devices in smart grids," *IET Generation, Transmission and Distribution*, vol. 10, pp. 1268-1277, Apr. 2016.
[6] X. Hu, T. Liu, C. He *et al.*, "A real-time power management technique for microgrid with flexible boundaries," *IET Generation, Transmission and Distribution*, vol. 14, no. 16, pp. 3161-3170, Aug. 2020.
[7] M. Batool, S. Islam, and F. Shahnia, "Market model for clustered microgrids optimisation including distribution network operations," *IET Generation, Transmission and Distribution*, vol. 13, no. 22, pp. 5139-5150, Nov. 2019.
[8] A. Alsabbagh and C. Ma, "Distributed charging management of electric vehicles considering different customer behaviors," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5119-5127, Aug. 2020.
[9] B. Chen, Z. Ye, C. Chen *et al.*, "Toward a MILP modeling framework for distribution system restoration," *IEEE Transactions on Power Systems*, vol. 34, no. 3, pp. 1749-1760, May 2019.
[10] Z. Cheng, J. Duan, and M.-Y. Chow, "Reliability assessment and comparison between centralized and distributed energy management system in islanding microgrid," in *Proceedings of 2017 North American Power Symposium (NAPS)*, Morgantown, USA, Sept. 2017, pp. 1-6.
[11] Z. Cheng, J. Duan, and M.-Y. Chow, "To centralize or to distribute: that is the question: a comparison of advanced microgrid management systems," *IEEE Industrial Electronics Magazine*, vol. 12, no. 1, pp. 6-24, Mar. 2018.
[12] A. Nedi and J. Liu, "Distributed optimization for control," *Robotics and Autonomous System*, vol. 1, no. 1, pp. 77-103, May 2018.
[13] T. Yang, X. Yi, J. Wu *et al.*, "A survey of distributed optimization," *Annual Reviews in Control*, vol. 47, pp. 278-305, Mar. 2019.
[14] Y. Zhang, N. Rahbari-Asr, J. Duan *et al.*, "Day-ahead smart grid cooperative distributed energy scheduling with renewable and storage integration," *IEEE Transactions on Sustainable Energy*, vol. 7, no. 4, pp. 1739-1748, Oct. 2016.
[15] A. Y. S. Lam, B. Zhang, and D. N. Tse, "Distributed algorithms for optimal power flow problem," in *Proceedings of 2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, Maui, USA, Feb. 2012, pp. 430-437.
[16] E. Dall'Anese, H. Zhu, and G. B. Giannakis, "Distributed optimal power flow for smart microgrids," *IEEE Transactions on Smart Grid*, vol. 4, no. 3, pp. 1464-1475, Sept. 2013.
[17] T. Erseghe, "Distributed optimal power flow using ADMM," *IEEE Transactions on Power Systems*, vol. 29, no. 5, pp. 2370-2380, Sept. 2014.
[18] Q. Peng and S. H. Low, "Distributed optimal power flow algorithm for radial networks, I: balanced single phase case," *IEEE Transactions on Smart Grid*, vol. 9, no. 1, pp. 111-121, Jan. 2018.
[19] R. Baldick, B. Kim, C. Chase *et al.*, "A fast distributed implementation of optimal power flow," *IEEE Transactions on Power Systems*, vol. 14, no. 3, pp. 858-864, Aug. 1999.
[20] D. Hur, J.-K. Park, and B. Kim, "Evaluation of convergence rate in the auxiliary problem principle for distributed optimal power flow," *IET Proceedings: Generation, Transmission and Distribution*, vol. 149, no. 5, p. 525, May 2002.
[21] F. J. Nogales, F. J. Prieto, and A. J. Conejo, "A decomposition methodology applied to the multi-area optimal power flow problem," *Annals of Operations Research*, vol. 120, pp. 99-116, Apr. 2003.
[22] J. Guo, G. Hug, and O. K. Tonguz, "Intelligent partitioning in distributed optimization of electric power systems," *IEEE Transactions on Smart Grid*, vol. 7, no. 3, pp. 1249-1258, May 2016.
[23] S. Kar, G. Hug, J. Mohammadi *et al.*, "Distributed state estimation and energy management in smart grids: a consensus + innovations approach," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 6, pp. 1022-1038, Dec. 2014.
[24] S. Mhanna, A. C. Chapman, and G. Verbi, "Component-based dual decomposition methods for the OPF problem," *Sustainable Energy, Grids and Networks*, vol. 16, pp. 91-110, Dec. 2018.
[25] M. Zhao, Q. Shi, Y. Cai *et al.*, "Distributed penalty dual decomposition algorithm for optimal power flow in radial networks," *IEEE Transactions on Power Systems*, vol. 35, no. 3, pp. 2176-2189, May 2020.
[26] A. Engelmann, Y. Jiang, T. Muhlpfordt *et al.*, "Toward distributed OPF using ALADIN," *IEEE Transactions on Power Systems*, vol. 34, no. 1, pp. 584-594, Jan. 2019.
[27] N. Meyer-Huebner, M. Suriyah, and T. Leibfried, "Distributed optimal power flow in hybrid ACDC grids," *IEEE Transactions on Power Systems*, vol. 34, no. 4, pp. 2937-2946, Jul. 2019.
[28] W. Lu, M. Liu, S. Lin *et al.*, "Incremental-oriented ADMM for distributed optimal power flow with discrete variables in distribution networks," *IEEE Transactions on Smart Grid*, vol. 10, no. 6, pp. 6320-6331, Nov. 2019.
[29] J. Huang, Z. Li, and Q. Wu, "Fully decentralized multiarea reactive

power optimization considering practical regulation constraints of devices," *International Journal of Electrical Power & Energy Systems*, vol. 105, pp. 351-364, Feb. 2019.
[30] S. H. Low, "Convex relaxation of optimal power flow part I: formulations and equivalence," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 1, pp. 15-27, Mar. 2014.
[31] Y. Nesterov, "Primal-dual subgradient methods for convex problems," *Mathematical Programming*, vol. 120, no. 1, pp. 221-259, Aug. 2009.
[32] M. R. Raju, K. R. Murthy, and K. Ravindra, "Direct search algorithm for capacitive compensation in radial distribution systems," *International Journal of Electrical Power & Energy Systems*, vol. 42, no. 1, pp. 24-30, Nov. 2012.
[33] D. Das, "Optimal placement of capacitors in radial distribution system using a fuzzy-GA method," *International Journal of Electrical Power & Energy Systems*, vol. 30, no. 6, pp. 361-367, Jul. 2008.
[34] D. Das, D. Kothari, and A. Kalam, "Simple and efficient method for load flow solution of radial distribution networks," *International Journal of Electrical Power & Energy Systems*, vol. 17, no. 5, pp. 335-346, Oct. 1995.
[35] H. Khodr, F. Olsina, P. D. O. D. Jesus *et al.*, "Maximum savings approach for location and sizing of capacitors in distribution systems," *Electric Power Systems Research*, vol. 78, no. 7, pp. 1192-1203, Jul. 2008.

Zheyuan Cheng received the B. S. degree in electrical engineering from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2015, and the Ph. D. degree in electrical engineering from North Carolina State University, Raleigh, USA, in 2020. He has been working as a Senior Engineer at Quanta Technology, LLC, Raleigh, USA, since 2021. He is a recipient of the 2021 Best Paper Award from IEEE Industrial Electronics Magazine. His research interests include distributed energy resources protection and control.

Mo-Yuen Chow is a Professor in the Department of Electrical and Computer Engineering at North Carolina State University, Raleigh, USA. He has established the Advanced Diagnosis, Automation, and Control Laboratory. He is an IEEE Fellow. He has received the IEEE Region-3 Joseph M. Biedenbach Outstanding Engineering Educator Award, the IEEE Industrial Electronics Society Anthony J Hornfeck Service Award. He is a Distinguished Lecturer of IEEE IES. He has been working several projects related to the cyber-physical microgrids since 2008. His research interests include distributed control and management on smart microgrids, batteries, and mechatronics systems.