Journal of Industrial and Management Optimization

https://www.aimspress.com/journal/jimo

JIMO, 22(6): 3050–3087.
DOI:10.3934/jimo.2026112
Received: 09 February 2026
Revised: 08 May 2026
Accepted: 20 May 2026
Published: 28 May 2026

Research article

A fully asynchronous distributed economic dispatch for electricity–heat energy management systems

Xiaolan Yuan[1], **Xiang Wu**[1,2,*] **and Haozheng Meng**[1]

[1] School of Mathematical Sciences, Guizhou Normal University, Guiyang, 550025, China
[2] School of Automation, Southeast University, Nanjing, 210096, China

* **Correspondence:** Email: xwu@gznu.edu.cn

Abstract: Distributed economic dispatch (DED) methods currently suffer from an over-reliance on synchronous iteration and ideal communication assumptions, which introduce a significant waiting overhead for heterogeneous devices with inconsistent updates and communication delays. Accordingly, this work proposes a fully asynchronous DED algorithm tailored for electricity–heat energy management systems (EH-EMS). Specifically, the EH-EMS model is first established under coupled operational constraints and imbalanced communication networks. Then an optimal response function is formulated to improve computational efficiency while the output constraints remain satisfied. On this basis, an asynchronous distributed dual-consensus (ADDC) algorithm is designed. It allows each device to iterate independently at its own pace, thereby eliminating the waiting time inherent in traditional synchronous schemes. Notably, the ADDC algorithm retains the same variable scale as synchronous DED methods, which further reduces the computational cost in practical applications. In addition, the convergence conditions and exact convergence speed of the ADDC algorithm are provided under the bounded delay and asynchronous update framework. Finally, the simulation results on the EH-EMS 65 test system show that the ADDC algorithm outperforms the compared asynchronous methods in terms of efficiency and performance.

Keywords: communication delay; asynchronous update; distributed algorithm; energy management system; economic dispatch
Mathematics Subject Classification: 90C90, 90B35, 68W15

1. Introduction

An integrated energy management system (IEMS) is an intelligent control system for various energy vectors, such as heat, electricity, gas, and energy storage. By coordinating multienergy optimization, it improves efficiency and economic welfare, but also introduces complex scheduling

and coupled constraints. The economic dispatch problem (EDP) coordinates energy production and consumption. Its core value is to optimize each unit's output to minimize costs, while meeting strict safety and stability constraints. Initially, the EDP is managed by centralized economic dispatch (CED) [1, 2]. However, CED often struggles with poor plug-and-play performance, risky data privacy, and heavy computational burdens at the central hub. As a result, distributed economic dispatch (DED), where agents are optimized using local data and achieve global coordination through local networks, has become the preferred alternative.

In recent decades, many DED algorithms have been applied to IEMSs and have achieved good performance. Among these, a significant number fall into the category of fully synchronous algorithms [1]. Specifically, this class of methods requires all devices to complete local computations within the same iteration cycle and synchronously exchange the latest decision variables. Leader–follower DED is one of the earliest fully synchronized methods [3]. This method selects a subset of nodes as leaders to collect the supply–demand mismatch. Then the incremental cost is treated as a consistency variable to achieve dispatch optimization. However, this structure requires the leaders to obtain a global supply–demand mismatch, which somewhat deviates from the original intent of distributed design. Motivated by this, [4] proposed a distributed gradient descent algorithm with a time-varying decreasing step size. However, a decreasing step sizegreatly slows convergence, although it simplifies algorithm construction. Subsequently, attention has been paid to DED algorithms with constant step sizes. Common strategies include incremental cost consensus methods [5,6], consensus + innovation methods [7–10], and approaches based on the alternating direction method of multipliers (ADMM) [11–13]. Among them, the consensus + innovation method is a typical application of distributed optimal consensus algorithms in the EDP. In the innovation component, it employs a gradient tracking mechanism to locally estimate the degree of supply–demand mismatch. In the consensus component, it uses a consensus protocol to coordinate the incremental costs of each device. The consensus+innovation technique adapts device output by incorporating estimated innovation value as feedback into the consensus part. Despite their remarkable performance, fully synchronous DED algorithms are based on a critical assumption: All devices update simultaneously, and the communication links are ideal. When devices are heterogeneous or suffer from communication delays, such assumptions become particularly restrictive and lead to significant waiting overhead.

In response to these limitations, researchers have developed weakly asynchronous DED algorithms. This type of method can utilize outdated information or inconsistent updates from certain devices. In [14–16], the DED algorithms achieve robustness against communication delays by using outdated consensus variables to replace real-time variables during synchronous iterative updates. The dgeneration-sideerence is that [14] only qualitatively discusses the impact of time delay on the algorithm, [15] proposes a convergence upper bound for constant delay, and [16] provides an accurate convergence proof for algorithms with random time but bounded delay. References [17–19] further allow for random disconnection of the communication links or permanent loss of data packets. In addition, event-triggered mechanisms have been widely studied due to their advantage of low resource consumption [20, 21]. References [22–24] integrate these mechanisms into DED algorithms, allowing devices to communicate only at specific trigger instants and rely on outdated information at other times. This practice improves communication efficiency and delay tolerance. However, event-triggered DED algorithms remain constrained by uniform computation intervals and synchronous activation times, thus implicitly relying on a global clock. Prominent examples of inconsistent device updates are found in [25] and [26]. Tailored to smart grid applications, these two studies propose two asynchronous DEDs based on the

gossip mechanism. In these asynchronous schemes, only a single pair of devices is updated during each iteration, while the other devices remain in a prolonged idle state until activated. Thus, it is evident that weakly asynchronous methods exhibit some robustness to communication delays and asynchronous behavior. Nevertheless, these methods remain dependent on specific synchronization points or a global clock. This underlying assumption makes it dgeneration-sideicult for all the abovementioned methods to address the issue of significant dgeneration-sideerences in iteration time.

To further overcome the constraints imposed by weak asynchrony, the academic world has turned its attention to fully asynchronous DED. Fully asynchronous algorithms must satisfy both of the following conditions simultaneously: (1) The devices can use outdated information; and (2) all devices iterate independently without synchronization waiting points. By integrating asynchronous iteration with various key technologies, [27–32] significantly reduced synchronous waiting time and achuieved fast convergence under substantial device computation time disparities. Specifically, [27, 28] used the ADMM scheme, [29] integrated the consensus + innovation approach, [30, 31] utilized the dual decomposition method, and [32] applied the augmented Lagrangian method. It is noted that the algorithm architecture in [29] is the closest to this work. Based on the asynchronous processing mechanism of [33], an asynchronous DED algorithm is designed for power systems. This algorithm allows each device to perform autonomous asynchronous updates according to its computational capability. A momentum acceleration mechanism is also introduced to improve the convergence speed of the algorithm. Dgeneration-sideerent from [27–31], only the work in [32] focuses on an IEMS as we do. Its activation mechanism is determined by two random probability numbers, which represent the subsystem and the communication link, respectively. As a result, it withstands link failures, packet loss, and communication delays. The authors proved that the algorithm converges to the optimal solution with probability one, provided that the mean traversal of the random communication network and the expected positive update of coupled variables hold. Several critical drawbacks still plague the aforementioned methods, despite their substantial progress in removing synchronization bottlenecks. Regarding communication graphs, both [29] and [32] required them to be balanced. In terms of algorithm convergence, [32] only did probabilistic convergence, while [29] does not clarify the convergence speed. Regarding constraints, all approaches fail to account for the coupled effects among dgeneration-sideerent energies. Additionally, the asynchronous algorithm in [29, 33] requires three auxiliary variables per decision variable, increasing the computational burden for practical implementation.

In our manuscript, a structurally simple and fully asynchronous DED is proposed for an integrated electricity–heat energy management system (EH-EMS). It allows devices to update asynchronously despite the presence of system delays, thus improving the overall efficiency of the solution. The main innovations are as follows.

- This study considers an EDP for an EH-EMS with coupled constraints and an unbalanced communication network, which improves the model's realism and practical applicability.
- The asynchronous distributed dual-consensus (ADDC) algorithm is designed, enabling each device to update independently at its own pace. Compared with synchronous and weakly asynchronous DED, it eliminates the need for a global synchronized clock and reduces the extensive waiting time.
- Compared with [29] and [33], which involve multiple intermediate steps and introduce two additional auxiliary variables to handle asynchronicity, the ADDC algorithm is simple to implement and involves the same number of decision variables as the synchronous algorithm. This

minimalist iterative design significantly reduces computational overhead and memory usage in actual deployments.

- By assigning two classes of virtual agents to each physical agent, independent delayed information forwarding mechanisms are constructed for the consensus and tracking variables, thereby reformulating the ADDC algorithm into an augmented synchronous algorithm. Subsequently, the exact convergence rate and convergence conditions of the ADDC algorithm are strictly proven under the setting of bounded delays and asynchronous updates.
- Validation is conducted on an EH-EMS test model comprising a 32-node Bali thermal network, an IEEE 33-bus power network, and two combined heat and power units. Numerical results demonstrate that the efficiency of the ADDC algorithm surpasses that of other asynchronous methods.

This paper proceeds as follows. Section 2 introduces the EDP of the EH-EMS. Section 3 carries out the preprocessing operations of the EDP. Section 4 details the design framework of the ADDC algorithm. In Section 5, a rigorous theoretical analysis is supplied to verify the convergence property. Numerical simulations and a corresponding performance analysis are carried out in Section 6. Finally, Section 7 summarizes the entire text and outlines the direction of future research.

Notation: The symbol $\lfloor\cdot\rfloor$ denotes the floor function. $\boldsymbol{I}$ is the identity matrix. $\mathbf{Z}^\top$ means the transpose of vector $\mathbf{Z}$. $\mathbf{1}$ and $\mathbf{0}$ are column vectors consisting entirely of 1s and 0s, respectively. Vector norms include the Euclidean norm $\|\cdot\|$, the Manhattan norm $\|\cdot\|_1$, and the $\boldsymbol{Q}$-weighted norm $\|\cdot\|_{\boldsymbol{Q}}$. $\boldsymbol{Q}_{rm}$ denotes the entry in the r-th row and m-th column of matrix $\boldsymbol{Q}$. $|\boldsymbol{Q}|$ is the determinant of matrix $\boldsymbol{Q}$. In set-valued analysis, $\mathbb{N}$ represents the set of natural numbers. $|\Omega|$ and $N_\Omega(x)$ stand for the cardinality of set Ω and the normal cone to the set Ω at point x, respectively. For a communication graph, an edge (n, j) means that agent n can unidirectionally send messages to agent j. $\mathcal{N}_n^+ = \{j \mid (j, n)\}$ and $\mathcal{N}_n^- = \{j \mid (n, j)\}$ represent the in-neighbor set and out-neighbor set of agent n, respectively. Finally, two scaled error bounds are defined as: $\delta^{\varrho,N} = \max\limits_{\ell=0,1,\ldots,N} \dfrac{\delta(\ell)}{\varrho^\ell}$ for a finite horizon and $\delta^{\varrho} = \sup\limits_{\ell\in\mathbb{N}} \dfrac{\delta(\ell)}{\varrho^\ell}$ for the infinite case.

2. Problem description

Consider an EH-EMS system comprising various energy supply devices and loads. Depending on the energy type, the devices can be classified into power-only devices (PODs), heat-only devices (HODs), and combined heat and power devices (CHPs). CHPs serve as a crucial link between the power grid and the heat network, reflecting the multienergy coupling nature of the system. Depending on their regulation characteristics, loads can be classified into fixed electrical loads, fixed thermal loads, flexible electrical loads (FELs), and flexible thermal loads (FTLs). Fixed loads maintain constant electrical and thermal demands during specific periods, while flexible loads can adjust their power output flexibly according to operational requirements. Flexible loads help reduce the required energy storage capacity and mitigate peak-to-trough load dgeneration-sideerences by participating in system scheduling. Furthermore, all types of devices and flexible loads are equipped with local controllers that can collect local data and update their own status through an unbalanced communication network. Figure 1 depicts a simplified EH-EMS architecture.

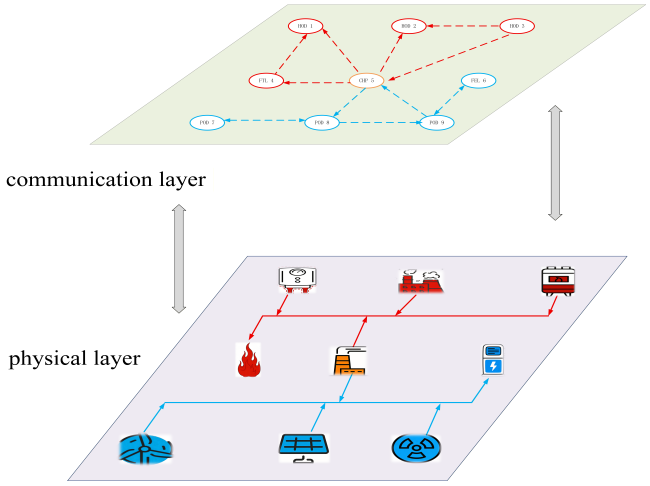


Figure 1. The architecture of an EH-EMS.

The EDP of the EH-EMS involves optimizing the power output to minimize the total operating costs while satisfying load demand and operational constraints. It is noteworthy that this goal is not a single reduction in production costs but a multiobjective equilibrium between generation-side costs and user-side benefits. A singular focus on production costs would overlook consumer gains and ultimately preclude the achievement of minimized total operating costs. The EDP of an EH-EMS can be given by

$$\begin{aligned} \mathrm{P}: \min \quad & G^{total} = G - H \\ s.t. \quad & \Delta X = 0, \qquad \Delta Y = 0, \\ & x_i \in \Omega_i, \qquad i \in I_p \cup I_e, \\ & y_i \in \Omega_i, \qquad i \in I_h \cup I_t, \\ & (x_i, y_i) \in \Omega_i, \qquad i \in I_c, \end{aligned} \tag{2.1}$$

where x_i and y_i are electrical and thermal power, respectively; Ω_i represents the inequality constraints for each device; ΔX and ΔY are the supply–demand constraints; I_p, I_h, I_c, I_e, and I_t are the index sets of PODs, HODs, CHPs, FELs, and FTLs, respectively; G^{total} is the total operating costs, G and H are generation-side costs and user-side benefits, expressed as

$$\begin{aligned} H &= \sum_{i \in I_e} H_i^e(x_i) + \sum_{i \in I_t} H_i^t(y_i), \\ G &= \sum_{i \in I_p} G_i^p(x_i) + \sum_{i \in I_h} G_i^h(y_i) + \sum_{i \in I_c} G_i^c(x_i, y_i). \end{aligned}$$

The specific local cost function and local constraint are given as follows.

Cost function. Since the PODs and the HODs are decoupled in terms of energy output, we develop independent operating costs for PODs and HODs, and simplify the cost functions to quadratic forms with respect to a single output variable. The general expressions are given by

$$\begin{aligned} G_i^p(x_i) &= \alpha_i (x_i)^2 + \beta_i x_i + \gamma_i, \quad i \in I_p, \\ G_i^h(y_i) &= a_i (y_i)^2 + b_i y_i + c_i, \quad i \in I_h. \end{aligned} \tag{2.2}$$

Note that $\alpha_i(a_i), \beta_i(b_i)$, and $\gamma_i(c_i)$ are all positive real numbers.

In a CHP, electricity and heat are both converted from the same energy, so a cross-term needs to be introduced to quantify this coupled relationship. The cost function is designed as

$$G_i^c(x_i, y_i) = \alpha_i(x_i)^2 + \beta_i x_i + \varpi_i x_i y_i + a_i(y_i)^2 + b_i y_i + \gamma_i, i \in I_c. \tag{2.3}$$

To ensure the convexity of the cost function, unless otherwise stated, the conditions $2\alpha_i > \varpi_i > 0$ and $2a_i > \varpi_i > 0$ are always satisfied.

Flexible loads typically have diminishing marginal utility in terms of user satisfaction, and increasing consumption after meeting the demand may not necessarily bring more benefits. Therefore, we introduce a piecewise and concave function to represent the benefits of flexible load. For each FEL $i, i \in I_e$, its benefit function is

$$H_i^e(x_i) = \begin{cases} -\alpha_i(x_i)^2 + \beta_i x_i & \text{if } x_i \le \frac{\beta_i}{2\alpha_i}, \\ \frac{(\beta_i)^2}{4\alpha_i} & \text{if } x_i > \frac{\beta_i}{2\alpha_i}. \end{cases} \tag{2.4}$$

For each FTL $i, i \in I_t$, its benefit function is

$$H_i^t(y_i) = \begin{cases} -a_i(y_i)^2 + b_i y_i & \text{if } y_i \le \frac{b_i}{2a_i}, \\ \frac{(b_i)^2}{4a_i} & \text{if } y_i > \frac{b_i}{2a_i}. \end{cases} \tag{2.5}$$

In (2.4) and (2.5), $\alpha_i(a_i)$ and $\beta_i(b_i)$ are the benefit parameters of the load unit, both of which are positive real numbers.

Constraint restriction. For the POD, HOD, FEL, and FTL, due to physical capacity limitations, their power must be within fixed upper and lower limits. That is

$$\begin{aligned} x_i^{min} \le x_i \le x_i^{max}, \quad i \in I_p \cup I_e, \\ y_i^{min} \le y_i \le y_i^{max}, \quad i \in I_h \cup I_t. \end{aligned} \tag{2.6}$$

Considering the upper saturation limit of flexible loads, it is assumed that

$$\begin{aligned} x_i^{min} < \frac{\beta_i}{2\alpha_i} < x_i^{max}, \quad i \in I_e, \\ y_i^{min} < \frac{b_i}{2a_i} < y_i^{max}, \quad i \in I_t. \end{aligned} \tag{2.7}$$

The CHP is an extraction condensing type that can adjust the extraction flow rate to regulate the electrical and thermal output within a certain range. Its feasible operating region can be characterized by four inequality constraints

$$\boldsymbol{A}_i \cdot \begin{bmatrix} x_i \\ y_i \end{bmatrix} \le \boldsymbol{B}_i, \quad i \in I_c, \tag{2.8}$$

where $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$ are 4×2- and 4×1- dimensional matrices, respectively.

Furthermore, the total power generated by the EH-EMS needs to match the user's electricity and heat consumption, thereby avoiding energy waste or power outages. The mathematical expression is

$$\begin{aligned} \Delta X = D_x - \sum_{i \in I_p \cup I_c} x_i + \sum_{i \in I_e} x_i = 0, \\ \Delta Y = D_y - \sum_{i \in I_h \cup I_c} y_i + \sum_{i \in I_t} y_i = 0, \end{aligned} \tag{2.9}$$

where D_x and D_y represent the fixed electrical and thermal loads of EH-EMS.

3. Problem preprocessing

In this section, an optimal response function is proposed by performing projection mapping on the incremental cost, thereby addressing the constraint limitations of problem P. On this basis, a distributed architecture is introduced to obtain the optimal distributed optimality conditions for the EH-EMS, which guides the design of subsequent distributed algorithms.

Let λ_x and λ_y denote the Lagrange multipliers, also known as incremental costs, associated with the constraints ΔX and ΔY. The Lagrangian function for problem P is constructed as

$$\Xi = G - H + \lambda_x \Delta X + \lambda_y \Delta Y.$$

From Karush–Kuhn–Tucker conditions in [34], the scalars $x_i^\star$ and $y_i^\star$ are optimal solutions to problem P if and only if $\lambda_x^\star$ and $\lambda_y^\star$ satisfying the following equations

$$-\frac{\partial G}{\partial x_i}(x_i^\star, y_i^\star) + \lambda_x^\star \in N_{\Omega_i}(x_i^\star, y_i^\star), \quad i \in I_p \cup I_c, \tag{3.1a}$$

$$-\frac{\partial G}{\partial y_i}(x_i^\star, y_i^\star) + \lambda_y^\star \in N_{\Omega_i}(x_i^\star, y_i^\star), \quad i \in I_h \cup I_c, \tag{3.1b}$$

$$\frac{\partial H}{\partial x_i}(x_i^\star, y_i^\star) - \lambda_x^\star \in N_{\Omega_i}(x_i^\star, y_i^\star), \quad i \in I_e, \tag{3.1c}$$

$$\frac{\partial H}{\partial y_i}(x_i^\star, y_i^\star) - \lambda_y^\star \in N_{\Omega_i}(x_i^\star, y_i^\star), \quad i \in I_t, \tag{3.1d}$$

$$\Delta X(x_i^\star) = 0, \qquad \Delta Y(y_i^\star) = 0. \tag{3.1e}$$

To circumvent the computational complexity arising from the optimality condition in (3.1), an optimal response function is proposed to transform it into an explicit projection mapping on the incremental cost. For uncoupled units such as the POD, HOD, FEL, and FTL, the derivatives of their respective cost functions are first derived. These derivatives are then set equal to the corresponding incremental costs to obtain the unconstrained optimal solution. Subsequently, this unconstrained solution is projected onto the feasible region to obtain the final solution. For coupled units like the CHP, the unconstrained solution is obtained in the same way. However, its cost function contains coupling terms, and the feasible region is a polygon. Thus, the final step is no longer a Euclidean projection but a weighted norm projection.

The optimal response function expression for each type of device is as follows. For any $i \in I_p$, the optimal response function is

$$\phi_{i,x}^p(\lambda_x) = \frac{1}{2\alpha_i}\lambda_x - \frac{\beta_i}{2\alpha_i}, \quad P_{i,x}^p(\lambda_x) = \begin{cases} x_i^{\min} & \text{if } \phi_{i,x}^p(\lambda_x) \le x_i^{\min}, \\ \phi_{i,x}^p(\lambda_x) & \text{if } x_i^{\min} < \phi_{i,x}^p(\lambda_x) < x_i^{\max}, \\ x_i^{\max} & \text{if } \phi_{i,x}^p(\lambda_x) \ge x_i^{\max}. \end{cases} \tag{3.2}$$

For any $i \in I_h$, the optimal response function is

$$\phi_{i,y}^h(\lambda_y) = \frac{1}{2a_i}\lambda_y - \frac{b_i}{2a_i}, \quad P_{i,y}^h(\lambda_y) = \begin{cases} y_i^{min} & \text{if } \phi_{i,y}^h(\lambda_y) \le y_i^{min}, \\ \phi_{i,y}^h(\lambda_y) & \text{if } y_i^{min} < \phi_{i,y}^h(\lambda_y) < y_i^{max}, \\ y_i^{max} & \text{if } \phi_{i,y}^h(\lambda_y) \ge y_i^{max}. \end{cases} \tag{3.3}$$

For any $i \in I_c$, the optimal response function is

$$\begin{aligned}
\phi^c_{i,x}(\lambda_x, \lambda_y) &= \frac{2a_i}{|\boldsymbol{Q}_i|}\lambda_x - \frac{\varpi_i}{|\boldsymbol{Q}_i|}\lambda_y + \frac{\varpi_i b_i - 2a_i\beta_i}{|\boldsymbol{Q}_i|},\\
\phi^c_{i,y}(\lambda_x, \lambda_y) &= -\frac{\varpi_i}{|\boldsymbol{Q}_i|}\lambda_x + \frac{2\alpha_i}{|\boldsymbol{Q}_i|}\lambda_y + \frac{\varpi_i\beta_i - 2\alpha_i b_i}{|\boldsymbol{Q}_i|},\\
P^c_i(\lambda_x, \lambda_y) &= \operatorname*{argmin}_{(x,y)\in\Omega_i} \|(x,y) - (\phi^c_{i,x}, \phi^c_{i,y})\|^2_{\boldsymbol{Q}_i}.
\end{aligned} \tag{3.4}$$

Here, P_i is a vector and $\boldsymbol{Q}_i$ is a matrix, written as

$$P^c_i = \begin{bmatrix} P^c_{i,x} \\ P^c_{i,y} \end{bmatrix}, \qquad \boldsymbol{Q}_i = \begin{bmatrix} 2\alpha_i & \varpi_i \\ \varpi_i & 2a_i \end{bmatrix}.$$

For any $i \in I_e$, the optimal response function is

$$\phi^e_{i,x}(\lambda_x) = \frac{\beta_i}{2\alpha_i} - \frac{1}{2\alpha_i}\lambda_x, \quad P^e_{i,x}(\lambda_x) = \begin{cases} x_i^{min} & \text{if } \phi^e_{i,x}(\lambda_x) \le x_i^{min}, \\ \phi^e_{i,x}(\lambda_x) & \text{if } x_i^{min} < \phi^e_{i,x}(\lambda_x) < \frac{\beta_i}{2\alpha_i}, \\ \frac{\beta_i}{2\alpha_i} & \text{if } \phi^e_{i,x}(\lambda_x) \ge \frac{\beta_i}{2\alpha_i}. \end{cases} \tag{3.5}$$

For any $i \in I_t$, the optimal response function is

$$\phi^t_{i,y}(\lambda_y) = \frac{b_i}{2a_i} - \frac{1}{2a_i}\lambda_y, \quad P^t_{i,y}(\lambda_y) = \begin{cases} y_i^{min} & \text{if } \phi^t_{i,y}(\lambda_y) \le y_i^{min}, \\ \phi^t_{i,y}(\lambda_y) & \text{if } y_i^{min} < \phi^t_{i,y}(\lambda_y) < \frac{b_i}{2a_i}, \\ \frac{b_i}{2a_i} & \text{if } \phi^t_{i,y}(\lambda_y) \ge \frac{b_i}{2a_i}. \end{cases} \tag{3.6}$$

Even in the absence of equality constraints, solving problem P via (3.1) is still dependent on the cost functions of all devices. This means that a central unit is needed to collect and broadcast global information. However, this centralized architecture is highly susceptible to single points of failure and has poor scalability. In this case, a distributed architecture is a good choice. Therefore, the local estimates of incremental costs, denoted as $\lambda_{i,x}$ and $\lambda_{i,y}$, are introduced. Local electric and thermal loads, denoted as $D_{i,x}$ and $D_{i,y}$, are defined, which satisfy

$$\begin{aligned}
\sum_{i\in I_p} D^p_{i,x} + \sum_{i\in I_c} D^c_{i,x} &= D_x,\\
\sum_{i\in I_h} D^h_{i,y} + \sum_{i\in I_c} D^c_{i,y} &= D_y.
\end{aligned}$$

On the basis of the distributed architecture and optimal response functions, we provide the distributed optimality conditions for problem P in Lemma 1.

Lemma 1. *If the local incremental costs $\lambda^\star_{i,x}$ and $\lambda^\star_{i,y}$ satisfy the following equations:*

$$\lambda^\star_{i,x} = \lambda^\star_{j,x}, \qquad \forall i, j \in I_p \cup I_c \cup I_e, \tag{3.7a}$$

$$\lambda^\star_{i,y} = \lambda^\star_{j,y}, \qquad \forall i, j \in I_h \cup I_c \cup I_t, \tag{3.7b}$$

$$\sum_{i\in I_p} D^p_{i,x} - P^{p,\star}_{i,x} + \sum_{i\in I_c} D^c_{i,x} - P^{c,\star}_{i,x} + \sum_{i\in I_e} P^{e,\star}_{i,x} = 0, \tag{3.7c}$$

$$\sum_{i\in I_h} D^h_{i,y} - P^{h,\star}_{i,y} + \sum_{i\in I_c} D^c_{i,y} - P^{c,\star}_{i,y} + \sum_{i\in I_t} P^{t,\star}_{i,y} = 0, \tag{3.7d}$$

then $P^{e,\star}_{i,x}, P^{p,\star}_{i,x}, P^{c,\star}_{i,x}$, *and* $P^{t,\star}_{i,y}, P^{h,\star}_{i,y}, P^{c,\star}_{i,y}$ *are the optimal electrical and thermal power for problem P, respectively.*

Proof. It follows from (3.7a) and (3.7b) that $\lambda^\star_x$ and $\lambda^\star_y$ exist such that

$$\lambda^\star_{i,x} = \lambda^\star_x, \quad \forall i \in I_p \cup I_c \cup I_e,$$
$$\lambda^\star_{i,y} = \lambda^\star_y, \quad \forall i \in I_h \cup I_c \cup I_t.$$

When Ω is a closed convex set and the function $g(y)$ is dgeneration-sideerentiable and convex, $-\partial g(y^\star) \in N_\Omega(y^\star)$ if and only if $y^\star = \operatorname{argmin}_{\Omega} g(y)$. Therefore, for any $i \in I_c$, it holds that

$$(x^\star_i, y^\star_i) = \operatorname*{argmin}_{(x,y)\in\Omega_i} \left(G^c_i - x\lambda^\star_x - y\lambda^\star_y\right) = \operatorname*{argmin}_{(x,y)\in\Omega_i} \left(G^c_i - x\lambda^\star_{i,x} - y\lambda^\star_{i,y}\right).$$

From Equation (2.2), it holds that

$$G^c_i - x\lambda^\star_{i,x} - y\lambda^\star_{i,y} = \left\|(x,y) - (\phi^c_{i,x}, \phi^c_{i,y})\right\|^2_{\boldsymbol{Q}_i} + M,$$

with $M = \gamma_i - (\phi^c_{i,x}, \phi^c_{i,y})\boldsymbol{Q}_i(\phi^c_{i,x}, \phi^c_{i,y})^\top$. The constant M is independent of (x, y), so for any CHP unit i, we have

$$(x^\star_i, y^\star_i) = (P^\star_{i,x}, P^\star_{i,y}).$$

For the FEL i, if $x^{min}_i < x^\star_i < \frac{\beta_i}{2\alpha_i}$, then $N_{\Omega_i}(x^\star_i) = 0, \quad \frac{\partial H}{\partial x_i}(x^\star_i, y^\star_i) = -2\alpha_i x^\star + \beta_i$. Substituting them into (3.1c) gives

$$x^\star_i = \phi^{e,\star}_{i,x}, \quad x^{min}_i < \phi^{e,\star}_{i,x} \le \frac{\beta_i}{2\alpha_i}.$$

If $\frac{\beta_i}{2\alpha_i} \le x^\star_i < x^{max}_i$, then $N_{\Omega_i}(x^\star_i) = 0, \quad \frac{\partial H}{\partial x_i}(x^\star_i, y^\star_i) = 0$. From (3.1c), we get $\lambda^\star_x = 0$, which means $\phi^{e,\star}_{i,x} = \frac{\beta_i}{2\alpha_i}$. In order to minimize costs and $x^\star_i$ can take any value in $[\frac{\beta_i}{2\alpha_i}, x^{\max}_i)$, let

$$x^\star_i = \phi^{e,\star}_{i,x} = \frac{\beta_i}{2\alpha_i}.$$

If $x^\star_i = x^{min}_i$, then $N_{\Omega_i}(x^\star_i) \le 0$, and $\frac{\partial H}{\partial x_i}(x^\star_i, y^\star_i) = -2\alpha_i x^\star + \beta_i$. Substituting into (3.1c), we get

$$\phi^{e,\star}_{i,x} \le x^{min}_i.$$

If $x^\star_i = x^{max}_i$, then $N_{\Omega_i}(x^\star_i) \ge 0$, and $\frac{\partial H}{\partial x_i}(x^\star_i, y^\star_i) = 0$. Substituting into (3.1c), one has

$$\phi^{e,\star}_{i,x} \ge \frac{\beta_i}{2\alpha_i}.$$

Given the expression above, it is evident that for any $i \in I_e$, we have

$$x_i^\star = P_{i,x}^{e,\star}.$$

The same procedure is applied to the remaining units, it holds that

$$x_k^\star = P_{k,x}^{p,\star}, \quad y_i^\star = P_{i,y}^{h,\star}, \quad y_j^\star = P_{j,y}^{t,\star}.$$

Substituting these results into (3.1), Lemma 1 can be directly derived. □

Remark 1. *The key distinction between this study and those involving only decoupled units lies in the inclusion of the CHP. The constraint treatment of the CHP directly affects the optimality of the results and computational efficiency. This paper discusses three possible methods to illustrate the principles and advantages of our design in* (3.4)*.*

- ***Naive Euclidean projection.*** *A straightforward approach is used to first calculate the unconstrained solution* $(\phi_{i,x}^c, \phi_{i,y}^c)$ *by solving*

$$\frac{\partial G_i^c}{\partial x_i} = \lambda_x, \quad \frac{\partial G_i^c}{\partial y_i} = \lambda_y.$$

We then project it onto the feasible region using standard Euclidean projection

$$P_i^{Euclid}(\lambda_x, \lambda_y) = \underset{(x,y)\in\Omega_i}{\operatorname{argmin}} \|(x, y) - (\phi_{i,x}^c, \phi_{i,y}^c)\|.$$

While this method yields a feasible solution, it loses its optimality. This is because the cost function of the CHP in (2.3) *introduces a weighted norm of* $\boldsymbol{Q}_i$ *through the cross-term* $\varpi_i x_i y_i$*. Unless* $\boldsymbol{Q}_i$ *is a scalar multiple of the identity matrix, the projection under Euclidean distance is not equivalent to the projection under the* $\boldsymbol{Q}_i$*-weighted distance. However, this condition usually does not hold due to the coupling coefficient* $\varpi_i \neq 0$*.*

- ***Active set method of [8].*** *This method first checks whether the unconstrained solution satisfies all constraints. If not, it solves for the candidate point and its Lagrange multiplier by treating each constraint as an equation. It then verifies whether the multiplier is non-negative and whether the candidate point satisfies the remaining constraints. If still not satisfied, it checks the vertices of the feasible region. While this method ensures optimality, its computation is cumbersome and requires repeated trials and verifications.*

- ***Optimal response function in*** (3.4)***.*** *The optimal response function projects the unconstrained solution onto the feasible region in the sense of the* $\boldsymbol{Q}_i$*-weighted norm. This operation has two key advantages. First, it preserves optimality because the distance metric in the projection is exactly the same as the metric in the objective function. Second, for the convex polyhedral regions* Ω_i *described by linear inequalities, this* $\boldsymbol{Q}_i$*-weighted norm projection can often be computed by simple algorithms (e.g., solving a small quadratic programming problem), thus improving computational efficiency.*

4. Algorithm design

In practical environments, systems inevitably encounter network latency and device heterogeneity. Therefore, this section proposes an ADDC algorithm to solve problem P in an asynchronous distributed manner. Subsequently, an auxiliary transformation is constructed to present the cumulative form of the ADDC algorithm and further provide the mathematical foundation for the convergence analysis that follows.

4.1. The ADDC algorithm

Algorithm 1 ADDC algorithm – a global perspective

1: Initialize the state variables $\boldsymbol{\lambda}(0)$ and $\mathbf{Z}(0)$ arbitrarily and obtain $\boldsymbol{\mathcal{F}}(0)$ according to (4.7).
2: Initialize the delayed state variables

$$\boldsymbol{\lambda}(-1) = \mathbf{Z}(-1) = 0,\ \boldsymbol{\mathcal{F}}(-1) = \mathbf{Z}(0),\ T_{ij}^{-1} = -1, \forall i, j \in \mathcal{N}_x \cup \mathcal{N}_y. \tag{4.1}$$

3: Send the initial variables $\boldsymbol{\lambda}(0)$ and $\mathbf{Z}(0)$ to neighboring devices.
4: Initialize the global clock $\ell = 0$.
5: Determine the active device i^ℓ and obtain the active agent indices n^ℓ and s^ℓ according to (4.6).
6: Perform local calculations.

- If $n = n^\ell$ and $n^\ell \neq \infty$, then update the timestamp $T_{nn}^\ell = \ell$ and execute

$$\begin{aligned}
\mathbf{Z}_{n,x}(\ell+1) &= \mathbf{C}_{nn}^x \mathbf{Z}_{n,x}(\ell) + \sum_{j\in\mathcal{N}_n^+} \mathbf{C}_{nj}^x \mathbf{Z}_{j,x}(T_{nj}^\ell) + \boldsymbol{\mathcal{F}}_{n,x}(\ell),\\
\boldsymbol{\lambda}_{n,x}(\ell+1) &= \mathbf{R}_{nn}^x \boldsymbol{\lambda}_{n,x}(\ell) + \sum_{j\in\mathcal{N}_n^+} \mathbf{R}_{nj}^x \boldsymbol{\lambda}_{j,x}(T_{nj}^\ell) + \varepsilon(\mathbf{Z}_{n,x}(\ell+1) - \mathbf{Z}_{n,x}(\ell)).
\end{aligned} \tag{4.2}$$

- If $s = s^\ell$ and $s^\ell \neq \infty$, then update the timestamp $T_{ss}^\ell = \ell$ and execute

$$\begin{aligned}
\mathbf{Z}_{s,y}(\ell+1) &= \mathbf{C}_{ss}^y \mathbf{Z}_{s,y}(\ell) + \sum_{j\in\mathcal{N}_s^+} \mathbf{C}_{sj}^y \mathbf{Z}_{j,y}(T_{sj}^\ell) + \boldsymbol{\mathcal{F}}_{s,y}(\ell),\\
\boldsymbol{\lambda}_{s,y}(\ell+1) &= \mathbf{R}_{ss}^y \boldsymbol{\lambda}_{s,y}(\ell) + \sum_{j\in\mathcal{N}_s^+} \mathbf{R}_{sj}^y \boldsymbol{\lambda}_{j,y}(T_{sj}^\ell) + \varepsilon(\mathbf{Z}_{s,y}(\ell+1) - \mathbf{Z}_{s,y}(\ell)).
\end{aligned} \tag{4.3}$$

- For all $n \neq n^\ell$ and $1 \leq n \leq N_x$, perform

$$\mathbf{Z}_{n,x}(\ell+1) = \mathbf{Z}_{n,x}(\ell), \quad \boldsymbol{\lambda}_{n,x}(\ell+1) = \boldsymbol{\lambda}_{n,x}(\ell), \quad T_{nj}^{\ell+1} = T_{nj}^\ell,\ j \in \mathcal{N}_n^+ \cup \{n\}. \tag{4.4}$$

- For all $s \neq s^\ell$ and $1 \leq s \leq N_y$, perform

$$\mathbf{Z}_{s,y}(\ell+1) = \mathbf{Z}_{s,y}(\ell), \quad \boldsymbol{\lambda}_{s,y}(\ell+1) = \boldsymbol{\lambda}_{s,y}(\ell), \quad T_{sj}^{\ell+1} = T_{sj}^\ell,\ j \in \mathcal{N}_s^+ \cup \{s\}. \tag{4.5}$$

7: Send state vectors $\boldsymbol{\lambda}_{n^\ell,x}(\ell+1)$ and $\mathbf{Z}_{n^\ell,x}(\ell+1)$ to the neighboring agents of agent n_x^ℓ. Send state vectors $\boldsymbol{\lambda}_{s^\ell,y}(\ell+1)$ and $\mathbf{Z}_{s^\ell,y}(\ell+1)$ to the neighboring agents of agent s_y^ℓ.
8: Check the stopping criteria. If it is met, output $\mathbf{P}_x(\ell)$ and $\mathbf{P}_y(\ell)$ based on (3.2)–(3.6). Otherwise, increase ℓ by 1 and return to Step (5).

Algorithm 1 outlines the specific iterative procedure of the ADDC algorithm. Note that a global counter ℓ is introduced in Algorithm 1. Its sole purpose is to theoretically define the temporal ordering of events and to quantify the degree of information delay. In practical implementations, however, the global counter ℓ does not exist. Instead, each device independently triggers and performs asynchronous updates based on its own computational capabilities. Specifically, each device i maintains only a local counter

ℓ_i. Whenever a local computation is completed, the value of this counter is immediately increased by 1. Throughout this process, the devices require neither synchronization operations nor any global coordination mechanisms. Figures 2 and 3 illustrate this asynchronous process from the perspectives of a single device and the global system, respectively.

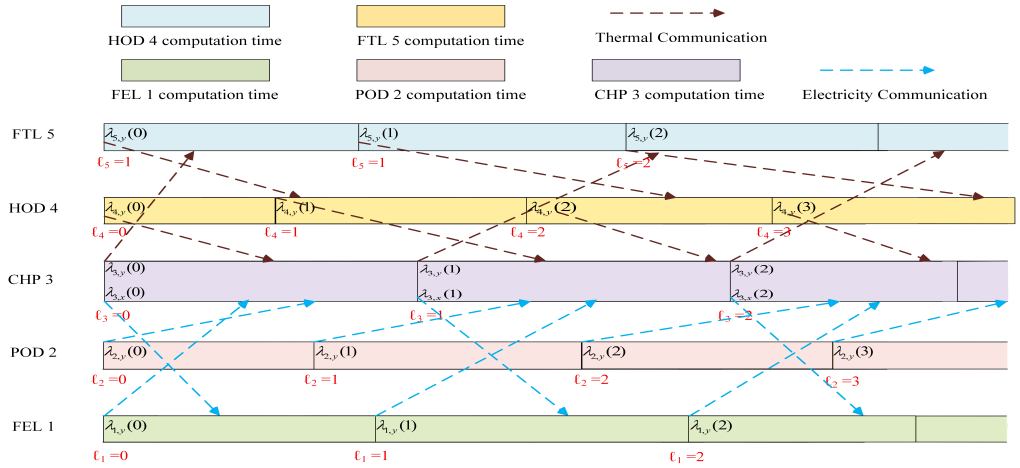


Figure 2. Fully asynchronous iteration from the perspective of an individual device.

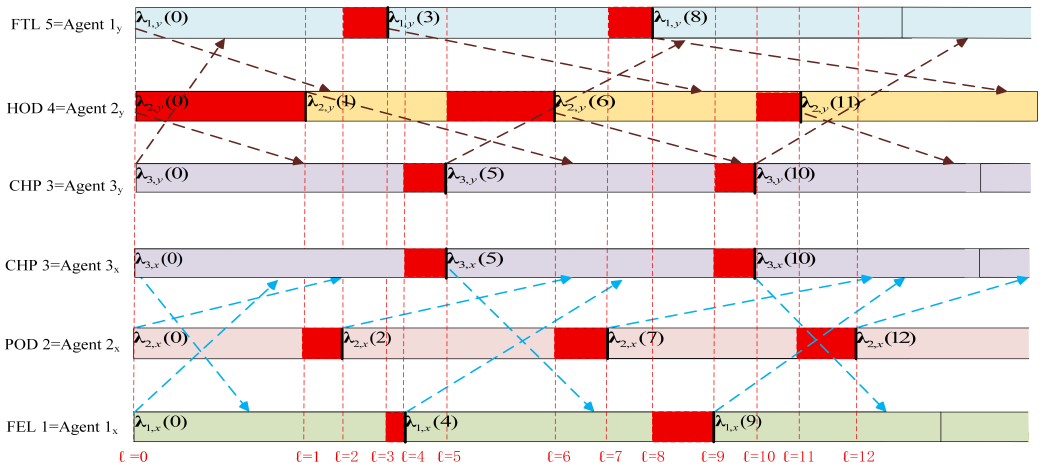


Figure 3. Fully asynchronous iteration from a global perspective.

According to Lemma 1, the optimal solution to problem P must simultaneously satisfy two conditions: The consistency of incremental costs and a balance between supply and demand. To this end, the ADDC algorithm uses a row-stochastic weight matrix **R** to achieve dual consensus on the local incremental costs and utilizes a column-stochastic weight **C** matrix to ensure the balance between supply and demand. It is important to emphasize that each device selects only the data with the most recent timestamp for computation, which prevents overly outdated information from blocking the consensus process.

Moreover, we introduce a specific auxiliary variable $\mathbf{Z}$ to track the cumulative amount of supply–demand mismatch and use its incremental change to reflect the equilibrium status. Compared with directly tracking supply–demand mismatch, it not only effectively eliminates sensitivity to initialization but also ensures the mean invariance of the increment $\mathbf{Z}(\ell+1)-\mathbf{Z}(\ell)$. The latter constitutes a crucial prerequisite for global supply–demand balance.

The notation used in the algorithm above is explained as follows. The topology of the EH-EMS is first divided into two regions, denoted as $\mathcal{G}_x$ and $\mathcal{G}_y$. Region $\mathcal{G}_x$ consists of N_x devices (e.g., FELs, PODs, and CHPs), while region $\mathcal{G}_y$ consists of N_y devices (e.g., CHPs, HODs, and FTLs). Then the devices in the $\mathcal{G}_x$ region are reordered according to the sequence of FELs, PODs, and CHPs, and the n-th device is relabeled as agent n_x with $n \in \mathcal{N}_x = \{1, \cdots, N_x\}$. Similarly, the devices in the $\mathcal{G}_y$ region are reordered according to the sequence of CHPs, HODs, and FTLs, and the s-th device is labeled as agent s_y with $s \in \mathcal{N}_y = \{1, \cdots, N_y\}$. The mathematical expression for the sequential index of device i mapped to regions $\mathcal{G}_x$ and $\mathcal{G}_y$ is given by

$$\Psi(i) = \begin{cases} (n, \infty), & if \quad i \in I_p \cup I_e, \\ (n, s), & if \quad i \in I_c, \\ (\infty, s), & if \quad i \in I_h \cup I_t. \end{cases} \tag{4.6}$$

The symbols $t(\ell)$ and $t_i(\ell_i)$ denote the physical time points at which the ℓ-th global iteration and the ℓ_i-th local iteration occur, respectively. Here, i^ℓ represents the device that completes a local iteration at $t(\ell+1)$. $\Psi(i^\ell) = (n^\ell, s^\ell)$ is the agent indices of the activated device i^ℓ in the $\mathcal{G}_x$ and $\mathcal{G}_y$ regions. T_{nj}^ℓ represents the latest iteration timestamp of the delayed information received by agent n from the neighbor agent j up to time $t(\ell)$. If $n = n^\ell$, the neighbor information used by agent n for updating is not real-time but a delayed version broadcast by j at historical time $t(T_{nj}^\ell)$. More importantly, this version is the most up-to-date information received by agent n from j up to time $t(\ell)$.

Define $\boldsymbol{\lambda}$ as the global incremental cost, $\mathbf{Z}$ as the auxiliary vector for tracking the cumulative supply–demand mismatch, $\mathbf{P}$ as the power vector, and $\mathcal{F}$ as a function of the supply–demand mismatch. Their constructions are as follows:

$$\begin{aligned} &\boldsymbol{\lambda} = [\boldsymbol{\lambda}_x; \boldsymbol{\lambda}_y], && \boldsymbol{\lambda}_x = [\boldsymbol{\lambda}_x^e; \boldsymbol{\lambda}_x^p; \boldsymbol{\lambda}_x^c], && \boldsymbol{\lambda}_y = [\boldsymbol{\lambda}_y^c; \boldsymbol{\lambda}_y^h; \boldsymbol{\lambda}_y^t], \\ &\mathbf{Z} = [\mathbf{Z}_x; \mathbf{Z}_y], && \mathbf{Z}_x = [\mathbf{Z}_x^e; \mathbf{Z}_x^p; \mathbf{Z}_x^c], && \mathbf{Z}_y = [\mathbf{Z}_y^c; \mathbf{Z}_y^h; \mathbf{Z}_y^t], \\ &\mathbf{P} = [\mathbf{P}_x; \mathbf{P}_y], && \mathbf{P}_x = [\mathbf{P}_x^e; \mathbf{P}_x^p; \mathbf{P}_x^c], && \mathbf{P}_y = [\mathbf{P}_y^c; \mathbf{P}_y^h; \mathbf{P}_y^t], \\ &\mathcal{F} = [\mathcal{F}_x; \mathcal{F}_y], && \mathcal{F}_x = [\mathbf{P}_x^e; \mathbf{D}_x^p - \mathbf{P}_x^p; \mathbf{D}_x^c - \mathbf{P}_x^c], && \mathcal{F}_y = [\mathbf{D}_y^c - P_y^c; \mathbf{D}_y^h - \mathbf{P}_y^h; \mathbf{P}_y^t]. \end{aligned} \tag{4.7}$$

Here, $\boldsymbol{\lambda}_x$ is the incremental cost of the $\mathcal{G}_x$ region; $\boldsymbol{\lambda}_x^e$, $\boldsymbol{\lambda}_x^p$, and $\boldsymbol{\lambda}_x^c$ are stacked from the incremental electricity costs $\lambda_{i,x}$ of the FELs, PODs, and CHPs, respectively; $\boldsymbol{\lambda}_y$ is the incremental cost of the $\mathcal{G}_y$ region; $\boldsymbol{\lambda}_y^c$, $\boldsymbol{\lambda}_y^h$, and $\boldsymbol{\lambda}_y^t$ are stacked from the thermal incremental costs $\lambda_{i,y}$ of the CHPs, POHs, and FTLs, respectively. Other vectors are similar and omitted here. $\mathbf{R}$ and $\mathbf{C}$ are block matrices, expressed as

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}^x & 0 \\ 0 & \mathbf{R}^y \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} \mathbf{C}^x & 0 \\ 0 & \mathbf{C}^y \end{bmatrix},$$

where $\mathbf{R}^x$ and $\mathbf{C}^x$ are weight matrices associated with $\mathcal{G}_x$, while $\mathbf{R}^y$ and $\mathbf{C}^y$ are the weight matrices associated with $\mathcal{G}_y$.

Remark 2. *To avoid confusion, bold symbols such as $\boldsymbol{\lambda}_{n,x}$ represent the n-th element of the vector $\boldsymbol{\lambda}_x$, whereas nonbold symbols such as $\lambda_{i,x}$ represent the incremental power cost of device i. Other symbols follow a similar convention.*

4.2. Cumulative form

From (4.1), $\mathbf{Z}_{n,x}(0)$ is expressed as

$$\mathbf{Z}_{n,x}(0) = \mathbf{C}^x_{nn}\mathbf{Z}_{n,x}(-1) + \sum_{j\in\mathcal{N}_n^+} \mathbf{C}^x_{nj}\mathbf{Z}_{j,x}(T^{-1}_{nj}) + \boldsymbol{\mathcal{F}}_{n,x}(-1). \tag{4.8}$$

Combining (4.4) and (4.8), $\mathbf{Z}_{n,x}(\ell+1)$ can be equivalently written as

$$\mathbf{Z}_{n,x}(\ell+1) = \mathbf{C}^x_{nn}\mathbf{Z}_{n,x}(\hat{\ell}) + \sum_{j\in\mathcal{N}_n^+} \mathbf{C}^x_{nj}\mathbf{Z}_{j,x}(T^{\hat{\ell}}_{nj}) + \boldsymbol{\mathcal{F}}_{n,x}(\hat{\ell}). \tag{4.9}$$

Here, $\hat{\ell} \le \ell$ is the activation point closest to ℓ for agent n_x. Recalling the definition of T^{ℓ}_{nj}, one has

$$\hat{\ell} = T^{\hat{\ell}}_{nn} = T^{\ell}_{nn}, \quad T^{\hat{\ell}}_{nj} = T^{\ell}_{nj}. \tag{4.10}$$

Let $\boldsymbol{\omega}_x(\ell) = \mathbf{Z}_x(\ell+1) - \mathbf{Z}_x(\ell)$. If agent n_x remains unawakened at $\ell+1$, the result from (4.4) gives

$$\boldsymbol{\omega}_{n,x}(\ell+1) = \mathbf{Z}_{n,x}(\ell+2) - \mathbf{Z}_{n,x}(\ell+1) = 0, \quad n \neq n^{\ell+1}. \tag{4.11}$$

If agent n_x is awakened at $\ell+1$, from (4.9) and (4.10), it holds that

$$\begin{aligned}\boldsymbol{\omega}_{n,x}(\ell+1) = {}& \mathbf{C}^x_{nn}\boldsymbol{\omega}_{n,x}(T^{\ell}_{nn}) + \boldsymbol{\mathcal{F}}_{n,x}(T^{\ell+1}_{nn}) - \boldsymbol{\mathcal{F}}_{n,x}(T^{\ell}_{nn}) \\ & + \sum_{j\in\mathcal{N}_n^+} \mathbf{C}^x_{nj}[\mathbf{Z}_{j,x}(T^{\ell+1}_{nj}) - \mathbf{Z}_{j,x}(T^{\ell}_{nj})],\end{aligned} \tag{4.12}$$

where $T^{\ell+1}_{nn} = \ell+1$ and $\mathbf{Z}_{n,x}(\ell+1) = \mathbf{Z}_{n,x}(T^{\ell}_{nn}+1)$ are used. Regarding the last item, one has

$$\mathbf{C}^x_{nj}[\mathbf{Z}_{j,x}(T^{\ell+1}_{nj}) - \mathbf{Z}_{j,x}(T^{\ell}_{nj})] = \mathbf{C}^x_{nj} \sum_{k=T^{\ell}_{nj}}^{T^{\ell+1}_{nj}-1} \boldsymbol{\omega}_{j,x}(k). \tag{4.13}$$

Substituting (4.13) into (4.12), $\boldsymbol{\lambda}_x$ in the ADDC algorithm are rewritten as

$$\begin{aligned}\boldsymbol{\lambda}_{n,x}(\ell+1) &= \mathbf{R}^x_{nn}\boldsymbol{\lambda}_{n,x}(\ell) + \sum_{j\in\mathcal{N}_n^+} \mathbf{R}^x_{nj}\boldsymbol{\lambda}_{j,x}(T^{\ell}_{nj}) + \varepsilon\boldsymbol{\omega}_{n,x}(\ell), && n = n^{\ell}, \\ \boldsymbol{\omega}_{n,x}(\ell+1) &= \mathbf{C}^x_{nn}\boldsymbol{\omega}_{n,x}(T^{\ell}_{nn}) + \sum_{j\in\mathcal{N}_n^+}\sum_{k=T^{\ell}_{nj}}^{T^{\ell+1}_{nj}-1} \mathbf{C}^x_{nj}\boldsymbol{\omega}_{j,x}(k) + \boldsymbol{\mathcal{F}}_{n,x}(T^{\ell+1}_{nn}) - \boldsymbol{\mathcal{F}}_{n,x}(T^{\ell}_{nn}), && n = n^{\ell+1},\end{aligned} \tag{4.14}$$

and

$$\begin{aligned}&\boldsymbol{\lambda}_{n,x}(\ell+1) = \boldsymbol{\lambda}_{n,x}(\ell), && n \neq n^{\ell}, \\ &\boldsymbol{\omega}_{n,x}(\ell+1) = 0, && n \neq n^{\ell+1}.\end{aligned} \tag{4.15}$$

Similarly, if we $\boldsymbol{\omega}_y(\ell) = \mathbf{Z}_y(\ell+1) - \mathbf{Z}_y(\ell)$, $\boldsymbol{\lambda}_y$ is rewritten as

$$\begin{aligned}
\boldsymbol{\lambda}_{s,y}(\ell+1) &= \mathbf{R}^y_{ss}\boldsymbol{\lambda}_{s,y}(\ell) + \sum_{j\in\mathcal{N}_s^+} \mathbf{R}^y_{sj}\boldsymbol{\lambda}_{j,y}(T^\ell_{sj}) + \varepsilon\boldsymbol{\omega}_{s,y}(\ell), && s = s^\ell,\\
\boldsymbol{\omega}_{s,y}(\ell+1) &= \mathbf{C}^y_{ss}\boldsymbol{\omega}_{s,y}(T^\ell_{ss}) + \sum_{j\in\mathcal{N}_s^+}\sum_{k=T^\ell_{sj}}^{T^{\ell+1}_{sj}-1} \mathbf{C}^y_{sj}\boldsymbol{\omega}_{j,y}(k) + \boldsymbol{\mathcal{F}}_{s,y}(T^{\ell+1}_{ss}) - \boldsymbol{\mathcal{F}}_{s,y}(T^\ell_{ss}), && s = s^{\ell+1},
\end{aligned} \tag{4.16}$$

and

$$\begin{aligned}
&\boldsymbol{\lambda}_{s,y}(\ell+1) = \boldsymbol{\lambda}_{s,y}(\ell), && s \neq s^\ell,\\
&\boldsymbol{\omega}_{s,y}(\ell+1) = 0, && s \neq s^{\ell+1}.
\end{aligned} \tag{4.17}$$

Remark 3. *Since the system cannot predict which agents n_x and s_y will be active at time $t(\ell)$, the cumulative forms* (4.14)–(4.17) *of the ADDC algorithm cannot be directly used in actual operation. Their introduction here serves a theoretical purpose. Specifically, they reveal the dgeneration-sideerence between the ADDC algorithm and its synchronous counterpart, and they provide the mathematical foundation for the convergence analysis that follows.*

Remark 4. *If the update is forcibly synchronized and communication is delay-free, then $T^\ell_{nj} = \ell$ and all agents are activated at each step. Substituting this into* (4.14)–(4.17)*, the synchronous algorithm, defined as D-ADDC, is obtained as*

$$\begin{aligned}
\boldsymbol{\lambda}(\ell+1) &= \mathbf{R}\boldsymbol{\lambda}(\ell) + \varepsilon\boldsymbol{\omega}(\ell),\\
\boldsymbol{\omega}(\ell+1) &= \mathbf{C}\boldsymbol{\omega}(\ell) + \boldsymbol{\mathcal{F}}(\ell+1) - \boldsymbol{\mathcal{F}}(\ell),
\end{aligned} \tag{4.18}$$

where $\boldsymbol{\omega}(\ell) = \mathbf{Z}(\ell+1) - \mathbf{Z}(\ell)$. Clearly, for the variable $\boldsymbol{\lambda}$, the ADDC algorithm permits the use of delayed information, whereas the D-ADDC algorithm requires all information to be real-time. For the variable $\boldsymbol{\omega}$, the ADDC algorithm draws on two additional types of information beyond the function increment term. One is the message the agent itself broadcasts at its last activation. The other is the cumulative information arriving from the neighboring agents during the last activation interval.

Remark 5. *In fact, the synchronous version of the ADDC algorithm, defined as S-ADDC, is given by*

$$\begin{aligned}
\mathbf{Z}(\ell+1) &= \mathbf{C}\mathbf{Z}(\ell) + \boldsymbol{\mathcal{F}}(\ell),\\
\boldsymbol{\lambda}(\ell+1) &= \mathbf{R}\boldsymbol{\lambda}(\ell) + \varepsilon(\mathbf{Z}(\ell+1) - \mathbf{Z}(\ell)).
\end{aligned} \tag{4.19}$$

The D-ADDC algorithm (4.18) *is an extended version of [7]. The only dgeneration-sideerence between the two synchronous algorithms is the auxiliary transformation $\boldsymbol{\omega}(\ell) = \mathbf{Z}(\ell+1) - \mathbf{Z}(\ell)$. Under synchronous communication, this dgeneration-sideerence only constrains how the auxiliary variables are initialized, without affecting the algorithm's feasibility. This conclusion, however, breaks down when an asynchronous mechanism is adopted. As later experiments show, a direct asynchronous modification of the algorithm* (4.18) *causes convergence and optimality to decouple.*

5. Convergence analysis

A rigorous convergence proof of the ADDC algorithm is provided in this section. Firstly the assumptions regarding the communication network and asynchronous updates are established. Next, an augmented network is used to reformulate the ADDC algorithm into an augmented algorithm. Finally, the convergence of the ADDC algorithm is demonstrated indirectly via the augmented network.

5.1. Basic assumptions

Assumption 1. *The electrical network among PODs, CHPs, and FELs is unbalanced but strongly connected. The thermal network among HODs, CHPs, and FTLs is unbalanced but strongly connected.*

Assumption 2. *Matrices* $\mathbf{R}^x$ *and* $\mathbf{R}^y$ *are row-stochastic, while matrices* $\mathbf{C}^x$ *and* $\mathbf{C}^y$ *are column-stochastic.*

Assumption 3. *The computation time and communication delay of all devices are bounded, which satisfy the following properties.*

3-1. For any two devices i and j, the information transmitted from device j to device i is received with a delay of at most θ time units.
3-2. The positive numbers $\overline{\theta}$ *and* $\underline{\theta}$ *exist such that the computation time spent by any device to complete a local update that satisfies* $\underline{\theta} \le t_i(\ell_i + 1) - t_i(\ell_i) \le \overline{\theta},\ \forall \ell_i \in \mathbb{N}$.

Lemma 2 provides a bounded upper bound for the delay and inconsistent updates in the global iterative process.

Lemma 2. *Based on Assumption 3, the following propositions hold.*

2-1. Each agent is activated at least once in the time interval $[t(\ell), t(\ell + \tau)]$ *with* $\tau = (N_x + N_y - |I_c| - 1)\lfloor \overline{\theta}/\underline{\theta} \rfloor + 1$.
2-2. The information used by each agent is delayed by at most $\sigma = \tau + (N_x + N_y - |I_c|)\lfloor \theta/\underline{\theta} \rfloor$ *global iterations.*

Proof. (1) According to the definition of i^ℓ, the device i^ℓ completes one update before $t(\ell + 1)$. Furthermore, by Assumption 3-2, devices other than i^ℓ complete at most $\lfloor \overline{\theta}/\underline{\theta} \rfloor$ updates before $t(\ell) + \overline{\theta}$. Consequently, one has $t(\ell + \tau) \ge t(\ell) + \overline{\theta}$. Since every device performs at least one update in the interval $[t(\ell), t(\ell) + \overline{\theta}]$, it follows that in $[t(\ell), t(\ell + \tau)]$, we have

$$\bigcup_{k=\ell}^{\ell+\tau} i^k = \mathcal{N},$$

where $\mathcal{N} = I_p \cup I_h \cup I_c \cup I_e \cup I_t$ and $|\mathcal{N}| = N_x + N_y - |I_c|$. Combining (4.6), the Lemma 2-1 can be directly derived.

(2) Suppose a neighboring device sends a message to device i at time $t(\ell_0)$, and device i receives it during the interval $[t(\ell_1),\ t(\ell_1 + 1))$. By Assumption 3-1, it has $t(\ell_1) - t(\ell_0) \le \theta$. From Assumption 3-2, each device performs at most $\lfloor \theta/\underline{\theta} \rfloor$ updates within the time interval $[t(\ell_0), t(\ell_1)]$. Equivalently,

$$\ell_1 - |\mathcal{N}|\lfloor \theta/\overline{\theta} \rfloor \le \ell_0 < \ell_1.$$

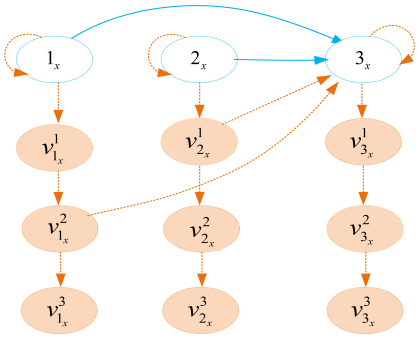
Note that device i is not necessarily activated at time $t(\ell_1)$, so the received message may not be used immediately. By Lemma 2-1, device i must be activated no later than $t(\ell_1 + \tau)$. If we let $\ell = \ell_1 + \tau$, we obtain $\ell - \sigma \le \ell_0 < \ell$. The mapping in (4.6) merely relabels devices, and thus

$$\ell - \sigma \le T_{nj}^{\ell} < \ell, \quad \ell - \sigma \le T_{sj}^{\ell} < \ell.$$

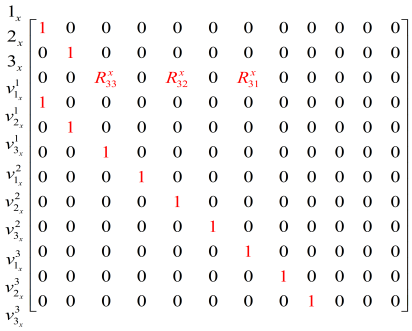
Lemma 2-2 is thus derived. □

5.2. *Augmented algorithm*

For each real agent n_x in region $\mathcal{G}_x$, two types of virtual agents are assigned. The first type, called v-type agents and denoted as $v_{n_x}^d, d = 1, \cdots, \sigma$, is introduced to handle the updates of $\boldsymbol{\lambda}_{n,x}$. The second type, called u-type agents and denoted as $u_{n_x}^d, d = 1, \cdots, \sigma$, is used to handle the updates of $\boldsymbol{\omega}_{n,x}$. Based on these, time-varying augmented networks are constructed, leading to the corresponding time-varying row-stochastic and column-stochastic matrices. The detailed design is described below.

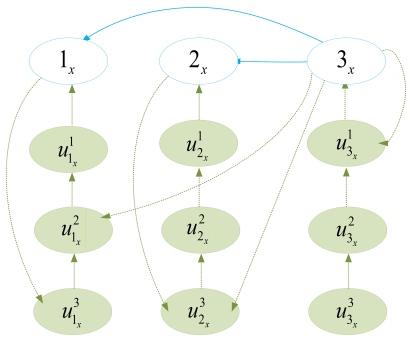


(a) The v-type time-varying augmented network

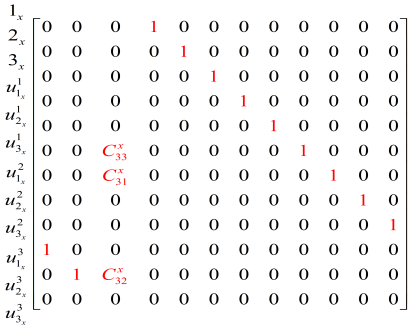


(b) Row-stochastic matrix $\tilde{\boldsymbol{R}}^x(\ell)$

Figure 4. An example of a v-type augmented network.



(a) The u-type time-varying augmented network



(b) Column-stochastic matrix $\tilde{\boldsymbol{C}}^x(\ell)$

Figure 5. An example of a u-type augmented network.

For the v-type augmented network, the agent set $\mathcal{N}_v^x$ contains the real agents N_x and the virtual agent $v_{N_x}^d$, amounting to $N_x(1+\sigma)$ agents in total. The edge set $\mathcal{E}_x^v(\ell)$ is time-varying. However, each real agent n_x always sends information to itself and the virtual agent $v_{n_x}^1$. Each virtual agent $v_{n_x}^d, d = 1, \cdots, \sigma - 1$ transmits information to the subsequent virtual agent $v_{n_x}^{d+1}$. In other words, $\mathcal{E}_x^v(\ell)$ always includes the edges (n_x, n_x), $(n_x, v_{n_x}^1)$, $(v_{n_x}^1, v_{n_x}^2)$, $\cdots$, $(v_{n_x}^{\sigma-1}, v_{n_x}^{\sigma})$. If the real agent n_x is activated at iteration ℓ, and $T_{nj}^{\ell} = \ell - d_j^{\ell}$, then $(v_{j_x}^{d_j^{\ell}}, n_x) \in \mathcal{E}_x^v(\ell)$.

As a simple example, suppose the region $\mathcal{G}_x$ contains three real agents and the delay parameter $\sigma = 3$. The real agent 3_x is activated at iteration ℓ, and uses the delayed information $\boldsymbol{\lambda}_{1,x}(\ell-2)$ and $\boldsymbol{\lambda}_{2,x}(\ell-1)$ from agents 1_x and 2_x when updating $\boldsymbol{\lambda}_{3,x}(\ell+1)$. It follows from $\sigma = 3$ that each real agent adds three

virtual v-type agents. In addition to the fixed edges $(v_{n_x}^d, v_{n_x}^{d+1})$, the new edges $(v_{1_x}^2, 3_x)$ and $(v_{2_x}^1, 3_x)$ are added. Figure 4a illustrates how the v-type augmented network is constructed based on the original network. Figure 4b depicts the row-stochastic matrix $\tilde{\boldsymbol{R}}^x(\ell)$ corresponding to the augmented network $\mathcal{E}_x^v(\ell)$ under this setting.

For the u-type augmented network, the agent set defined as $\mathcal{N}_u^x$ includes the real agent n_x and the virtual agents $u_{n_x}^d, d = 1, \cdots, \sigma$, amounting to $N_y(1+\sigma)$ agents in total. The edge set $\mathcal{E}_x^u(\ell)$ is also time-varying. The dgeneration-sideerence is that each virtual agent $u_{n_x}^d$ transmits information to the previous one. In other words, $\mathcal{E}_x^u(\ell)$ always includes the edges $(u_{n_x}^1, n_x)$, $(u_{n_x}^2, u_{n_x}^1)$, $\cdots$, $(u_{n_x}^\sigma, u_{n_x}^{\sigma-1})$. If agent n_x is not activated, then it transmits information to $u_{n_x}^\sigma$, i.e., $(n_x, u_{n_x}^\sigma) \in \mathcal{E}_x^u(\ell)$. If agent n_x is activated and its message $\boldsymbol{\omega}_{n,x}(\ell)$ is used by itself at iteration $\ell + d_n^\ell$ and by agent j_x at iteration $\ell + d_j^\ell$, then the edges $(n_x, u_{n_x}^{b_n^\ell})$ and $(n_x, u_{j_x}^{b_j^\ell})$ belong to $\mathcal{E}_x^u(\ell)$.

Figure 5a illustrates how the u-type augmented network is constructed, based on the original network. Since $\sigma = 3$, nine additional u-type agents are added. At iteration ℓ, the original agent 3_x is activated, and the information $\boldsymbol{\omega}_{3,x}(\ell)$ is used by itself at $\ell+1$, as well as by real agents 1_x and 2_x at $\ell+2$ and $\ell+3$, respectively. Therefore, the augmented network $\mathcal{E}_x^u(\ell)$ includes edges $(3_x, u_{3_x}^1)$, $(3_x, u_{1_x}^2)$, and $(3_x, u_{2_x}^3)$. Figure 5b depicts the column-stochastic matrix $\tilde{\boldsymbol{C}}^x(\ell)$ corresponding to the augmented network $\mathcal{E}_x^u(\ell)$.

Based on the augmented network described above, the augmented algorithm for $\mathcal{G}_x$ is written as

$$\begin{cases} \tilde{\boldsymbol{\lambda}}_x(\ell+1) = \tilde{\boldsymbol{R}}^x(\ell)\tilde{\boldsymbol{\lambda}}_x(\ell) + \varepsilon\tilde{\boldsymbol{E}}^x(\ell)\tilde{\boldsymbol{\omega}}_x(\ell), \\ \tilde{\boldsymbol{\omega}}_x(\ell+1) = \tilde{\boldsymbol{C}}^x(\ell)\tilde{\boldsymbol{\omega}}_x(\ell) + \tilde{\boldsymbol{\mathcal{F}}}_x(\ell+1) - \tilde{\boldsymbol{\mathcal{F}}}_x(\ell). \end{cases} \tag{5.1}$$

Here

$$\begin{cases} \tilde{\boldsymbol{\lambda}}_x(\ell) &= \left[\boldsymbol{\lambda}_x(\ell); \boldsymbol{\lambda}_x^1(\ell); \cdots; \boldsymbol{\lambda}_x^\sigma(\ell)\right], \\ \boldsymbol{\lambda}_x^d(\ell) &= \left[\boldsymbol{\lambda}_{1,x}^d(\ell); \cdots; \boldsymbol{\lambda}_{N^x,x}^d(\ell)\right], d \in \{1 \cdots \sigma\}, \\ \tilde{\boldsymbol{\omega}}_x(\ell) &= \left[\boldsymbol{\omega}_x(\ell); \boldsymbol{\omega}_x^1(\ell); \cdots; \boldsymbol{\omega}_x^\sigma(\ell)\right], \\ \boldsymbol{\omega}_x^d(\ell) &= \left[\boldsymbol{\omega}_{1,x}^d(\ell); \cdots; \boldsymbol{\omega}_{N^x,x}^d(\ell)\right], d \in \{1 \cdots \sigma\}, \\ \tilde{\boldsymbol{\mathcal{F}}}_x(\ell) &= [\boldsymbol{\mathcal{F}}_{1,x}(T_{11}^\ell); \cdots; \boldsymbol{\mathcal{F}}_{N_x,x}(T_{N^xN^x}^\ell); \mathbf{0}]. \end{cases}$$

$\tilde{\boldsymbol{R}}^x(\ell)$, $\tilde{\boldsymbol{C}}^x(\ell)$, and $\tilde{\boldsymbol{E}}^x(\ell)$ are time-varying and $\tilde{N}_x \times \tilde{N}_x$ dimensional matrices with $\tilde{N}_x = (\sigma+1)N^x$. Each matrix can be written explicitly as

$$\left[\tilde{\boldsymbol{R}}^x(\ell)\right]_{rm} = \begin{cases} \mathbf{R}_{n^\ell n^\ell}^x, & \text{if } r = m = n^\ell n^\ell, \\ \mathbf{R}_{n^\ell j}^x, & \text{if } r = n^\ell,\ m = j + N_x(\ell - T_{n^\ell j}^\ell), j \in \mathcal{N}_{n^\ell}^+, \\ 1, & \text{if } r = m, m \in \{\mathcal{N}_x / \{n^\ell\}\}, \\ 1, & \text{if } m = r - N_x,\ m \notin \mathcal{N}_x, \\ 0, & \text{otherwise.} \end{cases} \tag{5.2}$$

$$\left[\tilde{\boldsymbol{C}}^x(\ell)\right]_{rm} = \begin{cases} \mathbf{C}_{n^\ell n^\ell}^x, & \text{if } m = n^\ell, r = N^x d + n^\ell, T_{n^\ell n^\ell}^{\ell+d} \le \ell \le T_{n^\ell n^\ell}^{\ell+d+1} - 1, \\ \mathbf{C}_{jn^\ell}^x, & \text{if } m = n^\ell, r = N^x d + j, T_{jn^\ell}^{\ell+d} \le \ell \le T_{jn^\ell}^{\ell+d+1} - 1, j \in \mathcal{N}_{n^\ell}^-, \\ 1, & \text{if } r = m - N_x, m \notin \mathcal{N}_x, \\ 1, & \text{if } r = m + \sigma N_x, m \in \{\mathcal{N}_x / \{n^\ell\}\}, \\ 0, & \text{otherwise.} \end{cases} \tag{5.3}$$

$$\left[\tilde{\boldsymbol{E}}^x(\ell)\right]_{rm} = \begin{cases} 1, & \text{if } r = m = n^\ell, \\ 0, & \text{otherwise.} \end{cases} \tag{5.4}$$

By performing similar operations on the region $\mathcal{G}_y$, a augmented algorithm with $\tilde{N}_y = N_y(\sigma + 1)$ is obtained in the following way:

$$\begin{cases} \tilde{\boldsymbol{\lambda}}_y(\ell + 1) = \tilde{\boldsymbol{R}}^y(\ell)\tilde{\boldsymbol{\lambda}}_y(\ell) + \varepsilon\tilde{\boldsymbol{E}}^y(\ell)\tilde{\boldsymbol{\omega}}_y(\ell), \\ \tilde{\boldsymbol{\omega}}_y(\ell + 1) = \tilde{\boldsymbol{C}}^y(\ell)\tilde{\boldsymbol{\omega}}_y(\ell) + \tilde{\boldsymbol{\mathcal{F}}}_y(\ell + 1) - \tilde{\boldsymbol{\mathcal{F}}}_y(\ell). \end{cases} \tag{5.5}$$

Combining Equation (5.1) and (5.5), the complete augmented algorithm is obtained as follows:

$$\begin{aligned} \tilde{\boldsymbol{\lambda}}(\ell + 1) &= \tilde{\boldsymbol{R}}(\ell)\tilde{\boldsymbol{\lambda}}(\ell) + \varepsilon\tilde{\boldsymbol{E}}(\ell)\tilde{\boldsymbol{\omega}}(\ell), \\ \tilde{\boldsymbol{\omega}}(\ell + 1) &= \tilde{\boldsymbol{C}}(\ell)\tilde{\boldsymbol{\omega}}(\ell) + \tilde{\boldsymbol{\mathcal{F}}}(\ell + 1) - \tilde{\boldsymbol{\mathcal{F}}}(\ell). \end{aligned} \tag{5.6}$$

In the equation above, $\tilde{\boldsymbol{\lambda}}$, $\tilde{\boldsymbol{\omega}}$, and $\tilde{\boldsymbol{\mathcal{F}}}$ are $(\tilde{N}_x + \tilde{N}_y)$ -dimensional vectors, expressed as

$$\tilde{\boldsymbol{\lambda}}(\ell) = \begin{bmatrix} \tilde{\boldsymbol{\lambda}}_x(\ell) \\ \tilde{\boldsymbol{\lambda}}_y(\ell) \end{bmatrix}, \quad \tilde{\boldsymbol{\omega}}(\ell) = \begin{bmatrix} \tilde{\boldsymbol{\omega}}_x(\ell) \\ \tilde{\boldsymbol{\omega}}_y(\ell) \end{bmatrix}, \quad \tilde{\boldsymbol{\mathcal{F}}}(\ell) = \begin{bmatrix} \tilde{\boldsymbol{\mathcal{F}}}_x(\ell) \\ \tilde{\boldsymbol{\mathcal{F}}}_y(\ell) \end{bmatrix}.$$

$\tilde{\boldsymbol{R}}$, $\tilde{\boldsymbol{C}}$, and $\tilde{\boldsymbol{E}}$ are $(\tilde{N}_x + \bar{N}_y) \times (\tilde{N}_x + \bar{N}_y)$ dimensional matrices, which is given by

$$\tilde{\boldsymbol{R}}(\ell) = \begin{bmatrix} \tilde{\boldsymbol{R}}^x(\ell) & 0 \\ 0 & \tilde{\boldsymbol{R}}^y(\ell) \end{bmatrix}, \quad \tilde{\boldsymbol{C}}(\ell) = \begin{bmatrix} \tilde{\boldsymbol{C}}^x(\ell) & 0 \\ 0 & \tilde{\boldsymbol{C}}^y(\ell) \end{bmatrix}, \quad \tilde{\boldsymbol{E}}(\ell) = \begin{bmatrix} \tilde{\boldsymbol{E}}^x(\ell) & 0 \\ 0 & \tilde{\boldsymbol{E}}^y(\ell) \end{bmatrix}. \tag{5.7}$$

Note that if $s^\ell = \infty$, then

$$\tilde{\boldsymbol{R}}^y(\ell) = \tilde{\boldsymbol{C}}^y(\ell) = \boldsymbol{I}, \quad \tilde{\boldsymbol{E}}^y(\ell) = \boldsymbol{0}.$$

If $n^\ell = \infty$, then

$$\tilde{\boldsymbol{R}}^x(\ell) = \tilde{\boldsymbol{C}}^x(\ell) = \boldsymbol{I}, \quad \tilde{\boldsymbol{E}}^x(\ell) = \boldsymbol{0}.$$

Remark 6. *Although both ADDC and Asy-SONATA in [33] consider asynchronous iteration of the agents and utilize augmented graph techniques for convergence analysis, they dgeneration-sideer fundamentally in their algorithm design philosophy and augmented graph construction.*

- *The update rules in Asy-SONATA are complex and involve multiple intermediate steps. However, ADDC is very straightforward in practical deployment, maintaining a clear update equation without intermediate steps.*
- *In ADDC, each device requires only two main variables, which is comparable with synchronous algorithms. This minimal variable set significantly reduces the computational overhead in practical deployment. In contrast, Asy-SONATA introduces two additional auxiliary variables for each device to handle asynchronicity, thus increasing the memory footprint and computational complexity.*
- *Asy-SONATA augments graphs by placing virtual nodes on edges, turning each latency-carrying link into a chain of virtual nodes. ADDC uses a node-centric approach, assigning two dgeneration-sideerent types of virtual agents to each real agent. This dual-proxy design creates independent information paths for the consensus and tracking variables, enabling a more structured analysis of their coupling dynamics. Compared with Asy-SONATA, ADDC's augmented graph structure expands less, and its theoretical analysis is simpler and easier to understand.*

5.3. Important conclusion

Lemma 3. *For any* $(\lambda_x^1, \lambda_y^1)$ *and* $(\lambda_x^2, \lambda_y^2)$*, the following conclusion holds.*

3-1. A constant $\mathcal{L} > 0$ *exists such that*

$$\|\Delta g\|_1 \leq \mathcal{L}\,\|\Delta\lambda\|_1 . \tag{5.8}$$

3-2. If $\varepsilon \in (0, \frac{2}{\mathcal{L}})$*, a constant* $0 < \varrho < 1$ *exists such that*

$$\|\Delta\lambda + \varepsilon\Delta g\|_1 \leq \varrho\,\|\Delta\lambda\|_1 . \tag{5.9}$$

Here

$$\Delta\lambda = \begin{bmatrix} \lambda_x^1 - \lambda_x^2 \\ \lambda_y^1 - \lambda_y^2 \end{bmatrix}, \quad \Delta g = \begin{bmatrix} g_x(\lambda_x^1, \lambda_y^1) - g_x(\lambda_x^2, \lambda_y^2) \\ g_y(\lambda_x^1, \lambda_y^1) - g_y(\lambda_x^2, \lambda_y^2) \end{bmatrix},$$
$$g_x(\lambda_x, \lambda_y) = D_x + \sum_{i\in I_e} \phi_{i,x}^e(\lambda_x) - \sum_{i\in I_p} \phi_{i,x}^p(\lambda_x) - \sum_{i\in I_c} \phi_{i,x}^c(\lambda_x, \lambda_y),$$
$$g_y(\lambda_x, \lambda_y) = D_y + \sum_{i\in I_t} \phi_{i,y}^t(\lambda_y) - \sum_{i\in I_h} \phi_{i,y}^h(\lambda_y) - \sum_{i\in I_c} \phi_{i,y}^c(\lambda_x, \lambda_y).$$

Proof. (1) It folllows from (3.2)–(3.6) that

$$\Delta g = \boldsymbol{T}\Delta\lambda, \quad \boldsymbol{T} = \begin{bmatrix} -\sum\limits_{i\in I_e\cup I_p} \frac{1}{2\alpha_i} - \sum\limits_{i\in I_c} \frac{2a_i}{|\boldsymbol{Q}_i|} & \sum\limits_{i\in I_p} \frac{\varpi_i}{|\boldsymbol{Q}_i|} \\ \sum\limits_{i\in I_p} \frac{\varpi_i}{|\boldsymbol{Q}_i|} & -\sum\limits_{i\in I_t\cup I_h} \frac{1}{2a_i} - \sum\limits_{i\in I_c} \frac{2\alpha_i}{|\boldsymbol{Q}_i|} \end{bmatrix}.$$

Taking the Manhattan norm on both sides, we get

$$\|\Delta g\|_1 = \|\boldsymbol{T}\Delta\lambda\|_1 \leq \|\boldsymbol{T}\|_1\|\Delta\lambda\|_1.$$

Thus, (5.8) is clearly valid for any $\mathcal{L} \geq \|\boldsymbol{T}\|_1$.

(2) Furthermore,

$$\|\Delta\lambda + \varepsilon\Delta g\|_1 = \|\Delta\lambda + \varepsilon\boldsymbol{T}\Delta\lambda\|_1 \leq \|1 + \varepsilon\boldsymbol{T}\|_1\|\Delta\lambda\|_1.$$

According to Lemma 3-2, it is necessary to find a ε such that $\|1 + \varepsilon\boldsymbol{T}\|_1 < 1$, which means

$$|1 + \varepsilon\boldsymbol{T}_{11}| + |\varepsilon\boldsymbol{T}_{12}| < 1, \quad (a),$$
$$|1 + \varepsilon\boldsymbol{T}_{22}| + |\varepsilon\boldsymbol{T}_{21}| < 1, \quad (b).$$

When $1 + \varepsilon\boldsymbol{T}_{11} \geq 0$, namely $0 \leq \varepsilon < \frac{1}{|\boldsymbol{T}_{11}|}$, Condition (a) holds if $|\boldsymbol{T}_{11}| > |\boldsymbol{T}_{12}|$. When $1 + \varepsilon\boldsymbol{T}_{11} < 0$, namely $\varepsilon > \frac{1}{|\boldsymbol{T}_{11}|}$, Condition (a) holds if $\varepsilon < \frac{2}{|\boldsymbol{T}_{12}|+|\boldsymbol{T}_{11}|}$. For the admissible range of ε to be nonempty, we have

$$\frac{1}{|\boldsymbol{T}_{11}|} < \frac{2}{|\boldsymbol{T}_{12}| + |\boldsymbol{T}_{11}|},$$

which simplifies to $|\boldsymbol{T}_{11}| > |\boldsymbol{T}_{12}|$. Therefore, whenever $|\boldsymbol{T}_{11}| > |\boldsymbol{T}_{12}|$, Condition (a) is satisfied for all

$$\varepsilon \in \Big(0, \frac{2}{|\boldsymbol{T}_{12}| + |\boldsymbol{T}_{11}|}\Big).$$

The same argument applies to Condition (a). If $|\boldsymbol{T}_{22}| > |\boldsymbol{T}_{21}|$, then Condition (a) holds for all

$$\varepsilon \in \Big(0, \frac{2}{|\boldsymbol{T}_{21}| + |\boldsymbol{T}_{22}|}\Big).$$

Since $2a_i > \varpi_i$ and $2\alpha_i > \varpi_i$ are assumed, it is clear that $|\boldsymbol{T}_{22}| > |\boldsymbol{T}_{21}|$ and $|\boldsymbol{T}_{11}| > |\boldsymbol{T}_{12}|$. Furthermore, $\|\boldsymbol{T}\|_1 > \max\{|\boldsymbol{T}_{21}| + |\boldsymbol{T}_{22}|, |\boldsymbol{T}_{12}| + |\boldsymbol{T}_{11}|\}$, so there must be a constant $\varrho = \|1 + \varepsilon \boldsymbol{T}\|_1 < 1$ such that (5.9) holds when $\varepsilon \in (0, \frac{2}{\mathcal{L}})$. □

Lemma 4. *[33] If Assumptions 1–3 are satisfied, then the constants $M_c > 1$, $M_\zeta > 0$, and $0 < \rho < 1$ exist such that for any $\ell \in \mathbb{N}$, $n \in \mathcal{N}_x$, and $s \in \mathcal{N}_y$, we have*

$$\|\tilde{\boldsymbol{\omega}}_{n,x}(\ell+1) - \boldsymbol{\xi}^c_{n,x}(\ell)\mathbf{1}^\top \tilde{\boldsymbol{\mathcal{F}}}_x(\ell+1)\| \le M_c\left(\rho^\ell \|\tilde{\boldsymbol{\omega}}_x(0)\| + \sum_{k=0}^{\ell} \rho^{\ell-k}\|\Delta_x(k)\|\right),$$

$$\|\tilde{\boldsymbol{\omega}}_{s,y}(\ell+1) - \boldsymbol{\xi}^c_{s,y}(\ell)\mathbf{1}^\top \tilde{\boldsymbol{\mathcal{F}}}_y(\ell+1)\| \le M_c\left(\rho^\ell \|\tilde{\boldsymbol{\omega}}_y(0)\| + \sum_{k=0}^{\ell} \rho^{\ell-k}\|\Delta_y(k)\|\right),$$

where $\boldsymbol{\xi}^c_x(\ell) \in [M_\zeta, 1)$ and $\boldsymbol{\xi}^c_y(\ell) \in [M_\zeta, 1]$ are random vectorsas well as

$$\Delta_x(k) = \tilde{\boldsymbol{\mathcal{F}}}_x(k+1) - \tilde{\boldsymbol{\mathcal{F}}}_x(k), \quad \Delta_y(k) = \tilde{\boldsymbol{\mathcal{F}}}_y(k+1) - \tilde{\boldsymbol{\mathcal{F}}}_y(k).$$

Lemma 5. *[33] If Assumptions 1–3 are satisfied, a constant $M_r > 1$ and the random vectors $\boldsymbol{\xi}^r_x(\ell) \in [M_\zeta, 1)$ and $\boldsymbol{\xi}^r_y(\ell) \in [M_\zeta, 1)$ exist such that for any $\ell \in \mathbb{N}$, we have*

$$\|\tilde{\boldsymbol{\lambda}}_x(\ell+1) - \mathbf{1}\overline{\lambda}_x(\ell+1)\| \le M_r\left(\rho^\ell \|\tilde{\boldsymbol{\lambda}}_x(0) - \mathbf{1}\overline{\lambda}_x(0)\| + \sum_{k=0}^{\ell} \rho^{\ell-k}\|\Lambda_x(k)\|\right),$$

$$\|\tilde{\boldsymbol{\lambda}}_y(\ell+1) - \mathbf{1}\overline{\lambda}_y(\ell+1)\| \le M_r\left(\rho^\ell \|\tilde{\boldsymbol{\lambda}}_y(0) - \mathbf{1}\overline{\lambda}_y(0)\| + \sum_{k=0}^{\ell} \rho^{\ell-k}\|\Lambda_y(k)\|\right),$$

where constants M_ζ and ρ come from Lemma 4, and

$$\begin{aligned}
&\Lambda_x(\ell) = \varepsilon \tilde{\boldsymbol{E}}_x(\ell)\tilde{\boldsymbol{\omega}}_x(\ell), \qquad \Lambda_y(\ell) = \varepsilon \tilde{\boldsymbol{E}}_y(\ell)\tilde{\boldsymbol{\omega}}_y(\ell),\\
&\overline{\lambda}_x(0) = (\boldsymbol{\xi}^r_x(0))^\top \tilde{\boldsymbol{\lambda}}(0), \quad \overline{\lambda}_x(\ell+1) = \overline{\lambda}_x(0) + \sum_{k=1}^{\ell+1} (\boldsymbol{\xi}^r_x(k))^\top \Lambda_x(k-1),\\
&\overline{\lambda}_y(0) = (\boldsymbol{\xi}^r_y(0))^\top \tilde{\boldsymbol{\lambda}}(0), \quad \overline{\lambda}_y(\ell+1) = \overline{\lambda}_y(0) + \sum_{k=1}^{\ell+1} (\boldsymbol{\xi}^r_y(k))^\top \Lambda_y(k-1).
\end{aligned} \tag{5.10}$$

Theorem 1. *The sequences $\{\boldsymbol{\lambda}_x(\ell)\}$ and $\{\boldsymbol{\lambda}_y(\ell)\}$ of the ADDC algorithm are identical to those produced by* (5.6) *when the initial conditions of* (5.6) *satisfy*

$$\begin{aligned}
&\tilde{\boldsymbol{\lambda}}(0) = [\boldsymbol{\lambda}_x(0); \mathbf{0}; \boldsymbol{\lambda}_y(0); \mathbf{0}],\\
&\tilde{\boldsymbol{\omega}}(0) = \tilde{\boldsymbol{C}}(-1)\tilde{\boldsymbol{\omega}}(-1) + \tilde{\boldsymbol{\mathcal{F}}}(0) - \tilde{\boldsymbol{\mathcal{F}}}(-1).
\end{aligned} \tag{5.11}$$

In (5.11)*,* $\tilde{\boldsymbol{\omega}}(-1)$ *and* $\tilde{\boldsymbol{\mathcal{F}}}(-1)$ *are given by*

$$\tilde{\boldsymbol{\omega}}(-1) = \tilde{\boldsymbol{\mathcal{F}}}(-1) = [\mathbf{Z}_x(0); \mathbf{0}; \mathbf{Z}_y(0); \mathbf{0}],$$

and $\tilde{\boldsymbol{C}}(-1)$ *is a special matrix of* (5.7)*, expressed as*

$$\left[\tilde{\boldsymbol{C}}^x(-1)\right]_{rm} = \begin{cases} C_{nj}, & \text{if } m = j \text{ and } r = Nd + n, T_{nj}^{d-1} \le -1 \le T_{nj}^{d} - 1, \forall n, j \in \mathcal{N}_x, \\ 1, & \text{if } r = m - N_x, m \notin \mathcal{N}_x, \\ 0, & \text{otherwise.} \end{cases}$$

$$\left[\tilde{\boldsymbol{C}}^y(-1)\right]_{rm} = \begin{cases} C_{sj}, & \text{if } m = j \text{ and } r = Nd + s, T_{sj}^{d-1} \le -1 \le T_{sj}^{d} - 1, \forall s, j \in \mathcal{N}_y, \\ 1, & \text{if } r = m - N_y, m \notin \mathcal{N}_y, \\ 0, & \text{otherwise.} \end{cases}$$

Proof. As shown in Section 4.2, Theorem 1 follows immediately if $\{\boldsymbol{\lambda}(\ell)\}$ and $\{\boldsymbol{\omega}(\ell)\}$ of (5.6) are identical to those of form (4.14)–(4.17). Since the regions $\mathcal{G}_x$ and $\mathcal{G}_y$ share the same structure, we restrict attention to $\mathcal{G}_x$ without loss of generality.

(1) The consistency of sequence $\{\boldsymbol{\omega}_{n,x}\}$ will be proven. From (5.1) and (5.11), it holds that

$$\boldsymbol{\omega}_{n,x}(\ell + 1) = \boldsymbol{\omega}_{n,x}^1(\ell) + \boldsymbol{\mathcal{F}}_{n,x}\left(T_{nn}^{\ell+1}\right) - \boldsymbol{\mathcal{F}}_{n,x}\left(T_{nn}^{\ell}\right), \quad \forall\, \ell \ge -1. \tag{5.12}$$

Moreover, for $d \in [1, \sigma - 1]$, one has

$$\boldsymbol{\omega}_{n,x}^d(\ell - d + 1) = \boldsymbol{\omega}_{n,x}^{d+1}(\ell - d) + \mathbf{C}_{nj}^x\, \boldsymbol{\omega}_{j,x}(\ell - d)\chi(\ell, d, n, j), \quad j = n^{\ell-d}. \tag{5.13}$$

Here, $\chi(\ell, d, n, j)$ is an indicator function satisfying

$$\chi(\ell, d, n, j) = \begin{cases} 1, & \text{if } j = n^{\ell-d}, T_{nj}^{\ell} \le \ell - d \le T_{nj}^{\ell+1} - 1, \\ 0, & \text{otherwise.} \end{cases} \tag{5.14}$$

Since $\boldsymbol{\omega}_{n,x}(\ell) = 0$ with $n \ne n^{\ell}$, so $\boldsymbol{\omega}_{n,x}^{\sigma}(\ell - \sigma + 1)$ can be expressed as

$$\boldsymbol{\omega}_{n,x}^{\sigma}(\ell - \sigma + 1) = \mathbf{C}_{nj}^x\, \boldsymbol{\omega}_{j,x}(\ell - \sigma)\chi(\ell, \sigma, n, j), \quad j = n^{\ell-\sigma}. \tag{5.15}$$

Depending whether the agent n_x is activated at time $\ell + 1$, $\boldsymbol{\omega}_x(\ell)$ is divided into two categories, and their consistency is proven separately.

(1-a) If $n \ne n^{\ell+1}$, then

$$T_{nj}^{\ell} = T_{nj}^{\ell+1}, \quad j \in \mathcal{N}_n^+. \tag{5.16}$$

It follows from (5.14) that $\chi(\ell, d, n, j) = 0$ for any $j \in \mathcal{N}_x$ and $d \in [1, \sigma]$. As a result, we have

$$\boldsymbol{\omega}_{n,x}^1(\ell) = \boldsymbol{\omega}_{n,x}^2(\ell - 1) = \cdots = \boldsymbol{\omega}_{n,x}^{\sigma}(\ell - \sigma + 1) = 0. \tag{5.17}$$

Substituting (5.16) and (5.17) into (5.12) yields

$$\boldsymbol{\omega}_{n,x}(\ell + 1) = 0, \quad i \ne n^{\ell+1}, \tag{5.18}$$

which is consistent with (4.15).

(1-b) If $n = n^{\ell+1} \neq \infty$, then

$$T_{nn}^{\ell} \leq T_{nn}^{\ell+1} - 1 = \ell.$$

Define $d_n^{\ell} = \ell - T_{nn}^{\ell}$. If $d_n^{\ell} = \sigma$, it follows from $n^{\ell - d_n^{\ell}} = n$ and (5.15) that

$$\boldsymbol{\omega}_{n,x}^{\sigma}(\ell - \sigma + 1) = \mathbf{C}_{nn}^{x} \boldsymbol{\omega}_{n,x}\left(T_{nn}^{\ell}\right).$$

If $d_n^{\ell} \neq \sigma$, from (5.13), it yields

$$\boldsymbol{\omega}_{n,x}^{d_n^{\ell}}(\ell - d_n^{\ell} + 1) = \boldsymbol{\omega}_{n,x}^{d_n^{\ell}+1}(\ell - d_n^{\ell}) + \mathbf{C}_{nn}^{x} \boldsymbol{\omega}_{n,x}\left(T_{nn}^{\ell}\right). \tag{5.19}$$

For some $j \in \mathcal{N}_n^+$ that satisfies $j = n^k$ with $T_{nj}^{\ell} \leq k \leq T_{nj}^{\ell+1} - 1$, one has

$$\boldsymbol{\omega}_{n,x}^{d_j^k}(\ell - d_j^k + 1) = \boldsymbol{\omega}_{n,x}^{d_j^k+1}(\ell - d_j^k) + \mathbf{C}_{nj}^{x} \boldsymbol{\omega}_{j,x}(k), \quad d_j^k = \ell - k. \tag{5.20}$$

For the remaining $d \in [1, \sigma - 1] \setminus \{d_l^{\ell}, d_j^k\}$, one has

$$\boldsymbol{\omega}_{n,x}^{d}(\ell - d + 1) = \boldsymbol{\omega}_{n,x}^{d+1}(\ell - d). \tag{5.21}$$

Substituting (5.19)–(5.21) into (5.13) yields

$$\boldsymbol{\omega}_{n,x}^{1}(\ell) = \mathbf{C}_{nn}^{x} \boldsymbol{\omega}_{n,x}(T_{nn}^{\ell}) + \sum_{j \in \mathcal{N}_n^+} \sum_{k=T_{nj}^{\ell}}^{T_{nj}^{\ell+1}-1} \mathbf{C}_{nj}^{x} \boldsymbol{\omega}_{j,x}(k), \tag{5.22}$$

where $\boldsymbol{\omega}_{j,x}(k) = 0$ for $j \neq n^k$ is used. We can obtain (4.14) by substituting (5.22) into (5.12).

(2) The consistency of sequence $\{\boldsymbol{\lambda}_{n,x}\}$ will be proven. From (5.2), when $n \neq n^{\ell}$, it holds that

$$\boldsymbol{\lambda}_{n,x}(\ell + 1) = \boldsymbol{\lambda}_{n,x}(\ell),$$

which coincides with the iteration in (4.15).

When $n = n^{\ell}$, we have $T_{nn}^{\ell} = \ell$ and

$$\boldsymbol{\lambda}_{n,x}(\ell + 1) = \mathbf{R}_{nn}^{x} \boldsymbol{\lambda}_{n,x}(\ell) + \sum_{j \in \mathcal{N}_n^+} \mathbf{R}_{nj}^{x} \boldsymbol{\lambda}_{j,x}^{d_j^{\ell}}(\ell) + \varepsilon \boldsymbol{\omega}_{n,x}(\ell). \tag{5.23}$$

Here, $d_j^{\ell} = \ell - T_{nj}^{\ell} \leq \sigma$. Under the initial condition (5.11), for any $\ell \geq 0$ and $1 \leq d \leq \sigma$,

$$\boldsymbol{\lambda}_{j,x}^{d}(\ell) = \boldsymbol{\lambda}_{j,x}(\ell - d). \tag{5.24}$$

Thus, (5.23) is rewritten as

$$\boldsymbol{\lambda}_{n,x}(\ell + 1) = \mathbf{R}_{nn}^{x} \boldsymbol{\lambda}_{n,x}(\ell) + \sum_{j \in \mathcal{N}_n^+} \mathbf{R}_{nj}^{x} \boldsymbol{\lambda}_{j,x}\left(T_{nj}^{\ell}\right) + \varepsilon \boldsymbol{\omega}_{n,x}(\ell),$$

which is identical to the update of $\boldsymbol{\lambda}_{n,x}$ in (4.14). □

Theorem 2. *Define the error vector* $\boldsymbol{\delta}(\ell) = (\boldsymbol{\delta}_1(\ell), \boldsymbol{\delta}_2(\ell), \boldsymbol{\delta}_3(\ell), \boldsymbol{\delta}_4(\ell))^\top$, *given by*

$$\begin{aligned}
\boldsymbol{\delta}_1(\ell) &= \|\tilde{\boldsymbol{\lambda}}_x(\ell) - \mathbf{1}\overline{\lambda}_x(\ell)\| + \|\tilde{\boldsymbol{\lambda}}_y(\ell) - \mathbf{1}\overline{\lambda}_y(\ell)\|,\\
\boldsymbol{\delta}_2(\ell) &= |\overline{\lambda}_x(\ell) - \lambda_x^\star| + |\overline{\lambda}_y(\ell) - \lambda_y^\star|,\\
\boldsymbol{\delta}_3(\ell) &= |\tilde{\boldsymbol{\omega}}_{n^\ell,x}(\ell) - \boldsymbol{\xi}^c_{n^\ell,x}(\ell-1)\mathbf{1}^\top\tilde{\boldsymbol{\mathcal{F}}}_x(\ell)|,\\
&\quad + |\tilde{\boldsymbol{\omega}}_{s^\ell,y}(\ell) - \boldsymbol{\xi}^c_{s^\ell,y}(\ell-1)\mathbf{1}^\top\tilde{\boldsymbol{\mathcal{F}}}_y(\ell)|,\\
\boldsymbol{\delta}_4(\ell) &= |\tilde{\boldsymbol{\omega}}_{n^\ell,x}(\ell)| + |\tilde{\boldsymbol{\omega}}_{s^\ell,y}(\ell)|.
\end{aligned}$$

Under Assumptions 1–3, if the step size ε *belongs to* $(0, \frac{2}{\mathcal{L}})$, *the error vector* $\boldsymbol{\delta}(\ell)$ *satisfies the following recurrence relation*

$$\boldsymbol{\delta}_1(\ell+1) \le M_r\boldsymbol{\delta}_1(0)\rho^\ell + \varepsilon M_r\sum_{k=0}^{\ell}\rho^{\ell-k}\boldsymbol{\delta}_4(k), \tag{5.25}$$

$$\boldsymbol{\delta}_2(\ell+1) \le \varrho^\ell\boldsymbol{\delta}_2(0) + \sum_{k=0}^{\ell}\varrho^{\ell-k}[\mathcal{L}\varepsilon\boldsymbol{\delta}_1(k) + \varepsilon\boldsymbol{\delta}_3(k)], \tag{5.26}$$

$$\boldsymbol{\delta}_3(\ell+1) \le 2M_c\rho^\ell\tilde{\boldsymbol{\omega}}(0) + \sum_{k=0}^{\ell}\rho^{\ell-k}(6M_c\mathcal{L}\boldsymbol{\delta}_1(k) + 2M_c\mathcal{L}\varepsilon\boldsymbol{\delta}_4(k)), \tag{5.27}$$

$$\boldsymbol{\delta}_4(\ell) \le \boldsymbol{\delta}_3(\ell) + \mathcal{L}\boldsymbol{\delta}_1(\ell) + \mathcal{L}\boldsymbol{\delta}_2(\ell). \tag{5.28}$$

Here, the constants $\mathcal{L}$ *and* ϱ *are defined in Lemma 3; the others in Lemmas 4 and 5.*

Proof. (1) It follows from (5.10) that $\|\Lambda_x(\ell)\| = \varepsilon|\tilde{\boldsymbol{\omega}}_{n^\ell,x}(\ell)|$, $\|\Lambda_y(\ell)\| = \varepsilon|\tilde{\boldsymbol{\omega}}_{s^\ell,y}(\ell)|$. Combining this with Lemma 5, (5.25) can be directly obtained.

(2) From (5.10), it holds that

$$\begin{aligned}
\overline{\lambda}_x(\ell+1) &= \overline{\lambda}_x(\ell) + \varepsilon\boldsymbol{\xi}^r_{n^\ell,x}(\ell+1)\tilde{\boldsymbol{\omega}}_{n^\ell}(\ell),\\
\overline{\lambda}_y(\ell+1) &= \overline{\lambda}_y(\ell) + \varepsilon\boldsymbol{\xi}^r_{s^\ell,y}(\ell+1)\tilde{\boldsymbol{\omega}}_{s^\ell}(\ell).
\end{aligned}$$

Substituting these into $\|\overline{\lambda}_x(\ell+1) - \lambda_x^\star\|$ and $\|\overline{\lambda}_y(\ell+1) - \lambda_y^\star\|$, one has

$$\begin{aligned}
|\overline{\lambda}_x(\ell+1) - \lambda_x^\star| &\le |\overline{\lambda}_x(\ell) + \varepsilon\boldsymbol{\xi}^r_{n^\ell,x}(\ell+1)\boldsymbol{\xi}^c_{n^\ell,x}(\ell-1)g_x(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell)) - \lambda_x^\star|\\
&\quad + \varepsilon\boldsymbol{\xi}^r_{n^\ell,x}(\ell+1)\boldsymbol{\xi}^c_{n^\ell,x}(\ell-1)|\mathbf{1}^\top\tilde{\boldsymbol{\mathcal{F}}}_x(\ell) - g_x(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell))|\\
&\quad + \varepsilon|\boldsymbol{\xi}^r_{n^\ell,x}(\ell+1)|\,|(\tilde{\boldsymbol{\omega}}_{n^\ell,x}(\ell) - \boldsymbol{\xi}^c_{n^\ell,x}(\ell-1)\mathbf{1}^\top\tilde{\boldsymbol{\mathcal{F}}}_x(\ell)|,
\end{aligned}$$

$$\begin{aligned}
|\overline{\lambda}_y(\ell+1) - \lambda_y^\star| &\le |\overline{\lambda}_y(\ell) + \varepsilon\boldsymbol{\xi}^r_{s^\ell,y}(\ell+1)\boldsymbol{\xi}^c_{s^\ell,y}(\ell-1)g_y(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell)) - \lambda_y^\star|\\
&\quad + \varepsilon\boldsymbol{\xi}^r_{s^\ell,y}(\ell+1)\boldsymbol{\xi}^c_{s^\ell,y}(\ell-1)|\mathbf{1}^\top\tilde{\boldsymbol{\mathcal{F}}}_y(\ell) - g_y(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell))|\\
&\quad + \varepsilon|\boldsymbol{\xi}^r_{s^\ell,y}(\ell+1)|\,|(\tilde{\boldsymbol{\omega}}_{s^\ell,y}(\ell) - \boldsymbol{\xi}^c_{s^\ell,y}(\ell-1)\mathbf{1}^\top\tilde{\boldsymbol{\mathcal{F}}}_y(\ell)|.
\end{aligned}$$

According to Lemma 3-2, $\boldsymbol{\xi}^c_{n,x} < 1$, $\boldsymbol{\xi}^r_{n,x} < 1$, $\boldsymbol{\xi}^c_{s,y} < 1$, and $\boldsymbol{\xi}^r_{s,y} < 1$, we have

$$\boldsymbol{\delta}_2(\ell+1) \le \varrho\boldsymbol{\delta}_2(\ell) + \varepsilon\boldsymbol{\delta}_3(\ell) + \varepsilon(|\mathbf{1}^\top\tilde{\boldsymbol{\mathcal{F}}}_x(\ell) - g_x(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell))| + |\mathbf{1}^\top\tilde{\boldsymbol{\mathcal{F}}}_y(\ell) - g_y(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell))|). \tag{5.29}$$

It follows from (5.24) that for any $n \in \mathcal{N}_x$ and $s \in \mathcal{N}_y$, $i_n \in \tilde{\mathcal{N}}_x$ and $i_s \in \tilde{\mathcal{N}}_y$ exist such that

$$\mathbf{P}_{n,x}(\boldsymbol{\lambda}_{n,x}(T^{\ell}_{nn})) = \mathbf{P}_{n,x}(\tilde{\boldsymbol{\lambda}}_{i_n,x}(\ell)), \quad \mathbf{P}_{s,y}(\boldsymbol{\lambda}_{s,y}(T^{\ell}_{ss})) = \mathbf{P}_{s,y}(\tilde{\boldsymbol{\lambda}}_{i_s,y}(\ell)).$$

According to the distance shrinkage of the projection and the Cauchy–Schwartz inequality, it holds that

$$\begin{aligned}
&|\mathbf{1}^{\top}\tilde{\boldsymbol{\mathcal{F}}}_x(\ell) - g_x(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell))| + |\mathbf{1}^{\top}\tilde{\boldsymbol{\mathcal{F}}}_y(\ell) - g_y(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell))| \\
&\leq \sum_{n=1}^{N_x} \mathcal{L}^x_n |\tilde{\boldsymbol{\lambda}}_{i_n,x}(\ell) - \overline{\lambda}_x(\ell)| + \sum_{s=1}^{N_y} \mathcal{L}^y_s |\tilde{\boldsymbol{\lambda}}_{i_s,y}(\ell) - \overline{\lambda}_y(\ell)| \\
&\leq \sum_{n=1}^{N_x} \mathcal{L}^x_n \|\tilde{\boldsymbol{\lambda}}_x(\ell) - \mathbf{1}\overline{\lambda}_x(\ell)\| + \sum_{s=1}^{N_y} \mathcal{L}^y_s \|\tilde{\boldsymbol{\lambda}}_y(\ell) - \mathbf{1}\overline{\lambda}_y(\ell)\| \\
&\leq \mathcal{L}\boldsymbol{\delta}_3(\ell).
\end{aligned} \tag{5.30}$$

Here, $\mathcal{L}^x_n$ and $\mathcal{L}^y_s$ are the first-order coefficients of the optimal response function. Substitute (5.30) into (5.29) to obtain

$$\boldsymbol{\delta}_2(\ell+1) \leq \varrho\boldsymbol{\delta}_2(\ell) + \mathcal{L}\varepsilon\boldsymbol{\delta}_1(\ell) + \varepsilon\boldsymbol{\delta}_3(\ell).$$

By repeatedly iterating, we obtain Equation (5.26).

(3) According to Lemma 4,

$$\boldsymbol{\delta}_3(\ell+1) \leq M_c\left(\rho^{\ell}(\|\tilde{\boldsymbol{\omega}}_x(0)\| + \|\tilde{\boldsymbol{\omega}}_y(0)\|) + \sum_{k=0}^{\ell} \rho^{\ell-k}(\|\Delta_x(k)\| + \|\Delta_y(k)\|)\right). \tag{5.31}$$

Simplifying and rearranging the last item, this yields

$$\begin{aligned}
\|\Delta_x(\ell)\| + \|\Delta_y(\ell)\| &= \|\tilde{\boldsymbol{\mathcal{F}}}_x(\ell+1) - \tilde{\boldsymbol{\mathcal{F}}}_x(\ell)\| + \|\tilde{\boldsymbol{\mathcal{F}}}_y(\ell+1) - \tilde{\boldsymbol{\mathcal{F}}}_y(\ell)\| \\
&\leq 2\mathcal{L}(\|\tilde{\boldsymbol{\lambda}}_x(\ell+1) - \tilde{\boldsymbol{\lambda}}_x(\ell)\| + \|\tilde{\boldsymbol{\lambda}}_y(\ell+1) - \tilde{\boldsymbol{\lambda}}_y(\ell)\|) \\
&\leq 2\mathcal{L}(\|(\mathbf{R}^x(\ell) - \boldsymbol{I})\| \, \|\tilde{\boldsymbol{\lambda}}_x(\ell) - \mathbf{1}\overline{\lambda}_x(\ell)\| + \varepsilon|\tilde{\boldsymbol{\omega}}_{n^{\ell},x}(\ell)|) \\
&+ 2\mathcal{L}(\|(\mathbf{R}^y(\ell) - \boldsymbol{I})\| \, \|\tilde{\boldsymbol{\lambda}}_y(\ell) - \mathbf{1}\overline{\lambda}_y(\ell)\| + \varepsilon|\tilde{\boldsymbol{\omega}}_{s^{\ell},y}(\ell)|) \\
&\leq 6\mathcal{L}\boldsymbol{\delta}_1(\ell) + 2\mathcal{L}\varepsilon\boldsymbol{\delta}_4(\ell).
\end{aligned} \tag{5.32}$$

In this derivation, the first inequality uses the nonexpansive property of the projection operator, and the last inequality uses $\|(\mathbf{R}^x(\ell) - \boldsymbol{I})\| < \|(\mathbf{R}^x(\ell)\| + \|\boldsymbol{I}\| < 3$. Equation (5.27) can be obtained by combining (5.31) and (5.32).

(4) It is clear that

$$\begin{aligned}
|\tilde{\boldsymbol{\omega}}_{n^{\ell}}(\ell)| &\leq |\tilde{\boldsymbol{\omega}}_{n^{\ell},x}(\ell) - \boldsymbol{\xi}^{c}_{n^{\ell},x}(\ell-1)\mathbf{1}^{\top}\tilde{\boldsymbol{\mathcal{F}}}_x(\ell)| + |\mathbf{1}^{\top}\tilde{\boldsymbol{\mathcal{F}}}_x(\ell) - g_x(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell))| \\
&\quad + |g_x(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell)) - g_x(\lambda^{\star}_x, \lambda^{\star}_y)|,
\end{aligned}$$

$$\begin{aligned}
|\tilde{\boldsymbol{\omega}}_{s^{\ell}}(\ell)| &\leq |\tilde{\boldsymbol{\omega}}_{s^{\ell},y}(\ell) - \boldsymbol{\xi}^{c}_{s^{\ell},y}(\ell-1)\mathbf{1}^{\top}\tilde{\boldsymbol{\mathcal{F}}}_y(\ell)| + |\mathbf{1}^{\top}\tilde{\boldsymbol{\mathcal{F}}}_y(\ell) - g_y(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell))| \\
&\quad + |g_y(\overline{\lambda}_x(\ell), \overline{\lambda}_y(\ell)) - g_y(\lambda^{\star}_x, \lambda^{\star}_y)|.
\end{aligned}$$

From Lemma 3, we obtain (5.28). □

Theorem 3. *Under Assumptions 1–3, if the step size satisfies*

$$0 < \varepsilon < \frac{(1-\rho)^2(1-\varrho)}{240M_c^2M_r\mathcal{L}} < \frac{2}{\mathcal{L}},$$

the sequences $\{\mathbf{P}_x(\ell)\}$ *and* $\{\mathbf{P}_y(\ell)\}$ *of ADDC algorithm converge linearly to the optimal solution of problem P.*

Proof. Let $\eta = \frac{1+\max\{\rho,\,\varrho\}}{2} < 1$. Dividing both sides of (5.25) by $\eta^{\ell+1}$ yields

$$\frac{\boldsymbol{\delta}_1(\ell+1)}{\eta^{\ell+1}} \le \frac{M_r}{\eta}\Big(\frac{\rho}{\eta}\Big)^{\ell}\boldsymbol{\delta}_1(0) + \varepsilon\frac{M_r}{\eta}\sum_{k=0}^{\ell}\Big(\frac{\rho}{\eta}\Big)^{\ell-k}\frac{\boldsymbol{\delta}_4(k)}{\eta^k}. \tag{5.33}$$

As a result, we have

$$\boldsymbol{\delta}_1^{\eta,N} \le \frac{M_r\boldsymbol{\delta}_1(0)}{\eta} + \frac{\varepsilon M_r}{\eta}\boldsymbol{\delta}_4^{\eta,N}\sum_{k=0}^{\ell}\Big(\frac{\rho}{\eta}\Big)^{k}.$$

After simplification and rearrangement, we get

$$\boldsymbol{\delta}_1^{\eta,N} \le \frac{M_r\boldsymbol{\delta}_1(0)}{\eta} + \frac{\varepsilon M_r}{\eta-\rho}\boldsymbol{\delta}_4^{\eta,N}. \tag{5.34}$$

Applying the same scaling to (5.26) and (5.27) leads to

$$\begin{aligned}
\boldsymbol{\delta}_2^{\eta,N} &\le \frac{\boldsymbol{\delta}_2(0)}{\eta} + \frac{\varepsilon\mathcal{L}}{\eta-\varrho}\boldsymbol{\delta}_1^{\eta,N} + \frac{\varepsilon}{\eta-\varrho}\boldsymbol{\delta}_3^{\eta,N},\\
\boldsymbol{\delta}_3^{\eta,N} &\le \frac{2M_r\tilde{\boldsymbol{\omega}}(0)}{\eta} + \frac{6M_c\mathcal{L}}{\eta-\rho}\boldsymbol{\delta}_1^{\eta,N} + \frac{2M_c\mathcal{L}\varepsilon}{\eta-\rho}\boldsymbol{\delta}_4^{\eta,N}.
\end{aligned} \tag{5.35}$$

Next, dividing (5.28) by η^{ℓ} gives

$$\frac{\boldsymbol{\delta}_4(\ell)}{\eta^{\ell}} \le \frac{\boldsymbol{\delta}_3(\ell)}{\eta^{\ell}} + \mathcal{L}\frac{\boldsymbol{\delta}_1(\ell)}{\eta^{\ell}} + \mathcal{L}\frac{\boldsymbol{\delta}_2(\ell)}{\eta^{\ell}}. \tag{5.36}$$

Hence

$$\boldsymbol{\delta}_4^{\eta,N} \le \mathcal{L}\boldsymbol{\delta}_1^{\eta,N} + \mathcal{L}\boldsymbol{\delta}_2^{\eta,N} + \boldsymbol{\delta}_3^{\eta,N}. \tag{5.37}$$

Combining (5.34)–(5.37), it holds that

$$\boldsymbol{\delta}^{\eta,N} \le \boldsymbol{H}\boldsymbol{\delta}^{\eta,N} + \boldsymbol{\delta}_0 \le \boldsymbol{H}^N\boldsymbol{\delta}^{\eta,N} + \sum_{k=0}^{N-1}\boldsymbol{H}^k\boldsymbol{\delta}_0, \tag{5.38}$$

where $\mu_1 = \eta - \rho$, $\mu_2 = \eta - \varrho$, and

$$\boldsymbol{H} = \begin{bmatrix} 0 & 0 & 0 & \frac{\varepsilon M_r}{\mu_1} \\ \frac{\mathcal{L}\varepsilon}{\mu_2} & 0 & \frac{\varepsilon}{\mu_2} & 0 \\ \frac{6M_c\mathcal{L}}{\mu_1} & 0 & 0 & \frac{2M_c\mathcal{L}\varepsilon}{\mu_1} \\ \mathcal{L} & \mathcal{L} & 1 & 0 \end{bmatrix}, \quad \boldsymbol{\delta}_0 = \begin{bmatrix} \frac{M_r\boldsymbol{\delta}_1(0)}{\eta} \\ \frac{\boldsymbol{\delta}_2(0)}{\eta} \\ \frac{2M_c\|\tilde{\boldsymbol{\omega}}(0)\|}{\eta} \\ 0 \end{bmatrix}.$$

When $0 < \varepsilon \leq \frac{(1-\rho)^2(1-\varrho)}{240M_c^2M_r\mathcal{L}}$, a positive vector $\boldsymbol{X} = (\frac{\mu_1\mu_2}{6M_c\mathcal{L}}, \frac{1}{6\mathcal{L}\mu_2}, 2\mu_2, 5M_c)^\top$ exists such that

$$\boldsymbol{HX} - \boldsymbol{X} < \begin{bmatrix} 0 \\ \frac{(12M_c\mu_1^2+\mu_1^3)\mu_2^2-30M_c^3M_r}{180M_c^3\mathcal{L}M_r\mu_2} \\ -\frac{\mu_2(3M_r-\mu_1)}{3M_r} \\ \frac{(12M_c\mu_1+\mu_1)\mu_2-30M_c^2M_r+\frac{M_c}{\mu_2}}{6M_c\mu_2} \end{bmatrix} < \begin{bmatrix} 0 \\ 13M_c - 30M_c^3M_r \\ 0 \\ 9M_c - 30M_c^2 \end{bmatrix} < \mathbf{0}.$$

In this derivation, the first inequality uses $\varepsilon < \frac{\mu_1^2\mu_2}{30M_c^2M_r\mathcal{L}}$, and the second inequality uses $\max\{\mu_1, \mu_2\} < 1 < M_c$. Therefore, the spectral radius of matrix $\boldsymbol{H}$ is less than 1 when $0 < \varepsilon \leq \frac{(1-\rho)^2(1-\varrho)}{240M_c^2M_r\mathcal{L}}$. In this case,

$$\boldsymbol{H}^\infty = \mathbf{0}, \quad \sum_{k=0}^{\infty} \boldsymbol{H}^k = (\mathbf{I} - \boldsymbol{H})^{-1}.$$

Taking the limit as $N \to \infty$ in (5.38), one obtains

$$\boldsymbol{\delta}^\eta \leq \boldsymbol{H}^\infty\boldsymbol{\delta}^\eta + \sum_{k=0}^{\infty} \boldsymbol{H}^k\boldsymbol{\delta}_0 \leq (\mathbf{I} - \boldsymbol{H})^{-1}\boldsymbol{\delta}_0.$$

The inequality above implies that there is a constant $M_1 > 0$ such that

$$\|\tilde{\boldsymbol{\lambda}}_x(\ell) - \mathbf{1}\boldsymbol{\lambda}_x^\star\| + \|\tilde{\boldsymbol{\lambda}}_y(\ell) - \mathbf{1}\boldsymbol{\lambda}_y^\star\| \leq M_1\eta^\ell.$$

Moreover

$$\begin{aligned} &\|\mathbf{P}_x(\ell) - \mathbf{P}_x^\star\| + \|\mathbf{P}_y(\ell) - \mathbf{P}_y^\star\| \\ &< \mathcal{L}\Big(\|\boldsymbol{\lambda}_x(\ell) - \mathbf{1}\boldsymbol{\lambda}_x^\star\| + \|\boldsymbol{\lambda}_y(\ell) - \mathbf{1}\boldsymbol{\lambda}_y^\star\|\Big) \\ &\leq \mathcal{L}M_1\eta^\ell. \end{aligned}$$

As a result, $\{\mathbf{P}_x(\ell)\}$ and $\{\mathbf{P}_y(\ell)\}$ of the ADDC algorithm linearly convergent to the optimal solution of problem P. □

6. Simulation

This section aims to confirm both the effectiveness and the superiority of the ADDC algorithm using the EH-EMS 65 model as a benchmark. Specifically, Subsection 6.1 introduces the EH-EMS 65 test model and its parameters. Subsection 6.2 evaluates stability, constraint satisfaction, and convergence performance of ADDC. Following this, a demonstration of the algorithm's robustness in handling plug-and-play operations and load jumps is shown in Subsection 6.3. Finally, Subsection 6.4 presents a detailed analysis that underscores its superior convergence efficiency.

6.1. Model settings

The EH-EMS 65 system consists of an IEEE 33-node power system, a Bali 32-node thermal system, and two CHP units. In the power system, the 33 nodes are classified into 3 POD nodes, 2 FEL nodes,

and 26 fixed electrical load nodes. The thermal system consists of 32 nodes, including 4 HOD nodes, 2 FTL nodes, and 24 fixed thermal load nodes. One CHP connects electrical node 2 and thermal node 1, and another CHP connects electrical node 13 and thermal node 9. The detailed topology structure of the EH-EMS 65 system is depicted in Figure 6.

The communication network of the EH-EMS 65 system adopts a distributed architecture, as shown in Figure 7. It includes a built-in protocol conversion module that can convert device data from dgeneration-sideerent protocols, such as IEC 61850, Modbus-TCP, and OPC unified architecture, into a unified data stream, ensuring the interoperability of electrical and thermal data.

The detailed parameters of each device are listed in Tables 1–3, and the system operates under fixed electrical and thermal loads of 600 kW and 900 kW. The computation time of device follows a uniform distribution in [5, 20] ms, which can vary across iterations. Similarly, the information delay in communication is uniformly distributed in [0, 8] ms and can also vary. To ensure a fair comparison, all algorithms adopt the same step size of 0.011 unless otherwise specified.

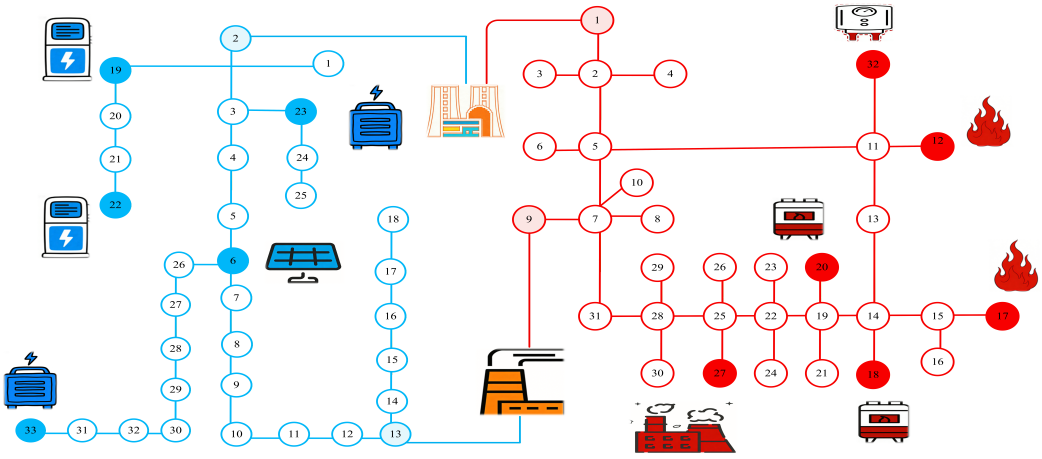


Figure 6. The topology of the EH-EMS 65 system.

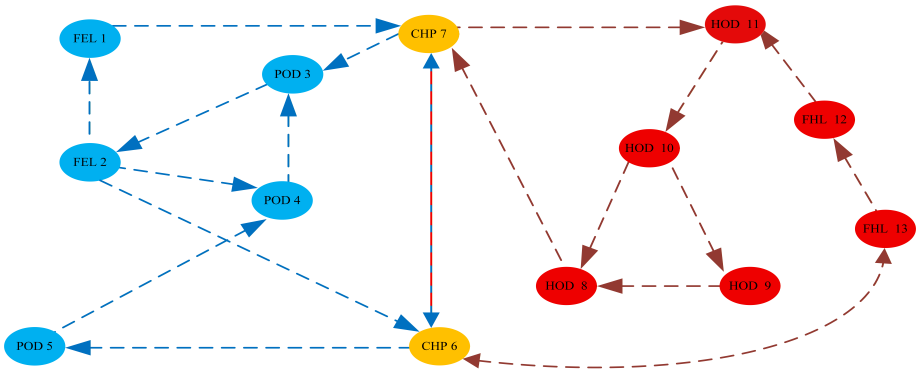


Figure 7. Communication network of the EH-EMS 65 system.

Table 1. The FEL and POD parameters.

FEL	α_i	β_i	$--$	x_i^{min}	x_i^{max}
1	0.079	36.97	$--$	10	170
2	0.072	25.5	$--$	30	140
POD	α_i	β_i	γ_i	x_i^{min}	x_i^{max}
3	0.051	11.15	10.1	10	250
4	0.062	10.3	8.9	40	200
5	0.065	22.4	7.8	20	210

Table 2. The CHP parameters.

CHP	α_i	β_i	γ_i	a_i	b_i	ρ_i
6	0.048	22.3	17.8	0.20	6.4	0.032
7	0.04	18.1	16	0.023	5.9	0.042
CHP	$A_i^\top$			$B_i^\top$		
6	$\begin{bmatrix} -33 & -14 & 49 & -1 \\ -1 & 41 & -2 & -19 \end{bmatrix}$			$\begin{bmatrix} -345 & 3710 & 9790 & -295 \end{bmatrix}$		
7	$\begin{bmatrix} -192 & 5 & 74 & -6 \\ 5 & 24 & -45 & -155 \end{bmatrix}$			$\begin{bmatrix} -920 & 4850 & 11750 & -1270 \end{bmatrix}$		

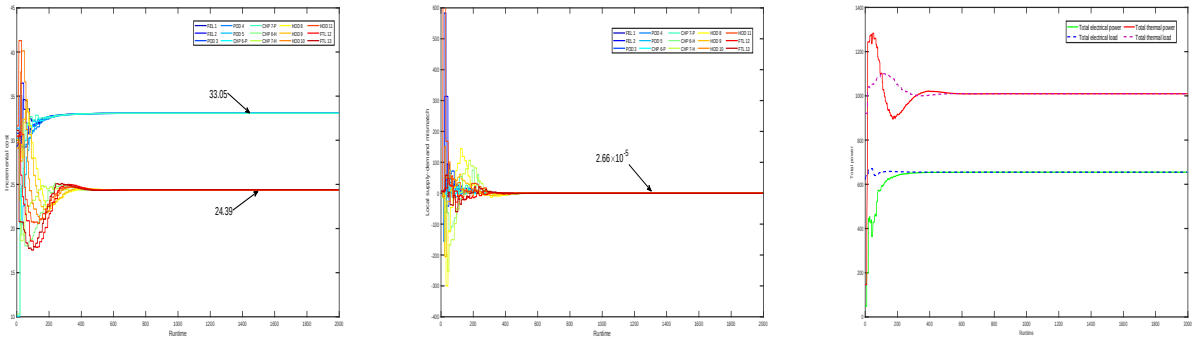
Table 3. The HOD and FTL parameters.

HOD	a_i	b_i	c_i	y_i^{min}	y_i^{max}
8	0.031	14.5	20.1	20	300
9	0.04	13	10.4	30	210
10	0.028	11.4	8.56	15	300
11	0.036	12.7	20.56	40	230
FTL	a_i	b_i	$--$	y_i^{min}	y_i^{max}
12	0.062	29.7	$--$	20	210
13	0.063	32.81	$--$	32	240

6.2. Basic performance testing

The stability and constraint satisfaction of the ADDC algorithm are verified in this part. Figure 8 presents the time evolution of the key system variables. In Figure 8a, the incremental costs $\boldsymbol{\lambda}_x$ and $\boldsymbol{\lambda}_y$ adjust rapidly within the first 500 ms and converge to approximately 33.05 and 24.39, respectively. Figure 8b depicts the supply–demand mismatch $\boldsymbol{\omega}$, which diminishes quickly and stabilizes near zero. Figure 8c confirms that the ADDC's power output meets the total electrical and thermal loads. Figure 9

then verifies constraint satisfaction. The green polygons represent the feasible operating regions, and the red dots denote the power outputs of ADDC. All operating points lie within their respective feasible regions, demonstrating the effectiveness of the optimal response function in handling constraints.



(a) Incremental cost **(b)** Supply–demand mismatch **(c)** Total supply–demand mismatch

Figure 8. Time evolution of the ADDC algorithm.

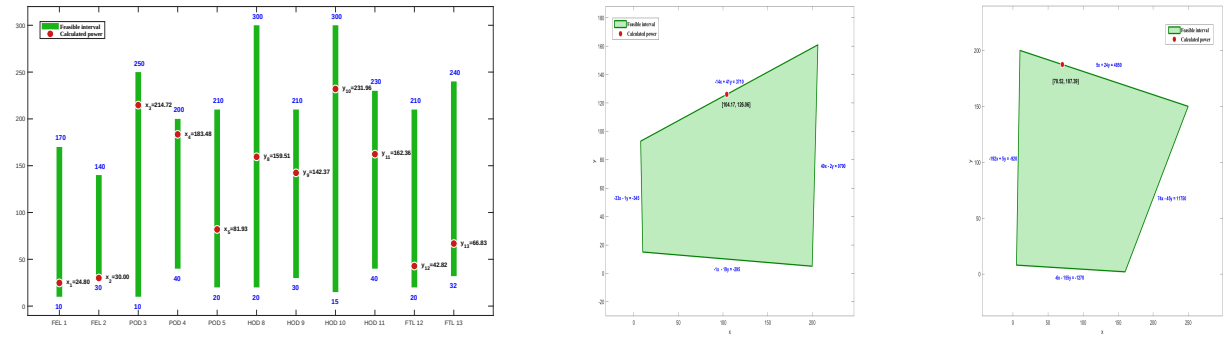
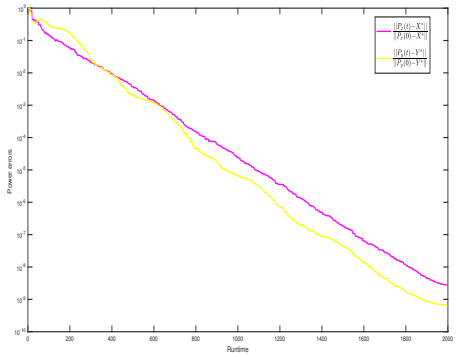
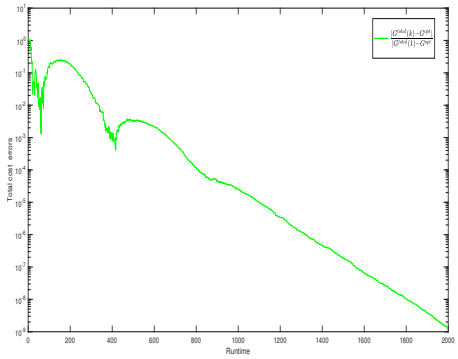


Figure 9. Feasibility of the device.

To further illustrate the steady-state accuracy of the ADDC algorithm, we used the centralized approach to compute the optimal power and optimal operating costs. Figure 10a illustrates the convergence curves of the relative errors for power and total cost. As can be observed, the algorithm achieves high-precision convergence within 2000 ms. Specifically, the relative electrical and thermal errors were as low as 2.73×10^{-9} and 6.46×10^{-10}, respectively, while the total cost error dropped to 1.26×10^{-9}. Furthermore, the logarithmic scale in Figure 10b demonstrates the linear convergence rate of the ADDC algorithm, which aligns with the theoretical results presented in Theorem 3.



(a) Relative errors of power output

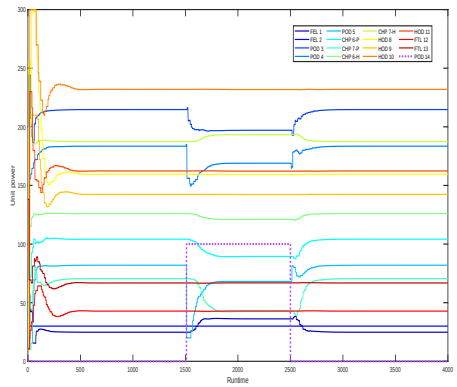


(b) Relative errors of total cost

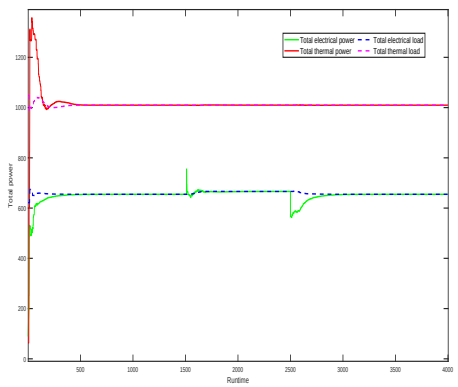
Figure 10. Two comparisons between the ADDC and centralized algorithms.

6.3. Plug-and-play and load jump transition

First, the plug-and-play capability of the algorithm was validated. At t=1500 ms, a new wind power generation unit, namely POD 14 with $\alpha_{14} = 0.002$, $\beta_{14} = 5$, $\gamma_{14} = 0$, $x_{14}^{min} = 0$, and $x_{14}^{max} = 100$, was integrated into the system. At t=2500 ms, POD 14 was disconnected due to adverse weather conditions. The results of Figure 11 demonstrate that the newly added POD 14 compensated for a portion of the power generation, and the outputs converged to a new optimal operating point after a period. When POD 14 was removed, the output of the original POD unit increased, and the power of all units returned to the initial optimal value.

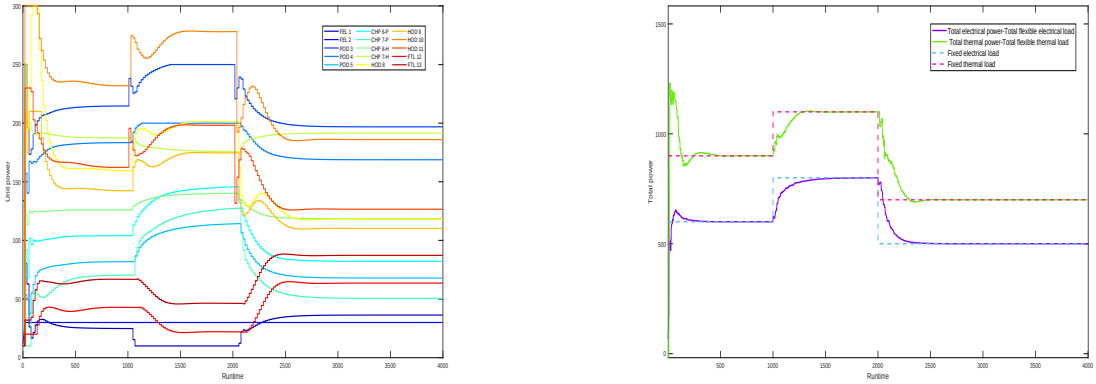


(a) Power output



(b) Total supply–demand mismatch

Figure 11. Plug-and-play properties.



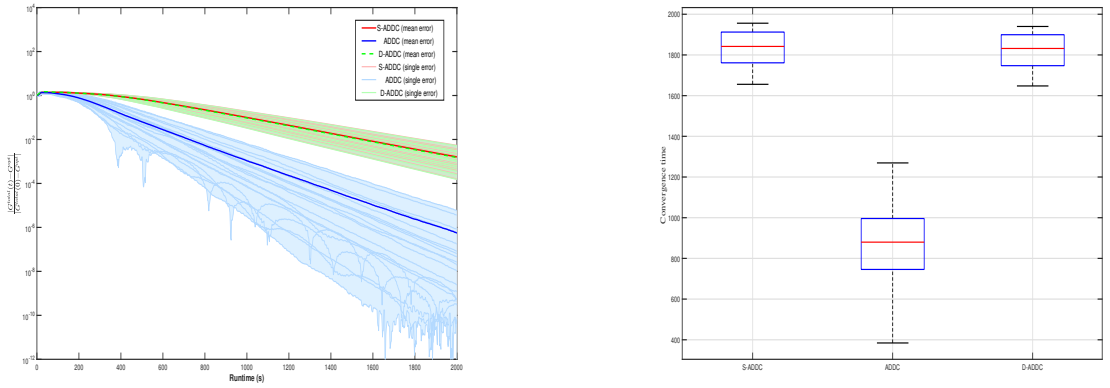
(a) Power output

(b) Total supply–demand mismatch

Figure 12. Load jump transition.

Subsequently, jump transitions in the daily load profiles were simulated. At $t = 1000$ ms, the fixed electric and thermal load demands were raised to 800 kW and 1100 kW, respectively. At $t = 2000$ ms, the fixed electric and thermal loads were reduced to 500 kW and 700 kW, respectively. As shown in Figure 12, each time the total load changed, the ADDC algorithm autonomously adjusted the electric and thermal power of the units, enabling the system to converge to new optimal operating state.

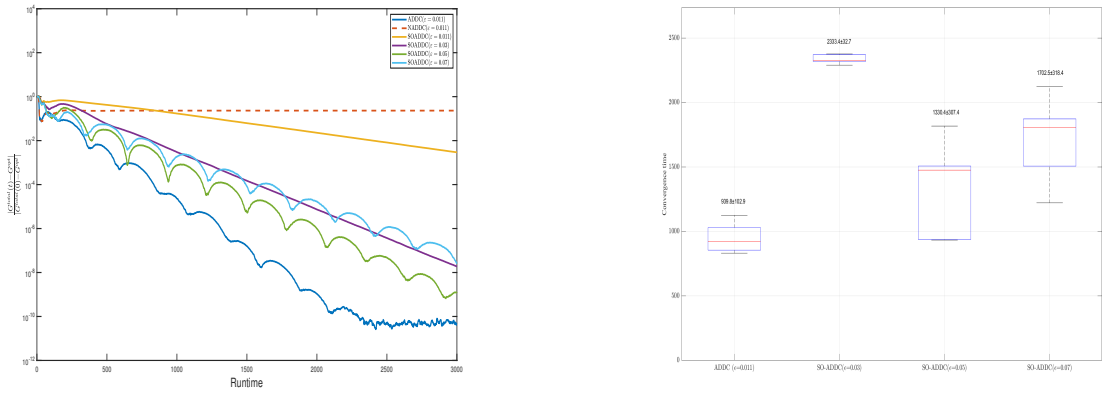
6.4. Algorithm comparison



(a) Relative error curves

(b) Convergence time to 10^{-3}

Figure 13. Synchronous algorithm comparison over 20 experiments.

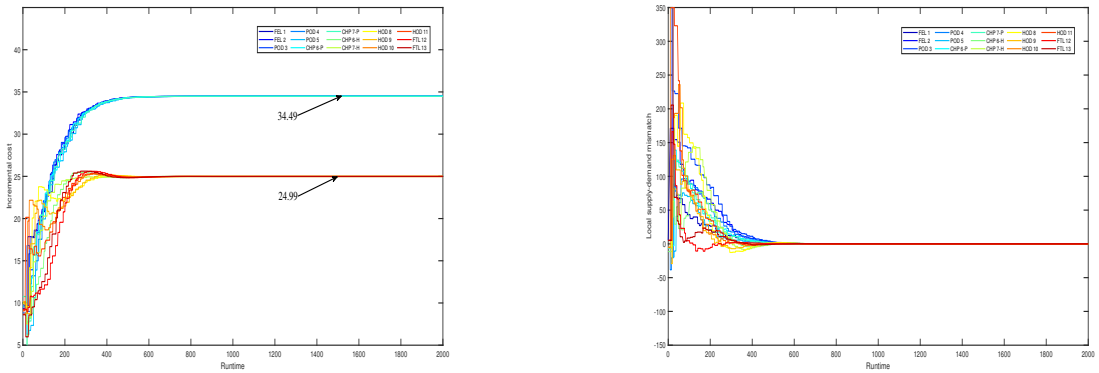


(a) Mean relative error curves

(b) Convergence time to 10^{-6}

Figure 14. Asynchronous algorithm comparison over 10 experiments.

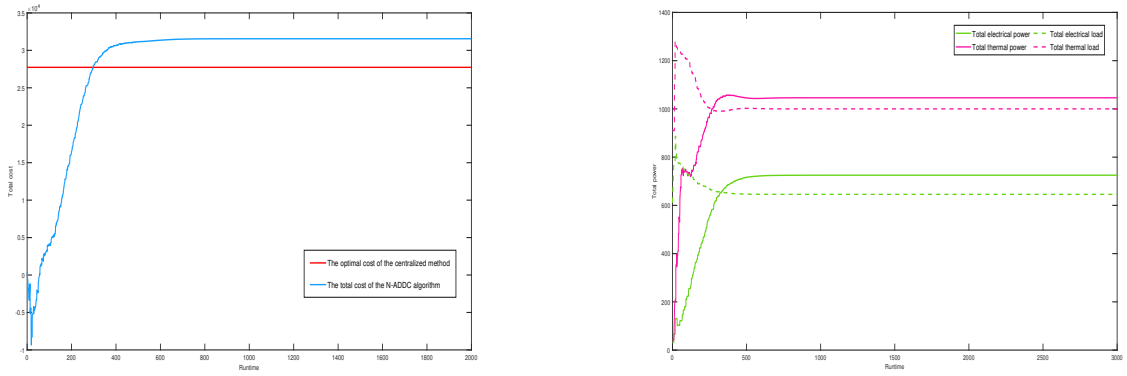
In this part, the ADDC algorithm will be compared with the D-ADDC and S-ADDC synchronization algorithms. For simplicity, all devices are assumed to have identical computation and communication delays per iteration. Figure 13a shows the relative cost error over 20 independent runs. ADDC converges significantly faster than both synchronous algorithms. Meanwhile, S-ADDC and D-ADDC exhibit nearly identical performance, which is consistent with Remark 5. For quantitative comparison, Figure 13b presents box plots of the convergence time to an accuracy of 10^{-3}. The median convergence time of ADDC is approximately 930 ms, much lower than that of S-ADDC and D-ADDC (both around 1850 ms). Note that the convergence time distribution of ADDC is more dispersed, while that of S-ADDC and D-ADDC is more concentrated. This is because ADDC is an asynchronous framework, requiring no synchronous waiting, resulting in faster convergence but sensitivity to latency and heterogeneity. S-ADDC and D-ADDC are synchronous frameworks, requiring iteration to wait for the slowest node, thus converging more slowly but with higher stability.



(a) Incremental cost

(b) Supply–demand mismatch

Figure 15. Time evolution of the N-ADDC algorithm.



(a) Total cost comparison

(b) Total supply–demand mismatch

Figure 16. Convergence results of the N-ADDC algorithm.

We next compare ADDC with the N-ADDC and SO-ADDC asynchronous algorithms. Here, N-ADDC is the ADDC algorithm without the incremental vector, and SO-ADDC is an asynchronous variant adapted from [33]. Figure 14a depicts the mean relative cost error over 10 runs. N-ADDC fails to reach the optimum. In contrast, ADDC and SO-ADDC with four dgeneration-sideerent step sizes all converge linearly, with ADDC achieving the fastest convergence. Figure 14b further provides box plots of convergence time to an accuracy of 10^{-6} for ADDC and SO-ADDC. ADDC achieves a median convergence time of about 930 ms, whereas the median convergence time of SO-ADDC—using various large step sizes—ranges from 1470 ms to 2400 ms. A plausible explanation is that ADDC assigns a small number of virtual nodes proportional to the number of physical nodes, resulting in a compact structure with short information propagation paths and thus lower effective delay. In contrast, SO-ADDC associates virtual nodes with edges; since the number of edges far exceeds the number of nodes, this leads to a larger virtual network, longer propagation paths, and consequently slower convergence.

To further explain the key role of the auxiliary transformation, the trajectory of the N-ADDC algorithm is described. As shown in Figure 15, the incremental cost achieves dual consensus convergence and the local supply–demand mismatch converges to zero. However, the total cost of N-ADDC algorithm is higher than the optimal value in Figure 16a. This behavior can be explained as follows. A small step size provides strong damping, mitigating oscillations induced by asynchronous delays and partial activations, and thus guarantees convergence. However, asynchrony breaks the invariance of the local supply–demand estimates without the auxiliary transformation. Consequently, the total energy supply cannot accurately track the aggregate demand in Figure 16b. Hence, the N-ADDC algorithm converges but sacrifices optimality.

7. Conclusion

This study proposes a fully asynchronous ADDC algorithm for an EH-EMS with coupled electricity and heat energy and asymmetric communication topologies. The first issue is handling complex local constraints, for which an optimal response function is constructed to ensure constraint satisfaction and computational efficiency. To remove the reliance on a global clock, each device is allowed to

activate autonomously according to its computational capabilities and perform asynchronous updates using the latest but potentially outdated information. A further issue is that asynchronous computation disrupts the initial conditions required for maintaining supply–demand balance. Accordingly, this paper innovatively introduces an auxiliary variable to track accumulated rather than instantaneous supply–demand mismatch, thereby eliminating sensitivity to initialization. Additionally, the uncertainty of the update time makes traditional analysis methods unsuitable. In response, two types of virtual agents are assigned to each real agent, and then the ADDC algorithm is transformed into a time-varying synchronization algorithm. Based on this transformation, a rigorous convergence proof has finally been established. Simulation results based on the EH-EMS 65 test system demonstrate that the ADDC algorithm exhibits excellent performance in both convergence speed and engineering feasibility.

Future work will proceed along two directions. First, the deterministic model will be extended to incorporate the uncertainty of renewable energy generation, thereby enhancing its adaptability to real-world energy systems. Second, communication optimization strategies under bandwidth constraints will be investigated. Specifically, event-triggered mechanisms will be combined with asynchronous updates to conserve communication resources, though ensuring network connectivity within fixed time bounds remains a challenging issue. In addition, protection mechanisms against malicious data injection attacks will be introduced to further enhance the algorithm's security.

Author contributions

Xiaolan Yuan: Conceptualization, formal analysis, funding acquisition, validation, writing–original draft, writing–review and editing; Xiang Wu: Funding acquisition, methodology, supervision, validation, writing–review and editing; Haozheng Meng: Formal analysis, validation, writing–review and editing.

Use of Generative–AI tools declaration

During the writing process, the AI tool Doubao was used to correct grammatical errors and refine the language. The authors guarantee that all generated content has been rigorously reviewed and rewritten, and they assume full responsibility for the final version.

Acknowledgments

This work was supported by the Graduate Research Fund Project of Guizhou Province (2024YJSKYJJ184) and the National Natural Science Foundation of China (62363005).

Conflict of interest

The authors declare that there are no conflicts of interest.

References

1. W. Y. Zheng, H. Lu, M. L. Zhang, Q. W. Wu, Y. H. Hou, J. Z. Zhu, Distributed energy management of multi-entity integrated electricity and heat systems: a review of architectures, optimization algorithms, and prospects, *IEEE Trans. Smart Grid,* **15** (2023), 1544–1561. https://doi.org/10.1109/TSG.2023.3310947
2. X. B. Chen, Y. Liu, B. Li, Adjustable robust optimization in enabling optimal day-ahead economic dispatch of CCHP-MG considering uncertainties of wind-solar power and electric vehicle, *J. Ind. Manag. Optim.,* **17** (2021), 1639–1661. https://doi.org/10.3934/jimo.2020038
3. Z. Zhang, M. Y. Chow, Convergence analysis of the incremental cost consensus algorithm under dgeneration-sideerent communication network topologies in a smart grid, *IEEE Trans. Power Syst.,* **27** (2012), 1761–1768. https://doi.org/10.1109/TPWRS.2012.2188912
4. Z. Q. Yang, J. Xiang, Y. J. Li, Distributed consensus based supply–demand balance algorithm for economic dispatch problem in a smart grid with switching graph, *IEEE Trans. Ind. Electron.,* **64** (2016), 1600–1610. https://doi.org/10.1109/TIE.2016.2615037
5. W. M. Zhang, X. He, Stealthy attack detection and solution strategy for consensus-based distributed economic dispatch problem, *Int. J. Electr. Power Energy Syst.,* **103** (2018), 233–246. https://doi.org/10.1016/j.ijepes.2018.05.028
6. U. E. Alvi, W. Ahmed, M. Rehan, S. Ahmed, R. Ahmad, I. Ahmed, A novel incremental cost consensus approach for distributed economic dispatch over directed communication topologies in a smart grid, *Soft Comput.,* **26** (2022), 6685–6700. https://doi.org/10.1007/s00500-022-07061-4
7. C. j. Li, X. H, Yu, T. W. Huang, X. He, Distributed optimal consensus over resource allocation network and its application to dynamical economic dispatch, *IEEE Trans. Neural Netw. Learn. Syst.,* **29** (2017), 2407–2418. https://doi.org/10.1109/TNNLS.2017.2691760
8. Q. Y. Sun, R. Y. Fan, Y. S. Li, B. N. Huang, D. Z. Ma, A distributed double-consensus algorithm for residential we-energy, *IEEE Trans. Ind. Inform.,* **15** (2019), 4830–4842. https://doi.org/10.1109/TII.2019.2921431
9. L. N. Liu, G. H. Yang, Distributed optimal energy management for integrated energy systems, *IEEE Trans. Ind. Inform.,* **18** (2022), 6569–6580. https://doi.org/10.1109/TII.2022.3146165
10. W. Chen, Z. Wang, H. Dong, J. Mao, G. P. Liu, Privacy-preserving distributed economic dispatch of microgrids over directed networks via state decomposition: A fast consensus algorithm, *IEEE Trans. Ind. Inform.,* **20** (2023), 4092–4102. https://doi.org/10.1109/TII.2023.3321027
11. G. Chen, J. Y. Li, A fully distributed ADMM-based dispatch approach for virtual power plant problems, *Appl. Math. Model.,* **58** (2018), 300–312. https://doi.org/10.1016/j.apm.2017.06.010
12. Y. J. Wu, C. W. Wang, Y. L. Wang, Cooperative game optimization scheduling of multi-region integrated energy system based on ADMM algorithm, *Energy,* **302** (2024), 131728. https://doi.org/10.1016/j.energy.2024.131728
13. S. Zhu, T. Ding, C. Chen, M.Y. Chow, X. Guan, DEED-ADMM: a scalable distributed algorithm for economic dispatch in multienergy systems with energy storage, *IEEE Trans. Autom. Sci. Eng.,* **22** (2025), 11431–11443. https://doi.org/10.1109/TASE.2025.3534171

14. C. X. Dou, D. Yue, J. M. Guerrero, X. P. Xie, S. L. Hu, Multiagent system-based distributed coordinated control for radial DC microgrid considering transmission time delays, *IEEE Trans. Smart Grid,* **8** (2016), 2370–2381. https://doi.org/10.1109/TSG.2016.2524688

15. G. C. Chen, Z. Y. Zhao, Delay effects on consensus-based distributed economic dispatch algorithm in microgrid, *IEEE Trans. Power Syst.,* **33** (2017), 602–612. https://doi.org/10.1109/TPWRS.2017.2702179

16. M. Yu, C. Song, S. X. Feng, W. Tan, A consensus approach for economic dispatch problem in a microgrid with random delay effects, *Int. J. Electr. Power Energy Syst.,* **118** (2020), 105794. https://doi.org/10.1016/j.ijepes.2019.105794

17. T. Qian, X. Y. Chen, Y. L. Xin, W. H. Tang, L. X. Wang, Resilient decentralized optimization of chance constrained electricity-gas systems over lossy communication networks, *Energy,* **239** (2022), 122158. https://doi.org/10.1016/j.energy.2021.122158

18. A. A. Eladl, M. I. El-Afifi, M. M. El-Saadawi, B. E. Sedhom, Distributed optimal dispatch of smart multi-agent energy hubs based on consensus algorithm considering lossy communication network and uncertainty, *CSEE J. Power Energy Syst.,* **11** (2023), 352–364. https://doi.org/10.17775/CSEEJPES.2023.00670

19. J. F. Wu, T. Yang, D. Wu, K. Kalsi, K. H. Johansson, Distributed optimal dispatch of distributed energy resources over lossy communication networks, *IEEE Trans. Smart Grid,* **8** (2017), 3125–3137. https://doi.org/10.1109/TSG.2017.2720761

20. G. L. Liu, H. J. Liang, R. Wang, Z. Q. Sui, Q. Y. Sun, Adaptive event-triggered output feedback control for nonlinear multiagent systems using output information only, *IEEE Trans. Syst. Man Cybern. Syst.*, **21** (2025), 7639–7650. https://doi.org/10.1109/TSMC.2025.3597016

21. Z. J. Hu, R. J. Ma, Adaptive event-triggered tracking control via switching functions, *Automatica,* **185** (2026), 112813. https://doi.org/10.1016/j.automatica.2025.112813

22. C. J. Li, X. H. Yu, W. W. Yu, T. W. Huang, Z. W. Liu, Distributed event-triggered scheme for economic dispatch in smart grids, *IEEE Trans. Ind. Informatics.,* **12** (2016), 1775–1785. https://doi.org/10.1109/TII.2015.2479558

23. J. Liu, C. Wang, J. Liu, P. Xie, Attack resilient strategy for event-triggered distributed control scheme of multienergy systems, *J. Frankl. Inst.,* **360** (2023), 5333–5353. https://doi.org/10.1016/j.jfranklin.2023.03.043

24. L. Liu, G. H. Yang, Distributed dynamic event-triggered optimal economic dispatch for multienergy systems with prescribed-time convergence, *IEEE Trans. Ind. Informatics.,* **21** (2025), 4533–4542. https://doi.org/10.1109/TII.2025.3545042

25. R. Wang, Q. Q. Li, Y. Shi, L. H. Wang, A gossip-based asynchronous distributed algorithm for economic dispatch problem with transmission losses, *Proc. 2019 IEEE Innov. Smart Grid Technol*, 2019, 770–775. https://doi.org/10.1109/isgt-asia.2019.8881641

26. Y. M. Zhang, Y. L. Zhang, Z. X. Liu, Z. Q. Chen, Gossip-based algorithm for economic dispatch of microgrids integrating isolated and grid-connected modes, *Sci. China Inf. Sci.,* **68** (2025), 132204. ttps://doi.org/10.1007/s11432-024-4140-5

27. Y. Zheng, Y. Song, D. J. Hill, Y. X. Zhang, Multiagent system based microgrid energy management via asynchronous consensus ADMM, *IEEE Trans. Energy Convers.,* **33** (2018), 886–888. https://doi.org/10.1109/TEC.2018.2799482

28. A. Mohammadi, A. Kargarian, Learning-aided asynchronous admm for optimal power flow, *IEEE Trans. Power Syst.,* **37** (2021), 1671–1681. https://doi.org/10.1109/TPWRS.2021.3120260

29. Z. Wang, G. Chen, H. Q. Li, An efficient distributed algorithm for economic dispatch considering communication asynchrony and time delays, *Energy Convers. Econ.,* **3** (2022), 214–226. https://doi.org/10.1049/enc2.12061

30. Y. F. Su, S. Y. Wu, Z. J. Wang, Z. Y. Bao, F. Liu, Hierarchically distributed energy management in distribution systems: an error-tolerant and asynchronous approach, *IEEE Trans. Smart Grid,* **15** (2023), 2909–2920. https://doi.org/10.1109/TSG.2023.3328955

31. Z. J. Wang, L. J. Chen, F. Liu, P. Yi, M. Cao, S. C. Deng, et al., Asynchronous distributed power control of multimicrogrid systems, *IEEE Trans. Control Netw. Syst.,* **7** (2020), 1960–1973. https://doi.org/10.1109/TCNS.2020.3018703

32. W. Ananduta, C. Ocampo-Martinez, A. Nedić, A distributed augmented Lagrangian method over stochastic networks for economic dispatch of large-scale energy systems, *IEEE Trans. Sustain. Energy,* **12** (2021), 1927–1934. https://doi.org/10.1109/TSTE.2021.3073510

33. Y. Tian, Y. Sun, G. Scutari, Achieving linear convergence in distributed asynchronous multiagent optimization, *IEEE Trans. Autom. Control,* **65** (2020), 5264–5279. https://doi.org/10.1109/TAC.2020.2977940

34. D. P. Bertsekas, Nonlinear programming, *J. Oper. Res. Soc.,* **48** (1997), 334. https://doi.org/10.1057/palgrave.jors.2600425

AIMS Press