Distributed Bisection Method for Economic Power Dispatch in Smart Grid

Hao Xing, *Student Member, IEEE*, Yuting Mou, Minyue Fu, *Fellow, IEEE*, and Zhiyun Lin, *Senior Member, IEEE*

***Abstract*—In this paper, we present a fully distributed bisection algorithm for the economic dispatch problem (EDP) in a smart grid scenario, with the goal to minimize the aggregated cost of a network of generators, which cooperatively furnish a given amount of power within their individual capacity constraints. Our distributed algorithm adopts the method of *bisection*, and is based on a consensus-like iterative method, with no need for a central decision maker or a leader node. Under strong connectivity conditions and allowance for local communications, we show that the iterative solution converges to the globally optimal solution. Furthermore, two stopping criteria are presented for the practical implementation of the proposed algorithm, for which *sign consensus* is defined. Finally, numerical simulations based on the IEEE 14-bus and 118-bus systems are given to illustrate the performance of the algorithm.**

***Index Terms*—Consensus, distributed algorithm, distributed convex optimization, economic power dispatch, smart grid.**

I. Introduction

THE economic dispatch problem (EDP) has been actively studied in the electric power industry for optimal operation and planning of energy resources. This problem is usually formulated as an optimization problem [1]. The classic EDP is mainly concerned with the economic dispatch of fossil-fired power generation systems to achieve minimum operational costs within capacity limits. In this scenario, the operation and planning for power generation systems can be done by one or several central decision makers. Many types of cost functions are available. A convex and piecewise linear cost function is used in [2], but a quadratic cost function is usually preferred [1]. Many centralized solutions have been proposed to solve EDP. In [1], the conventional Lagrangian relaxation approach and first order gradient method are given. In [3], a strategy based on a direct search method with multi-level convergence is proposed to solve EDP with transmission capacity constraints. In [4], an algorithm based on evolutionary programming (EP), tabu search (TS), and quadratic programming (QP) methods is proposed to solve the non-convex economic dispatch problem (NEDP). A parallel micro genetic algorithm (PMGA) is employed in [5] to solve ramp-rate constrained EDP with non-monotonically and monotonically increasing incremental cost functions.

Distributed algorithms for control, estimation, and optimization have been intensively investigated for large-scale systems [6]. Spatially distributed large-scale systems interconnected by a communication network are ubiquitous in the real world, where the traditional centralized control algorithms are inefficient. A smart grid with distributed renewable power generation is a typical such large-scale system. A lot of work has been done about distributed optimization (e.g., [7]–[9]) using distributed gradient method, distributed sub-gradient method, alternating direction method of multipliers (ADMM), and so on. In general, compared with centralized algorithms, distributed algorithms have many advantages, including enhanced robustness, reduction in communication between agents, and uniform power consumption for each agent.

To meet environmental targets, to accommodate a greater emphasis on demand response (DR) [10], and to support plug-in hybrid electric vehicles (PHEVs) [11], [12], distributed generation, and storage capabilities, traditional power grids need to become "smart grids". This is an area being heavily studied in recent years [13]. In a smart grid integrating distributed generation, renewable power sources, and a communication network, it is desirable to solve the EDP in a distributed fashion. In fact, a lot of such work has been done so far. In [14] and [15], the authors propose a consensus based algorithm to realize decentralized economic dispatch, where a master node aware of the total power demand is required to ensure the equality between the total power supply and demand. In [16], the authors present a ratio consensus based decentralized algorithm to find the optimal incremental cost, under the assumption that each node (i.e., generator) knows the parameters of all the nodes. In [17], an algorithm based on a consensus and innovation framework is proposed, where the consensus term makes all the nodes agree with each other to realize their common goal of estimating the global price index, while the innovation term makes all the nodes estimate the index according to the local knowledge of loads. In [18], the authors propose an algorithm for EDP with a quadratic cost function, which can be treated as a distributed

Manuscript received February 28, 2014; revised July 22, 2014, October 09, 2014, and November 19, 2014; accepted November 25, 2014. Date of publication December 18, 2014; date of current version August 03, 2015. This work was supported in part by the National Natural Science Foundation of China under Grant 61273113 and in part by Zhejiang Provincial Natural Science Foundation of China LR13F030002. Paper no. TPWRS-00303-2014.

H. Xing and Y. Mou are with the State Key Laboratory of Industrial Control Technology and the Department of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: haoxing@zju.edu.cn; ytmou@zju.edu.cn).

M. Fu is with the School of Electrical Engineering and Computer Science, University of Newcastle, NSW 2308 Australia, and also with the State Key Laboratory of Industrial Control Technology and the Department of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: minyue.fu@newcastle.edu.au).

Z. Lin is with the State Key Laboratory of Industrial Control Technology and the College of Electrical Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: linz@zju.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TPWRS.2014.2376935

implementation of standard Lambda-Iteration method, without requiring other nodes' parameters.

In this paper, a *distributed bisection algorithm* based on a consensus-like iterative method is proposed to solve the EDP. Compared with the existing work, our algorithm has the following features.

1) Our algorithm requires no global information of the system. In [18], a distributed algorithm is proposed, but global parameters including network topology and generators' parameters are needed to design an appropriate learning gain to guarantee convergence. In [16], each node needs to know some parameters of all other nodes, which implies that the computation and communication package size grow at least linearly with the network size, while in our algorithm each node only needs to know its local parameters.
2) No master or leader node aware of the total power demand is needed in our algorithm, whereas such a node is required in [14]–[16], [19]. In our algorithm, none of the nodes knows the total demand, yet the demand and supply balance is guaranteed by the algorithm. For that purpose, every bus in the power grid, with pure load, pure generation, or both, is modelled as a node, which merely knows its local power demand (the demand from loads attached to it). However, in [18], the nodes only represent buses with a generator aware of its associated power demand (including the power demand of the pure generation buses in its neighborhood).
3) Our algorithm only assumes that the communication network is a strongly connected directed graph, whereas in [14], [15], and [17], an undirected graph is assumed. Communications may be subject to packet loss, device failure, or asymmetric bandwidth allocations, which makes the directed graph model more reasonable and general. However, it is well-known in the field of distributed control and computation that convergence analysis is much more challenging in the directed graph setting.
4) The algorithm developed in this paper can handle EDP with general convex cost functions, whereas the algorithm proposed in [18] is hard to be applied to EDP with general convex functions, due to the difficulty in guaranteeing its convergence in such a situation.

II. Preliminaries and Problem Formulation

In this section, we first give some basic concepts and results in graph theory, and then present a consensus-like iterative method. Then we formulate the EDP as a convex optimization problem with both equality and inequality constraints, and introduce its centralized solution.

A. Graph Theory

A directed graph (or just digraph) $G = (V, E)$ consists of a non-empty finite set of nodes $V = \{1, 2, \ldots, n\}$ and a finite set of ordered edges $E \subseteq V \times V$. For node $i \in V$, its in-neighbor set and out-neighbor set are denoted by $N_i^- = \{j \in V - \{i\} : (j, i) \in E\}$ and $N_i^+ = \{j \in V - \{i\} : (i, j) \in E\}$, i.e., node i receives information from its in-neighbors and sends out information to its out-neighbors. The in-degree and out-degree of node i are the cardinalities of N_i^- and N_i^+, denoted by $d_i^- = |N_i^-|$ and $d_i^+ = |N_i^+|$, respectively. A path in graph G is a finite sequence of edges in E connecting a sequence of distinct nodes in V, and the length of a path is the number of its edges. The diameter of a connected directed graph G, denoted by D, is defined as the length of the longest among the shortest paths connecting any two nodes. The $period\ d$ of G is defined as the greatest common divisor of all the lengths of cycles in G. We call the graph is $d - periodic$ if $d > 1$ and $aperiodic$ if $d = 1$. We assume that each node can communicate with itself, i.e., $\forall i \in V, (i, i) \in E$, thus $d = 1$ and the graph is aperiodic. A graph is strongly connected if there is a path from any node to any other node in the graph, which is assumed throughout the paper. We also say that a non-negative matrix $A \in \mathbb{R}^{n\times n}$ is associated with graph G, where $[A]_{ij} > 0$ if and only if $(j, i) \in E$.

B. Consensus-Like Iterative Algorithm

We start by introducing a result about non-negative matrices. Then we present a consensus-like iterative algorithm that was first introduced in [20].

Lemma 1: [21] A non-negative matrix $A \in \mathbb{R}^{n\times n}$ is primitive, if and only if its associated graph G is strongly connected and aperiodic.

For a strongly connected digraph $G = (V, E)$, define a *normalized adjacency matrix* $Q \in \mathbb{R}^{n\times n}$ as

$$[Q]_{ij} = q_{ij} = \begin{cases} \frac{1}{d_j^+ + 1} & \text{if } (j, i) \in E, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

One can easily verify that Q is associated with G and is column stochastic, i.e., Q^T is (row) stochastic. From the properties of stochastic matrix, we have $\rho(Q) = \rho(Q^T) = 1$ and $Q^T\underline{1} = 1\underline{1}$, where $\underline{1} = [1, 1, \ldots, 1]^T$. Since G is strongly connected with self-loops, from Lemma 1, Q is primitive, and thus Q^T is also primitive. From Perron-Frobenius Theorem [22], we have

$$\lim_{t\to\infty} Q^t = \lim_{t\to\infty} ((Q^T)^t)^T = (\underline{1}\eta^T)^T = \eta\underline{1}^T \tag{2}$$

where $\eta = [\eta_1, \eta_2, \ldots, \eta_n]^T$ is the right eigenvector associated to the eigenvalue 1 of Q, with the properties $\eta_i > 0$ for all i and $\underline{1}^T\eta = 1$.

Endow each node i in the graph $G = (V, E)$ with a local variable $\pi_i \in \mathbb{R}$, and denote the global variable by the column vector $\pi = [\pi_1, \pi_2, \ldots, \pi_n]^T$. Let us consider the following consensus-like iterative algorithm with the iteration index denoted by t and initial value $\pi(0)$:

$$\pi(t + 1) = Q\pi(t). \tag{3}$$

This can be implemented in a distributed form, i.e.,

$$\pi_i(t + 1) = q_{ii}\pi_i(t) + \sum_{j\in N_i^-} q_{ij}\pi_j(t). \tag{4}$$

From (2) and (3), the iterative algorithm has the following property:

$$\pi^* = \lim_{t\to\infty} \pi(t) = \eta\underline{1}^T\pi(0) = \left(\sum_{i=1}^{n} \pi_i(0)\right)\eta \tag{5}$$

where π^* denotes an equilibrium point of system (3).

We call this algorithm "consensus-like" because, if the matrix Q is such that $\eta_i = 1/n$ (which happens when Q is doubly stochastic, i.e., both row and column stochastic), all the $\pi_i(t)$'s reach the average consensus asymptotically, i.e., they all converge to the same average of the initial $\pi(0)$ [23].

Remark 1: To guarantee the convergence of the "consensus like" iteration, a strongly connected digraph with self-loops is required. Besides, each node i needs to know its out-degree d_i^+ for sake of the distributed implementation. As for communication, each node i sends out $\pi_i(t)/(d_i^+ + 1)$, instead of $\pi_i(t)$, which is slightly different from the consensus algorithm. This consensus-like iterative algorithm is also used for the study of *ratio consensus* in [16], [20], and [24].

C. Problem Formulation

The EDP we study in this paper is to minimize the aggregate cost of all the generators in the power grid on the premise that all the n generators cooperatively provide a required amount of power $P^\star$ within their individual generation capacities. We only consider active power in this paper and we ignore power transmission loss and transmission capacity constraints, which is valid for many power networks. Each generator is associated with a local variable $x_i \geqslant 0$, i.e., the (active) power generated by generator i, and a cost function $C_i(x_i)$.

In this paper, we deal with the EDP with general cost functions satisfying the assumption below:

Assumption 1: For every $1 \leqslant i \leqslant n$, $C_i(x_i) : \mathbb{R}_+ \to \mathbb{R}_+$ is strictly convex and twice continuously differentiable with

$$\frac{d^2 C_i(x_i)}{dx_i^2} \geqslant 0, \quad \forall x \in \mathbb{R}_+$$

where $\mathbb{R}_+$ denotes the set of nonnegative real numbers, and the equality holds at isolated points only.

One can easily verify that the commonly used quadratic cost functions, given as follow, are special cases satisfying Assumption 1:

$$C_i(x_i) = a_i x_i^2 + b_i x_i + c_i \tag{6}$$

where $a_i > 0$, b_i, and c_i are cost parameters. For simplicity of expression in the following sections, we use an equivalent function by changing a constant term:

$$C_i(x_i) = \frac{(x_i - \alpha_i)^2}{2\beta_i} \tag{7}$$

with constants α_i and $\beta_i > 0$.

The following is an example of a non-quadratic cost function with an natural exponential term [1]:

$$C_i(x_i) = a_i x_i^2 + b_i x_i + c_i + d_i \exp(\frac{x_i - e_i}{o_i})$$

with $a_i > 0$, $d_i > 0$, and $o_i > 0$.

We remark that many generators, especially distributed energy resources (DER) feeding on renewable energy, are often uncontrollable. For such a generator j, we may assume that a fixed amount of power x_j^{fix} [17]. These generators can be viewed as negative loads and added to the positive loads, i.e., the load P_j for the associated bus j will be replaced with $P_j - x_j^{\text{fix}}$. With the above convention, we can assume, without loss of generality, that every generator has variable generation capacity.

Denote the total number of buses in the grid by m and the number of buses with power generators by n. In general, each bus can be with a generator only, loads only, or both. Since not all the buses are attached to power generators, we have $m > n$. Denoting by P_j the power demand (load) of bus j, the aggregate power demand $P^\star$ is given by

$$\sum_{j=1}^{m} P_j = P^\star$$

where $P_j = 0$, if bus j is a pure generation bus.

For generators with variable generation capacities, denoting by $\underline{x}_i$ and $\bar{x}_i$ the lower and upper bounds of x_i, we have

$$0 \leqslant \underline{x}_i \leqslant x_i \leqslant \bar{x}_i.$$

In the framework assumed above, the EDP can be formulated as follows:

$$\min \quad \sum_{i=1}^{n} C_i(x_i), \tag{8}$$

$$\text{s.t.} \quad \underline{x}_i \leqslant x_i \leqslant \bar{x}_i, \ \forall i = 1, 2, \ldots, n, \tag{9}$$

$$\sum_{i=1}^{n} x_i = P^\star. \tag{10}$$

It is obvious that the EDP is feasible if and only if

$$\sum_{i=1}^{n} \underline{x}_i \leqslant P^\star \leqslant \sum_{i=1}^{n} \bar{x}_i. \tag{11}$$

Note that the optimization problem (8)–(10) also finds application in other problems, including optimal resource allocation problem (ORAP) for parallel computing [7], and demand side management (DSM) for power systems, especially for direct load control of smarter control systems [25].

Throughout the paper, we assume that communication networks are imposed on the power grid so that each bus corresponds to a node in the communication network. Here we set up two communication networks, denoted by $G_m = (V_m, E_m)$ and $G_n = (V_n, E_n)$, respectively. The node set V_m consists of all the m buses in the grid, while V_n consists of all the n generation buses, i.e., $V_m = \{1, 2, \ldots, n, n+1, \ldots, m\}$ and $V_n = \{1, 2, \ldots, n\}$. Define E_m and E_n as the sets of directional communication paths between nodes in G_m and G_n, respectively. It is assumed that both G_m and G_n are strongly connected digraphs with self loops. To make our distributed solution meaningful, we also assume that G_m and G_n are sparse graphs in the sense that

$$\max_{1 \leqslant i \leqslant m} d_{m,i}^+ \ll m, \ \max_{1 \leqslant i \leqslant n} d_{n,i}^+ \ll n,$$

$$\max_{1 \leqslant i \leqslant m} d_{m,i}^- \ll m, \ \max_{1 \leqslant i \leqslant n} d_{n,i}^- \ll n.$$

Except for the ability of exchanging information, the ability of local computation is also required for each node. Besides, every node i knows its local power demand P_i, cost function $C_i(x_i)$ and capacity constraints $\underline{x}_i$ and $\bar{x}_i$, but they do not need to know other nodes' parameters.

D. Centralized Solution to EDP

It is clear that the EDP is a convex optimization problem, and Assumption 1 guarantees a unique optimal solution for the problem (8)–(10). Many centralized algorithms have been developed for the convex optimization problem [26]. Furthermore, the cost function is strictly convex and Slater condition holds due to affine constraint (10). Thus, strong duality is guaranteed, which allows us to solve the primal problem by solving its Lagrange dual problem [26]. The centralized solution is as follows.

Denote the incremental cost of generator i by

$$u_i(x_i) = \frac{dC_i(x_i)}{dx_i},\ \forall i \in V_n$$

which is continuous and strictly increasing with respect to x_i according to Assumption 1. Thus the inverse function of $u_i(x_i)$, denoted by u_i^{-1}, exists and is also continuous and strictly increasing. Note that u_i^{-1} may not have a closed-form expression, but its numerical solution can be obtained using a bisection method due to its continuity and strict monotonicity.

The Lagrange dual problem is given by

$$\max \sum_{i=1}^{n} C_i^\star(\lambda) + \lambda P^\star \tag{12}$$

where

$$C_i^\star(\lambda) = \begin{cases} C_i(\underline{x}_i) - \lambda \underline{x}_i & \lambda < u_i(\underline{x}_i), \\ C_i(u_i^{-1}(\lambda)) - \lambda u_i^{-1}(\lambda) & u_i(\underline{x}_i) \leqslant \lambda < u_i(\bar{x}_i), \\ C_i(\bar{x}_i) - \lambda \bar{x}_i & u_i(\bar{x}_i) \leqslant \lambda \end{cases} \tag{13}$$

and $\lambda \in \mathbb{R}$ is the Lagrange multiplier.

From the above, we have

$$g_i(\lambda) = \frac{dC_i^\star(\lambda)}{d\lambda} = \begin{cases} -\underline{x}_i & \lambda < u_i(\underline{x}_i), \\ -u_i^{-1}(\lambda) & u_i(\underline{x}_i) \leqslant \lambda < u_i(\bar{x}_i), \\ -\bar{x}_i & u_i(\bar{x}_i) \leqslant \lambda. \end{cases} \tag{14}$$

If the primal solution is feasible, the Lagrange dual problem (12) has a unique optimal solution λ^*, satisfying

$$P^\star + \sum_{i=1}^{n} g_i(\lambda^*) = 0.$$

Accordingly, the primal EDP has a unique optimal solution given by $x_i^* = -g_i(\lambda^*), i = 1, 2, \ldots, n$, i.e.,

$$x_i^* = \begin{cases} \underline{x}_i & \lambda^* < u_i(\underline{x}_i), \\ u_i^{-1}(\lambda^*) & u_i(\underline{x}_i) \leqslant \lambda^* < u_i(\bar{x}_i), \\ \bar{x}_i & u_i(\bar{x}_i) \leqslant \lambda^*. \end{cases} \tag{15}$$

III. Main Results

In this section, we present a distributed bisection method to obtain the optimal Lagrange multiplier λ^* for the problem (8)–(10). This is done based on the iterative algorithm (3), with no need for a central decision maker or a leader node. We first propose a distributed algorithm for gathering the aggregate power demand, then show distributed feasibility test of EDP, and finally give a distributed bisection algorithm for EDP.

Intuitively speaking, to solve the EDP in a fully distributed fashion, the total power demand $P^\star = \sum_{j=1}^{m} P_j$ shall be obtained by all the generators using some distributed algorithm. We term such a problem as the *sum consensus* problem, i.e., every node gets a common value equal to the sum of all the nodes' initial values, using a distributed method. A special instance of the sum consensus problem is the so-called *network size* problem, where the goal is to use a fully distributed algorithm to find out the number of nodes in a connected network [27], [28]. This is a special instance of the sum consensus problem by setting all the initial values to 1. It is known that if each node has bounded memory, communication and computation, and the network is anonymous (i.e., each node does not have a unique global identifier), then a sufficiently large network size can not be computed using a fully distributed algorithm [28]. A challenge for us is to get over this technical difficulty.

A. Distributed Algorithm for Aggregate Demand

The first step of solving EDP is to collect the aggregate power demand $P^\star = \sum_{j=1}^{m} P_j$. From our discussion above with regard to the sum consensus problem, we understand that it is a difficult task to compute $P^\star$ directly. Instead, our algorithm is to make every node i (generation bus) in V_n get a value $y_i^\star$ such that $\sum_{i=1}^{n} y_i^\star = P^\star$. As we will show in the next subsection, it turns out that such values of $y_i^\star$ will be sufficient to solve the EDP. Our algorithm is developed based on the aforementioned consensus-like algorithm, with novelty in how to transfer the aggregate power demand $P^\star$ from the graph G_m to G_n using a fully distributed algorithm, i.e., we go from $\sum_{j=1}^{m} P_j = P^\star$ to $\sum_{i=1}^{n} y_i^\star = P^\star$.

For $G_m = (V_m, E_m)$, define an associated *normalized adjacency matrix* $Q \in \mathbb{R}^{m \times m}$ as

$$[Q]_{ij} = q_{ij} = \begin{cases} \frac{1}{d_{m,j}^+ + 1} & \text{if } (j,i) \in E_m \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

where $d_{m,j}^+$ is the out-degree of node $j \in G_m$. Similarly, for $G_n = (V_n, E_n)$, define an associated *normalized adjacency matrix* $R \in \mathbb{R}^{n \times n}$ as

$$[R]_{ij} = r_{ij} = \begin{cases} \frac{1}{d_{n,j}^+ + 1} & \text{if } (j,i) \in E_n \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

where $d_{n,j}^+$ is the out-degree of node $j \in G_n$.

For every node $i \in V_m$, we first establish an auxiliary variable $p_i(t)$ with initial value $p_i(0) = P_i$, and then run the following iterations until convergence:

$$p_i(t+1) = q_{ii} p_i(t) + \sum_{j \in N_{m,i}^-} q_{ij} p_j(t) \tag{18}$$

where $N_{m,i}^-$ denotes the in-neighbor set of node i in G_m. Denoting $p^\star = \lim_{t \to \infty} p(t)$, from (5), we have

$$p_i^\star = P^\star \eta_i,\ \forall i \in V_m \tag{19}$$

where $\eta = [\eta_1, \eta_2, \ldots, \eta_m]^T$ is the right eigenvector for the eigenvalue 1 of Q, with the properties $\eta_i > 0$ for all i and $\underline{1}^T \eta = 1$. In words, $p_i^\star$'s are the scaled local power demands. Using the auxiliary variables p_i's, the demand information is gathered in G_m.

Once (18) converges, for any node $i \in V_m$, we establish an auxiliary variable $s_i(t)$ initialized with

$$s_i(0) = \begin{cases} p_i^\star & i = 1, 2, \ldots, n, \\ 0 & i = n+1, n+2, \ldots, m \end{cases}$$

and then run the following iterations until convergence:

$$s_i(t+1) = q_{ii} s_i(t) + \sum_{j \in N_{m,i}^-} q_{ij} s_j(t). \tag{20}$$

Denoting $s^\star = \lim_{t\to\infty} s(t)$, from (5), we have

$$s_i^\star = \left(\sum_{j=1}^{n} \eta_j \right) P^\star \eta_i, \ \forall i \in V_m. \tag{21}$$

Variables $s_i^\star$'s are the scaling ratios between graph G_m and G_n. We will use them to transfer the demand information from G_m to G_n. Next, for every node $i \in V_n$, we establish an auxiliary variable $y_i(t)$ with initial value

$$y_i(0) = \frac{(p_i^\star)^2}{s_i^\star} = \frac{P^\star \eta_i}{\sum_{j=1}^{n} \eta_j} \tag{22}$$

and then run the following iterations until convergence:

$$y_i(t+1) = r_{ii} y_i(t) + \sum_{j \in N_{n,i}^-} r_{ij} y_j(t) \tag{23}$$

where $N_{n,i}^-$ denotes the in-neighbor set of node i in G_n. Denoting $y^\star = \lim_{t\to\infty} y(t)$, from (5) and (22), we have

$$y_i^\star = \left(\sum_{j=1}^{n} y_j(0) \right) \gamma_i = P^\star \gamma_i, \ \forall i \in V_n \tag{24}$$

where $\gamma = [\gamma_1, \gamma_2, \ldots, \gamma_n]^T$ is the right eigenvector for the eigenvalue 1 of R, with the properties $\gamma_i > 0$ for all i and $\underline{1}^T \gamma = 1$.

Variables $y_i^\star$ are the scaled power demand held by the generator buses only. Using the above procedures, we can get the needed demand information $y_i^\star$'s in a distributed fashion. We summarize below the distributed algorithm above.

Remark 2: It is clear that the algorithm above is fully distributed because each node only uses local information and information from its neighbors without any central processing or leader node. We will thereinafter show that based on Algorithm 1, our distributed bisection algorithm does not need a central information collector to compute $P^\star$, and the total power demand $P^\star$ is not needed explicitly for solving the EDP.

Algorithm 1: Distributed Algorithm for $P^\star$

Require: P_i : the local power demand, $\forall i \in V_m$;

Ensure: $y_i^\star, \ \forall i \in V_n$;

1: In graph G_m, each node runs (18) to get $p_i^\star$ in (19);

2: In graph G_m, each node runs (20) to get $s_i^\star$ in (21);

3: In graph G_n, each node runs (23) to get $y_i^\star$ in (24);

B. Distributed Algorithm for Feasibility Test

Before proceeding to the distributed solution to EDP, we propose a distributed iterative algorithm based on (4) for feasibility test.

Recall the EDP is feasible if and only if (11) holds. For any node $i \in V_n$, consider two variables $\underline{y}_i(t)$, $\bar{y}_i(t)$, with their initial values given by

$$\underline{y}_i(0) = \underline{x}_i, \ \bar{y}_i(0) = \bar{x}_i.$$

Run the following iterations simultaneously:

$$\underline{y}_i(t+1) = r_{ii} \underline{y}_i(t) + \sum_{j \in N_{n,i}^-} r_{ij} \underline{y}_j(t), \tag{25}$$

$$\bar{y}_i(t+1) = r_{ii} \bar{y}_i(t) + \sum_{j \in N_{n,i}^-} r_{ij} \bar{y}_j(t). \tag{26}$$

It is clear that the above converges. Denote their asymptotic values by $\underline{y}_i^\star$ and $\bar{y}_i^\star$, respectively. From (5), we have

$$\underline{y}_i^\star = \left(\sum_{j=1}^{n} \underline{x}_j \right) \gamma_i, \tag{27}$$

$$\bar{y}_i^\star = \left(\sum_{j=1}^{n} \bar{x}_j \right) \gamma_i \tag{28}$$

for all $i \in V_n$. Since every $\gamma_i > 0$, the feasibility condition (11) holds if and only if

$$\underline{y}_i^\star \leqslant y_i^\star \leqslant \bar{y}_i^\star \tag{29}$$

for any $i \in V_n$, where $y_i^\star$ is obtained from (24). Moreover, if the above holds for one node, then it holds for all other nodes.

For clarity, the result above is summarized in the following algorithm.

Algorithm 2: Distributed Algorithm for Feasibility Test

Require: $y_i^\star$, $\underline{x}_i$ and $\bar{x}_i$, $\forall i \in V_n$;

Ensure: feasible or not;

1: **Initialization:** $\underline{y}_i(0) = \underline{x}_i$, $\bar{y}_i(0) = \bar{x}_i$;

2: Each node runs iteration (25) and (26);

3: **if** Inequality (29) is true **then**

4: Feasible

5: **else**

6: Infeasible

7: **end if**

C. Distributed Bisection Algorithm for EDP

We now propose a distributed algorithm for EDP, which is operated in graph G_n. Following from Algorithm 1, we assume here that each node i contains $y_i^\star$ with $\sum_{i=1}^{n} y_i^\star = P^\star$.

Let $k \geqslant 0$ denote the iteration index for the bisection method. We let each node establish two variables $\lambda^-(k)$ and $\lambda^+(k)$, representing the lower and upper bounds of the Lagrange multiplier. Their initial values are given such that $\lambda^-(0)$ is sufficiently small and $\lambda^+(0)$ is sufficiently large, or alternatively given by

$$\lambda^-(0) = \min_{i \in V_n} u_i(\underline{x}_i),$$
$$\lambda^+(0) = \max_{i \in V_n} u_i(\bar{x}_i).$$

We will explain how to compute these initial values using a distributed algorithm later.

Define a variable $\lambda(k)$, which acts as an approximation of the Lagrange multiplier, as

$$\lambda(k) = \frac{(\lambda^+(k) + \lambda^-(k))}{2}. \tag{30}$$

Each node $i \in V_n$ takes

$$x_i(k) = -g_i(\lambda(k)), \tag{31}$$

establishes a local variable $z_i(t)$ initialized by $z_i(0) = x_i(k)$, and runs the following iterations:

$$z_i(t+1) = r_{ii} z_i(t) + \sum_{j \in N^-_{n,i}} r_{ij} z_j(t). \tag{32}$$

Denoting $z^\star = \lim_{t \to \infty} z(t)$ and using (5), we have

$$z_i^\star = \left(\sum_{j=1}^{n} x_j(k) \right) \gamma_i, \ \forall i \in V_n \tag{33}$$

where $z_i^\star$'s are the scaled generator outputs associated with $\lambda(k)$. Every node updates $\lambda^+(k+1)$ and $\lambda^-(k+1)$ by comparing $y_i^\star$ and $z_i^\star$ as follows:

$$\begin{cases} \lambda^+(k+1) = \lambda(k), \ \lambda^-(k+1) = \lambda^-(k) & \text{for } z_i^\star > y_i^\star, \\ \lambda^+(k+1) = \lambda^+(k), \ \lambda^-(k+1) = \lambda(k) & \text{for } z_i^\star \leqslant y_i^\star. \end{cases} \tag{34}$$

Although the update of $\lambda^+(k+1)$ and $\lambda^-(k+1)$ is done locally, every node computes the same $\lambda^+(k+1)$ and $\lambda^-(k+1)$ because (24), (33), and $\gamma_i > 0$ imply that

$$\operatorname{sgn}(z_i^\star - y_i^\star) = \operatorname{sgn}(z_j^\star - y_j^\star), \ \forall i, j \in V_n$$

where $\operatorname{sgn}(\cdot)$ is the sign function.

It is clear from (30) and (34) that $\lambda^\star = \lim_{k \to \infty} \lambda(k)$ exists and that each node obtains locally its optimal solution from (31), i.e.,

$$x_i^* = -g_i(\lambda^*), \ \forall i \in V_n.$$

For clarity, we summarize the distributed bisection method for this scenario in Algorithm 3. The convergence property of Algorithm 3 is formally stated below.

Algorithm 3: Distributed Bisection Method for EDP

Require: P_i: local power demand, $\forall i \in V_m$;

Ensure: x_i^*: power assignment, $\forall i \in V_n$;

1: Each node gets $y_i^\star$ using Algorithm 1;

2: Each node runs feasibility test using Algorithm 2;

3: Initialization $\lambda^-(0)$ and $\lambda^+(0)$;

4: **if** feasible **then**

5: **for** $k = 0, 1, 2, \ldots$ **do**

6: Each node computes $\lambda(k) = 1/2(\lambda^-(k) + \lambda^+(k))$;

7: Each node computes $x_i(k) = -g_i(\lambda(k))$;

8: Each node runs iteration (32) to get $z_i^\star$;

9: Each node computes $\lambda^+(k+1)$ and $\lambda^-(k+1)$ according to (34).

10: **end for**

11: Each node computes $x_i^* = -g_i(\lambda^*)$;

12: **end if**

Theorem 1: Under the assumption that the EDP (8)–(10) is feasible, Algorithm 3 converges to the unique optimal solution as $k \to \infty$.

Proof: For all $i \in V_n$, the function $g_i(\lambda)$ is monotonically decreasing with respect to λ, therefore $-g_i(\lambda)$ is monotonically increasing. Especially, $-g_i(\lambda)$ is strictly increasing with respect to λ for

$$u_i(\underline{x}_i) \leqslant \lambda \leqslant u_i(\bar{x}_i).$$

Define

$$\bar{\lambda} = \max_{i \in V_n} u_i(\bar{x}_i),$$
$$\underline{\lambda} = \min_{i \in V_n} u_i(\underline{x}_i).$$

Since the problem is feasible, the optimal Lagrange multiplier must satisfy

$$\underline{\lambda} \leqslant \lambda \leqslant \bar{\lambda}.$$

Therefore, $-\sum_{i=1}^n g_i(\lambda)$ is strictly increasing with respect to $\lambda \in [\underline{\lambda}, \bar{\lambda}]$. Thus, for every node i, $\left(\sum_{j=1}^n x_j(k)\right) \gamma_i$ is strictly increasing with respect to $\lambda(k) \in [\underline{\lambda}, \bar{\lambda}]$. Therefore, Algorithm 3 converges. Moreover, since the optimal solution is unique, Algorithm 3 converges to the unique one. ■

Remark 3: The algorithm above is fully distributed due to the following properties. Information exchange between nodes occurs only when running the consensus-like iteration (32). All

the computations are performed locally. Also, each node only requires knowledge of local parameters α_i and β_i, without need for knowledge of other nodes' parameters.

Remark 4: Besides getting around the difficulty of directly obtaining $P^\star$ in a distributed fashion, another benefit of Algorithm 1 is that it reduces the communication and computation burden of the nodes representing buses with pure loads. The utilization of the two networks G_m and G_n is motivated by the fact that in reality generator buses only account for a relatively small percentage in power systems. For instance, there are only 5 and 14 generator buses in the IEEE 14-bus and 118-bus test systems [29]. Intuitively, it seems unnecessary for the non-generator buses to be involved in the overall process. But on the other hand, the power demand is spatially distributed at almost all the buses. To deal with this situation, two communication networks G_m and G_n are constructed and Algorithm 1 is developed. After $y_i^\star$'s are computed using Algorithm 1, the remaining part of Algorithm 3 only involves generation buses, i.e., the bisection steps are performed in G_n only. As we will show in the simulations, the utilization of the two networks greatly reduces the aggregate communication volume.

We now address the issue of how to choose the initial values $\lambda^-(0)$ and $\lambda^+(0)$. Intuitively, the closer $\lambda^+(0)$ and $\lambda^-(0)$ are to the optimal multiplier λ^*, the fewer steps of bisection are needed. For Algorithm 3, if the EDP is feasible, we can initialize $\lambda^-(0)$ and $\lambda^+(0)$ using a minimum/maximum consensus algorithm [30]. Note that for a strongly connected network, minimum/maximum consensus algorithm is a fully distributed iterative algorithm.

Algorithm 4: Initialization of $\lambda^+(0)$ and $\lambda^-(0)$

1: Each node $i \in V_n$ initializes $\overline{\lambda}_i = u_i(\overline{x}_i)$ and $\underline{\lambda}_i = u_i(\underline{x}_i)$;

2: Run maximum consensus so that each node i achieves

$$\overline{\lambda}_i := \max_{j \in V_n} \overline{\lambda}_j.$$

3: Run minimum consensus so that each node i achieves

$$\underline{\lambda}_i := \min_{j \in V_n} \underline{\lambda}_j.$$

4: Each node i sets $\lambda^+(0) = \overline{\lambda}_i$, $\lambda^-(0) = \underline{\lambda}_i$;

D. Convergence and Stopping Criteria

Now we give analysis on convergence of Algorithm 3 and offer stopping criteria for a practical implementation of the algorithms. Two stopping criteria are needed, one for the consensus-like iterations at each bisection step, and the other for the bisection iterations.

Firstly, we give the stopping criterion for the consensus-like iterations. The consensus like iteration (3) converges asymptotically, but it is impossible to run it for infinite time. In most scenarios, when implementing iteration (3), a stopping criterion might be setting a fixed number of iterations t^* such that

$$\frac{\|\pi(t) - \pi^*\|_2}{\|\pi(0) - \pi^*\|_2} < \epsilon, \forall t \geqslant t^* \tag{35}$$

i.e., for some preset $\epsilon > 0$, which is small enough to assume the iteration converges to π^*. Therefore, there accordingly exists t^*, such that when $t = t^*$, the iterations are assumed to have converged. The stopping criterion in a distributed manner to judge whether consensus is reached is also studied in [31].

However, such stopping criterion is not suitable for the consensus-like iterations when used on (32) in Algorithm 3. According to (35), when convergence is assumed to be reached for $t > t^*$, it is not necessarily that the nodes make the same decision on how to bisect their incremental cost intervals, which may happen when the iterations stop at some time t^*, with $z_i^\star > y_i^\star$ for some nodes, while $z_i^\star \leqslant y_i^\star$ for the other nodes. Consequently, different decisions on bisection are made. In fact, at each bisection step, we only need to run iteration (32) until every node reaches agreement on the direction in which they shall bisect their incremental cost intervals.

For clarity, we modify our Algorithm 4 as follows. Reinitializing $z_i(0) = x_i(k) - y^\star$, and then running iterations (32), it is easy to verify that (34) is equivalent to

$$\begin{cases} \lambda^+(k+1) = \lambda(k),\ \lambda^-(k+1) = \lambda^-(k) & \text{for } z^\star > 0, \\ \lambda^+(k+1) = \lambda^+(k),\ \lambda^-(k+1) = \lambda(k) & \text{for } z^\star \leqslant 0. \end{cases} \tag{36}$$

Define $\sigma_i(t) = \text{sgn}(z_i(t))$, where $\text{sgn}(\cdot)$ is the sign function. We say that iteration (32) reaches *sign consensus*, if $\sigma_i(t) = \sigma_j(t), \forall i, j$. It's easily verified that for iterations (32), sign consensus can always be reached in finite time, i.e., there exists $t^\dagger \geqslant 0$, such that for all $t > t^\dagger$, $\sigma_i(t) = \sigma_j(t), \forall i, j$, unless $z^\star = 0$.

Now we give a distributed method to judge whether sign consensus is reached or not, which is based on the maximum/minimum consensus algorithm [30]. We assume each node has an estimate of (or knows) the diameter of G_n, denoted by D_n. At bisection, each node i establishes three auxiliary variables, $\underline{\sigma}_i$, $\overline{\sigma}_i$, and $\sigma_i^\dagger$. For $\sigma_i^\dagger$, we define

$$\sigma_i^\dagger(t) = \begin{cases} 1 & \text{if } z_i(\lfloor \frac{t}{D_n} \rfloor D_n) > 0, \\ 0 & \text{if } z_i(\lfloor \frac{t}{D_n} \rfloor D_n) \leqslant 0 \end{cases} \tag{37}$$

where $\lfloor \cdot \rfloor$ is the floor function. At $t = 0$, initialize $\underline{\sigma}_i(0) = \sigma_i^\dagger(0)$, $\overline{\sigma}_i(0) = \sigma_i^\dagger(0)$. Then for $t \geqslant 1$, we update $\underline{\sigma}_i(t)$ and $\overline{\sigma}_i(t)$ by

$$\underline{\sigma}_i(t+1) = \begin{cases} \sigma_i^\dagger(t) & \text{if } \lfloor \frac{t}{D_n} \rfloor D_n = t, \\ \min\limits_{j \in \{N_{n,i}^-, i\}} \underline{\sigma}_i(t) & \text{otherwise,} \end{cases} \tag{38}$$

$$\overline{\sigma}_i(t+1) = \begin{cases} \sigma_i^\dagger(t) & \text{if } \lfloor \frac{t}{D_n} \rfloor D_n = t, \\ \max\limits_{j \in \{N_{n,i}^-, i\}} \overline{\sigma}_i(t) & \text{otherwise.} \end{cases} \tag{39}$$

The stopping criterion for iteration (32) is that for every node i, if there exists some positive integer τ^* such that

$$\underline{\sigma}_i(\tau^* D_n) = \overline{\sigma}_i(\tau^* D_n) \tag{40}$$

then sign consensus is already reached. Then every node can stop the consensus-like iteration (32) and make the same decision at $t = \tau^* D_n$. Besides, $\underline{\sigma}_i(\tau^* D_n) = \overline{\sigma}_i(\tau^* D_n) = 1$ implies that $\lambda^+(k+1) = \lambda(k)$ and $\lambda^-(k+1) = \lambda^-(k)$, while $\underline{\sigma}_i(\tau^* D_n) = \overline{\sigma}_i(\tau^* D_n) = 0$ implies that $\lambda^+(k+1) = \lambda^+(k)$ and $\lambda^-(k+1) = \lambda(k)$.

The theoretical basis of the procedure (37)–(40) is that maximum/minimum consensus is bound to be reached within D_n steps, as D_n is the diameter of the strongly connected graph G_n (see [31] for more details). From (37), with $\tau = 1, 2, \ldots$, we can see that the nodes in fact only update $\sigma_i^\dagger$'s at $t = \tau D_n$, and otherwise the $\sigma_i^\dagger$'s remain unchanged. Moreover, from (38) and (39), the nodes update $\underline{\sigma}_i$ and $\bar{\sigma}_i$ at time $t = (\tau - 1)D_n + 1, (\tau - 1)D_n + 2, \ldots, \tau D_n$, using only maximum/minimum consensus algorithm. Therefore, at $t = \tau D_n$, the nodes are able to know whether they have reached sign consensus at $t = (\tau - 1)D_n$.

Remark 5: Another benefit of using the stopping criterion (40) is the probable reduction of iteration steps needed for convergence. A special situation is that the signs are the same initially, e.g., for all i, $z_i(0) > 0$. But due to the absence of a leader or master node, the nodes still have to run D_n steps of iteration (32). In comparison, even if the nodes can make the same decision when the consensus-like iteration (32) stops according to (35), it takes t^* steps of iteration. Note that t^* will be a large number when ϵ is sufficiently small, so it is often the case that $t^* \gg D_n$.

Next, we present the stopping criterion for the bisection steps. It is clear from (30) that each iteration of k halves the interval $[\lambda^-(k), \lambda^+(k)]$, thus the convergence of $\lambda(k)$ is very rapid. Since solving EDP is actually finding the optimal incremental cost λ^*, a stopping criterion can be established by either setting a fixed number of iterations K, or using

$$|\lambda(k) - \lambda^*| \leqslant \frac{\epsilon}{2} \tag{41}$$

for some sufficiently small $\epsilon > 0$. Since λ^* is not available, an alternative can be

$$|\lambda^+(k) - \lambda^-(k)| \leqslant \epsilon \tag{42}$$

which can be easily achieved in a distributed fashion, provided sign consensus is always reached in each bisection step.

IV. Simulation

In this section, we show the performance of the distributed bisection algorithm using several numerical experiments based on the IEEE 14-bus and 118-bus systems [29]. Five cases are simulated. Case 1 demonstrates our algorithm using a quadratic cost function whereas Case 2 does so using a non-quadratic cost function. In Case 3 the convergence speed of our algorithm is analyzed. We compare our algorithm with the algorithm proposed in [17] in Case 4 with regard to the convergence speed, the total computation load, and the aggregate communication volume. The above four cases use the IEEE 14-bus system. Finally in Case 5 we apply our algorithm to the IEEE 118-bus system.

For the numerical simulations on the IEEE 14-bus system, generators buses are $\{1, 2, 3, 6, 8\}$, and load buses are $\{2, 3, 4, 5, 6, 9, 10, 11, 12, 13, 14\}$. Note that the power transmission grid is not necessarily the same with the information communication network, so we do not assign a node to bus 7. The two strongly connected directed graphs with self-loops

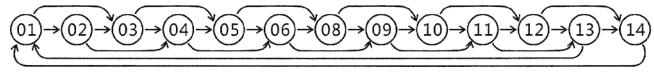


Fig. 1. Illustration of graph G_m, where self-loops are not shown.

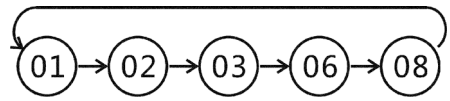


Fig. 2. Illustration of graph G_n, where self-loops are not shown.

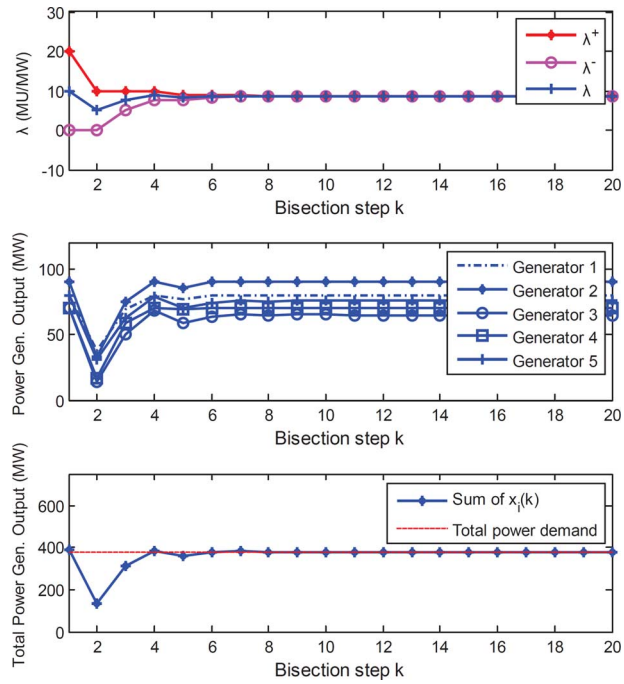


Fig. 3. Results for EDP with quadratic cost functions only.

G_m and G_n are established, as shown in Figs. 1 and 2. One can easily verify that G_m and G_n are sparse graphs.

A. Case 1: EDP With Quadratic Cost Functions Only

In this case, we solve the EDP with quadratic cost functions only, as it is most commonly assumed for EDP. The generator parameters are listed in Table I, which are adopted from [17]. We set $\underline{x}_i = 10$ MW for all i, so they are not shown in the table. We take $\epsilon = 0.005$ for the stopping criterion.

The local power demands are: $P_1 = 0$ MW, $P_2 = 9$ MW, $P_3 = 56$ MW, $P_4 = 55$ MW, $P_5 = 27$ MW, $P_6 = 46$ MW, $P_8 = 0$ MW, $P_9 = 8$ MW, $P_{10} = 24$ MW, $P_{11} = 53$ MW, $P_{12} = 46$ MW, $P_{13} = 16$ MW, and $P_{14} = 40$ MW. The total demand $P^\star = \sum_{i \in V_m} P_i = 380$ MW, which is not known to the individual nodes.

We set $\lambda^+(0) = 20$ MU/MW and $\lambda^-(0) = 0$ MU/MW, which is sufficient to guarantee $\lambda^* \in [\lambda^-(0), \lambda^+(0)]$. The result is shown in Fig. 3. The upper subplot of Fig. 3 shows the evolution of $\lambda(k)$, the middle subplot shows the evolution of each $x_i(k)$, and the lower subplot shows the evolution of $\sum_{i \in V_n} x_i$. We artificially set the iteration step to be 20, while the stopping criterion is already satisfied at $k = 12$. Taking the results at $k = 12$ to be the optimal solution, we have $x_1^* = 80.00$

TABLE I
GENERATOR PARAMETERS (MU = MONETARY UNIT)

Generator	Bus	α_i (MW)	β_i (MW^2/MU)	$\bar{x}_i$ (MW)
1	1	-25	12.5	80
2	2	-50	16.67	90
3	3	-57.14	14.29	70
4	6	-66.67	16.67	70
5	8	-31.25	12.5	80

MW, $x_2^* = 90.00$ MW, $x_3^* = 64.48$ MW, $x_6^* = 70.00$ MW, $x_8^* = 75.35$ MW, and $\sum_{i\in V_n} x_i^* = 380.03$ MW. The optimal incremental cost $\lambda^* = 8.5278$ MU/MW, and the optimal solution x^* stays within the capacity constraints, where x_i^* of generator 1, 2, and 4 take their upper bounds of capacity constraints, respectively. Note that our distributed bisection algorithm is based on the Lagrange dual method, so the equality constraint will not be truly satisfied until infinite bisections. Therefore using our algorithm, there is a tolerable gap between demand and supply, i.e., $\sum_{i\in V_n} x_i^* - P^\star = 380.03 - 380 = 0.03$ MW.

B. Case 2: EDP With Non-Quadratic Cost Functions

We now demonstrate a case where some of the generators have non-quadratic cost functions and some generators have a fixed amount of power. In particular, we replace generator 4 (at bus 6) with a fixed generation of $x_4^{\text{fix}} = 100$ MW and take

$$C_1(x_1) = \frac{(x_1+25)^2}{25} + 50\exp(\frac{x_1+40}{100}),$$
$$C_3(x_3) = \frac{(x_3+57.14)^2}{28.58} + 7\times 10^{-6}x_3^4.$$

Using the same $\lambda^+(0)$, $\lambda^-(0)$, and total power demand $P^\star$ as those in Case 1, the result is shown in Fig. 4. After 13 steps of bisection, the algorithm converges to $x_1^* = 64.86$ MW, $x_2^* = 90.00$ MW, $x_3^* = 48.62$ MW, $x_6^* = 100.00$ MW, $x_8^* = 76.46$ MW, and $\sum_{i\in V_n} x_i^* = 379.92$ MW. The optimal incremental cost $\lambda^* = 8.6157$ MU/MW.

C. Case 3: Convergence Speed Analysis

Now we study how the starting values $\lambda^+(0)$ and $\lambda^-(0)$ would affect the convergence speed. For simplicity, we still set $\lambda^-(0) = 0$ MU/MW and only vary $\lambda^+(0)$, i.e., we run Case 1 again with different values of $\lambda^+(0)$ from 10 MU/MW to 150 MU/MW. The result is shown in Fig. 5.

Apparently, a larger $\lambda^+(0)$ leads to more bisection steps. To make convergence fast, Algorithm 4 can be adapted to choose good $\lambda^+(0)$ and $\lambda^-(0)$. Furthermore, since $\lambda^\star$ is monotonically increasing with respect to $P^\star$, historical data can be useful provided the current EDP and the previous EDP share the same configuration of generators and parameters, i.e., the only difference must be the total power demand. If so, take the historical $\lambda^\star$ as $\lambda^-(0)$ of the current problem if the current total demand is larger than previous demand, and vice versa.

D. Case 4: Comparison With the Algorithm in [17]

In this case we compare our algorithm with previous work, and show the benefits of using two communication networks

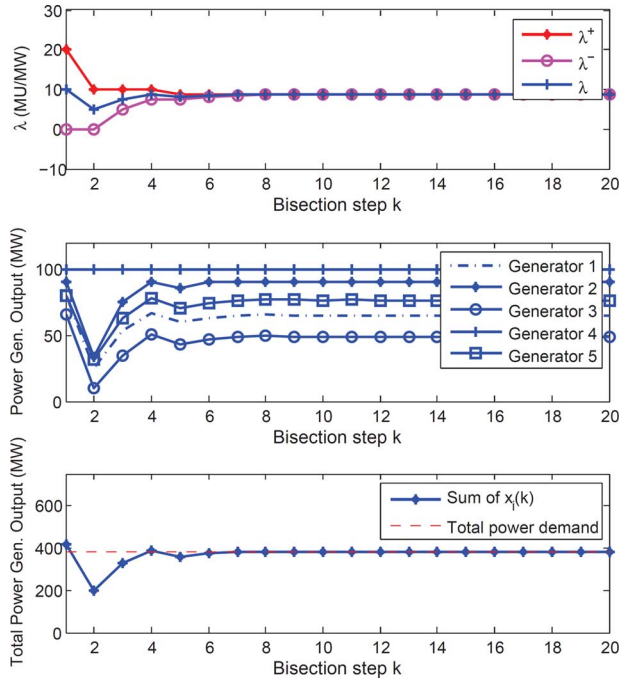


Fig. 4. Results of EDP with non-quadratic cost functions.

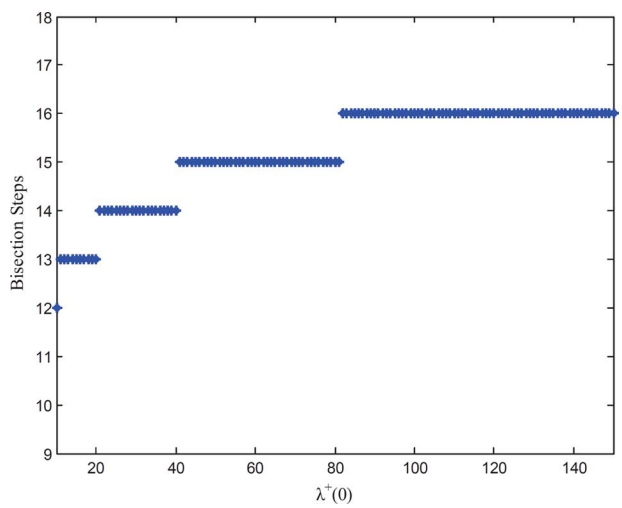


Fig. 5. Relationship between $\lambda^+(0)$ and bisection steps needed.

and the stopping criterion based on the sign consensus. Although other references listed in the introduction are also highly related to this paper, we mainly make comparisons with the algorithm proposed in [17] because [17] has the most similarities with ours with regard to the problem setup. Specifically, both the algorithms in our paper and in [17] are fully distributed without relying on a leader node, and all the buses in the grid are involved in both algorithms' implementation. The comparison is primarily about the convergence speed, the computation load, and the communication volume.

We first review the results in Case 1 using our algorithm. With $\epsilon = 0.005$, the iteration steps needed for iterations (18), (20), and (23) are 29, 50, and 24, respectively. During the bisections, the consensus-like iteration steps needed for each bisection step are shown in Fig. 6. With the stopping criterion based on the

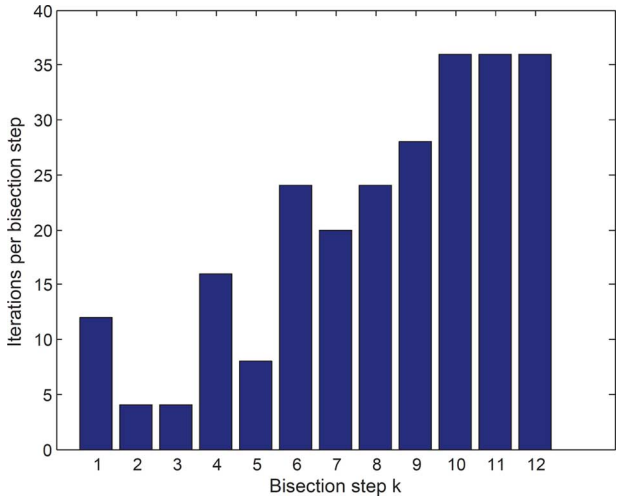


Fig. 6. Consensus-like iterations needed for each bisection step.

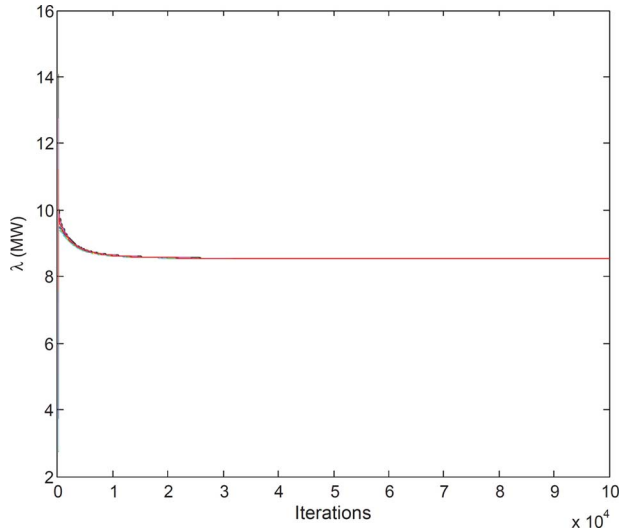


Fig. 7. Results for EDP with quadratic cost functions only (Case 1) using the algorithm in [17].

sign consensus in Section III-D, not much consensus-like iterations are needed for the bisections. Particularly, for the 2nd and 3rd bisection steps, only 4 consensus-like iterations are needed. Note that the diameter of graph G_n in Case 1 is 4. According to Remark 5, it follows that in fact the sign consensus is already reached without the need of running the consensus-like iteration. The computation time is counted in terms of iteration steps rather than seconds, as the time which an iteration takes to compute depends on the CPU speed of the nodes. For our algorithm, the computation time of this simulation example is 351 steps.

We then resolve the problem in Case 1 using the algorithm proposed in [17]. Learning gains of the algorithm are adopted from the numerical experiments in [17]. Since undirected graphs are assumed in [17], we use the undirected graph by replacing the unidirectional arrows in Fig. 1 with bidirectional ones to implement the algorithm in [17]. The results are shown in Fig. 7. One can see that the convergence of the algorithm in [17] is very slow. Though the λ's held and updated by the buses reach consensus fast, at that time the λ's do not reach λ^*. It still takes almost 1×10^5 steps to converge to the optimal Lagrange multiplier. Therefore the algorithm in [17] converges much slower than our algorithm.

Besides, in the algorithm in [17] there are totally 13 buses assigned with agents in this case. All those buses are involved in the algorithm's implementation until the convergence. Since the iterations practically account for most of the computational load, we directly use the iterations needed in total to describe the computational load. Similarly, the total number of numerical values exchanged between nodes is regarded as the aggregate communication volume. So the total computation load using the algorithm in [17] is 1.3×10^6. Each node communicates bidirectionally with other 4 nodes, so the total communication volume is $4 \times 13 \times 10^5 = 5.2 \times 10^6$, which is rather huge. As for our algorithm, although there are also 13 buses assigned nodes, we only involve them all when running iterations (18) and (20), and the remaining steps are implemented in the smaller graph G_n. So the total computation load using our algorithm is $13 \times (29+50)+5 \times (24+248) = 2487$. In our algorithm, each node in G_m and G_n communicates unidirectionally with other 4 and 2 nodes, respectively. So the total communication volume of our algorithm is $2 \times 13 \times (29+50)+1 \times (24+248) = 2326$. The comparisons are summarized in Table II.

TABLE II
SUMMARY OF THE COMPARISONS BETWEEN OUR ALGORITHM AND THE ALGORITHM PROPOSED IN [17]

	Time	Computation load	Communication volume
Our algorithm	3.5×10^2	2.5×10^3	2.3×10^3
The algorithm in [17]	1×10^5	1.3×10^6	5.2×10^6

From the data above, one can see that our algorithm converges much faster than the algorithm in [17]. Also, the computation load and communication volume using our algorithm are far less. The fast convergence and low operational cost of our algorithm are a cooperative feature of bisection, the utilization of the two communication networks, and the stopping criterion based on the sign consensus. We also remark that to guarantee the convergence of the algorithm in [17], one needs to properly design the gains, referred to as α_t and β_t in [17]. Those gains also affect the speed of convergence. However, the determination of α_t and β_t depends on global information and cannot be implemented in a distributed manner. As aforementioned in Remark 3, no such gains in our algorithm need to be pre-determined using global information.

E. Case 5: Implementation on IEEE 118-Bus System

In this case we apply our algorithm to the IEEE 118-bus test system [29] to further investigate our algorithm's performance. The generator parameters are adopted from [32]. The total power demand is 950 MW. We set $\lambda^+(0) = 50$ MU/MW and $\lambda^-(0) = 0$ MU/MW, which is sufficient to guarantee $\lambda^* \in [\lambda^-(0), \lambda^+(0)]$. The result is shown in Fig. 8. We artificially set the iteration number to 20, but in the simulation the stopping criterion is already satisfied at $k = 13$. The optimal Lagrange multiplier $\lambda^* = 29.8332$ MU/MW.

From this case we can find that our algorithm is also applicable to large power grids such as the IEEE 118-bus test system.

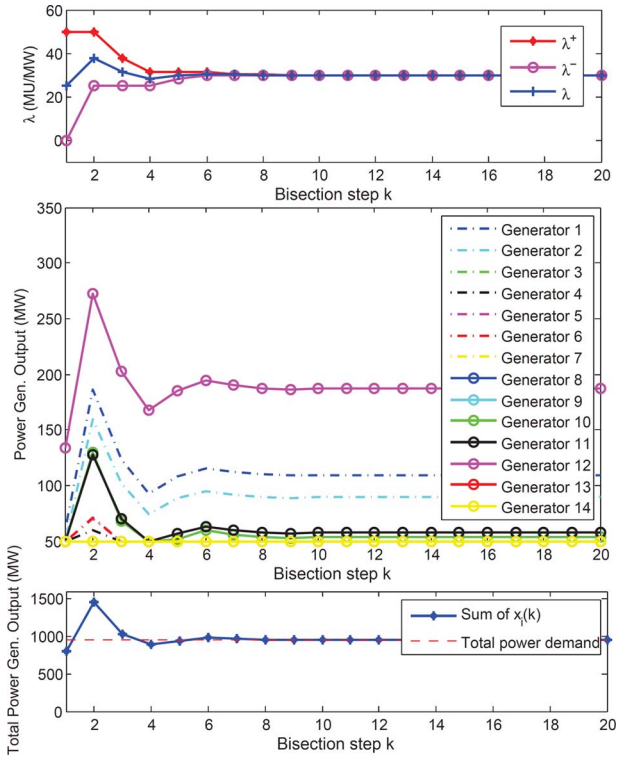


Fig. 8. Results for EDP with IEEE 118-bus system.

Moreover, the convergence is still fast due to the nature of bisection, and is insensitive to the network size.

V. Conclusion

In this paper, we have proposed a distributed bisection method based on a consensus-like iteration to solve EDP, where the cost functions can be general convex functions (i.e., they are not restricted to quadratic functions). The algorithm is fully distributed, with no need for a master node or leader, in which a strongly connected digraph with self-loops is sufficient for communication. Also, each node only requires its local parameters, without knowledge of the global information. Convergence of our algorithm is proved, and by simulations we illustrate the performance of the algorithm. Future work would be to extend the proposed distributed bisection algorithm to other applications, e.g., extended EDP with line loss and line capacity constraints considered.

References

[1] A. J. Wood and B. F. Wollenberg, *Power Generation, Operation, and Control*. : Wiley, 1996.

[2] R. A. Jabr, A. H. Coonick, and B. J. Cory, "A homogeneous linear programming algorithm for the security constrained economic dispatch problem," *IEEE Trans. Power Syst.*, vol. 15, no. 3, pp. 930–936, Aug. 2000.

[3] C. Chen and N. Chen, "Direct search method for solving economic dispatch problem considering transmission capacity constraints," *IEEE Trans. Power Syst.*, vol. 16, no. 4, pp. 764–769, Nov. 2001.

[4] W.-M. Lin, F.-S. Cheng, and M.-T. Tsay, "Nonconvex economic dispatch by integrated artificial intelligence," *IEEE Trans. Power Syst.*, vol. 16, no. 2, pp. 307–311, May 2001.

[5] J. Tippayachai, W. Ongsakul, and I. Ngamroo, "Parallel micro genetic algorithm for constrained economic dispatch," *IEEE Trans. Power Syst.*, vol. 17, no. 3, pp. 790–797, Aug. 2002.

[6] L. Bakule, "Decentralized control: An overview," *Annu. Rev. Control*, vol. 32, no. 1, pp. 87–98, 2008.

[7] L. Xiao and S. Boyd, "Optimal scaling of a gradient method for distributed resource allocation," *J. Optimiz. Theory Applicat.*, vol. 129, no. 3, pp. 469–488, 2006.

[8] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, 2009.

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

[10] H. Xing, Y. Mou, Z. Lin, and M. Fu, "Fast distributed power regulation method via networked thermostatically controlled loads," in *Proc. 19th World Congr. Int. Federation of Automatic Control*, Cape Town, South Africa, 2014, pp. 5439–5444.

[11] Y. Mou, H. Xing, Z. Lin, and M. Fu, "A new approach to distributed charging control for plug-in hybrid electric vehicles," in *Proc. 33rd Chinese Control Conf. (CCC)*, Nanjing, China, 2014, pp. 8118–8123.

[12] Y. Mou, H. Xing, Z. Lin, and M. Fu, "Decentralized optimal demand-side management for PHEV charging in a smart grid," *IEEE Trans. Smart Grid*, accepted for publication.

[13] A. Ipakchi and F. Albuyeh, "Grid of the future," *IEEE Power Energy Mag.*, vol. 7, no. 2, pp. 52–62, 2009.

[14] Z. Zhang and M.-Y. Chow, "Incremental cost consensus algorithm in a smart grid environment," in *Proc. IEEE Power and Energy Soc. General Meeting*, San Diego, CA, USA, Jul. 24–29, 2011, pp. 1–6.

[15] Z. Zhang, X. Ying, and M.-Y. Chow, "Decentralizing the economic dispatch problem using a two-level incremental cost consensus algorithm in a smart grid environment," in *Proc. IEEE North Amer. Power Symp. (NAPS)*, Boston, MA, USA, Aug. 4–6, 2011, pp. 1–7.

[16] A. D. Dominguez-Garcia, S. T. Cady, and C. N. Hadjicostis, "Decentralized optimal dispatch of distributed energy resources," in *Proc. IEEE Conf. Decision and Control*, Maui, HI, USA, Dec. 10–13, 2012, pp. 3688–3693.

[17] S. Kar and G. Hug, "Distributed robust economic dispatch in power systems: A consensus + innovations approach," in *Proc. IEEE Power and Energy Soc. General Meeting*, San Diego, CA, USA, Jul. 2012, pp. 1–8.

[18] S. Yang, S. Tan, and J. X. Xu, "Consensus based approach for economic dispatch problem in a smart grid," *IEEE Trans. Power Syst.*, vol. 28, no. 4, pp. 4416–4426, Nov. 2013.

[19] H. Xing, Y. Mou, M. Fu, and Z. Lin, "Consensus based bisection approach for economic power dispatch," in *Proc. 53rd IEEE Conf. Decision and Control (CDC)*, Los Angeles, CA, USA, 2014.

[20] A. D. Dominguez-Garcia and C. N. Hadjicostis, "Coordination and control of distributed energy resources for provision of ancillary services," in *Proc. 1st IEEE Int. Conf. Smart Grid Commun. (SmartGridComm)*, Gaithersburg, MD, USA, 2010, pp. 537–542.

[21] Z. Lin, *Distributed Control and Analysis of Coupled Cell Systems*. Saarbrücken, Germany: VDM Publishing, 2008.

[22] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1985.

[23] A. Bemporad, M. Heemels, and M. Johansson, *Networked Control Systems*. New York, NY, USA: Springer, 2010, vol. 406.

[24] A. D. Dominguez-Garcia and C. N. Hadjicostis, "Distributed algorithms for control of demand response and distributed energy resources," in *Proc. IEEE Conf. Decision and Control and Eur. Control Conf. (CDC-ECC)*, Orlando, FL, USA, 2011, pp. 27–32.

[25] G. Strbac, "Demand side management: Benefits and challenges," *Energy Policy*, vol. 36, no. 12, pp. 4419–4426, 2008.

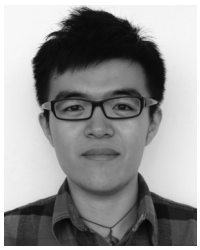
[26] S. P. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[27] I. Shames, T. Charalambous, C. N. Hadjicostis, and M. Johansson, "Distributed network size estimation and average degree estimation and control in networks isomorphic to directed graphs," in *Proc. 50th Annu. IEEE Allerton Conf. Commun., Control, and Computing*, 2012, pp. 1885–1892.

[28] J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis, "Distributed anonymous discrete function computation," *IEEE Trans. Autom. Control*, vol. 56, no. 10, pp. 2276–2289, 2011.

[29] Power System Test Case Archive [Online]. Available: http://www.ee.washington.edu/research/pstca/

[30] J. Cortés, "Distributed algorithms for reaching consensus on general functions," *Automatica*, vol. 44, no. 3, pp. 726–737, 2008.
[31] V. Yadav and M. V. Salapaka, "Distributed protocol for determining when averaging consensus is reached," in *Proc. 45th Annu. Allerton Conf*, 2007, pp. 715–720.
[32] L. Wu, Y. Wang, X. Yuan, and S. Zhou, "Environmental/economic power dispatch problem using multi-objective differential evolution algorithm," *Elect. Power Syst. Res.*, vol. 80, no. 9, pp. 1171–1181, 2010.



Hao Xing (S'14) received the Bachelor's degree from Zhejiang University, China, in 2012. He is currently pursuing the Ph.D. degree in the Department of Control Science and Engineering, Zhejiang University, China.

His research interests include distributed control and optimization in smart grid.

Yuting Mou received the Bachelor's degree from Jilin University, China, in 2012. He is currently pursuing the Master's degree in the Department of Control Science and Engineering, Zhejiang University, China.

His research interests are mainly in the areas of smart grid and distributed optimization.

Minyue Fu (F'03) received the Bachelor's degree from the University of Science and Technology of China in 1982 and the M.S. and Ph.D. degrees from the University of Wisconsin-Madison in 1983 and 1987, respectively.

From 1987 to 1989, he was an Assistant Professor at Wayne State University, Detroit, MI, USA. He joined the University of Newcastle, Australia, in 1989 and was promoted to a Chair Professor in 2002. He has served as the Head of Department and Head of School for over 7 years. He has been a Visiting Professor at the University of Iowa, Nanyang Technological University, and Tokyo University. He has held a Changjiang Visiting Professorship at Shandong University and a Qian-ren Professorship at Zhejiang University. His main research interests include control systems, signal processing, and communications. His current research projects include networked control systems as well as smart electricity networks. He has published nearly 350 papers, including 120 or so journal papers.

Dr. Fu has been an Associate Editor for the IEEE TRANSACTIONS ON AUTOMATIC CONTROL, the IEEE TRANSACTIONS ON SIGNAL PROCESSING, *Automatica*, and *Journal of Optimization and Engineering*.

Zhiyun Lin (SM'10) received the Bachelor's degree in electrical engineering from Yanshan University, China, in 1998, the Master's degree in electrical engineering from Zhejiang University, China, in 2001, and the Ph.D. degree in electrical and computer engineering from the University of Toronto, Canada, in 2005.

He was a Postdoctoral Research Associate in the Department of Electrical and Computer Engineering, University of Toronto, Canada, from 2005 to 2007. He joined the College of Electrical Engineering, Zhejiang University, China, in 2007. Currently, he is a Professor of Systems Control in the same department. He is also affiliated with the State Key Laboratory of Industrial Control Technology at Zhejiang University. He held visiting professor positions at several universities including The Australian National University (Australia), the University of Cagliari (Italy), the University of Newcastle (Australia), and the University of Technology Sydney (Australia) . His research interests focus on distributed control, estimation and optimization, coordinated and cooperative control of multi-agent systems, hybrid and switched system theory, and locomotion control of biped robots.

Dr. Lin is currently an associate editor for *Hybrid Systems: Nonlinear Analysis* and the *International Journal of Wireless and Mobile Networking*.