A Fully Decentralized Approach for Solving the Economic Dispatch Problem

Wael T. Elsayed, *Student Member, IEEE*, and Ehab F. El-Saadany, *Senior Member, IEEE*

***Abstract*—A new decentralized approach for solving the economic dispatch problem is presented in this paper. The proposed approach consists of either two or three stages. In the first stage, a flooding-based consensus algorithm is proposed in order to achieve consensus among the agents with respect to the units and system data. In the second stage, a suitable algorithm is used for solving the economic dispatch problem in parallel. For cases in which a nondeterministic method is used in the second stage, a third stage is applied to achieve consensus about the final solution of the problem, with a flooding-based consensus algorithm for sharing the information required during this stage. The proposed approach is highly effective for solving the non-convex formulation of the economic dispatch problem and for incorporating transmission losses accurately in a fully decentralized manner. Three case studies that were examined for validation purposes are described. The results obtained demonstrate that the proposed approach aggregates many of the advantages of both centralized and fully decentralized mechanisms for solving the economic dispatch problem.**

***Index Terms*—Flooding-based consensus algorithm, fully decentralized approach, non-convex economic dispatch, smart grid, transmission losses.**

I. INTRODUCTION

SOLVING the economic dispatch problem (EDP) is one of the most important tools of power system operation. The objective of solving it is to minimize the total generation cost of the generation units while satisfying numerous constraints associated with both the units and the system. The simplest formulation of the EDP involves a convex cost function and neglects many practical cost functions, such as those involving the valve-point effect or prohibited operating zones and those for multiple fuel units. The simplest formulation is either an approximated formulation or, when many practical non-convex cost functions are present in the actual system, a completely improper formulation. Such an improper formulation and the solution based on it may result in monetary losses in the order of millions of dollars per year. A more practical EDP formulation is based on treating the problem as a non-convex optimization problem in which practical non-convex cost functions are taken into consideration. Almost all previously presented solutions to the non-convex economic dispatch problem (NCEDP) are centralized solutions: the problem is solved in a central authority. Recent studies directed at enabling smart grids have led to a new research trend: the development and investigation of solutions to the EDP based on decentralized and distributed mechanisms [1]–[5]. The primary motivations for this trend are as follows:

1) The extensive employment of smart grid concepts will lead to communication congestion and complexity in central management systems. The complexity inherent in centralized controllers may make it difficult for system operators to act on information collected from smart grid sensors in an appropriate time frame [1], and the resultant communication congestion requires the implementation of a high-bandwidth communication infrastructure [5].
2) A fully decentralized system does not give rise to concerns about reliability issues related to single point failure [1]–[5].
3) Distributed and decentralized systems are more scalable and more flexible with respect to system changes than centralized systems and hence can more effectively accommodate the variable topology and the plug-and-play feature associated with smart grids [4].

Based on these factors, the need for consideration and investigation of decentralized EDP solutions is apparent. However, the literature describes only a few attempts to solve the NCEDP in a distributed manner. Almost all previously proposed distributed and decentralized algorithms are based on an approach involving consensus about the incremental cost [1]–[4]. This approach can be used only for solving the EDP with convex cost functions in a distributed manner, and transmission losses cannot be appropriately or accurately incorporated. Previous attempts to incorporate transmission losses, such as in [1] and [5], are based on the assumption that the loss coefficients are always constant or are provided to the agents by a central authority. Another attempt to incorporate transmission losses into a distributed algorithm is that proposed in [4]; however, the results produced by the first case study reported in [4] show that the methodology used to incorporate the transmission losses yields inaccurate results, especially when a substantial change in system state occurs between one dispatching period and the next, such as a significant change in system load. The only attempt to solve the NCEDP using a distributed algorithm is that reported in [5]; however, in addition to the disadvantage of the assumption of constant loss coefficients or central authority assistance in computing the loss coefficients, another inadequacy is that better-quality solutions for the NCEDP can be obtained if an efficient metaheuristic technique were used rather than the

Manuscript received June 02, 2014; revised August 18, 2014; accepted September 22, 2014. Date of publication October 07, 2014; date of current version June 16, 2015. Paper no. TPWRS-00741-2014.

The authors are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: wtahagha@uwaterloo.ca; ehab@uwaterloo.ca).

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.2014.2360369

deterministic algorithm proposed by the authors in [5]. The innovation in this paper is presenting a new fully decentralized approach for solving the EDP. This new approach utilizes the flooding based consensus algorithm over a ring communication graph for information sharing between the agents. The advantages of the approach presented in this paper are as follows:

1) The proposed approach is fully decentralized, with no need of a central authority to compute the total number of agents, the total system load, or the transmission losses.
2) The proposed approach can be adapted for solving both the convex and the practical non-convex formulation of the EDP.
3) Transmission losses are incorporated effectively.
4) Because the proposed approach can share transmission line data and bus data among the agents in a reasonable timespan that is equal to or less than a few seconds, the proposed fully decentralized approach can be adapted for solving other power system optimization problems in a fully decentralized manner: security-constrained economic dispatch, unit commitment, and optimal power flow.

The development of a fully decentralized approach that provides the above advantages for solving the EDP entails two assumptions: each bus considered in the power network must have a distinct number that characterizes it from other buses considered, and the system load can be aggregated for a specific set of buses in which each bus is responsible for estimating or forecasting its connected load for the next dispatch period. The second assumption has been implicitly taken into account in previous reported studies in which fully distributed algorithms are proposed, such as [2], [4], and [5]. For future power systems, i.e., smart grids, a set of intelligent buses for which the total system load can be aggregated, with the buses estimating and forecasting their load and then communicating with one another, is a reasonable assumption.

The paper is organized as follows. The formulation of the EDP is discussed in Section II. The proposed approach is introduced in Section III, and the simulation results are provided in Section IV. Section V includes a discussion of the practical issues and limitations related to the proposed approach, followed by the presentation of the conclusions in Section VI.

II. Formulation of the Economic Dispatch Problem

A. Simple Formulation

The following represents a simple formulation of the EDP:

$$\text{Minimize } C_T = \sum_{i=1}^{n} C_i(P_i) \tag{1}$$

Subject to

$$P_i^{\min} \le P_i \le P_i^{\max} \tag{2}$$

$$\sum_{i=1}^{n} P_i = P_D + P_L \tag{3}$$

where $C_i(P_i)$ is the cost function of generation unit i; $P_i$is the power output of generation unit i; n is the number of generators; P_D is the total system load; $P_i^{\min}$, $P_i^{\max}$ are the lower and upper limits of generation unit i, respectively; and P_L is the total system losses computed using Kron's loss formula, as follows:

$$P_L = \sum_{i=1}^{n}\sum_{j=1}^{n} P_i B_{ij} P_j + \sum_{i=1}^{n} B_{0i} P_i + B_{00} \tag{4}$$

where B, B_0, and B_{00} are the loss coefficients. The generation cost function is modelled with the following quadratic formula:

$$C_i(P_i) = a_i + b_i P_i + c_i P_i^2 \tag{5}$$

where $\mathrm{a}_i, \mathrm{b}_i$, and c_i are the fuel cost coefficients for unit i.

B. EDP With Consideration of the Valve-Point Effect

Real input-output cost curves for the generator units are non-convex due to the valve-point effect. A cost function that includes the valve-point effect can be expressed as follows [6]:

$$C_i(P_i) = a_i + b_i\, P_i + c_i\, P_i^2 + \left| d_i\, \sin(\mathrm{e}_i \times (P_i^{\min} - P_i)) \right| \tag{6}$$

where $\mathrm{a}_i, \mathrm{b}_i, \mathrm{c}_i, \mathrm{d}_i$, and e_i are the fuel cost coefficients for unit i.

C. EDP With Consideration of Prohibited Operating Zones

Physical operating limitations may result in generation units with prohibited operating zones. To model these zones, the following constraints must be added to the problem formulation:

$$\begin{aligned} &P_i^{\min} \le P_i \le P_{i,1}^l \quad (i = 1, 2, \ldots, \mathrm{n}) \\ &P_{i,j-1}^u \le P_i \le P_{i,j}^l \quad (j = 2, 3, \ldots, nj)\ (i = 1, 2, \ldots, \mathrm{n}) \\ &P_{i,nj}^u \le P_i \le P_i^{\max} \quad (i = 1, 2, \ldots, \mathrm{n}) \end{aligned} \tag{7}$$

where $P_{i,j}^l$ is the lower bound of the jth prohibited operating zone of unit i, $P_{i,j}^u$ is the upper bound of the jth prohibited operating zone of unit i, and nj is the total number of prohibited operating zones in unit i.

D. EDP With Consideration of Ramp Rate Limits

In actual systems, thermal units cannot change their output power instantaneously. A change in unit output power from one specific interval to the next cannot exceed a specified limit, as expressed using the following equations. As the output power increases

$$P_i - P_{i0} \le UR_i. \tag{8}$$

As the output power decreases

$$P_i - P_{i0} \le DR_i. \tag{9}$$

Combining (8) and (9) with (2) enables (2) to be written as

$$\max(P_i^{\min}, P_{i0} - DR_i) \le P_i \le \min(P_i^{\max}, P_{i0} + UR_i) \tag{10}$$

where P_{i0} is the unit output power at the previous interval, and UR_i and DR_i are the up-ramp and down-ramp limits of unit i, respectively.

III. Proposed Approach

The approach presented in this section is proposed primarily for solving the NCEDP in a fully decentralized manner. With some adaptation, the approach can also be applied for solving the EDP with convex cost functions in a fully decentralized manner. It is assumed that each power generation station in the power system has an agent responsible for solving the EDP and that this agent is embedded in the control system of the power station. An undirected ring communication graph is assumed to connect the agents. The use of a ring topology is not a restrictive assumption for the operation of the proposed flooding-based consensus (FBC) algorithm. The advantage of the ring graph is that it provides a simple and fast stopping condition for the proposed flooding algorithm, as explained later in this section. For connecting the agents, any other undirected and connected communication graph topology can be assumed. A general stopping criterion that can be used with any graph topology is based on defining a certain timing condition for each agent. The stopping criterion based on a timing condition works as follows. Once an agent sends its first message, the agent sets a time counter. When the agent receives a new different message, it then resets this time counter to zero. If the time in this time counter reached a predefined certain time, which means that the agent did not receive any new different messages for this time period, the agent will stop and will start the next stage. This predefined time period can be as small as 0.15 s. During this time period, the agent keeps switching the messages normally between its neighbors, but it does not reset the time counter or store these repeated messages in its dynamic matrix if the received message is not a different message compared to the messages it has collected already in its dynamic matrix. In the proposed algorithm, each agent communicates only with its neighbors in the graph.

The proposed approach consists of three stages. In the first stage, the agents reach consensus about the problem data, using a flooding-based consensus (FBC) algorithm. The FBC algorithm is based on the concept of flooding algorithms [7], [8], which are routing algorithms used in data networks for broadcasting messages or data packets between network nodes. Flooding algorithms, whose application in computer networks and ad hoc wireless networks is well known, function in the following way. In the first iteration, each node sends its data to its neighbors. In the following iterations, each node in the network that receives the data then stores a copy and resends them to all of its neighbors except the one from which it received the data. The algorithm stops when all nodes have received and transmitted the data only once. Since a ring communication graph is assumed in this work, and since this graph consists of one loop only, each agent will receive a repeated message only if this message has already been transmitted by all of the other nodes in the network. This condition can be used as a stopping criterion for each agent under the synchronized mode of operation. The synchronized mode of operation means that the agents start the algorithm according to synchronized clocks and keep switching the messages between their neighbors following a certain predefined time step. For stopping the algorithm in the case of asynchronous mode of operation of the proposed FBC algorithm, the timing based stopping condition discussed above can be defined for each agent.

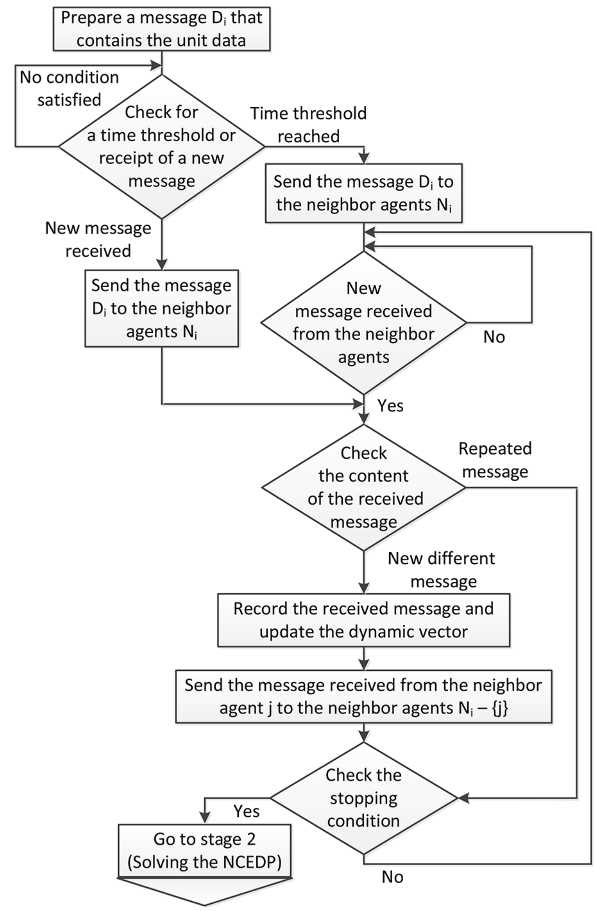


Fig. 1. Stage 1: flooding-based consensus (FBC) algorithm.

When each agent receives a new message that contains unit data, it copies and stores it in a dynamic matrix. Fig. 1 shows a flowchart that clarifies the operation of the FBC algorithm; D_i is the message that contains the data from agent i, and N_i is the set of neighboring agents for agent i.

After the agents have reached consensus with respect to the problem data, the second stage begins, during which each agent solves the EDP locally using a suitable algorithm. For solving the NCEDP, an appropriate metaheuristic technique can be applied during this stage. A flowchart that clarifies the operation of the second stage that involves the metaheuristic technique is shown in Fig. 2. In this figure, nr is the run counter, nrt is the total number of runs, and g is the iteration counter.

Since metaheuristic techniques are stochastic in nature, with each run of a metaheuristic algorithm possibly producing a different solution, the agents will produce different solutions for the problem. To address this issue, a third stage is incorporated to enable the agents to reach consensus about the best solution to the problem. Fig. 3 shows a flowchart that illustrates stage 3, during which each agent prepares a message S_i that contains the best solution the agent has obtained. An example of this message is a tuple $S_i = \langle \mathrm{bus_{nu.}}, x_i \rangle$, where $\mathrm{bus_{nu.}}$ is the bus number; and X_i is a vector defined as $x_i = [\mathrm{Total}\cos t_i, P_1, \ldots, P_n]$,

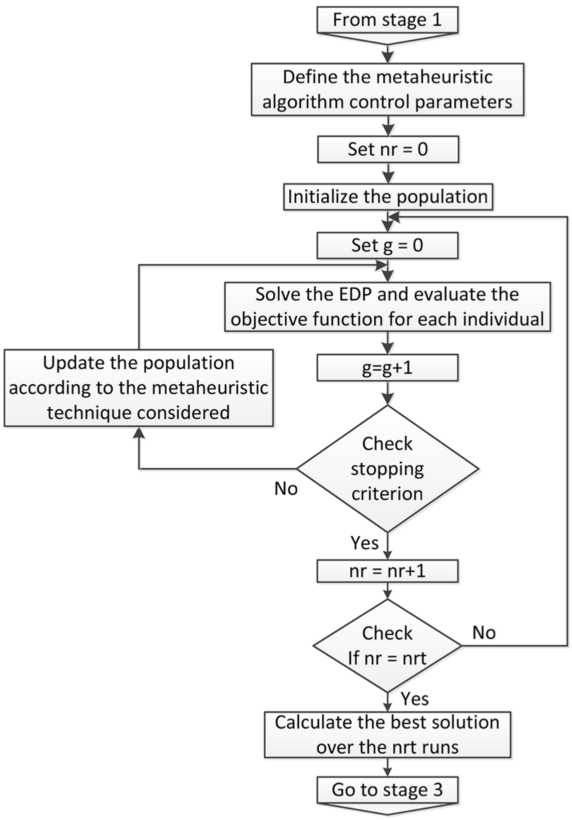


Fig. 2. Stage 2: use of a metaheuristic technique for enabling each agent to solve the NCEDP locally.

where Total cost_i is the best cost computed by agent i, and $\mathrm{P}_1, \ldots, \mathrm{P}_\mathrm{n}$ are the corresponding power levels produced by the generation units. After the agents apply the FBC algorithm so that they can reach agreement about the best solutions, each agent computes a final solution to the problem using the following equations:

$$i^* = \arg\min_{i \in n}(\mathrm{Total}\cos t_i) \tag{11}$$

$$x_i = x_{i^*} \tag{12}$$

where n is the total number of generating units in the network, and i^* is the generator index that provides the best solution to the problem. An example that clarifies the operation of the FBC on the physical level for four-bus network is shown in Fig. 4.

To compute transmission losses in a fully decentralized manner, the loss coefficients in (4) must be computed locally by each agent, which means that each agent must solve the power flow problem locally [9]. This step can be performed if each agent has knowledge of the complete power network data represented in the transmission line data and the bus data, in the following way. It is assumed that each agent knows its own bus data plus the data related to the transmission lines connected to its bus, and each agent then prepares a message that contains a data matrix in which the first row contains the bus number, the bus stamp, and the generator data. The second row contains bus data such as the bus type; whether it is a slack, constant power, or load bus; the bus voltage; and the active and reactive load power. The next rows, one row per line, contain the data for the transmission lines connected to the bus: the resistance, the reactance, and one-half of the total line susceptance. After this message is prepared, the FBC algorithm is applied, stopping when each agent has all the data it needs to solve the power flow problem locally. At this stage, each agent can include power flow equations as constraints while solving the EDP; however, handling power flow equations in a metaheuristic technique demands substantial computational power. For this reason, after the power flow problem has been solved locally, the loss coefficients are computed from the power flow results using the technique from [9]. Kron's loss formula is then applied by each agent in order to calculate the transmission losses for each candidate solution in the metaheuristic algorithm.

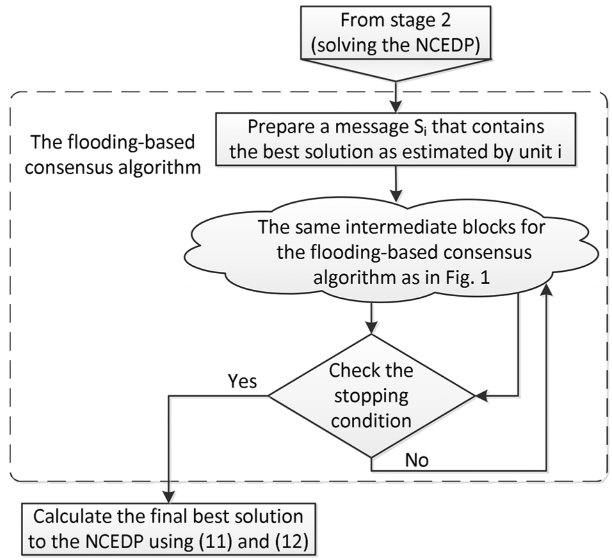


Fig. 3. Stage 3: achieving consensus about the best solution to the NCEDP.

IV. Results and Discussion

A. Modeling Methodology and Assumptions

In this paper, the MATLAB has been used for validating the operation of the proposed approach; however, a different modeling methodology than that used by the previous literature [1]–[5] is used in this paper. This modeling methodology is more suitable for validating the operation of the proposed flooding-based consensus algorithm and hence validating the proposed approach. In this modeling methodology, each agent has been modeled as a separate MATLAB function defined in a separate MATLAB program (m-file). The local data for each agent has been defined only for that agent. Each agent also has its own dynamic matrix. In order to model the capability of each agent to communicate with its neighbors, the names of the MATLAB functions that represent the neighbor agents have been defined to each agent. Only two MATLAB functions have been defined to each agent. Thereafter, in order for an agent to send a message to one of its neighbors, it calls the function that represents this neighbor and passes to it the message which can be a matrix or a vector as an argument. Once an agent receives a message from one of its neighbors, it checks the content of the message through comparing the message content with that

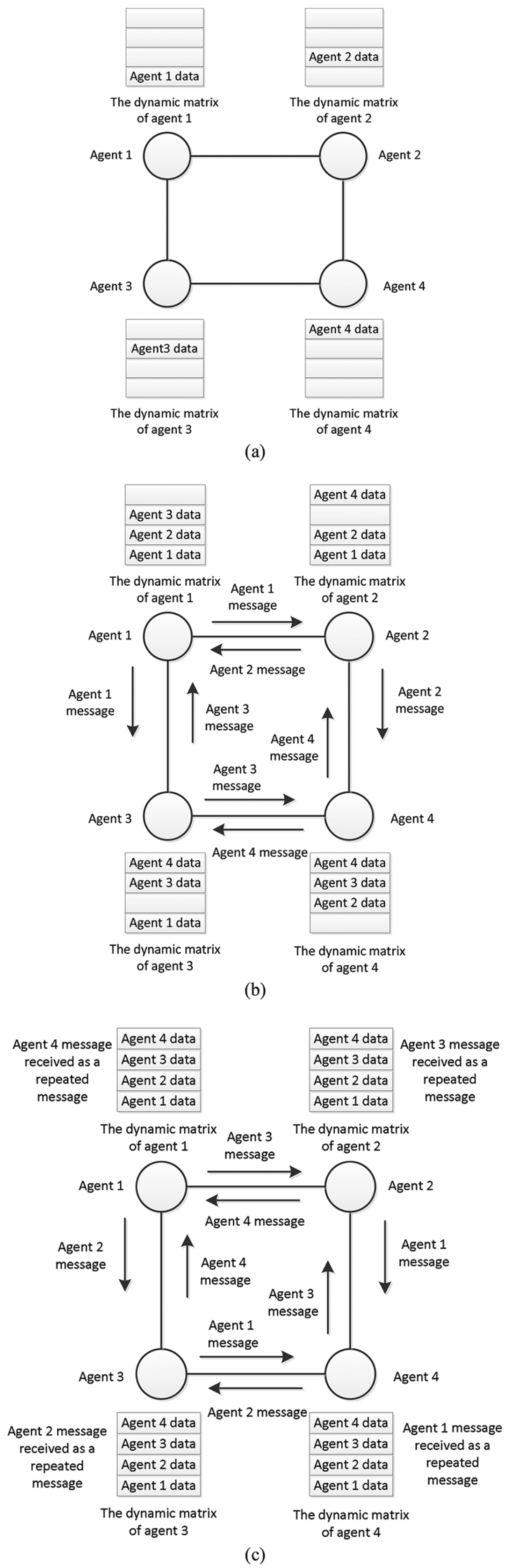


Fig. 4. FBC algorithm operation for 4-bus network. (a) Before the first iteration. (b) First iteration. (c) Second iteration.

of the previously received messages, which are stored in its dynamic matrix. If the received message is a new different message, the agent stores it in the dynamic matrix as a new row or a set of new rows according to the number of rows in the received message. After that, the agent passes the received message to the other neighbor using a bus stamp idea. In this idea, each message contains a bus stamp element that is used as follows. First, when each agent prepares its message, it adds a bus stamp element equal to its bus number. Any agent in the network that receives a message changes the bus stamp element in the message to match its own bus number before relaying the message to its neighbors. The bus stamp enables each agent to determine which neighbor sent the message and to which neighbor it must transmit this message. In multi-agent software such as JADE, each message sent by any agent has the name and address of the sender agent attached to this message. In this case, the bus stamp idea is not needed. The bus stamp idea has been used while modelling the agents in MATLAB. After running the algorithm, the algorithm stops autonomously with each agent has all the system data in its own dynamic matrix. The previously mentioned modeling methodology has been used to validate the operation of stage one and three of the proposed approach. To run the second stage, all the meta-heuristic technique runs have been done sequentially on one computer. After finishing the first stage, one agent is chosen randomly to run the total number of runs of the meta-heuristic technique based on the data it has in its dynamic matrix and which are the same for all the other agents. After the agent finishes the total number of runs, the final solutions of the runs are distributed over all the agents and then the third stage is validated in the same way used for stage 1.

B. Experimentation With JADE and Approximating the Communication Time

JADE software was used as a means of approximating the communication time between agents using the following experimental setup. Two personal computers were located 2.4 km apart. An agent was launched on each computer using the JADE software. The internet was used for the creation of the communication link between the two agents. To determine the time required to send a message between two agents, the time difference between the sending of the message by a specific agent and receipt of a response message by the same agent was recorded. This time is equivalent to the sending of two messages. With the above experimental setup, the average time required for a message to be sent between two agents was found to be 0.046 s, which also includes the time required for the JADE software to prepare for sending a message and to check the content of the received message. In the above experimental setup, one of the computers has a Core i7 (2.4 GHz) processor and 8 GB of RAM. The second computer has a Core i5 (3 GHz) processor and 4 GB of RAM. The messages sent between the agents were containing 1 scalar number during this experiment. When the experiment has been repeated with the same specifications except that instead of sending one scalar number in each message, a matrix containing 64 scalar numbers has been used in each message the average time for sending a message in this case was found to be the same as in the previous experiment (0.046 s). When the same experiment has been repeated between two computers one has a Core i5

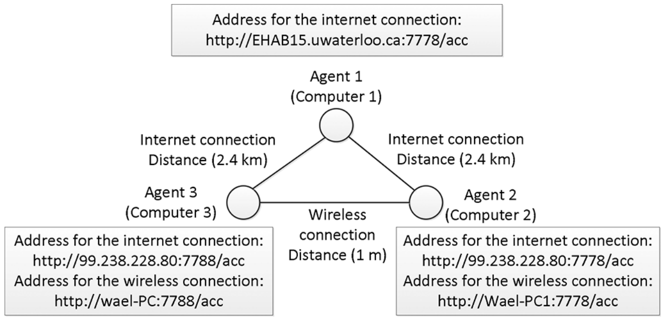


Fig. 5. Schematic diagram of the experimental setup for validating the FBC algorithm.

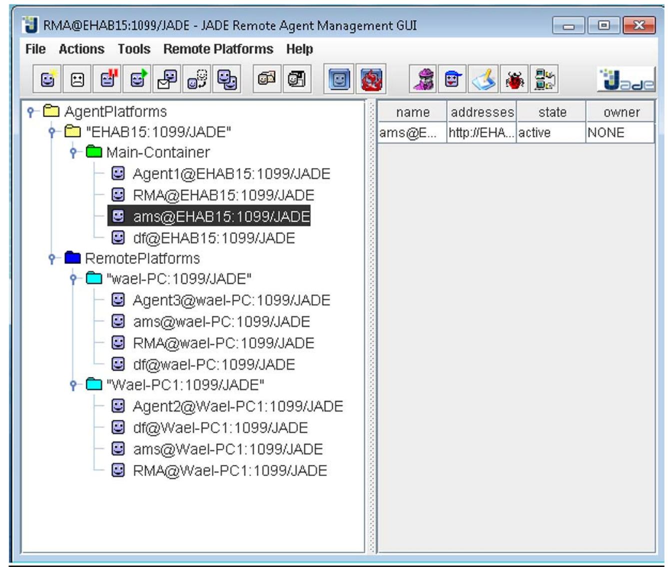


Fig. 6. JADE GUI from computer 1.

processor (3 GHz) and 4 GB of RAM, and the second has a Core2 Duo processor (2.13 GHz) and 3 GB of RAM, the average communication time to send a message between two agents was found to be 0.080 s.

The above experimentation for approximating the communication time of sending a message between two agents has been extended to validate the operation of the FBC algorithm over a ring communication network that connects three different personal computers. The specifications of these computers are as follows:

— Computer 1: Core i5 processor (3 GHz) and 4 GB of RAM
— Computer 2: Core i7 processor (2.4 GHz) and 8 GB of RAM
— Computer 3: Core2 Duo processor (2.13 GHz) and 3 GB of RAM

Due to the differences in the specifications between the computers, the communication time required to send a message over the three communication links is not equal. This creates an unbalanced communication network. A schematic diagram that clarifies this ring communication network is shown in Fig. 5. Fig. 6 shows a screenshot of JADE GUI from computer 1. As shown in Fig. 6, the platform of agent 1 is able to detect the existence of the remote platforms using the internet. In this experiment, The FBC algorithm has been validated under the asynchronous mode of operation. Testing the operation of the FBC algorithm under the asynchronous mode of operation and an unbalanced communication network represents a tough testing condition compared to testing the algorithm under a synchronized mode of operation and a balanced communication network. The algorithm started by agent 1 that sent two messages contain its data to its two neighbor agents. The algorithm has stopped successfully with each agent has the data of the other two agents in addition to its own data. The time measured by each agent is the time difference between the time of sending its first message and the time of detecting the receipt of the first repeated message. Using this experimental setup, the operation of the FBC algorithm has been tested for 10 consecutive times and the minimum, average and maximum of the time recorded by each agent is shown in Table I.

C. Calculating the Time Required by the Proposed Approach

Calculating the time needed for stage 1 and stage 3 is based on the assumption that stage 1 and stage 3 are completed over consecutive iterations under the synchronized mode of operation.

TABLE I
TOTAL COMMUNICATION TIME RECORDED BY EACH NODE

	Minimum time (milli-second)	Average time (milli-second)	Maximum time (milli-second)
Agent 1	187	234	266
Agent 2	119	140.3	167
Agent 3	47	93.5	156

With each iteration, each agent receives two messages from its two neighbors, which also means that each agent sends two messages to those two neighbors per iteration. The total number of iterations required for the information to be shared among the agents can be computed as follows:

$$\text{Number of iterations} = \begin{cases} \frac{n}{2} & \text{if } n \text{ is even} \\ \frac{n+1}{2} & \text{if } n \text{ is odd} \end{cases} \tag{13}$$

where n is the total number of nodes in the system. It is also assumed that the time required per iteration is equivalent to the time required for sending one message between two agents.

The time required for sending a number of consecutive messages that matches the total number of iterations is assumed to be equal to the time required for stage 1. The same methodology was also applied for approximating the time required for stage 3. The total time required for solving the EDP using the proposed approach can be computed from the following formula:

$$\text{total time} = \sum_{i=1}^{4} t_i \tag{14}$$

where t_1 is the time required for stage 1; t_2 is the time required for stage 2, as calculated using (15); t_3 is the time required for stage 3; and t_4 is the time required for solving the power flow problem and computing the loss coefficients, if needed:

$$t_2 = \max_j(time_j) \quad j = 1, 2, \ldots, n \tag{15}$$

where $time_j$ is the total time required for agent j to solve the EDP for a specified number of runs, and n is the total number of generator agents.

D. Differential Evolution Algorithm

To solve the NCEDP locally by each agent, the differential evolution (DE) algorithm, proposed in [10], has been utilized in all the case studies in this paper which involve non-convex cost functions. The main reasons for choosing the DE algorithm are its simplicity of implementation and its satisfactory convergence speed. The results obtained by the differential evolution algorithm with the constraints handling techniques adapted in this paper were found to be satisfactory compared to the results presented in the previous literature. This section aims at reviewing the DE algorithm. The differential evolution is a stochastic search method that works in the general framework of the evolutionary algorithms. The operators used by the differential evolution are the mutation, crossover and selection. The DE algorithm starts by initializing the population vectors within their limits. For each vector in the population X_i, three other vectors $X_{\mathrm{j}}, X_{\mathrm{k}}$, and X_{m} are selected randomly from the population such that $\mathrm{j} \neq \mathrm{k} \neq \mathrm{m}$, and then these three vectors are employed to create a new mutated vector U_i as follows:

$$U_i = X_j + F \cdot (X_k - X_m) \tag{16}$$

where F is a scaling factor. After the previous mutation process, the binomial crossover process is applied for mating the vector U_i and the vector X_i according to the crossover probability C to produce a new offspring Z_i. The crossover operation can be expressed as

$$Z_{ij} = \begin{cases} U_{ij} & \text{if } \mathrm{rand}(j) \le C \\ X_{ij} & \text{otherwise} \end{cases} \tag{17}$$

where Z_{ij}, U_{ij}, and X_{ij} are the jth components of the vectors Z_i, U_i, and X_i, respectively. $rand(j)$ is the jth evaluation of a uniformly distributed random variable between 0 and 1. Finally, the selection process is applied to choose one survivor between the vector X_i and the vector Z_i for the next generation. If the fitness value corresponding to Z_i is higher than that of X_i, then Z_i will replace X_i in the next generation, otherwise, X_i will remain the same for the next generation.

E. Initialization and Handling the Constraints

In order to initialize the population, each component of the individuals is generated randomly as follows:

$$P_i = P_i^{\min} + (P_i^{\max} - P_i^{\min}) \times \mathrm{rand} \tag{18}$$

where rand is a random number between 0 and 1 that obeys the uniform distribution. For handling the inequality constraints such as the prohibited operating zone, the generator limits and the ramp rate limits, the penalty function method has been used. In this method, if any of the generators output power has violated any of the above inequality constraints, the problem solution that contains this output power is penalized with a very large positive constant. For handling the equality constraints, the concept of the dependent source [11] and [12] has been used. In this concept, a dependent unit is chosen randomly, and the power mismatch is added to the output power of this unit. If this dependent unit violates its generation limits due to the additional mismatch added to it, the unit is fixed to its violated limit, and the remaining of the mismatch is compensated from another randomly selected unit. This process continues till the mismatch vanishes. In the case of a system in which the transmission losses are not considered, the mismatch is computed as follows:

$$\mathrm{mismatch} = P_D - \sum_{i=1}^{n} P_i. \tag{19}$$

In the case of considering the transmission losses, the mismatch is computed as follows:

$$\mathrm{mismatch} = P_D + P_L - \sum_{i=1}^{n} P_i. \tag{20}$$

For the DE algorithm, the crossover probability used in this paper is 0.5, and the scaling factor is 0.09.

F. Case Study 1

The system used in this case study consists of 26 buses, six thermal units, and 46 transmission lines. The ramp rate limits, the prohibited operating zones and the transmission losses are taken into account. The transmission line and bus data are as provided in [9], and the generator data are as given in [13]. After the proposed approach was applied, all of the agents reached consensus about the solution shown in Table II. To solve the power flow problem and to compute the loss coefficients, the MATLAB toolbox presented in [9] was used. The method that has been used to solve the power flow problem in this case study is the Newton-Raphson method. A ring communication network connects the 26 buses. It is assumed that the generator buses are responsible for solving the EDP and that each generator agent runs the differential evolution algorithm for three runs, so the total number of runs is thus 18. The number of iterations used by each agent $= 200$, and the population size $= 200$. Table II shows a comparison of the results obtained by the DE algorithm and some of those reported in the literature. The total losses computed in [16]–[19] are lower than those obtained using Kron's loss formula. This discrepancy means that the total power generated as shown in those studies is lower than the actual value, which results in a lower total generation cost. When the prohibited operating zones and the ramp rate limits are not considered, the resultant problem has a convex cost function and one optimal solution. This problem has been solved in [9], and the global optimal solution has a total cost equal to 15 447.72 \$/h. When the prohibited operating zones and the ramp rate limits are considered, the global optimal solution should have a total generation cost either equal to 15 447.72 \$/h or higher than this value but not lower. From Table II, it can be noted that the solution provided by the DE algorithm has a mismatch equal to 0.02 MW, which is due to the rounding up of the power generated by each unit to two decimal places. This mismatch helps produce a slightly lower total cost, which might also be the case in [15], in which the power generation from each unit is also rounded up to two decimal places, and a mismatch of 0.01 MW exists. When the DE algorithm is used without rounding up the output power computed for the units, the best solution found has a total cost of 15 449.89 \$/h, with a mismatch equal to $-4.8e$-10 MW. The loss coefficients used in all of the references cited in Table II are

TABLE II
SIX-GENERATOR TEST SYSTEM: COMPARISON OF THE DE ALGORITHM SOLUTION WITH SOME OF THOSE REPORTED IN THE LITERATURE

Unit power output	aBBOmDE [19]	RDPSO [18]	BBO [17]	SOH-PSO [16]	**DE**	GAAPI [15]	NPSO-LRS [14]	PSO [13]	GA [13]
P_1	447.3944	445.2541	447.3997	438.21	**448.27**	447.12	446.96	447.4970	474.8066
P_2	173.4968	172.7916	173.2392	172.58	**172.96**	173.41	173.3944	173.3221	178.6363
P_3	263.2259	263.5285	263.3163	257.42	**263.44**	264.11	262.3436	263.4745	262.2089
P_4	138.8915	141.0687	138.0006	141.09	**139.3**	138.31	139.512	139.0594	134.2826
P_5	165.1239	163.8578	165.4104	179.37	**165.28**	166.02	164.7089	165.4761	151.9039
P_6	87.2793	88.8558	87.07979	86.88	**86.68**	87.00	89.0162	87.1280	74.1812
Total power output	1275.4121	1275.3565	1275.446	1275.55	**1275.93**	1275.97	1275.94	1276.01	1276.03
Total losses	12.412*	12.3598*	12.446*	12.55*	**12.95**	12.98	12.9361	12.9584	13.0217
Minimum cost ($/hr)	15442.673	15442.7575	15443.096	15446.02	**15449.5826**	15449.7	15450	15450	15459
Mean cost ($/hr)	15442.83	15445.0245	15443.0964	15497.35	**15449.6171**	15449.81	15450.5	15454	15469
Max cost ($/hr)	15442.9930	15455.2936	15443.096	15609.64	**15449.6508**	15449.85	15452	15492	15524

*The total losses computed with Kron's loss formula are greater than these values.

TABLE III
40-GENERATOR TEST SYSTEM: COMPARISON OF THE DE SOLUTION WITH SOME OF THOSE REPORTED IN THE LITERATURE

Method	Min ($/hr)	Average ($/hr)	Max ($/hr)
IFEP [20]	122,624.35	123,382.00	125,740.63
AA [5]		121,788.70	
CPSO–SQP [21]	121,458.54	122,028.16	NA
FCASO-SQP [22]	121,456.98	122,026.21	NA
DE/BBO [23]	121,420.90	121,420.90	121,420.90
aBBOmDE [19]	121,414.87	121,487.85	121,568.32
CE-SQP [24]	121,412.88	121,423.65	NA
DE	**121,412.68**	**121,439.89**	**121,479.63**
FAPSO–VDE [25]	121,412.56	121,412.61	121,412.78
CSA [26]	121,412.54	121,520.41	121,810.25

four decimal places rounded up from the original loss coefficients. When each agent solves the power flow problem and computes the original loss coefficients without rounding up, the solution obtained with the DE algorithm entails a total cost of 15 447.72369 $/h, which is approximately equal to that provided in [9] when a convex cost function is considered. After each agent has computed the loss coefficients, the loss coefficients are rounded up to four decimal places in order to provide a fair comparison with the results from other studies. The total time required for the proposed approach to solve the above problem, as computed using (14) and (15), is 3.634 s, where $t_1 = t_3 = 0.598$ s, $t_2 = 2.064$ s (maximum time for running the DE algorithm for three runs over all of the generator agents), and $t_4 = 0.374$ s.

G. Case Study 2

A 40-unit system was considered in this case study. The data for the system are listed in [20]. In this system, the cost function is non-convex due to the valve-point effect. The number of iterations used by each agent $= 1500$, and the population size $= 500$. Table III shows a comparison of the results obtained using the DE algorithm and those obtained with other algorithms reported in the literature. Total number of agents in this case study is 40.

Each agent runs the metaheuristic technique for 3 runs. The minimum total cost in Table III is the value on which the agents reached consensus.

All of the solutions shown in Table III are centralized except that from [5], which is distributed, and that obtained using the new approach, which is decentralized. The summation of the total power generated with the best solution obtained by the DE algorithm is 10 500 MW, and the mismatch is zero. The total time required for the proposed approach to solve the above problem as computed using (14) and (15) is 31.5037 s, where $t_1 = t_3 = 0.92$ s, $t_2 = 29.6637$ s, and $t_4 = 0$ s.

H. Case Study 3

1) Part A: Most previously proposed decentralized and distributed algorithms for solving the EDP are based on achieving consensus with respect to the lambda variable. One such algorithm is the one proposed in [2], which has been re-simulated for comparison with the approach proposed in this paper. The same test system used in [2], which consists of four units, was also used in this case study. The authors of [2] claimed that their proposed communication graph, which is shown in Fig. 7(a), is sufficient to enable information sharing between the agents and is less restrictive than an undirected communication graph. These points are true; however, if a failure occurs at a specific communication link between two agents, such as link 4-1 as shown in Fig. 7(b), the whole algorithm fails. The approach proposed in this paper has been applied to solve the same problem, and the equal incremental cost method has been used by each agent to solve the problem locally during stage 2. Stage 3 is not needed in this case. The problem has been solved for two cases: one with a loss of one communication link, and the second without this loss. For both cases, the agents reached consensus about the following solution: $\lambda = 8.8397$ $/MWh, $p_1 = 577.3547, p_2 = 577.3547, p_3 = 255.0741$, and $p_4 = 90.2165$. An inspection of the communication graph in Fig. 7(a) reveals that the summation of the out-degree over all of the nodes in this graph is equal to 5, so five messages are assumed to be sent by the nodes for each communication iteration. Each message contains two scalars: the estimated lambda value and the power mismatch estimated by agent i. The total number of scalar values that flow over the network is equal to 10 per iteration. The total number of iterations required for solving the above problem using the consensus on lambda approach is approximately 20. With the algorithm proposed in [2], the total number of messages that flow over the four-bus system is therefore $5 \times 20 = 100$, and the total number of scalar values transferred over the network is $10 \times 20 = 200$. The time required with the approach proposed in [2] is equal to the time required for 20 communication iterations, which is considered equal to the time required for sending 20 consecutive messages between

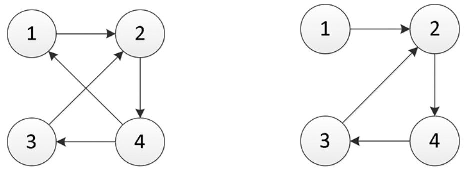


Fig. 7. Communication graph proposed in [2]. (a) Original graph. (b) After the removal of communication link 4-1.

two agents, or 0.92 sec. In contrast, if the above system is assumed for the approach proposed in this paper, each message consists of a tuple that contains 8 scalar values, as follows:

$$\langle \text{Bus number},\ \text{Bus stamp},\ a_i, b_i, c_i, P_{Li}, P_i^{\min}, P_i^{\max} \rangle$$

where P_{Li} is the load at bus i. Each node sends two messages per iteration and receives two messages per iteration. For the four-bus system, the total messages sent per iteration equals $2 \times 4 = 8$, and the total number of iterations, based on (13), is equal to 2. With the proposed algorithm, the total number of messages that flow over the four-bus system is $2 \times 8 = 16$, and the total number of scalar values transferred over the network is $16 \times 8 = 128$. The total time required for the proposed approach to solve the above problem is 0.282 s: the time required for stage 1 is 0.092 s, and the time required for each agent to solve the problem locally using the equal incremental cost method is 0.19 s using a computer with Core2 Duo processor (2.13 GHz) and 3 GB of RAM.

2) Part B: In [4], the authors proposed a technique for incorporating transmission losses while solving the EDP using the consensus on lambda approach. They used a six-bus, three-generator, and 11-line system, the data for which is available in [27]. They also doubled the system load from 210 MW to 420 MW and then applied their proposed algorithm in order to dispatch the system at the 420 MW load. The results obtained by this proposed algorithm was $\mathrm{p}_1 = 127.33$ MW, $\mathrm{p}_2 = 151.48$ MW, and $\mathrm{p}_3 = 148$ MW. According to [4], with the above dispatching, the power mismatch becomes zero, and the generation-demand equality constraint is satisfied; however, summing the output power of the three generators gives a total power generated of 426.81 MW, which means that the total transmission losses are 6.81 MW. When the power flow problem has been solved with the new load values and output generator power, the total transmission losses are found to be 19.199 MW. The proposed approach in this paper has been applied to solve the economic dispatch problem for that system. In stage 2, the penalty factors have been used with the equal incremental cost method for incorporating the transmission losses [9], and there is no need to stage 3. To handle the significant change in load from one dispatching period to the next, the following strategy is applied. Each agent solves the EDP for a total system load of 420 MW, without the inclusion of the transmission losses. The power levels output from the units are then used as the initial values for the power flow problem in order to compute the loss coefficients, following which, each agent uses the equal incremental cost method with the penalty factors to resolve the EDP with the inclusion of the transmission losses. The output results from dispatching the units are shown in Table IV.

TABLE IV
RESULTS FOR THE 3-GENERATOR SYSTEM WITH A CONVEX COST FUNCTION

Output power		Output power		Output power	
P1	139.1997	P2	153.8351	P3	146.2829
Total power 439.3177		Total losses 19.3177		Total cost 5923.91 \$/h	

When the power flow problem is solved with the output power indicated in Table IV, the total transmission losses are found to be 19.2 MW. As with the first case study in [4], this case study does not consider the generation limits.

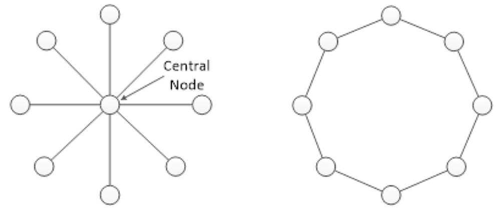


Fig. 8. Case study 3, part c. (a) Star topology for a centralized system. (b) Ring topology for a decentralized system.

3) Part C: This section provides a comparison of the new decentralized approach and the centralized approach with respect to solving the EDP from a communication perspective. Although more data flow over the communication network than in a centralized system, the decentralized approach has the advantage of no single system node is subject to a communication bottleneck or increased management complexity relative to the other nodes. In centralized systems, if one node is assumed to be distinct from the other system nodes so that it provides central management for the system, as shown in Fig. 8(a), and if it is assumed that the central node needs to collect data from the other system nodes, the central node then solves the EDP and sends the results back to the other system nodes.

If the total number of nodes excluding the central node is n, then the total number of the messages communicated for the centralized system is 2n. All of these messages are either received or sent by the central node for the performance of one task such as solving the EDP. As well, n communication links end at the central node. On the other hand, if a fully decentralized operation with a ring communication graph is assumed, as shown in Fig. 8(b), each node has only two communication channels connected to it whatever the size of the system. Based on the assumption, as stipulated in the proposed decentralized approach, that each node needs to collect system data in order to process these data locally and produce a decision with the same efficiency as that in the centralized system, the number of messages then received by each node in the system equals two per iteration. The total number of iterations required for sharing the information between the agents is proportional to the system size and can be computed according to (13). The time required by the proposed decentralized approach to collect the data is very small, and is comparable with the time required by the central node in the centralized system to collect the data. The time required by the proposed decentralized approach to collect the data varies from few fractions of seconds in small systems to approximately 0.92 s in the 40-unit system.

V. Practical Considerations, Limitations, and Future Work

This section discusses possible limitations associated with the practical application of the proposed approach in real-life systems and suggests future research that could be expected to address these limitations. During the process of tuning the DE algorithm, it was noted that the DE algorithm control parameters are sensitive to the formulation and dimensions of the problem. If the DE algorithm has been used to solve other problems with different dimensions based on the same parameter settings described in this paper, it may therefore not provide a high-quality or satisfactory solution. This problem is a general problem associated with many metaheuristic techniques and with any centralized, decentralized, or distributed approach that utilizes such techniques. This problem can be addressed through the use of a more efficient metaheuristic technique that either relies on control parameters less sensitive to the problem formulation and dimensions or incorporates a self-tuning feature. Examples of already-existing self-tuning DE algorithms are those reported in [25] and [28].

Other challenges that may be associated with the practical implementation of the proposed approach are related to cyber-attacks. Sharing system and unit data between system nodes is not an issue because these data can be encrypted if necessary during their transfer between system nodes; however, injecting false data that will misdirect the system operation may be considered problematic. This drawback is associated with any distributed and decentralized approach, including the consensus on lambda approach, and can occur in any system that employs a communication layer for sharing data among system nodes because unauthorized individuals are able to access these data or communication networks. For this reason, although this research has included the successful use of the internet for creating a communication link between two nodes at different locations, it is expected that a practical implementation of the proposed approach would require a special dedicated communication network for connecting the system nodes, so that use of the internet could be avoided.

VI. Conclusion

This paper has presented a fully decentralized approach for solving the economic dispatch problem. Compared to previously proposed distributed and decentralized approaches, the new approach offers the ability to solve the non-convex economic dispatch problem and to incorporate the transmission losses accurately. With respect to solving the economic dispatch problem with convex cost functions, the proposed approach is competitive with the consensus on lambda technique. The proposed approach can also handle power flow constraints, which enables it to be extended for solving the security-constrained economic dispatch and the optimal power flow problems. In the proposed approach, a flooding-based consensus algorithm is applied as a means of sharing information among agents. An experimental setup has been employed for approximating the communication time required with the proposed approach. The three case studies described have been examined with the goals of validating the operation of the proposed approach and of providing a comparison with previously proposed approaches. The results indicate that the proposed approach offers significant advantages.

References

[1] V. Loia and A. Vaccaro, “Decentralized economic dispatch in smart grids by self-organizing dynamic agents,” *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 44, no. 4, pp. 397–408, 2014.

[2] 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.

[3] Z. Zhang and M.-Y. Chow, “Convergence analysis of the incremental cost consensus algorithm under different communication network topologies in a smart grid,” *IEEE Trans. Power Syst.*, vol. 27, no. 4, pp. 1761–1768, Nov. 2012.

[4] G. Binetti, A. Davoudi, F. L. Lewis, D. Naso, and B. Turchiano, “Distributed consensus-based economic dispatch with transmission losses,” *IEEE Trans. Power Syst.*, vol. 29, no. 4, pp. 1711–1720, Jul. 2014.

[5] G. Binetti, A. Davoudi, D. Naso, B. Turchiano, and F. Lewis, “A distributed auction-based algorithm for the non-convex economic dispatch problem,” *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1124–1132, 2014.

[6] D. C. Walters and G. B. Sheble, “Genetic algorithm solution of economic dispatch with valve point loading,” *IEEE Trans. Power Syst.*, vol. 8, no. 3, pp. 1325–1332, Aug. 1993.

[7] V. Saligrama and M. Alanyali, “A token-based approach for distributed computation in sensor networks,” *IEEE J. Select. Topics Signal Process.*, vol. 5, no. 4, pp. 817–832, Aug. 2011.

[8] D. P. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, NJ, USA: Prentice Hall, 1992.

[9] H. Saadat, *Power System Analysis*, 2nd ed. New York, NY, USA: McGraw-Hill, 2002, pp. 257–289.

[10] R. Storn and K. Price, “Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces,” *J. Global Optimiz.*, vol. 11, pp. 341–359, 1997.

[11] T. Niknam and F. Golestaneh, “Enhanced bee swarm optimization algorithm for dynamic economic dispatch,” *IEEE Syst. J.*, vol. 7, no. 4, pp. 754–762, 2013.

[12] X. Yuan, L. Wang, Y. Yuan, Y. Zhang, B. Cao, and B. Yang, “A modified differential evolution approach for dynamic economic dispatch with valve-point effects,” *Energy Convers. Manage.*, vol. 49, no. 12, pp. 3447–3453, Dec. 2008.

[13] Z. Gaing, “Particle swarm optimization to solving the economic dispatch considering the generator constraints,” *IEEE Trans. Power Syst.*, vol. 18, no. 3, pp. 1187–1195, Aug. 2003.

[14] I. Selvakumar and K. Thanushkodi, “A new particle swarm optimization solution to non-convex economic dispatch problems,” *IEEE Trans. Power Syst.*, vol. 22, no. 1, pp. 42–51, Feb. 2007.

[15] I. Ciornei and E. Kyriakide, “A GA-API solution for the economic dispatch of generation in power system operation,” *IEEE Trans. Power Syst.*, vol. 27, no. 1, pp. 233–242, Feb. 2012.

[16] K. T. Chaturvedi, M. Pandit, and L. Srivastava, “Self-organizing hierarchical particle swarm optimization for non-convex economic dispatch,” *IEEE Trans. Power Syst.*, vol. 23, no. 3, pp. 1079–1087, Aug. 2008.

[17] A. Bhattacharya and P. K. Chattopadhyay, “Biogeography-based optimization for different economic load dispatch problems,” *IEEE Trans. Power Syst.*, vol. 25, no. 2, pp. 1064–1077, May 2010.

[18] J. Sun, V. Palade, X.-J. Wu, W. Fang, and Z. Wang, “Solving the power economic dispatch problem with generator constraints by random drift particle swarm optimization,” *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 222–232, Feb. 2014.

[19] M. R. Lohokare, B. K. Panigrahi, S. S. Pattnaik, S. Devi, and A. Mohapatra, “Neighborhood search-driven accelerated biogeography-based optimization for optimal load dispatch,” *IEEE Trans. Syst., Man, Cybern. C, Applicat. Rev.*, vol. 42, no. 5, pp. 641–652, 2012.

[20] N. Sinha, R. Chakrabarti, and P. K. Chattopadhyay, “Evolutionary programming techniques for economic load dispatch,” *IEEE Trans. Evol. Computat.*, vol. 7, no. 1, pp. 83–94, Feb. 2003.

[21] J. Cai, Q. Li, L. Li, H. Peng, and Y. Yang, "A hybrid CPSO-SQP method for economic dispatch considering the valve-point effects," *Energy Convers. Manage.*, vol. 53, no. 1, pp. 175–181, 2012.

[22] J. Cai, Q. Li, L. Li, H. Peng, and Y. Yang, "A hybrid FCASO-SQP method for solving the economic dispatch problems with valve-point effects," *Energy*, vol. 38, no. 1, pp. 346–353, Feb. 2012.

[23] A. Bhattacharya and P. K. Chattopadhyay, "Hybrid differential evolution with biogeography-based optimization for solution of economic load dispatch," *IEEE Trans. Power Syst.*, vol. 25, no. 4, pp. 1955–1964, Nov. 2010.

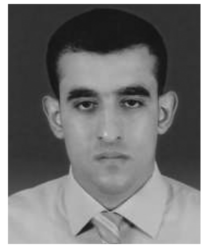
[24] M. S. P. Subathra, S. E. Selvan, T. A. A. Victoire, A. H. Christinal, and U. Amato, "A hybrid with cross-entropy method and sequential quadratic programming to solve economic load dispatch problem," *IEEE Trans. Power Syst.*, to be published.

[25] T. Niknam, H. D. Mojarrad, and H. Z. Meymand, "A novel hybrid particle swarm optimization for economic dispatch with valve-point loading effects," *Energy Convers. Manage.*, vol. 52, no. 4, pp. 1800–1809, 2012.

[26] D. N. Vo, P. Schegner, and W. Ongsakul, "Cuckoo search algorithm for non-convex economic dispatch," *IET Gen., Transm., Distrib.*, vol. 7, no. 6, pp. 645–654, 2013.

[27] A. J. Wood and B. F. Wollenberg*, Power Generation, Operation, and Control*. New York, NY, USA: Wiley, 1996.

[28] Y. Lu, J. Zhou, H. Qin, Y. Wang, and Y. Zhang, "Chaotic differential evolution methods for dynamic economic dispatch with valve-point effects," *Eng. Applicat. Artif. Intell.*, vol. 24, no. 12, pp. 378–387, Feb. 2011.



Wael T. Elsayed (GS'13) was born in Qaluobia, Egypt, in 1985. He received the B.Sc. (with honors) and M.Sc. degrees from the Shoubra Faculty of Engineering, Benha University, Egypt, in 2007 and 2010, respectively, and the M.Sc. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in August 2014.

In 2007, using a point-based evaluation criterion, he was ranked first in the Faculty of Engineering at Shoubra among 1600 students who graduated in that year and was honored by the President of Egypt at a party held by the Supreme Council of the Egyptian Universities. From 2007 to 2012, he was a research/teaching assistant in the Faculty of Engineering, Benha University. His current research interests are the economic dispatch problem and applications of decentralized and distributed algorithms in power systems.

Ehab F. El-Saadany (SM'05) was born in Cairo, Egypt, in 1964. He received the B.Sc. and M.Sc. degrees in electrical engineering from Ain Shams University, Cairo, Egypt, in 1986 and 1990, respectively, and the Ph.D. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 1998.

Currently, he is a Professor in the Department of Electrical and Computer Engineering, University of Waterloo. His research interests include smart grid operation and control, power quality, distributed generation, power electronics, digital signal processing applications for power systems, and mechatronics.