Distributed Algorithms for Optimal Power Flow Problem

Albert Y.S. Lam, Baosen Zhang, and David N. Tse

Abstract— **Optimal power flow (OPF) is an important problem for power generation and it is in general non-convex. With the employment of renewable energy, it will be desirable if OPF can be solved very efficiently so that its solution can be used in real time. With some special network structure, e.g. trees, the problem has been shown to have a zero duality gap and the convex dual problem yields the optimal solution. In this paper, we propose a primal and a dual algorithm to coordinate the smaller subproblems decomposed from the convexified OPF. We can arrange the subproblems to be solved sequentially and cumulatively in a central node or solved in parallel in distributed nodes. We test the algorithms on IEEE radial distribution test feeders, some random tree-structured networks, and the IEEE transmission system benchmarks. Simulation results show that the computation time can be improved dramatically with our algorithms over the centralized approach of solving the problem without decomposition, especially in tree-structured problems. The computation time grows linearly with the problem size with the cumulative approach while the distributed one can have size-independent computation time.**

I. Introduction

In the past, research on power systems mainly focused on the core of the network, i.e., from the generation, via transmission, to the substations. All of the control, planning and optimization were done by a single entity (e.g. an ISO). With the integration of renewal energy and energy storage, self-healing ability, and demand response, the focus is shifted toward the consumer side, i.e. distribution networks, and this new paradigm is called the smart grid [1]. The optimal power flow (OPF) is one of the most important problems in power engineering and it aims to minimize the generation cost subject to demand constraints and the network physical constraints, e.g. bus voltage limits, bus power limits, thermal line constraints, etc. Due to the quadratic relations between voltage and power, OPF is non-convex. In general, heuristic approaches have been employed to solve the OPF but they are not guaranteed to yield the optimal solution. To simplify the calculation, with assumptions on lossless power line, constant voltage and small voltage angles, OPF can be linearized and this approximation is also called direct current (DC)-OPF, which is not accurate under all circumstances [2]. For the complex OPF, [3] suggested solving the problem in its dual form and studied the conditions of the power network with zero duality gap. In [4], it was shown that the duality gap is always zero for network structures such as trees which model distribution networks well. [5], as an independent work of this paper, decomposes the OPF in terms of cycles and branches and formulates the problem as an second-order cone program for tree networks which is equivalent to that given in [6]. In traditional power systems, OPF is mainly for planning purpose. For example, it is used to determine the system state in the day-ahead market with the given system information. In the smart grid paradigm, due to highly intermittent nature of the renewable, the later the prediction is made, the more reliable it is. If OPF can be solved very efficiently, we may solve the OPF in real time thus mitigating some of the unpredictability.

We aim at solving OPF efficiently. When the system size (e.g. the number of buses) increases, solving the problem in a centralized manner is not practical (this will be verified in the simulation). One possible way is to tackle the problem distributedly by coordinating several entities in the system, each of which handle part of the problem and their collaborative effort solves the whole problem. To do this, a communication protocol is needed to define what information should be conveyed among the entities. We can learn from the networking protocol development to design a communication protocol for OPF. The earliest form of protocols for the Internet was proposed in 70's. They were designed to handle the increasing volume of traffic sent over the Internet in an ad hoc manner. In 1998, Kelly et al. studied Transmission Control Protocol (TCP), which is one of the core protocols in TCP/IP [7]. They showed that TCP can be analyzed with a fundamental optimization problem for rate control and the algorithms developed from the optimization fit the ad hoc designed variants of TCP. This lays down a framework to design communication protocols for complex systems with reasoning. In this framework, we start with an optimization problem representing the system. By optimization decomposition [8], the problem is decomposed into (simpler) subproblems which can be solved by different entities in the system independently. The coordination between the subproblems define the communication protocols (i.e., what and how the data exchange between the entities). [9] shows that many problems in communications and networking can be cast under this framework and protocols can be designed through primal and dual decomposition. In this paper, we study OPF by decomposing it into subproblems with primal and dual decomposition. Then we propose the primal and dual algorithms, respectively, to solve OPF in a distributed manner and the algorithms determine the communication protocols. Our algorithms do not assume the existence of a communication overlay with topology different from the power network. In other words, a bus only needs to communicate with its one-hop neighbors in the power network. The

A.Y.S. Lam is with the Department of Computer Science, Hong Kong Baptist University, Kowloon Tong, Hong Kong albertlam@ieee.org

B. Zhang and D.N. Tse are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA, 94720 USA {zhangbao,dtse}@eecs.berkeley.edu

algorithms can employed to any power network as long as the strong duality holds.

The rest of this paper is organized as follows. In Section II, we give the OPF formulation and the necessary background. Section III describes the primal and dual algorithms and we study the OPF with quadratic cost function in Section IV. We present the simulation results in Section V and Section VI discusses the characteristics of the algorithms. We conclude the paper in Section VII.

II. Preliminaries

A. Problem Formulation

Assume that there are n buses in the power network. For buses i and k, $i \sim k$ means that they are connected by a power line and $i \not\sim k$ otherwise. Let z_{ik} and y_{ik} be the complex impedance and admittance between i and k, respectively, and we have $y_{ik} = \frac{1}{z_{ik}}$. We denote $\mathbf{Y} = (Y_{ik}, 1 \le i, k \le n)$ as the admittance matrix, where

$$Y_{ik} = \begin{cases} \sum_{l \sim i} y_{il} & \text{if } i = k \\ -y_{ik} & \text{if } i \sim k \\ 0 & \text{if } i \not\sim k. \end{cases}$$

Let $\mathbf{v} = (V_1, V_2, \ldots, V_n)^T \in \mathbb{C}^n$ and $\mathbf{i} = (I_1, I_2, \ldots, I_n)^T \in \mathbb{C}^n$ be the voltage and current vectors, respectively. By Ohm's Law and Kirchoff's Current Law, we have $\mathbf{i} = \mathbf{Yv}$. The apparent power injected at bus i is $S_i = P_i + jQ_i = V_i I_i^H$, where P_i and Q_i are the real and reactive power, respectively, and H means Hermitian transpose. We have the real power vector $\mathbf{p} = (P_1, P_2, \ldots, P_n)^T = \text{Re}\{\text{diag}(\mathbf{vv}^H\mathbf{Y}^H)\}$, where $\text{diag}(\mathbf{vv}^H\mathbf{Y}^H)$ forms a diagonal matrix whose diagonal is $\mathbf{vv}^H\mathbf{Y}^H$. We define the cost function of Bus i as $\text{cost}_i(P_i) = c_{i2}P_i^2 + c_{i1}P_i + c_{i0}$, where $c_{i0}, c_{i1}, c_{i2} \in \mathbb{R}$ and $c_{i2} \ge 0, \forall i$. OPF can be stated as

$$\text{minimize} \sum_{i=1}^{n} \text{cost}_i(P_i) \tag{1a}$$

$$\text{subject to } \underline{V_i} \le |V_i| \le \overline{V_i}, \forall i \tag{1b}$$

$$\underline{P_i} \le P_i \le \overline{P}_i, \forall i \tag{1c}$$

$$P_{ik} \le \overline{P}_{ik}, \forall i, k \tag{1d}$$

$$\mathbf{p} = \text{Re}\{\text{diag}(\mathbf{vv}^H\mathbf{Y}^H)\} \tag{1e}$$

where $\underline{V_i}$, $\overline{V_i}$, $\underline{P_i}$, $\overline{P_i}$, and $\overline{P}_{ik}$ are the lower and upper voltage limits of bus i, the lower and upper power limits of bus i, and the real power flow limit between buses i and k, respectively. Eq. (1b) is the nodal voltage constraint limiting the magnitude of bus voltage. Eq. (1c) is the nodal power constraint limiting the real power generated or consumed and (1d) is the flow constraint. Eq. (1e) describes the physical properties of the network. In this formulation, $\mathbf{p}$ and $\mathbf{v}$ are the variables. Eqs. (1c) and (1d) are box constraints with respect to $\mathbf{p}$ which are the variables of the objective function (1a) and they are relatively easy to handle. Eq. (1b) together with (1e) make the problem non-convex and hard to solve. To illustrate the algorithms, we first consider a simplified version of OPF with $c_{i2} = c_{i0} = 0, \forall i$ and neglect (1c) and (1d). Having $c_{i0} = 0$ will not affect the optimal solution of the original problem. We will explain how to handle non-zero c_{i2} later. By introducing a $n \times n$ complex matrix $W = (W_{ik}, 1 \le i, k \le n) = \mathbf{vv}^H$, we can write the simplified OPF in the sequel:

$$\text{minimize} \sum_{i=1}^{n} c_{i1}P_i \tag{2a}$$

$$\text{subject to } \underline{V_i}^2 \le W_{ii} \le \overline{V_i}^2, \forall i \tag{2b}$$

$$\text{rank}(W) = 1 \tag{2c}$$

$$\mathbf{p} = \text{Re}\{\text{diag}(\mathbf{vv}^H\mathbf{Y}^H)\} \tag{2d}$$

Let $\mathbf{C} = \text{diag}(c_{11}, c_{21}, \ldots, c_{n1})$ and $\mathbf{M} = (M_{ik}, 1 \le i, k \le n) = \frac{1}{2}(\mathbf{Y}^H\mathbf{C} + \mathbf{CY})$. By relaxing the rank constraint (22c), we have the following semidefinite program (SDP):

$$\text{minimize } \text{Tr}(\mathbf{MW}) \tag{3a}$$

$$\text{subject to } \underline{V_i}^2 \le W_{ii} \le \overline{V_i}^2, \forall i \tag{3b}$$

$$\mathbf{W} \succeq 0 \tag{3c}$$

where $\text{Tr}(\cdot)$ is the trace operator. We can solve this SDP at a central control center. However, current algorithms for SDP, e.g. primal-dual interior-point methods [10], can only handle problems with size up to *several hundreds*. We will decompose the problem into smaller ones by exploring the network structure.

B. Zero Duality Gap

By [4], the simplified OPF and SDP share the equivalent optimal solution provided that the network has a tree structure, is a lossless cycle, or a combination of tree and cycle. For these kinds of network structures which are typically found in distribution networks, the optimal solution computed from (3) is exactly the same as that from (2). Targeting distribution networks, we can merely focus on (3). For completeness, the approach in [4] is outlined below.

The dual of (3) is given by

$$\text{maximize} \sum_{i=1}^{n} (-\overline{\lambda}_i \overline{V}_i^2 + \underline{\lambda}_i \underline{V_i}^2) \tag{4}$$

$$\text{subject to } \overline{\lambda}_i \ge 0, \underline{\lambda}_i \ge 0 \ \forall i$$

$$\Lambda + M \succcurlyeq 0,$$

where $\overline{\lambda}_i$ and $\underline{\lambda}_i$ are the Lagrangian multipliers associated with the constraints $W_{ii} \le \overline{V}_i^2$ and $\underline{V_i}^2 \le W_{ii}$ respectively. From the KKT conditions, [4] showed that (3) always has a solution that is rank 1.

C. Graph Structure

We will use the following graph structures to decompose SDP. Consider a graph $G = (V, E)$, where $V = \{i | 1 \le i \le n\}$ are vertices and $E = \{(i, k) \in V \times V\}$ are edges. Vertices i and k are adjacent if $(i, k) \in E$. A clique C is a subset of V whose induced subgraph is fully connected, i.e., $(i, k) \in E, \forall i, k \in C$. A clique is maximal if it cannot be extended to form a larger one by including any adjacent vertex to the clique. In other words, there does not exist a clique whose proper subset is a maximal clique. A chord is

an edge which connects two non-adjacent vertices in a cycle. A graph is chordal if each of its cycles with four or more vertices contains a chord. Thus a chordal graph does not have a cycle with four or more vertices. If G is not chordal, we can produce a corresponding chordal graph $\tilde{G} = (V, \tilde{E})$, where $\tilde{E} = E \cup E_f$ and $E_f = \{(i,j) \in V \times V - E\}$ are chords of G, called fill-in edges. $\tilde{G}$ is not unique. From $\tilde{G}$, we can compute the set of all possible maximal cliques $\mathcal{C} = \{C_1, \ldots, C_{|\mathcal{C}|}\}$, where $C_i = \{j \in V\}$ whose induced subgraph is complete and maximal. If G is a tree, each pair of vertices connecting by an edge forms a maximal clique. For a tree with n vertices, it can be decomposed into $n-1$ maximal cliques.

For $\mathbf{M}$ in (3a), we can induce the corresponding G by having $V = \{i | 1 \leq i \leq n\}$ and $E = \{(i,k) | M_{i,k} \neq 0\}$. G has a very close relationship with the power network structure because of $\mathbf{Y}$. If all $c_{i1}, \forall i$ are non-zero, G directly represents the network.

We use the following procedure to produce $\mathcal{C}$ from $\mathbf{M}$:

1) Construct a graph $G = (V, E)$ from $\mathbf{M}$.
2) From G, compute Maximum Cardinality Search [11] to construct an elimination ordering σ of vertices [12].
3) With σ, perform Fill-In Computation [13] to obtain a chordal graph $\tilde{G}$.
4) From $\tilde{G}$, determine the set of maximal cliques $\mathcal{C}$ by the Bron-Kerbosch algorithm [14].

Note that similar ideas about maximal cliques have been utilized to develop a parallel IPM for SDP [15]–[17]. However, we make use of the ideas to decompose SDP into smaller problems, which can be tackled by any appropriate SDP algorithm, not necessarily IPM. Therefore, our approach is more flexible on that any future efficient SDP algorithm can be incorporated into our framework.

III. Algorithms

A. Primal Algorithm

The objective function (3a) can be expressed as

$$\text{Tr}(\mathbf{MW}) = \sum_{i,k=1}^{n} M_{ik}^H W_{ik}, \tag{5}$$

where each term $M_{ik}^H W_{ik}$ can be classified into one of the following three categories:

1) Ignored terms: Each of which has $M_{ik} = 0$. Let $\mathcal{I} = \{(i,k) | M_{ik} = 0\}$;
2) Unique terms: For $M_{ik} \neq 0$, both i and k belong to a unique maximal clique. If $i, k \in C_l$, then $i, k \notin C_r, \forall r \neq l$. Let $\mathcal{U} = \{(i,k) | i, k \in C_l, \forall l, i, k \notin C_r, \forall r \neq l\}$; and
3) Shared terms: For $M_{ik} \neq 0$, both i and k belongs to more than one maximal clique.

As all the ignored terms can be ignored and each unique term is unique to each maximal clique, (5) becomes

$$\text{Tr}(\mathbf{MW}) = \sum_{\substack{i,k \in C_l, \forall C_l \in \mathcal{C} \\ |(i,k) \in \mathcal{U} - \mathcal{I}}} M_{ik}^H W_{ik} + \sum_{i,k | (i,k) \notin \mathcal{I} \cup \mathcal{U}} M_{ik}^H W_{ik}. \tag{6}$$

Eq. (3b) gives bounds to each $W_{ii}, 1 \leq i \leq n$ and it is equivalent to

$$\underline{V}_i^2 \leq W_{ii} \leq \overline{V}_i^2, \forall i \in C_l, \forall C_l \in \mathcal{C}. \tag{7}$$

By [16]–[18], a matrix can be completed into a positive semidefinite (PSD) matrix if and only if all its submatrices corresponding to the maximal cliques induced by the matrix are all PSD. Let $\mathbf{W}_{C_l}$ be the partial matrix of $\mathbf{W}$ with rows and columns indexed according to C_l. Eq. (3c) is equivalent to

$$\mathbf{W}_{C_l} \succeq 0, \forall C_l \in \mathcal{C}. \tag{8}$$

The semidefinite constraint (8) involves those variables which are not unique to C_l, i.e., W_{ik} such that $(i,k) \notin \mathcal{U}$. By introducing a slack variable $X_{ik,l} = W_{ik}$ for each shared W_{ik} in subproblem l, we define $\tilde{\mathbf{W}}_{C_l} = (\tilde{W}_{ik}, i, k \in C_l)$ and $\tilde{\mathbf{M}}_{C_l} = (\tilde{M}_{ik}, i, k \in C_l)$ where

$$\tilde{W}_{ik} = \begin{cases} W_{ik} & \text{if } (i,k) \in \mathcal{U} \\ X_{ik,l} & \text{otherwise} \end{cases}, \tilde{M}_{ik} = \begin{cases} M_{ik} & \text{if } (i,k) \in \mathcal{U} \\ 0 & \text{otherwise.} \end{cases}$$

If we fix all W_{ik} in the shared terms (those in the second summation in (6), the problem can be decomposed into $|\mathcal{C}|$ subproblems, each of which corresponds to a maximal clique. For C_l, we have the subproblem l, as follows:

$$\text{minimize } \text{Tr}(\tilde{\mathbf{M}}_{C_l} \tilde{\mathbf{W}}_{C_l}) \tag{9a}$$

$$\text{subject to } \underline{V}_i^2 \leq W_{ii} \leq \overline{V}_i^2, \forall i \in C_l, i \notin C_r, \forall r \neq l \tag{9b}$$

$$\tilde{\mathbf{W}}_{C_l} \succeq 0 \tag{9c}$$

$$X_{ik,l} = W_{ik}, \forall i, k | (i,k) \notin \mathcal{U}. \tag{9d}$$

Note that W_{ik}'s in (9d) are given to the subproblem. When given such W_{ik}'s, all subproblems are independent and can be solved in parallel. Let the domain of (9) be Φ_l. Given W_{ik} where $(i,k) \notin \mathcal{U}$, let $\phi_l(W_{ik} | (i,k) \notin \mathcal{U}) = \inf_{\tilde{\mathbf{W}}_{C_l} \in \Phi_k} \{\text{Tr}(\tilde{\mathbf{M}}_{C_l} \tilde{\mathbf{W}}_{C_l})\}$. The master problem is defined as

$$\text{minimize } \sum_{\forall C_l \in \mathcal{C}} \phi_l(W_{ik} | (i,k) \notin \mathcal{U}) + \sum_{i,k | (i,k) \notin \mathcal{U}} M_{ik}^H W_{ik} \tag{10a}$$

$$\text{subject to } \underline{V}_i^2 \leq W_{ii} \leq \overline{V}_i^2, \forall i | (i,i) \notin \mathcal{U}, \tag{10b}$$

which minimizes those W_{ik} shared by the maximal cliques. With those shared W_{ik} computed in (10), we minimize the W_{ik} unique to each subproblem given in (9).

In (10), for those shared W_{ik} corresponding to nodes (i.e. $i = k$), let $\lambda_{ii,l}$ be the Lagrangian multiplier for (9d) with $i = k$. The subgradient of W_{ii} with respect to subproblem l is $-\lambda_{ii,l}$ [19]. Thus the overall subgradient is $\sum_{l | i \in C_l} (-\lambda_{ii,l}) + M_{ii}^H$. At iteration t, we update W_{ii} by

$$W_{ii}[t+1] = Proj\left(W_{ii}[t] - \alpha[t] \left(\sum_{l | i \in C_l} (-\lambda_{ii,k}) + M_{ii}^H\right)\right), \tag{11}$$

where

$$Proj(x) = \begin{cases} \underline{V}_i^2 & \text{if } x < \underline{V}_i^2, \\ \overline{V}_i^2 & \text{if } x > \overline{V}_i^2, \\ x & \text{otherwise,} \end{cases}$$

$\alpha[t]$ is the step size at iteration t, and $W_{ii}[t]$ represents W_{ii} at iteration t.

Next we consider those W_{ik} corresponding to edges (i.e. $i \neq k$). Since W_{ik} and $X_{ik,l}$ are complex numbers, we can handle the real and imaginary parts separately, i.e. $\text{Re}\{X_{ik,l}\} = \text{Re}\{W_{ik}\}$ and $\text{Im}\{X_{ik,l}\} = \text{Im}\{W_{ik}\}$. Let $\lambda^{\text{Re}}_{ik,l}$ and $\lambda^{\text{Im}}_{ik,l}$ be their corresponding Lagrangian multipliers for subproblem l, respectively. In (5), $\forall i \neq k$, the ik and ki terms always come in a pair. We have $M^H_{ik}W_{ik} + M^H_{ki}W_{ki} = 2\text{Re}\{M^H_{ik}\}\text{Re}\{W_{ik}\} - 2\text{Im}\{M_{ik}\}\text{Im}\{W_{ik}\}$. At iteration t, we update $\text{Re}\{W_{ik}[t]\}$ and $\text{Im}\{W_{ik}[t]\}$ by

$$\text{Re}\{W_{ik}[t+1]\} = \text{Re}\{W_{ik}[t]\} - \alpha[t]\left(\sum_{l|i,k\in C_l}(-\lambda^{\text{Re}}_{ik,l}) + 2\text{Re}\{M^H_{ik}\}\right), \tag{12}$$

and

$$\text{Im}\{W_{ik}[t+1]\} = \text{Im}\{W_{ik}[t]\} - \alpha[t]\left(\sum_{l|i,k\in C_l}(-\lambda^{\text{Re}}_{ik,l}) - 2\text{Im}\{M^H_{ik}\}\right). \tag{13}$$

If W_{ik} is for a fill-in edge, we also make the update by (12) and (13) with $M_{ik} = 0$.

We can interpret the updating mechanism as follows: certain maximal cliques share a component W_{ik} (if $i = k$, it corresponds to a node; otherwise, it corresponds to an edge or a fill-in edge). W_{ik} represents electricity resources and $-M^H_{ik}$ is its default price. An agent (i.e. a node responsible for computing the update) which is common to all those maximal cliques sharing the resource determines how much resource should be allocated to each maximal clique. In other words, it fixes W_{ik} and every party gets this amount. $X_{ik,l}$ is the actual resource required by C_l and $\lambda_{ik,l}$ corresponds to the price of the resources when W_{ik} is allocated to it. If C_l requires more resource than those allocated, i.e., $X_{ik,l} > W_{ik}$, then $\lambda_{ik,l} > 0$. If the net price, i.e. $\sum_{l|i,k\in C_l}\lambda_{ik,l} - M_{ik}$, is positive, the agent should increase the amount of resource allocating to the maximal cliques because it can earn more. If the net price is negative, then supply is larger than demand and it should reduce the amount of allocated resources.

From (11)–(13), all shared W_{ik} can be updated independently. The update of each W_{ik} only involves those $\{C_l | C_l \in \mathcal{C}, i, k \in C_l\}$. In other words, (10) can be further computed separately according to those maximal cliques shared by each W_{ik}. The pseudocode of the primal algorithm is as Algorithm 1.

Algorithm 1 Primal Algorithm

Given $Q, \overline{V}, \underline{V}, \mathcal{C}$
1. Construct (9) for each maximal clique
2. **while** stopping criteria not matched **do**
   3. **for** each subproblem l (in parallel) **do**
      4. Given W_{ik} with $(i,k) \notin \mathcal{U}$, solve (9)
      5. Return $\lambda_{ik,l} \forall i,k | (i,k) \notin \mathcal{U}$
   6. **end for**
   7.Given $\lambda_{ik,l} \forall l | i,k \in C_l$, update the shared W_{ik} with (11)–(13) (in parallel)
8. **end while**

B. Dual Algorithm

Let $\Omega_{ik} = \{C_l | i,k \in C_l, \forall l\}$. Eq. (5) can be written as

$$\text{Tr}(\mathbf{MW}) = \sum_{C_l\in\mathcal{C}}\left(\sum_{i,k\in C_l|(i,k)\in\mathcal{U}} M^H_{ik}W_{ik} + \sum_{i,k\in C_l|(i,k)\notin\mathcal{U}}\frac{M^H_{ik}W_{ik}}{|\Omega_{ik}|}\right). \tag{14}$$

Similar to the primal algorithm, we can replace $\mathbf{W}_{C_l}$ with $\tilde{\mathbf{W}}_{C_l}$. For each $W_{ik}|(i,k) \notin \mathcal{U}$, let $X_{ik,l}$ be a copy of W_{ik} in $C_l \in \Omega_{ik}$. To make all $\tilde{\mathbf{W}}_{C_l}$ consistent, we should have

$$X_{ik,l_1} = X_{ik,l_2} = \cdots = X_{ik,l_{|\Omega_{ik}|}}, \forall l_r | C_{l_r} \in \Omega_{ik}, \quad \forall i,k | (i,k) \notin \mathcal{U}. \tag{15}$$

For each $(i,k) \notin \mathcal{U}$, (15) can be written into $|\Omega_{ik}| - 1$ equalities, e.g.,

$$X_{ik,l_1} = X_{ik,l_2}, X_{ik,l_2} = X_{ik,l_3}, \ldots, X_{ik,l_{|\Omega_{ik}|-1}} = X_{ik,l_{|\Omega_{ik}|}}. \tag{16}$$

As shown later, the update mechanism of the dual algorithm depends only on how we arrange (15) into equalities. In fact, there are many ways to express the $|\Omega_{ik}| - 1$ equalities provided that each $X_{ik,l}$ appears in at least one of the equalities. Suppose the rth equality be $\tilde{X}_{ik,r,(1)} = \tilde{X}_{ik,r,(2)}$. We assign a Lagrangian multiplier $\upsilon_{ik,r}$ to it. Then we have

$$\begin{aligned}
\upsilon_{ik,1}(\tilde{X}_{ik,1,(1)} - \tilde{X}_{ik,1,(2)}) &= 0 \\
\upsilon_{ik,2}(\tilde{X}_{ik,2,(1)} - \tilde{X}_{ik,2,(2)}) &= 0 \\
&\vdots \\
\upsilon_{ik,|\Omega_{ik}|-1}(\tilde{X}_{ik,|\Omega_{ik}|-1,(1)} - \tilde{X}_{ik,|\Omega_{ik}|-1,(2)}) &= 0
\end{aligned} \tag{17}$$

When we sum all these equalities up, each $X_{ik,l}$ will be associated with an aggregate Lagrangian multiplier $\tilde{\upsilon}_{ik,l}$, which is composed of all υ_{ik} associated with $X_{ik,l}$. For example, in (16), we have $\tilde{X}_{ik,1,(1)} = X_{ik,l_1}$, $\tilde{X}_{ik,1,(2)} = X_{ik,l_2}$, and $\tilde{X}_{ik,2,(1)} = X_{ik,l_2}$. Thus $\tilde{\upsilon}_{ik,l_1} = \upsilon_{ik,1}$ and $\tilde{\upsilon}_{ik,l_2} = \upsilon_{ik,2} - \upsilon_{ik,1}$.

In (17), $\forall 1 \leq i,k \leq n$, the corresponding rth equality for the (i,k) pair implies the one for the (k,i) pair, i.e.,

$$\tilde{X}_{ik,r,(1)} = \tilde{X}_{ik,r,(2)} \Rightarrow \tilde{X}_{ki,r,(1)} = \tilde{X}_{ki,r,(2)},$$

due the positive semidefinite property of $\mathbf{W}$ given in (3c). We have

$$\begin{aligned}
&\upsilon_{ik,r}\tilde{X}_{ik,r,(1)} = \upsilon_{ik,r}\tilde{X}_{ik,r,(2)} \\
\Rightarrow\ &\upsilon^H_{ik,r}\tilde{X}_{ki,r,(1)} = \upsilon^H_{ik,r}\tilde{X}_{ki,r,(2)}.
\end{aligned}$$

Thus the aggregate Lagrangian multiplier for $X_{ki,l}$ can be computed directly from that for $X_{ik,l}$, i.e., $\tilde{\upsilon}_{ki,l} = \tilde{\upsilon}^H_{ik,l}$.

Let $\tilde{\upsilon} = (\tilde{\upsilon}_{ik,l_r}, i,k \in C_l | (i,k) \notin \mathcal{U}, \forall C_l \in \mathcal{C}; l_r | C_{l_r} \in \Omega_{ik})$. We form function $d(\tilde{\upsilon}, \mathbf{W})$ by aggregating (15) into (14). We have

$$\begin{aligned}
&d(\tilde{\upsilon}, \mathbf{W}) \\
&= \sum_{C_l\in\mathcal{C}}\left(\sum_{\substack{i,k\in C_l \\ |(i,k)\in\mathcal{U}}} M^H_{ik}W_{ik} + \sum_{\substack{i,k\in C_l \\ |(i,k)\notin\mathcal{U}}}\left(\frac{M^H_{ik}W_{ik}}{|\Omega_{ik}|} + \tilde{\upsilon}_{ik,l}X_{ik,l}\right)\right) \\
&\triangleq \sum_{C_l\in\mathcal{C}} d(\tilde{\upsilon}, \tilde{\mathbf{W}}_{C_l})
\end{aligned} \tag{18}$$

Given $\tilde{v}$, the problem becomes

$$\text{minimize} \sum_{C_l \in \mathcal{C}} d(\tilde{v}, \tilde{\mathbf{W}}_{C_l}) \tag{19a}$$

$$\text{subject to } \underline{V}_i^2 \le W_{ii} \le \overline{V}_i^2, \forall i \in C_l, \forall C_l \in \mathcal{C} \tag{19b}$$

$$\tilde{\mathbf{W}}_{C_l} \succeq 0, \forall C_l \in \mathcal{C}, \tag{19c}$$

which can be divided into subproblems according to the maximal cliques and each of them is independent of each other. Subproblem l is stated as:

$$\text{minimize} \sum_{\substack{i,k \in C_l \\ |(i,k) \in \mathcal{U}}} M_{ik}^H W_{ik} + \sum_{\substack{i,k \in C_l \\ |(i,k) \notin \mathcal{U}}} \left(\frac{M_{ik}^H}{|\Omega_{ik}|} + \tilde{\upsilon}_{ik,l} \right) X_{ik,l} \tag{20a}$$

$$\text{subject to } \underline{V}_i^2 \le W_{ii} \le \overline{V}_i^2, \forall i \in C_l | (i,i) \in \mathcal{U}, \tag{20b}$$

$$\underline{V}_i^2 \le X_{ii,l} \le \overline{V}_i^2, \forall i \in C_l | (i,i) \notin \mathcal{U}, \tag{20c}$$

$$\tilde{\mathbf{W}}_{C_l} \succeq 0. \tag{20d}$$

By solving (20), we denote the optimal $\tilde{X}_{ik,r}(z)$ for the rth equality in (17) by $\tilde{X}_{ik,r}^{opt}(z)$, where $z \in \{1,2\}$. The gradient of $-d(\tilde{v}, \tilde{\mathbf{W}}_{C_l})$[1] with respect to $\upsilon_{ik,r}$ is

$$-\frac{\partial d}{\partial \upsilon_{ik,r}} = \tilde{X}_{ik,r,(2)}^{opt} - \tilde{X}_{ik,r,(1)}^{opt}.$$

Let $\upsilon_{ik,r}[t]$, $\alpha[t] > 0$, and $\tilde{X}_{ik,l,(z)}[t]$ be the Lagrangian multiplier of the rth equality associated with W_{ik}, the step size, $\tilde{X}_{ik,l,(z)}^{opt}$, respectively, at time t. Then we can update $\upsilon_{ik,r}$ in (17) by

$$\upsilon_{ik,r}[t+1] = \upsilon_{ik,r}[t] - \alpha[t] \left(\tilde{X}_{ik,r,(2)}[t] - \tilde{X}_{ik,r,(1)}[t] \right). \tag{21}$$

If $\tilde{X}_{ik,r,(2)}[t] < \tilde{X}_{ik,r,(1)}[t]$, then $\upsilon_{ik,r}[t+1] > \upsilon_{ik,r}[t]$. This will make the coefficient corresponding to $\tilde{X}_{ik,r,(1)}$ larger while making that corresponding to $\tilde{X}_{ik,r,(2)}$ smaller. At time $t+1$, the subproblem will obtain $\tilde{X}_{ik,r,(1)}[t+1] < \tilde{X}_{ik,r,(1)}[t]$ and $\tilde{X}_{ik,r,(2)}[t+1] > \tilde{X}_{ik,r,(2)}[t]$. Hence, $|\tilde{X}_{ik,r,(2)}[t+1] - \tilde{X}_{ik,r,(1)}[t+1]| < |\tilde{X}_{ik,r,(2)}[t] - \tilde{X}_{ik,r,(1)}[t]|$. On the other hand, if $\tilde{X}_{ik,r,(2)}[t] > \tilde{X}_{ik,r,(1)}[t]$, then $\upsilon_{ik,r}[t+1] < \upsilon_{ik,r}[t]$. This will make the coefficient corresponding to $\tilde{X}_{ik,r,(1)}$ smaller while making that corresponding to $\tilde{X}_{ik,r,(2)}$ larger. Then we will get $\tilde{X}_{ik,r,(1)}[t+1] > \tilde{X}_{ik,r,(1)}[t]$ and $\tilde{X}_{ik,r,(2)}[t+1] < \tilde{X}_{ik,r,(2)}[t]$. This will also make $|\tilde{X}_{ik,r,(2)}[t+1] - \tilde{X}_{ik,r,(1)}[t+1]| < |\tilde{X}_{ik,r,(2)}[t] - \tilde{X}_{ik,r,(1)}[t]|$. Therefore, (21) drives $X_{ik,l}$'s in (15) become closer to each other in value when the algorithm evolves. In other words, (21) tries to make equality (15) hold when the algorithm converges.

At any time before the algorithm converges, i.e., (15) does not hold, the solution $\mathbf{W}$ with the computed $X_{ik,l}, \forall i,k | (i,k) \notin \mathcal{U}, \forall l | C_l \in \Omega_{ik}$ is an infeasible solution. We can always construct a feasible $\hat{\mathbf{W}}$ with W_{ik} which is the average of all $X_{ik,l}$ in (15). This allows us to determine a good value for W_{ik} from the $X_{ik,l}$.

The purpose of (21) is to make the two entity $\tilde{X}_{ik,r,(1)}[t]$ and $\tilde{X}_{ik,r,(2)}[t]$ closer to each other. As long as $\tilde{X}_{ik,r,(1)}[t]$ and $\tilde{X}_{ik,r,(2)}[t]$ have been computed (from two subproblems), we can update $\upsilon_{ik,r}$ with (21). Thus different υ_{ik} can be updated asynchronously. Since the only co-ordination between subproblems is through (21), synchronization is not required in dual algorithm.

We can interpret the updating mechanism as follows: $\upsilon_{ik,r}$ is the price assigned to equality $\tilde{X}_{ik,r,(1)} = \tilde{X}_{ik,r,(2)}$. We can treat $\tilde{X}_{ik,r,(1)}$ and $\tilde{X}_{ik,r,(2)}$ as demand and supply of electricity resources, respectively. If the demand is larger than the supply, i.e., $\tilde{X}_{ik,r,(1)} > \tilde{X}_{ik,r,(2)}$, we should increase the price so as to suppress the demand and to equalize the supply and demand. On the other hand, if the supply is larger than the demand, i.e. $\tilde{X}_{ik,r,(1)} < \tilde{X}_{ik,r,(2)}$, we should reduce the price in order to boost the demand. The pseudocode of the dual algorithm is as Algorithm 2.

Algorithm 2 Dual Algorithm

Given $Q, \overline{V}, \underline{V}, \mathcal{C}$
1. Pair up slack variables for the shared variables into equalities
2. Construct (20) for each maximal clique
3. **while** stopping criteria not matched **do**
   4. **for** each subproblem l (in parallel) **do**
      5. Given $\tilde{v}_{ik}$, solve (20)
      6. Return $X_{ik,l}, \forall i,k | (i,k) \notin \mathcal{U}$
   7. **end for**
   8.Given X_{ik,l_r}, update the price υ_{ik,l_r} with (21) (in parallel and asynchronously)
9. **end while**

C. Computation of Voltage

When either the primal or the dual algorithm converges, assuming zero duality gap, we obtain the optimal $\mathbf{W} = \mathbf{v}\mathbf{v}^H$. To obtain each bus voltage and voltage flown on each line, we first compute the voltage magnitude at each bus, $|V_i| = \sqrt{W_{ii}}, 1 \le i \le n$. For $1 \le i,k \le n$, if there is a line between nodes i and k, the corresponding line voltage angle difference θ_{ik} can be found by solving $W_{ik} = |V_i||V_k|e^{j\theta_{ik}}$ at either bus i or k. For the former, bus k needs to send $|V_k|$ to bus i, and vice versa. By fixing the voltage angle of a particular bus to zero, the voltages of the whole network can be found subsequently.

IV. Quadratic Cost Function

Up to now we have focused on the OPF problem with a linear objective function. In practice, sometimes a quadratic cost function is used. If this is the case, the methods developed so far can be used as subroutines to solve the OPF problem by adding a outer loop to the iteration.

Let $\text{cost}_i(P_i) = c_{i2}P_i^2 + c_{i1}P_i$ be the cost function associated with P_i. We assume this function is convex for all buses, that is, $c_{i2} > 0 \ \forall i$. From (1e), $P_i = \text{Tr}(\mathbf{A_i}\mathbf{v}\mathbf{v}^\mathbf{H})$, where $\mathbf{A_i} = \frac{1}{2}((Y^H E_i) + \mathbf{E_i}\mathbf{Y})$ and $\mathbf{E_i}$ is the matrix with 1

[1]In the dual form, we maximize $\inf_{\tilde{\mathbf{W}}} d(\tilde{v}, \tilde{\mathbf{W}})$ over $\tilde{v}$. In minimization, we consider $-d(\tilde{v}, \tilde{\mathbf{W}}_{C_l})$.

in the (i,i)th entry and zero everywhere else. Now the OPF problem is (compare with (2))

$$\text{minimize} \quad \sum_{i=1}^{n} c_{i2}\text{Tr}(A_i W)^2 + c_{i1}\text{Tr}(A_i W) \tag{22a}$$

$$\text{subject to} \quad \underline{V_i}^2 \le W_{ii} \le \overline{V_i}^2, \forall i \tag{22b}$$

$$\text{rank}(W) = 1 \tag{22c}$$

$$\mathbf{p} = \text{Re}\{\text{diag}(\mathbf{v}\mathbf{v}^H \mathbf{Y}^H)\}. \tag{22d}$$

Using Schur's complement, we may write (22) equivalently as

$$\text{minimize} \quad \sum_{i=1}^{n}(t_i + c_{i1}\text{Tr}(A_i W) \tag{23a}$$

$$\text{subject to} \quad \begin{bmatrix} t_i & \sqrt{c_{i2}}\text{Tr}(\mathbf{A_i W}) \\ \sqrt{c_{i2}}\text{Tr}(\mathbf{A_i W}) & 1 \end{bmatrix} \succeq 0, \forall i \tag{23b}$$

$$\underline{V_i}^2 \le W_{ii} \le \overline{V_i}^2, \forall i$$

$$\text{rank}(W) = 1.$$

By relaxing the the rank-1 constraint and taking the dual, we get

$$\text{maximize} \quad \sum_{i=1}^{n}(-\overline{\lambda}_i \overline{V}_i^2 + \underline{\lambda}_i \underline{V}_i^2 - u_i) \tag{24a}$$

$$\text{subject to} \quad \sum_{i=1}^{n}(c_{i1}A_i - 2\sqrt{c_{i2}}z_i A_i) + \Lambda \succcurlyeq 0 \tag{24b}$$

$$\begin{bmatrix} 1 & z_i \\ z_i & u_i \end{bmatrix} \succcurlyeq 0 \ \forall i, \tag{24c}$$

where Constraint (24c) corresponds to the Schur's compliment constraint in (23). Constraint (24c) can be rewritten as $u_i \ge z_i^2$, for a given z_i, the maximizing u_i is z_i^2. Therefore the we may drop Constraints (24c) and replace the u_i in the objective function by z_i^2. If we fix the z_i's, then (24) becomes a function of z_i's

$$J(\mathbf{z}) = \text{maximize} \quad \sum_{i=1}^{n}(-\overline{\lambda}_i \overline{V}_i^2 + \underline{\lambda}_i \underline{V}_i^2 - z_i^2) \tag{25}$$

$$\text{subject to} \quad \sum_{i=1}^{n}(c_{i1}A_i - 2\sqrt{c_{i2}}z_i A_i) + \Lambda \succcurlyeq 0.$$

For fixed $\mathbf{z}$, (25) is in the form of (4). Therefore, $J(\mathbf{z})$ is a dual of the optimization problem with linear cost functions with costs $(c_{11}-2\sqrt{c_{12}}z_1, c_{21}-2\sqrt{c_{22}}z_2, \ldots, c_{n1}-2\sqrt{c_{n2}}z_n)$. To find the optimal solution of (25) we may use any of the algorithm in the previous sections. Let $W^*(\mathbf{z})$ denote the optimal solution to $J(\mathbf{z})$. To find the optimal $\mathbf{z}$, we use a gradient algorithm. The Lagrangian of (25) is

$$\mathcal{L}(\lambda, W) = \sum_{i=1}^{n}(-\overline{\lambda}_i \overline{V}_i^2 + \underline{\lambda}_i \underline{V}_i^2 - z_i^2) + \text{Tr}((\sum_{i=1}^{n}(c_{i1}A_i - 2\sqrt{c_{i2}}z_i A_i) + \Lambda)W). \tag{26}$$

TABLE I

NORMALIZED CPU TIME FOR DISTRIBUTION TEST FEEDERS[a]

Number of buses	Centralized	Cumulative		Distributed	
		Primal	Dual	Primal	Dual
8	1.85	7.21	5.62	1.52	1.00
34	298.68	37.79	33.89	1.94	1.70
123	–[b]	143.39	126.48	2.24	1.64

[a] The CPU times are normalized by 0.0857s.

[b] The solver cannot be applied because of the out-of-memory problem.

TABLE II

NORMALIZED CPU TIME FOR IEEE POWER TRANSMISSION SYSTEM BENCHMARKS[a]

Number of buses	Centralized	Cumulative dual	Distributed dual	iterations	Initial step size
14	5.38	5.38	1.00	1	30
30	45.29	58.60	5.38	6	30
57	1696.79	49.08	4.28	4	30
118	–[b]	704.46	13.51	9	300

[a] The simulations for this problem set are done on MacBook Pro with 2.4 GHz Intel core i5 and 4 GB RAM. The CPU times are normalized by 0.1410s.

[b] The solver cannot be applied because of the out-of-memory problem.

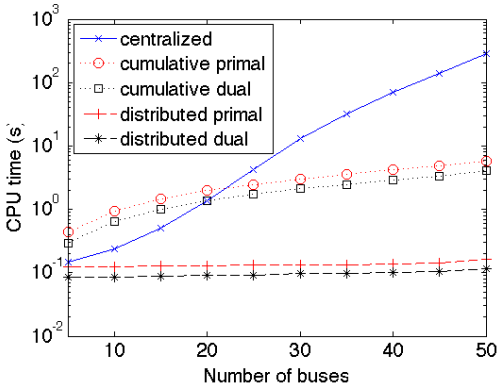


Fig. 1. CPU time of the various approaches on radial networks with bounde

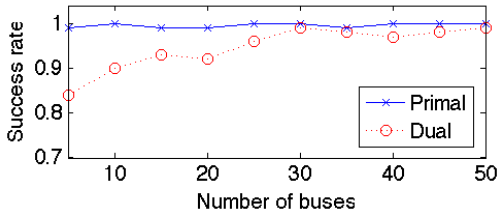


Fig. 2. Success rates of the primal and dual algorithms on radial networks with bounded voltages

By a standard result in convex programming, the gradient of $J(z)$ is given by $\frac{J(\mathbf{z})}{z_i} = \frac{\partial \mathcal{L}(\lambda^*, W^*)}{\partial z_i} = -2\text{Tr}(\sqrt{c_{i2}}A_i W^*) - 2z_i$, where (λ^*, W^*) is a pair of optimal dual-primal solutions (dependent on $\mathbf{z}$). Therefore, to solve the problem with quadratic cost functions, we add an additional outer loop to the solution algorithms for the linear cost functions.

V. SIMULATION RESULTS

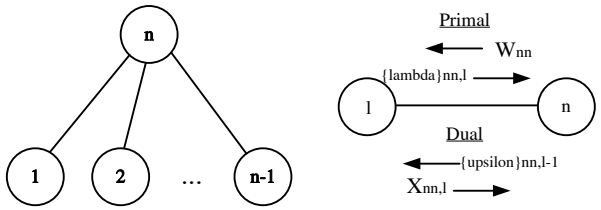
Recall that the primal or dual algorithm aims to divide the original problem into smaller ones and to coordinate the subproblems, which of each can be solved by any SDP solver

independently. When compared with those done by the SDP solver, the computation and ordination required solely by our algorithms are relatively far less stringent. As a whole, the bottleneck of computation should be at the SDP solver. In our simulation, we program the primal and dual algorithms in MATLAB and and solve each SDP with YALMIP [20] and SeDuMi [21]. To get rid of the dependence on the programming language and to simplify the comparison, we only count the CPU time spent on the SDP solver. Moreover, we can arrange the subproblems to be solved in a single node or distribute them to different nodes in the network. For the former, we assume the problems are handled sequentially and we call it the cumulative approach. The latter, named as the distributed approach, addresses the subproblems in parallel. Without our algorithms, the (original) problem will be solved in its original form in a centralized manner. Here we compare the CPU times required for the SDP(s) among the centralized approach, (primal and dual) cumulative approaches, and (primal and dual) distributed approaches.

We run the simulations on Dell PowerEdge 2650 with 2 $\times$ 3.06 GHz Xeon and 6 GB RAM (except those for the transmission system benchmarks in Table II).[2] In order to monitor the performance in each simulation run, we assemble the partial solutions (done by the subproblems) to form a complete one for the original problem and evaluate the corresponding objective value.[3] Our algorithms stop when the computed objective value falls in the range of $10^{-2}\times$ the global minimum. We assume that the dual algorithm is synchronized. In other words, all subproblems for the dual are solved in each iteration (but this is not required when implemented in real systems). The initial step size $\alpha[0]$ is set to one and it is updated by $\alpha[t] = \alpha[t-1]/t, \forall t > 0$. An algorithm is deemed successful if the stopping criterion is met in 100 iterations.

We perform simulations on three problem sets; the first two focuses on tree-like networks while the last one is about transmission networks. The first problem set is some distribution test feeder benchmarks [22]. As the data set does not specify the cost function of power production/consumption, we create a problem instance by randomly generating the costs. To do this, we first select one node, e.g. node i, to be the power source node with c_{i1} set randomly in the range $(0, 10)$. For other node $k \neq i$, c_{k1} set randomly in the range $(-10, 0)$. We create 100 instances for each network. Table I shows the averages of the normalized CPU times of the various approaches. All algorithms converge in 100 iterations for all instances.

The second problem set is the n-bus radial network of height equal to one.For each instance, the root is the power source with a random cost selected in $(0, 10)$ and each of the rest takes a random cost in $(-10, 0)$. For each node i, we specify a number ξ in $(0.9, 1.1)$ and set $\underline{V}_i = 0.95\xi$ and $\overline{V}_i = 1.05\xi$. For each line, the magnitudes of the conductance and susceptance are randomly assigned in $(0, 10)$. We produce 100 instances for each n and plot the average CPU times in logarithmic scale in Fig. 1.



(a) Toplogy (b) Communications between nodes l and n

Fig. 3. n-bus radial network

From the simulation results for these two problem sets, both the primal and dual algorithms converge very fast for distribution networks. The CPU time for the centralized approach grows very fast with the size of the network. The CPU time grows roughly linearly for the cumulative approach while it becomes independent of the size for the distributed approach. We define success rate as the fraction of the total number of simulation runs with stopping criterion met in 100 iterations, shown in Fig. 2. The success rate of the primal is almost 100% for all tested network sizes. The dual fails to converge in 100 iterations for a small fraction of small networks but the success rate grows to almost 100% with the network size. In general,the primal and dual algorithms are similar in performance but the dual requires a little bit less CPU time than the primal on the average.

The third problem set is some IEEE power transmission system benchmarks [23]. As pointed out in [3], these test cases have zero duality zap although they have network structures different from what we mention in Section II-B. Table II shows the CPU times required, the iterations for convergence, and the initial step sizes. The primal algorithm is not applied to this problem set and the reason will be given in the next section. For these transmission network topologies, the maximal cliques of the fill-in graphs are much irregular than those with tree-structured networks. There are many ways to construct the maximal cliques and different construction can result in different convergence speed. The study of the relationship between maximal clique construction and the algorithm performance is out of the scope of this paper. In this simulation, we randomly choose one maximal clique configuration and the step sizes are adjusted individually so as to have fast convergence. Nevertheless, the dual algorithm is more desirable than the centralized approach.

VI. Discussion

The primal algorithm requires less information sharing between buses than the dual and it favors situations with sensitive bus information. Consider an example given in Fig. 3(a) and its communication pattern is shown in Fig. 3(b). W_{nn} for bus n is common to all the subproblems. Bus n computes W_{nn} with its own $\underline{V}_n$, $\overline{V}_n$, and M_{nn}. Only the computed W_{nn} is required to transmit to other buses which do not require bus n's information. For the dual algorithm, for $1 \leq i \leq n-1$, node i requires $\underline{V}_n$, $\overline{V}_n$, and M_{nn} to

[2]The results in Tables I and II are normalized, and thus, they are comparable.

[3]The assembly of partial solutions is not required in real implementation.

solve its subproblem. The situation is similar if the common solution is for a link.

Our algorithms do not require an overlay of communication networks with different topology. All the communication is one-hop. Communication links need to be built along with existing transmission lines only.

The primal algorithm is suitable for networks with tree structure while the dual can also handle those with other network topolgoies. In fact, the primal is not very efficient to update the partial solutions for (fill-in) edges, i.e. (12)–(13), especially when their values are closed to the boundary of the feasible region. The dual algorithm does not have this problem when updating v with (21). As mentioned, we can always constrain a feasible solution by averaging the shared variables.

The dual algorithm is more resistant to communication delay than the primal. For the primal, an update of a shared variable requires λ from all involved subproblems and thus synchronization is required. Delay of computing or transmitting an λ from any subproblem can affect the whole algorithm proceed. On the other hand, an update of an v requires the $\tilde{X}$ from two pre-associated subproblems according to the arrangement of the inequalities in 15. Thus the dual algorithm is asynchronous.[4] Moreover, we can pair the variables in (15) into equalities differently and secretly whenever we start the dual algorithm. In some sense, the dual algorithm is more robust to attack stemmed from communication on the communication links.

VII. Conclusion

OPF is very important in planning the schedule of power generation. In the smart grid paradigm, more renewable energy sources will be incorporated into the system, especially in distribution networks, and the problem size will also grow tremendously. As problems with some special structures (e.g. trees for distribution networks) have a zero duality gap, we can find the optimal solution by solving the convex dual problem. In this paper, we propose the primal and dual algorithms (with respect to the primal and dual decomposition techniques) to speed up the computation of the convexified OPF problem. The problem is decomposed into smaller subproblems, each of which can be solved independently and effectively. The primal algorithm coordinates the subproblems by controlling the shared terms (related to electricity resources) while the dual one manages them by updating the prices. From the simulation results for tree-structure problems, the computation time grows linearly with the problem size if we solve the decomposed problem in a central node with our algorithms. The computation time becomes independent of the problem size when the subproblems are solved in parallel in different nodes. Even without nice network structure such as a tree, the dual algorithm outperforms the centralized approach without decomposition. Therefore, the primal and dual algorithms are excellent in addressing OPF, especially for distribution networks. In future, we will improve the algorithm by incorporate more constraints into the OPF problem and move to nonlinear objective functions.

[4]We adopt a similar approach in solving a voltage regulation problem in [24]. In [24], we verify by simulation that non-synchronousness does not prevent the algorithm from converging.

References

[1] P. P. Varaiya, F. F. Wu, and J. W. Bialek, "Smart operation of smart grid: Risk-limiting dispatch," *Proc. IEEE*, vol. 99, pp. 40–57, 2011.

[2] B. Stott, J. Jardim, and O. Alsac, "Dc power flow revisited," *IEEE Trans. Power Syst.*, vol. 24, pp. 1290–1300, 2009.

[3] J. Lavaei and S. H. Low, "Zero duality gap in optimal power flow problem," *IEEE Trans. Power Syst.*, vol. 27, no. 1, pp. 92–107, 2012.

[4] B. Zhang and D. Tse, "Geometry of feasible injection region of power networks," in *Proc. 49th Annual Allerton Conf. on Comm., Control, and Comput.*, 2011, pp. 1508–1515.

[5] S. Sojoudi and J. Lavaei, "Physics of power networks makes hard optimization problems easy to solve," in *Proc. IEEE Power & Energy Society General Meeting*, 2012.

[6] R. A. Jabr, "Radial distribution load flow using conic programming," *IEEE Trans. Power Syst.*, vol. 21, pp. 1458–1459, 2006.

[7] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *Journal of the Operational Research Society*, vol. 49, pp. 237–252, 1998.

[8] D. P. Bertsekas, *Nonlinear programming*, 2nd ed. Belmont, MA, USA: Athena Scientific, 1999.

[9] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: A mathematical theory of network architectures," *Proc. IEEE*, vol. 95, pp. 255–312, Jan. 2007.

[10] S. Wright, *Primal-dual interior-point methods*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1997.

[11] R. E. Tarjan and M. Yannakakis, *Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, July 1984, vol. 13.

[12] D. R. Fulkerson and O. A. Gross, "Incidence matrices and interval graph," *Pacific J. Math.*, vol. 15, no. 3, pp. 835–855, 1965.

[13] R. E. Neapolitan, *Probabilistic reasoning in expert systems: theory and algorithms*. New York, NY, USA: John Wiley & Sons, Inc., 1990.

[14] E. A. Akkoyunlu, "The enumeration of maximal cliques of large graphs," *SIAM Journal on Computing*, vol. 2, pp. 1–6, 1973.

[15] K. Nakata, M. Yamashita, K. Fujisawa, and M. Kojima, "A parallel primal-dual interior-point method for semidefinite programs using positive definite matrix completion," *Parallel Comput.*, vol. 32, pp. 24–43, Jan. 2006.

[16] M. Fukuda, M. Kojima, K. Murota, and K. Nakata, "Exploiting sparsity in semidefinite programming via matrix completion I: General framework," *SIAM J. on Optimization*, vol. 11, pp. 647–674, March 2000.

[17] K. Nakata, K. Fujisawa, M. Fukuda, M. Kojima, and K. Murota, "Exploiting sparsity in semidefinite programming via matrix completion II: implementation and numerical results," *Mathematical Programming*, vol. 95, pp. 303–327, 2003.

[18] R. Grone, C. R. Johnson, E. M. Sa, and H. Wolkowicz, "Positive definite completions of partial hermitian matrices," *Linear Algebra and Its Applications*, vol. 58, pp. 109–124, 1984.

[19] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

[20] J. Löfberg, "YALMIP : A toolbox for modeling and optimization in MATLAB," in *Proc. International Symposium on Computer Aided Control Systems Design*, Sep. 2004, pp. 284–289.

[21] J. F. Sturm, "Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones," 1998.

[22] W. H. Kersting, "Radial distribution test feeders," in *Proc. IEEE Power Engineering Society Winter Meeting*, vol. 2, 2001, pp. 908–912.

[23] University of washington, power systems test case archive. [Online]. Available: http://www.ee.washington.edu/research/pstca

[24] A. Y. S. Lam, B. Zhang, A. Dominguez-Garcia, and D. Tse, "Optimal distributed voltage regulation in power distribution networks," submitted for publication.