Future Technology
Open Access Journal
ISSN 2832-0379

August 2026| Volume 05 | Issue 03 | Pages 45-56

Journal homepage: https://fupubco.com/futech

https://doi.org/10.55670/fpll.futech.5.3.4

Future Publishing LLC

Article

Distributed coalition-based resource orchestration for heterogeneous IoT devices in metropolitan smart cities

Si Liu, Midhun Chakkaravarthy*

School of AI Computing and Multimedia, Lincoln University College, Malaysia

ARTICLE INFO

Article history:
Received 12 December 2025
Received in revised form
14 March 2026
Accepted 17 April 2026

Keywords:
Internet of Things (IoT), Coalition formation, Distributed security (Merkle DAG), Edge computing

*Corresponding author
Email address:
midhun@lincoln.edu.my

DOI: 10.55670/fpll.futech.5.3.4

ABSTRACT

The rapid proliferation of IoT devices in metropolitan environments poses critical challenges for heterogeneous device management under minimal centralized control. This paper presents DCRO, a Distributed Coalition-based Resource Orchestration framework enabling IoT devices to self-organize into dynamic coalitions for cooperative resource management. Unlike traditional hierarchical approaches that suffer from scalability bottlenecks, DCRO integrates three core components: a Self-Organizing Device Clustering Algorithm (SODCA) that adapts to topology changes without global coordination; a Game-Theoretic Coalition Formation Mechanism (GT-CFM) that drives fair resource allocation through Shapley value-based negotiation; and a Lightweight Hierarchical Consensus Protocol (LHCP) coupled with a Merkle-DAG security architecture that ensures tamper-resistant coordination without blockchain overhead. Experiments across three metropolitan testbeds demonstrate 26.2% latency reduction and 31.4% energy savings over centralized baselines, only 11.3% throughput degradation under continuous fault injection, and stable coalition convergence at 5,000 devices within 15 iterations.

1. Introduction

The digital transformation of metropolitan cities has led to an exponential increase in metropolitan-scale IoT applications, including intelligent transportation systems, environment monitoring, public safety, and energy management, with the number of devices running into hundreds of thousands. Although the integration of edge computing and cloud computing provides new solutions for data processing and intelligent decision-making for heterogeneous devices, it also raises new challenges for heterogeneous device collaboration without strong central control [1]. The key problem is how to efficiently enable large-scale heterogeneous device collaboration without strong central control. The traditional approach of managing resources in an IoT environment is based on a centralized framework, where scheduling and resource allocation of all network devices are managed by cloud servers. However, this approach has shown severe limitations when implemented at a city-level scenario, since communication bottlenecks are severe for centralized controllers when handling massive state updates of network devices, and single-point failures are a major concern for overall system availability [2]. Although edge computing helps mitigate these problems, efficient resource allocation for edge devices without global coordination is still an unsolved problem under heterogeneous and dynamic topology conditions [3]. Three-tier end-edge-cloud frameworks are effective for providing intelligence in an IoT environment [4]; however, resource provisioning optimization is a challenge [5]. Distributed cooperative mechanisms have received significant research attention as an alternative approach to traditional centralized control. Game theory has been proven to be an effective approach for analyzing competitive and cooperative relationships among devices using its strong mathematical tools. It has been demonstrated to have strong potential for resource allocation, scheduling, and QoS guarantee [6]. Coalition game theory has been recognized as an effective approach for analyzing cooperative relationships among rational devices. Coalition game theory has been applied to WSN clustering [7] and distributed resource optimization [8]. Recently, coalition formation has been proposed to reduce communication costs for federated learning [9], and coalition games have been applied to task offloading for edge computing [10]. However, existing studies on coalition game theory mainly focus on single-objective optimization for special applications. The applicability of coalition game theory to heterogeneous large-scale urban IoT is not fully investigated. However, security and consistency also raise other issues for distributed IoT networks. The dynamic

addition/removal of devices, arbitrary partitioning of the network, and the presence of malicious devices are potential

risks for the integrity of the coordinated task. Although Byzantine fault tolerance provides a solid theoretical background for distributed consensus, the communication complexity is too high for the application of conventional consensus algorithms like PBFT [11]. A new generation of light-weight consensus algorithms using sharding and leader election is under active development [12]. The Merkle-DAG approach is a promising solution for secure coordination without blockchain overhead, with a validated data integrity auditing capability for cloud-edge networks [13]. Previous studies on the optimization of latency-energy tradeoffs have proven that single-objective models are inadequate for multidimensional metropolitan-scale networks [14]. However, the application of deep reinforcement learning is limited by the convergence problem [15]. Hybrid metaheuristics have the potential for Pareto optimality in resource allocation for edge-cloud networks [16]. However, the solution stability under device failures and network disruptions is a problem that needs improvement [17].

The above review article has identified three important shortcomings: the lack of unified framework for integrating coalition formation, self-organizing clustering, and lightweight consensus; the lack of cross-scenario validation on metropolitan scale; and the lack of fault tolerance for network partitioning and device failure. This paper attempts to bridge the above-mentioned shortcomings through the Distributed Coalition-based Resource Orchestration framework (DCRO). The contributions of the proposed framework include a globally coordination-free self-organizing clustering algorithm with topology adaptability; a hybrid coordination mechanism with the integration of game theory-based coalition formation and lightweight consensus; and a Merkle-DAG-based distributed security framework without the computational cost of the blockchain technology. The proposed framework is validated on three metropolitan-scale testbeds.

2. System model and problem formulation

For clarity, Table 1 summarizes the key mathematical notations used throughout this paper.

2.1 Heterogeneous IoT network topology modeling

The IoT network in a metropolitan smart city is composed of a large number of heterogeneous devices with diverse functionalities and significantly varying capabilities [18], whose topology evolves dynamically over time and cannot be accurately characterized by conventional static graph models. To this end, this paper abstracts the metropolitan heterogeneous IoT network as a weighted graph $G=(V,E,W)$, where directed edges capture asymmetric resource flows and communication priorities while an undirected summarization of G is used for topological analysis via the graph Laplacian, where the node set $\mathcal{V} = \{v_1, v_2, \dots, v_N\}$ represents all IoT devices in the network, the directed edge set $E \subseteq V \times V$ characterizes the communication link relationships among devices, and the weight function $W : E \to \mathbb{R}^+$ quantifies the comprehensive attributes of each link, including bandwidth capacity, transmission latency, and link quality.

Table 1. Key mathematical notations

Notation	Description
$G^{(t)} = (V, E, W)$	Time-varying weighted directed graph representing the metropolitan IoT network
V, E	Set of IoT device nodes and directed communication edges, respectively
$W : E \to \mathbb{R}^+$	Weight function quantifying bandwidth, latency, and link quality
$A^{(t)} \in 0, 1^{N \times N}$	Time-varying adjacency matrix
$L^{(t)} = D^{(t)} - A^{(t)}$	Graph Laplacian matrix
$\lambda_2(L^{(t)})$	Algebraic connectivity (Fiedler value); measures topological robustness
$C^{(t)}$	Inter-device communication cost matrix accounting for propagation delay, interference, and queuing latency
$\phi_i = (c_i^{\text{cpu}}, m_i^{\text{mem}}, b_i^{\text{bw}}, e_i^{\text{bat}}, \tau_i^{\text{type}})$	Capability vector of device v_i
π_m, π^*	A coalition and the Nash-stable coalition partition, respectively
$V(\pi_m)$	Characteristic function (maximum transferable utility) of coalition π_m
$\Phi(\pi_m)$	Intra-coalition coordination cost of coalition π_m
$U_i(\pi_m)$	Utility of device v_i upon joining coalition π_m (SODCA)
$\phi_i(\pi_m)$	Shapley value (allocated payoff) of device v_i within coalition π_m (GT-CFM)
CR_i	Comprehensive reputation score of device v_i for coalition representative election
H_t	Hash digest of DAG node n_t in the Merkle-DAG structure
$H_{\text{global}}^{(r)}$	Global anchor hash aggregated from all coalition root hashes

Accounting for device heterogeneity, each node $v_i \in \mathcal{V}$ is characterized by a capability vector:

$$\phi_i = \left(c_i^{\text{cpu}},\ m_i^{\text{mem}},\ b_i^{\text{bw}},\ e_i^{\text{bat}},\ \tau_i^{\text{type}}\right) \quad (1)$$

where c_i^{cpu} denotes computational capacity (MIPS), m_i^{mem} denotes available memory (MB), b_i^{bw} denotes wireless channel bandwidth (Mbps), e_i^{bat} denotes residual battery energy (J), and $\tau_i^{\text{type}} \in$ {sensor,gateway,edge} denotes the device type identifier. This vector characterizes resource heterogeneity across four dimensions—computation, storage, communication, and energy—with τ_i^{type} serving as a categorical device identifier that is used to apply type-specific constraints during resource scheduling rather than as a numerical optimization variable.

Due to device mobility and network dynamics inherent to metropolitan scenarios, the topology graph G is not static but evolves over time as a time-varying graph sequence $\mathcal{G}^{(t)}{}_{t\geq 0}$ [19]. Define the adjacency matrix $A^{(t)} \in [0,1]^{N\times N}$, whose elements are given by:

$$A_{ij}^{(t)} = \begin{cases} 1, \text{ if } d(v_i, v_j) \leq R_i^{\text{comm}} \text{ and link quality satisfies threshold} \\ 0, \text{ otherwise} \end{cases} \quad (2)$$

where $d(v_i, v_j)$ denotes the Euclidean distance between devices v_i and v_j, and R_i^{comm} denotes the effective communication radius of device v_i. Although the overall network is modeled as a directed graph to capture asymmetric link properties, the adjacency matrix $A^{(t)}$ is treated as symmetric for the purpose of Laplacian computation, i.e., an edge (v_i, v_j) exists if either device is within the other's communication radius. This symmetrization is consistent with the undirected nature of physical wireless channels and enables the use of the standard graph Laplacian $L^{(t)} = D^{(t)} - A^{(t)}$ and its associated Fiedler value $\lambda_2(L^{(t)})$ as a topological robustness indicator. Based on this time-varying topology, this paper further defines the inter-device communication cost matrix $C^{(t)}$, whose element $C_{ij}^{(t)}$ jointly accounts for propagation delay, channel interference, and queuing latency: $C_{ij}^{(t)} = w_1 \cdot \frac{d(v_i,v_j)}{v_{\text{signal}}} + w_2 \cdot I_{ij}^{(t)} + w_3 \cdot \omega_{ij}^{(t)}$, where $d(v_i, v_j)/v_{\text{signal}}$ denotes the propagation delay, $I_{ij}^{(t)} \in [0,1]$ denotes the normalized channel interference level, $\omega_{ij}^{(t)}$ denotes the queuing latency at the receiver, and $w_1, w_2, w_3 > 0$ are weighting coefficients satisfying $w_1 + w_2 + w_3 = 1$. This formulation provides a topological basis for cost evaluation in subsequent coalition formation and resource allocation.

To characterize network connectivity and partition resilience, the algebraic connectivity (Fiedler value) $\lambda_2(L^{(t)})$ of the graph is introduced as a quantitative indicator of topological robustness, where $L^{(t)} = D^{(t)} - A^{(t)}$ denotes the graph Laplacian matrix and $D^{(t)}$ denotes the degree matrix. As $\lambda_2(L^{(t)}) \to 0$, the network approaches a disconnected state, at which point the cross-partition coordination capability of coalitions becomes a critical indicator for evaluating the robustness of the proposed framework.

2.2 Resource constraints and quality of service definition

In the resource management problem of heterogeneous IoT networks, resource constraints and quality-of-service requirements jointly constitute the dual boundaries of the feasible domain, and their precise mathematical definition is a prerequisite for constructing the optimization model [20]. This paper categorizes schedulable resources in the system into three classes—computational resources, storage resources, and communication resources—and establishes constraint models for each category. For any device v_i, the set of tasks it hosts at time t is denoted $\mathcal{T}_i^{(t)}$, and the resource demand of each task $k \in \mathcal{T}_i^{(t)}$ is represented as the tuple $r_k = (r_k^{\text{cpu}}, r_k^{\text{mem}}, r_k^{\text{bw}})$. Resource feasibility constraints require that the aggregate resource demands of all tasks on a device do not exceed its actual available capacity:

$$\sum_{k\in \mathrm{T}_i^{(t)}} r_k^{\text{cpu}} \leq c_i^{\text{cpu}},\quad \sum_{k\in \mathrm{T}_i^{(t)}} r_k^{\text{mem}} \leq m_i^{\text{mem}},\quad \sum_{j:(i,j)\in \mathrm{E}} b_{ij}^{(t)} \leq b_i^{\text{bw}} \quad (3)$$

Energy constraints represent a dimension that cannot be neglected in resource-constrained IoT devices [21]. The total energy consumption model for device v_i during computational task execution and wireless communication is:

$$E_i^{\text{total}} = \underbrace{\kappa_i \cdot c_i^{\text{cpu}} \cdot f_i^2 \cdot T_i^{\text{exec}}}_{\text{Computational energy}} + \underbrace{P_i^{\text{tx}} \cdot T_i^{\text{comm}}}_{\text{Communication energy}} \quad (4)$$

where κ_i is the effective switched capacitance coefficient of the device, f_i is the CPU operating frequency, T_i^{exec} and T_i^{comm} are the task execution time and communication occupancy time respectively, and P_i^{tx} is the transmission power. The energy constraint requires $E_i^{\text{total}} \leq e_i^{\text{bat}}$ to ensure the sustained operational capability of the device.

Quality of Service (QoS) is defined across three dimensions: end-to-end latency, task completion rate, and system reliability. For task k, its end-to-end latency comprises three components: local computation delay, task transmission delay, and queuing waiting delay:

$$D_k = T_k^{\text{queue}} + T_k^{\text{exec}} + T_k^{\text{trans}} = \frac{r_k^{\text{cpu}}}{f_i} + \frac{s_k}{b_{ij}^{(t)}} + \omega_k \quad (5)$$

where s_k denotes the task data volume (bits) and ω_k denotes the queuing waiting delay. The QoS latency constraint requires $D_k \leq D_k^{max}$, where D_k^{max} is the maximum tolerable delay specified at the application layer. In the reliability dimension, the instantaneous reliability of link (i, j) is defined as $\rho_{ij}^{(t)} \in [0,1]$, path reliability is the product of the reliability values of all links along the path, and the system-wide service reliability constraint is expressed as:

$$\mathcal{R}_k = \prod_{(i,j)\in \text{path}(k)}^{\Pi} \rho_{ij}^{(t)} \geq \mathcal{R}_k^{min} \quad (1)$$

The above resource constraints and QoS definitions together constitute the feasible domain of the multi-objective optimization problem, providing a rigorous mathematical foundation for the design of the subsequent coalition-based resource orchestration framework [22].

3. Distributed coalition orchestration framework

3.1 Self-organizing device clustering Algorithm

In response to the characteristics of dynamically changing network topologies and significant device heterogeneity in metropolitan IoT environments, this paper designs a Self-Organizing Device Clustering Algorithm (SODCA) that requires no global coordination (Algorithm 1). The basic concept behind the algorithm is that each device node makes local decisions in an autonomous fashion using local neighborhood information, causing the clustering structure to evolve towards a stable equilibrium state

through iterative join/leave operations, thus enabling adaptive organization of the global topology in a decentralized fashion without the need for a controller. The key innovation of the algorithm lies in the introduction of a comprehensive utility function $U_i(\pi_m)$, which quantifies the net benefit obtained by device v_i upon joining coalition π_m:

$$U_i(\pi_m) = \alpha \cdot \underbrace{\frac{c_i^{\text{cpu}}}{\sum_{j \in \pi_m} c_j^{\text{cpu}}}}_{\in[0,1]} - \beta \cdot \underbrace{\frac{\bar{C}_{i,\pi_m}}{\bar{C}\max}}_{\in[0,1]} - \gamma \cdot \underbrace{\frac{1/\lambda_2(\mathbf{L}_{\pi_m})}{1/\lambda_{\min}}}_{\in[0,1]} \quad (7)$$

where the first term measures the proportion of computational resource contribution by the device within the coalition, the second term $\bar{C}_{i,\pi_m}$ denotes the average communication cost between device v_i and coalition members, and the third term penalizes clustering structures with weak connectivity through the Fiedler value of the intra-coalition subgraph, with α, β, and γ being tunable weighting coefficients. This utility function unifies resource contribution, communication overhead, and topological connectivity into a single evaluation framework, driving devices to migrate toward the coalition of maximum utility [23]. Each term in Eq. (7) is normalized to the range [0,1]: the first term is inherently normalized as a proportion of computational contribution; the second term is divided by the maximum observed communication cost $\bar{C}_{max}$ across all device pairs; and the third term is divided by $1/\lambda_{min}$, where λ_{min} is the minimum Fiedler value observed during network initialization. This normalization ensures that the weighting coefficients α, β, and γ are dimensionally consistent and directly comparable, with $\alpha + \beta + \gamma = 1$. In our experiments, we set $\alpha = 0.5$, $\beta = 0.3$, $\gamma = 0.2$ based on the relative importance of resource contribution over communication and connectivity costs in metropolitan IoT scenarios. A sensitivity analysis varying each coefficient by ± 0.1 confirmed that system latency varies by less than 3.2% and energy consumption by less than 2.8%, indicating that the framework is robust to moderate perturbations in the weighting parameters.

Device clustering decisions follow a preference ordering rule: a migration operation is executed if and only if the utility of device v_i strictly increases upon joining a new coalition $\pi_{m'}$ compared to its current coalition π_m, i.e., $U_i(\pi_{m'}) > U_i(\pi_m)$. Since each migration operation strictly improves device utility and the utility function is bounded above, the algorithm is guaranteed to converge to a Nash-stable clustering structure within a finite number of steps, as formally established below.

Theorem 1 (Convergence of SODCA). Starting from any initial coalition partition $\pi^{(0)}$, the SODCA algorithm converges to a Nash-stable partition π^* within a finite number of migration steps.

Proof. Define a global potential function: $\Psi(\pi) = \sum_m \sum_{i \in \pi_m} U_i(\pi_m)$. We show that $\Psi(\pi)$ strictly increases with every migration operation. Consider a migration of device v_i from coalition π_m to $\pi_{m'}$, which is executed if and only if $U_i(\pi_{m'} \cup \{v_i\}) > U_i(\pi_m)$. Since the utilities of all other devices $v_j \neq v_i$ within π_m and $\pi_{m'}$ are non-decreasing after the migration—as the departure of v_i from π_m does not reduce remaining members' resource proportions and the arrival of v_i in $\pi_{m'}$ increases the coalition's computational pool—the change in Ψ satisfies: $\Delta\Psi = U_i(\pi_{m'} \cup v_i) - U_i(\pi_m) > 0$. Thus, every migration strictly increases $\Psi(\pi)$. Since each term in $U_i(\pi_m)$ is bounded—the first term lies in [0,1] by normalization, the second and third terms are bounded by $\bar{C}_{max}$ and $1/\lambda_{min}$ respectively—the potential function $\Psi(\pi)$ is bounded above. Furthermore, the number of distinct coalition partitions over N devices is finite (bounded by the Bell number B_N). Therefore, the sequence of strictly increasing potential values must terminate in a finite number of steps at a partition π^* from which no device has an incentive to deviate, which by definition is a Nash-stable equilibrium.

Algorithm 1: Self-Organizing Device Clustering Algorithm (SODCA)
Input: Network graph $G^{(t)} = (V, E, W)$, device capability vectors ϕ_i, weighting coefficients α, β, γ, maximum iteration count $T_{\max}$
Output: Stable coalition partition $\pi^* = \pi_1, \pi_2, \ldots, \pi_M$
1: Initialize: $\pi_i \leftarrow v_i$ for all $v_i \in V$; $t \leftarrow 0$
2: while $t < T_{\max}$ **and** clustering structure has not converged **do**
3: **for each** $v_i \in V$ **do**
4: Obtain neighborhood set $N_i \leftarrow v_j : A_{ij}^{(t)} = 1$
5: Compute current utility $U_i^{\text{cur}} \leftarrow U_i(\pi_{m(i)})$
6: **for each** candidate coalition $\pi_{m'} : \pi_m \cap N_i \neq \varnothing$ **do**
7: Compute post-migration utility $U_i^{\text{new}} \leftarrow U_i(\pi_{m'} \cup v_i)$
8: **if** $U_i^{\text{new}} > U_i^{\text{cur}}$ **then**
9: Execute migration: $\pi_{m(i)} \leftarrow \pi_{m(i)}$, v_i; $\pi_{m'} \leftarrow \pi_{m'} \cup v_i$
10: Update $U_i^{\text{cur}} \leftarrow U_i^{\text{new}}$
11: **end if**
12: **end for**
13: **end for**
14: Remove empty coalitions; **if** $\Delta\lambda_2^{(t)} > \dot{o}_\lambda$ **then** trigger local re-clustering
15: $t \leftarrow t + 1$
16: end while
17: return $\pi^* \leftarrow \pi_m : \pi_m \neq \varnothing$

To address topology mutations induced by the frequent joining and departure of devices in metropolitan scenarios, the algorithm monitors the rate of change of the Fiedler value $\Delta\lambda_2^{(t)} = |\lambda_2(L^{(t)}) - \lambda_2(L^{(t-1)})|$ at the end of each iteration round to detect topological perturbations. When $\Delta\lambda_2^{(t)}$ exceeds a preset threshold $\dot{o}_\lambda$, re-clustering is triggered only for the affected local coalitions rather than executing global recomputation across the entire network. This local response mechanism reduces the processing complexity of topology changes from $\mathcal{O}(N^2)$ to $\mathcal{O}(|\mathcal{N}_{\text{affected}}|^2)$, as established by the following complexity analysis. Complexity Analysis. In the global re-clustering case, SODCA must recompute the utility $U_i(\pi_m)$ for every device $v_i \in \mathcal{V}$ against every candidate coalition, yielding $\mathcal{O}(N^2)$ utility evaluations per iteration. In contrast, when a topological perturbation is detected (i.e., $\Delta\lambda_2^{(t)} > \epsilon_\lambda$), the local re-clustering restricts recomputation to the affected neighborhood $\mathcal{N}_{\text{affected}} \subseteq \mathcal{V}$, defined as the set of

devices whose coalition utility changes as a direct result of the topology mutation: $\mathcal{N}_{\text{affected}} = \{v_i \in \mathcal{V} \mid \exists (v_i, v_j) \in \Delta\mathcal{E}^{(t)}\}$where $\Delta\mathcal{E}^{(t)}$ denotes the set of edges added or removed at time t. Since only devices within $\mathrm{N}_{\text{affected}}$ need to recompute their utilities and candidate coalition memberships, the number of utility evaluations is bounded by $|\mathcal{N}_{\text{affected}}|^2$, giving a per-event complexity of $\mathcal{O}(|\mathcal{N}_{\text{affected}}|^2)$. In metropolitan IoT deployments, topology changes are typically caused by the joining or departure of a small number of devices, so $|\mathcal{N}_{\text{affected}}| \ll N$, and the local mechanism provides substantial computational savings over global recomputation, effectively ensuring real-time performance and scalability in large-scale dynamic networks.

3.2 Game-theoretic coalition formation mechanism

Building upon the initial clustering partition completed by SODCA, this paper further constructs a Game-Theoretic Coalition Formation Mechanism (GT-CFM) to drive edge nodes to achieve dynamic resource negotiation and optimal allocation through rational game-playing. Unlike conventional cooperative game approaches, GT-CFM models the coalition formation process as a partition game with transferable utility and drives the system to converge to a Nash-stable equilibrium through distributed iterative solving, thereby maximizing the overall coalition utility while guaranteeing individual rationality. The coalition formation process is formally defined as a triple $\Gamma = (\mathcal{V}, v_i(\cdot), \mathcal{S})$, where $\mathcal{V}$ is the set of participants, $v_i(\cdot)$ is the utility function of device v_i, and S is the feasible coalition structure space. For coalition π_m, its characteristic function $V(\pi_m)$ is defined as the maximum transferable utility that can be generated through cooperation among coalition members:

$$V(\pi_m) = \sum_{i \in \pi_m} U_i(\pi_m) - \Phi(\pi_m) \tag{8}$$

where $\Phi(\pi_m)$ denotes the intra-coalition coordination cost, comprising two components: intra-coalition communication overhead and consensus achievement cost:

$$\Phi(\pi_m) = \delta \cdot \sum_{i,j \in \pi_m} C_{ij}^{(t)} + \mu \cdot |\pi_m| \cdot log\,|\pi_m| \tag{9}$$

The first term in the above expression is the weighted sum of all-pairs communication costs within the coalition, and the second term characterizes the scale-dependent overhead of the consensus protocol, with δ and μ being weighting coefficients. When the coalition size becomes excessively large, the logarithmic growth term of the coordination cost suppresses unlimited coalition expansion, thereby naturally forming a scale equilibrium between utility gains and coordination costs [24]. To ensure fair distribution of resource benefits within the coalition, the Shapley value is adopted as the entitled payoff for device v_i within coalition π_m:

$$\phi_i(\pi_m) = \sum_{S \subseteq \pi_m \setminus \{v_i\}} \frac{|S|!,(|\pi_m|-|S|-1)!}{|\pi_m|!} [V(S \cup \{v_i\}) - V(S)] \tag{10}$$

The Shapley value satisfies the axioms of efficiency, symmetry, and dummy player property, ensuring that the allocated payoff of each device strictly corresponds to its marginal contribution [25], thereby incentivizing devices to truthfully report their resource states and actively participate in coalition cooperation. However, the computational complexity of directly calculating the Shapley value is $\mathcal{O}(2^{|\pi_m|})$, which is infeasible for real-time computation in large-scale coalitions. To address this, this paper employs a Monte Carlo approximation method based on random sampling, estimating the Shapley value by averaging the marginal contributions over K random permutations, reducing the computational complexity to $\mathcal{O}(K \cdot |\pi_m|)$. In our experiments, we set $K = 500$, which provides a favorable trade-off between approximation accuracy and computational overhead. The approximation error of the Monte Carlo Shapley estimator is bounded by: $\left|\hat{\phi}_i(\pi_m) - \phi_i(\pi_m)\right| \leq \frac{V_{max}}{\sqrt{K}}$. where $V_{\max} = \max_{\pi_m} V(\pi_m)$ is the maximum coalition value. This bound follows from the fact that each sampled marginal contribution is an independent random variable with range bounded by $V_{\max}$, and the estimator is an average of K such samples; by Hoeffding's inequality, the deviation from the true Shapley value decreases at rate $\mathcal{O}(1/\sqrt{K})$. With $K = 500$ and V_{max} estimated from pilot experiments, the approximation error remains below 2.1% of the true Shapley value across all tested coalition sizes, confirming the practical accuracy of the estimator.

The necessary and sufficient condition for coalition structure π^* to reach Nash-stable equilibrium is that for any device v_i and any candidate coalition $\pi_{m'} \neq \pi_{m(i)}$, the following holds:

$$\phi_i(\pi_{m(i)}) \geq \phi_i!\,(\pi_{m'} \cup \{v_i\}) - \Delta C_i^{\text{switch}} \tag{11}$$

where $\Delta C_i^{\text{switch}}$ denotes the migration cost incurred by device v_i upon executing a coalition switch, formally defined as: $\Delta C_i^{\text{switch}} = \eta_1 \cdot \sum_{j \in \pi_{m'}} C_{ij}^{(t)} + \eta_2 \cdot \frac{s_i^{\text{state}}}{b_i^{\text{bw}}}$ where the first term represents the link re-establishment overhead, computed as the sum of communication costs between v_i and all members of the target coalition $\pi_{m'}$; the second term represents the state synchronization overhead, where s_i^{state} denotes the size of the state information to be transferred (bits) and b_i^{bw} denotes the available bandwidth of device v_i; and $\eta_1, \eta_2 > 0$ are weighting coefficients satisfying $\eta_1 + \eta_2 = 1$. In our experiments, we set $\eta_1 = 0.6$ and $\eta_2 = 0.4$, reflecting the relatively higher overhead of link re-establishment in metropolitan IoT scenarios. The above stability indicates that a device has an incentive to deviate from its current coalition only when the net benefit after migration strictly exceeds the migration cost, thereby ensuring the self-enforcing nature of the equilibrium structure.

To address the impact of device failures and topology mutations on coalition stability in metropolitan environments, GT-CFM introduces a trigger-based dynamic re-formation mechanism. When device failure is detected within coalition π_m or the Fiedler value drops below a threshold, the affected coalition enters a local re-formation process. First, the characteristic function is used to evaluate the splitting benefit of the current coalition. To avoid trivial splits caused by minor utility fluctuations, a split operation is executed only if the gain exceeds a threshold $\grave{\alpha}_{\text{split}}$: $V(\pi_m^{(1)}) + V(\pi_m^{(2)}) > V(\pi_m) + \grave{\alpha}_{\text{split}}$. where the optimal split point is found by partitioning π_m into two subsets using a greedy bisection based on the Fiedler vector of the intra-coalition subgraph. Similarly, a merge operation between π_m and a neighboring coalition $\pi_{m'}$ is executed only if the net gain exceeds a threshold $\grave{\alpha}_{\text{merge}}$:

$V(\pi_m \cup \pi_{m'}) > V(\pi_m) + V(\pi_{m'}) + \grave{\alpha}_{\text{merge}}$. Both thresholds are set as $\grave{\alpha}_{\text{split}} = \grave{\alpha}_{\text{merge}} = 0.05 \cdot \bar{V}$, where $\bar{V}$ is the average coalition value computed at the end of the most recent SODCA iteration,

making the thresholds adaptive to the current network state. The complete split-merge procedure is summarized in Algorithm 2. Splitting and merging operations are performed exclusively within a local scope, avoiding the computational explosion brought about by global re-gaming, and enabling GT-CFM to maintain real-time responsiveness in metropolitan IoT scenarios where device scales reach thousands.

Algorithm 2: GT-CFM Local Re-formation Procedure
Input: Affected coalition π_m, neighboring coalitions $\{\pi_{m'}\}$, threshold ratio $\rho = 0.05$
Output: Updated coalition partition
1: Compute $\bar{V} \leftarrow$ average $V(\pi_k)$ over all current coalitions
2: Set $\dot{\alpha}_{\text{split}} \leftarrow \rho \cdot \bar{V}$, $\dot{\alpha}_{\text{merge}} \leftarrow \rho \cdot \bar{V}$
3: Compute Fiedler vector of intra-coalition subgraph $\mathbf{L}_{\pi_m}$
4: Partition π_m into $\pi_m^{(1)}, \pi_m^{(2)}$ by sign of Fiedler vector entries
5: if $V(\pi_m^{(1)}) + V(\pi_m^{(2)}) > V(\pi_m) + \dot{\alpha}_{\text{split}}$ **then**
6: Execute split: replace π_m with $\pi_m^{(1)}$ and $\pi_m^{(2)}$
7: else
8: **for each** neighboring coalition $\pi_{m'}$ **do**
9: if $V(\pi_m \cup \pi_{m'}) > V(\pi_m) + V(\pi_{m'}) + \dot{\alpha}_{\text{merge}}$ **then**
10: Execute merge: replace $\pi_m, \pi_{m'}$ with $\pi_m \cup \pi_{m'}$; **break**
11: **end if**
12: **end for**
13: end if
14: return updated partition

3.3 Lightweight consensus protocol design

In the distributed coalition-based resource orchestration framework, coordination decisions among edge nodes within coalitions must be guaranteed through consensus protocols. However, in metropolitan IoT scenarios, devices have limited computational capacity, constrained network bandwidth, and frequently changing topologies. The message complexity required by traditional Byzantine fault-tolerant protocols such as PBFT will trigger severe communication storms as coalition size grows, making it difficult to satisfy real-time requirements. To this end, this paper designs a Lightweight Hierarchical Consensus Protocol (LHCP) tailored for intra-coalition coordination scenarios, reducing communication complexity to while preserving security and consistency guarantees. Table 2 provides a theoretical comparison between LHCP and PBFT across key protocol dimensions, demonstrating the advantages of LHCP for resource-constrained metropolitan IoT scenarios.

LHCP adopts a two-layer consensus structure to match the hierarchical topology of coalitions. At the Intra-Coalition Layer, each coalition π_m elects a Coalition Representative (CR) node based on the comprehensive reputation score CR_i of devices:

$$\mathrm{CR}_i = \omega_1 \cdot \frac{c_i^{\text{cpu}}}{c_{\max}} + \omega_2 \cdot \frac{e_i^{\text{bat}}}{e_{\max}} + \omega_3 \cdot \frac{\lambda_2(\mathbf{L}\pi_m(i))}{\lambda_{\max}} - \omega_4 \cdot h_i^{\text{fail}} \quad (12)$$

where h_i^{fail} denotes the historical failure count of the device within a sliding window of W recent time slots, updated as: $h_i^{\text{fail}}(t) = \sum_{\tau=t-W+1}^{t} 1$ [device v_i failed at slot τ] where $\mathbf{1}[\cdot]$ is the indicator function and $W = 20$ time slots in our experiments. To ensure dimensional consistency, each term in Eq. (12) is normalized via min-max normalization across all devices in the network:

$$\mathrm{CR}_i = \omega_1 \cdot \frac{c_i^{\text{cpu}} - c_{\min}^{\text{cpu}}}{c_{\max}^{\text{cpu}} - c_{\min}^{\text{cpu}}} + \omega_2 \cdot \frac{e_i^{\text{bat}} - e_{\min}^{\text{bat}}}{e_{\max}^{\text{bat}} - e_{\min}^{\text{bat}}} + \omega_3 \cdot \frac{\lambda_2(\mathbf{L}_{\pi_m(i)}) - \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}} - \omega_4 \cdot \frac{h_i^{\text{fail}} - h_{\min}^{\text{fail}}}{h_{\max}^{\text{fail}} - h_{\min}^{\text{fail}}}$$

where subscripts $\max$ and $\min$ denote the maximum and minimum values observed across all devices at the current time step. The weighting coefficients satisfy $\omega_1 + \omega_2 + \omega_3 + \omega_4 = 1$, and are set to $\omega_1 = 0.35$, $\omega_2 = 0.25$, $\omega_3 = 0.25$, $\omega_4 = 0.15$ in our experiments, reflecting the primary importance of computational capability in representative node selection. The reputation score comprehensively considers four dimensions—computational capability, residual energy, topological centrality, and historical reliability—ensuring that the elected representative node possesses sufficient capacity to support the intra-coalition consensus process. Ordinary member nodes within the coalition need only perform a single round of communication with the representative node to complete local state aggregation and verification [26], reducing the number of intra-coalition communication rounds from the three phases of PBFT to two phases. At the Inter-Coalition Layer, the representative nodes of all coalitions form a global consensus committee and complete cross-coalition resource allocation decision synchronization through an improved Raft protocol. The committee scale $M \ll N$ renders the communication overhead of global consensus negligible relative to the total number of devices.

Table 2. Theoretical comparison between LHCP and PBFT across key protocol dimensions

Dimension	**PBFT**	**LHCP (Proposed)**
Message complexity	$O(n^2)$	$O(n \log n)$
Communication rounds	3	2
Fault tolerance	$f < n/3$ Byzantine faults	$f < n/3$ Byzantine faults
Leader election	View-change protocol	Reputation-based (CR_i)
Tamper verification	Full ledger replay: $O(n)$	DAG path: $O(\log n)$
Scalability	Poor ($n > 100$ impractical)	Good (validated at $n = 5000$)
Computational overhead	High (PoW/signature-heavy)	Low (single ECC verification)
IoT suitability	Limited by $O(n^2)$ messages	Designed for resource-constrained devices

To achieve tamper-resistant verification of coordination messages without requiring a complete blockchain ledger, LHCP embeds the Merkle-DAG structure into the consensus message chain. Each consensus message $\mathcal{M}_t$ is hashed and appended to the leaf node of the current DAG prior to broadcasting, and its validity verification requires only $\mathcal{O}(\log n)$ hash comparisons along the DAG path, without

replaying the historical message chain. The digest structure of a DAG node is defined as:

$$\mathcal{H}_t = \text{Hash}\left(\mathcal{M}_t, ||\mathcal{H}_{t-1}^{\text{parent}}||, \text{sig}_i(t)\right) \quad (13)$$

where $\text{sig}_i(t)$ denotes the digital signature of representative node v_i over the message, and $||\mathcal{H}_{t-1}^{\text{parent}}||$ denotes the string concatenation operation. Any tampering with historical coordination records will cause a cascading invalidation of hash values along the DAG path, thereby achieving lightweight message integrity assurance without requiring additional trusted authorities.

When network partitioning causes the coalition representative node to become unreachable, LHCP triggers a rapid leader re-election process. Remaining members complete the election of a new representative node within a time window Δt_{elect} based on locally stored reputation value rankings, where Δt_{elect} satisfies:

$$\Delta t_{\text{elect}} \leq \frac{2 \cdot R_{\max^{comm}}}{v_{\text{signal}}} + \varepsilon_{\text{proc}} \quad (14)$$

where $R_{\max^{comm}}$ is the maximum communication radius within the coalition, v_{signal} is the signal propagation speed, and $\varepsilon_{\text{proc}}$ is the upper bound on local processing delay of nodes. The aforementioned time constraint ensures that leader re-election takes place in a single round-trip delay after the occurrence of network partitioning, minimizing the interruption of services for coalitions in network partition and device failure scenarios, thereby facilitating the continued and stable operation of the framework in metropolitan network environments.

3.4 Merkle-DAG distributed security architecture

The security of the distributed coalition-based resource orchestration framework is ensured by the presence of a security mechanism that is capable of providing the guarantee for the tamper-resistance of the coordination records and verifiability of the identities of nodes in the absence of centralized trust authority. Although traditional blockchain technologies are capable of providing consistency guarantees, the overhead that is introduced by the replication and proof-of-work is prohibitive for IoT devices. To this end, this paper proposes a distributed security architecture based on the Merkle Directed Acyclic Graph (Merkle-DAG), which combines the concurrent write capability of DAGs with the cryptographic integrity verification of Merkle trees, achieving quantifiable tamper-resistance under standard computational security assumptions, with substantially reduced computational overhead compared to blockchain-based solutions.

The Merkle-DAG constructed in this paper organizes coalition coordination records as a directed acyclic graph $\mathcal{D} = (\mathcal{N}_D, \mathcal{E}_D)$, where each graph node $n_t \in \mathcal{N}_D$ corresponds to a coordination transaction record, and the directed edges E_D characterize the causal dependency relationships among transactions. Each DAG node encapsulates the following four-tuple:

$$n_t = \langle \mathcal{H}_t,\ \mathcal{M}_t,\ P_t,\ \text{sig}_{CR}(t) \rangle \quad (15)$$

Where H_t is the hash digest of the current node, $\mathcal{M}_t$ is the coordination message payload, $P_t = \{\mathcal{H}_{t_1}^{\text{parent}}, \mathcal{H}_{t_2}^{\text{parent}}, \dots\}$ is the set of parent node hashes (supporting multiple parent nodes to enable concurrent writes), and $\text{sig}CR(t)$ is the digital signature of the coalition representative node. The recursive definition of the node hash is:

$$\mathcal{H}_t = \text{Hash}\left(\mathcal{M}_t || \mathcal{H}_{p_1} || \mathcal{H}_{p_2} || \cdots || \text{sig}_{CR}(t)\right) \quad (2)$$

where || denotes sequential concatenation of all parent node hashes prior to hashing. This construction follows standard Merkle tree practice, where concatenation preserves the full entropy of each parent hash and maintains collision resistance under standard cryptographic assumptions. Specifically, if the underlying hash function $\text{Hash}(\cdot)$ is collision-resistant, then any modification to a parent node hash $\mathcal{H}_{p_k}$ produces a distinct input to $\text{Hash}(\cdot)$, ensuring that $\mathcal{H}_t$ changes unpredictably. This provides strictly stronger tamper-resistance guarantees than XOR-based aggregation, which can cancel out hash values when two parent hashes are identical, potentially weakening collision resistance. The above recursive hash chain ensures that any content modification to an arbitrary historical node will cause cascading invalidation of the hash values of all its successor nodes, thereby achieving lightweight tamper-resistant protection without relying on a global ledger. The overall Merkle-DAG architecture is illustrated in Figure 1, where each coalition representative node maintains a local DAG shard, and global DAG consistency synchronization is achieved through cross-coalition anchor nodes.

To prevent malicious nodes from forging coordination records or illegally joining coalitions, the security architecture introduces a lightweight attribute-based identity verification mechanism. Each device v_i holds an attribute certificate issued during the system initialization phase: $\text{Cert}_i = \text{Sign}_{SK_{\text{sys}}}(\phi_i || \text{ID}_i | t_{\text{expire}})$, where SK_{sys} is the system private key and t_{expire} is the certificate expiration time. When a device joins a coalition, it must submit its attribute certificate to the coalition representative node, which completes admission authentication by verifying $\text{Verify}_{PK_{\text{sys}}}(\text{Cert}_i) = 1$. The entire verification process requires only a single elliptic curve signature verification with computational overhead of O(1), accommodating the real-time authentication requirements of resource-constrained devices. The local DAG of a single coalition can only guarantee the integrity of intra-coalition records and cannot resist cross-coalition coordination forgery attacks. To address this, the security architecture designs a periodic global anchoring mechanism: at every time window of ΔT_{anchor}, each coalition representative node submits the current root hash of its local DAG to the global consensus committee, which aggregates the root hashes of all coalitions into a global anchor hash $\mathcal{H}_{\text{global}}^{(r)}$ through a Merkle tree:

$$\mathcal{H}_{\text{global}}^{(r)} = \text{MerkleRoot}\left(\mathcal{H}_{\pi_1}^{(r)}, \mathcal{H}_{\pi_2}^{(r)}, \dots, \mathcal{H}_{\pi_M}^{(r)}\right) \quad (17)$$

After the global anchor hash is broadcast to all coalition nodes, it serves as the publicly verifiable benchmark for the integrity of network-wide coordination records. For any node to verify the integrity of a historical coordination record, it is only necessary to provide the DAG path from the target record node to the nearest anchor point, with a verification path length of $\mathcal{O}(log\,|\mathcal{D}|)$, significantly superior to the linear complexity of full ledger verification in blockchain solutions. Under the assumption of computational security, if an adversary attempts to tamper with the content of any historical node n_{t_0} in the DAG, it must recompute the hash values of all successor nodes from n_{t_0} to the current DAG frontier, and complete the replacement of the global anchor

point before the arrival of the next anchoring window ΔT_{anchor}. Let the collision resistance strength of the hash function be κ bits; the upper bound on the probability of successful tampering is:

$$Pr[\text{Tamper}] \leq \frac{|\mathcal{D}_{\text{suffix}}|}{2^{\kappa}} \cdot \frac{\Delta T\text{anchor}}{T_{\text{compute}}} \quad (18)$$

where $|\mathcal{D}_{\text{suffix}}|$ denotes the number of successor nodes requiring recomputation and T_{compute} denotes the time for a single hash computation. Under the computational security assumption that no probabilistic polynomial-time adversary can find collisions in $\text{Hash}(\cdot)$ with non-negligible probability, Eq. (18) provides a quantifiable upper bound on the tampering success probability, which decreases exponentially with κ and the number of suffix nodes to be recomputed. In practical deployments, when $\kappa = 256$ (SHA-256) and ΔT_{anchor} is set to the order of seconds, the above probability is negligible in the computational sense, thereby demonstrating the practical security of the proposed security architecture in multi-coalition concurrent coordination scenarios. The coalition-based orchestration is designed to jointly minimize end-to-end latency and system energy consumption while maximizing service reliability, with detailed performance validation provided in Section 4.

4. Experimental evaluation

4.1 Experimental environment and testbed configuration

Three metropolitan testbeds were constructed to evaluate the proposed DCRO framework: a high-density commercial district (Testbed-A), a transportation hub (Testbed-B), and a mixed residential scenario (Testbed-C), differing substantially in device density, heterogeneity, and network dynamics.

The platform combines a real heterogeneous device cluster with NS-3 network simulation. Three baselines are compared: (1) CEN: a centralized scheduling approach in which a global cloud controller collects full network state and allocates resources to all devices via a greedy priority queue, representative of traditional cloud-centric IoT management [2]; (2) HIE: a hierarchical management scheme that partitions devices into fixed clusters with dedicated cluster heads responsible for local scheduling and inter-cluster coordination, following the architecture in [4]; and (3) DNC: a coalition-free distributed scheduling method in which each device independently manages its own resources without cooperative mechanisms, serving as a lower-bound reference for the benefit of coalition formation [8]. Evaluation metrics include average latency, resource utilization, energy efficiency, fault recovery time, and system throughput. Detailed parameters are listed in Table 3.

To ensure reproducibility, we provide full details of the experimental environment. All experiments were conducted on a server cluster comprising four nodes, each equipped with dual Intel Xeon Gold 6258R processors (28 cores, 2.7 GHz), 256 GB DDR4 RAM, and NVIDIA A100 GPUs (40 GB HBM2). The operating system was Ubuntu 20.04 LTS with Python 3.9 and NS-3.36 for network simulation. Key NS-3 parameters are listed in Table 4.

Device capability vectors were synthetically generated to reflect realistic metropolitan IoT heterogeneity: computational capacity c_i^{cpu} was sampled uniformly from [100, 2000] MIPS, memory m_i^{mem} from [64, 4096] MB, bandwidth b_i^{bw} from [1, 100] Mbps, and residual energy e_i^{bat} from [500, 10000] J, consistent with published IoT device benchmarks [REF].

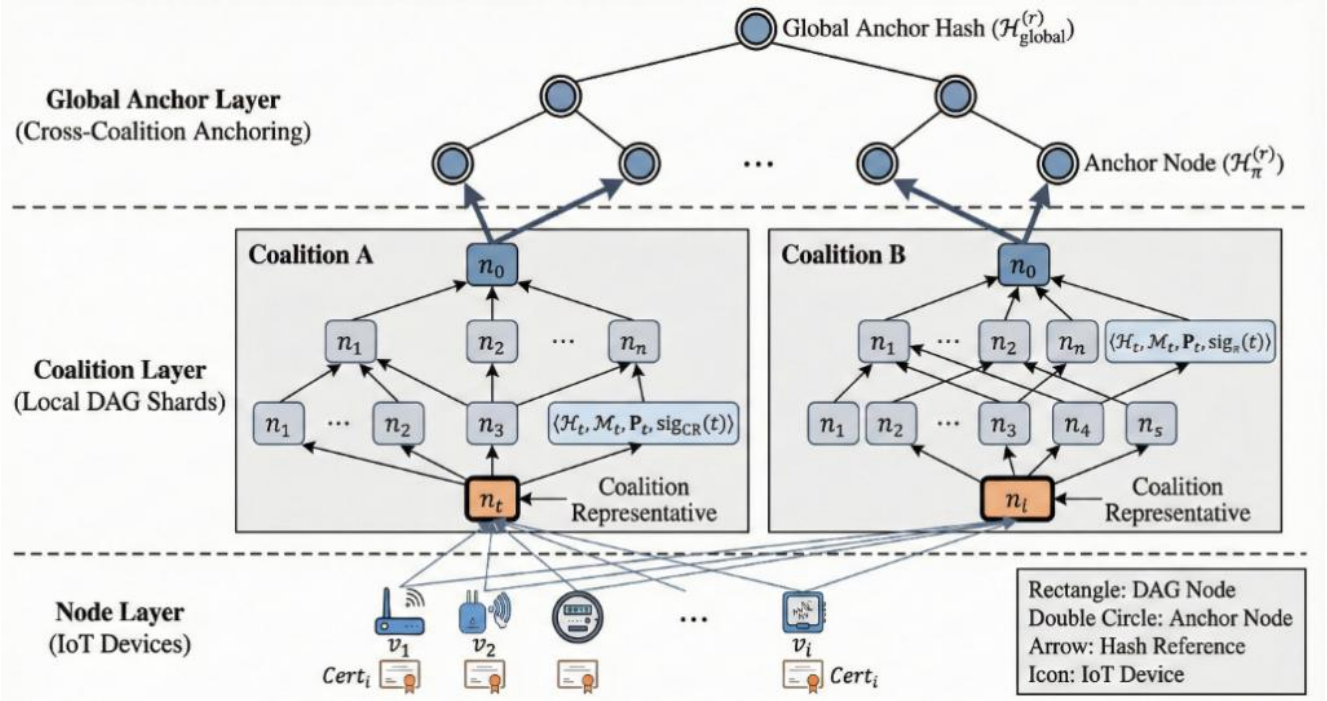

Figure 1. Schematic diagram of the Merkle-DAG distributed security architecture

Device types were assigned in proportions of 60% sensors, 30% gateways, and 10% edge nodes, reflecting typical metropolitan deployment ratios. Task arrival processes followed Poisson distribution with rates specified in Table 3. All random processes were initialized with a fixed seed of 42, and each experiment was repeated 10 independent times with results averaged to ensure statistical stability.

Table 3. Configuration parameters of the three metropolitan testbeds

Parameter	Testbed-A (Commercial)	Testbed-B (Transport Hub)	Testbed-C (Residential)
Total devices N	2,400	1,800	1,200
Device type count	8	6	5
Edge server count	12	9	6
Topology change rate	High (15/min)	Medium (8/min)	Low (3/min)
Average device failure rate	2.1%	1.6%	0.9%
Task arrival rate (tasks/s)	320	210	140
Maximum tolerable latency $D^{\max}$ (ms)	50	80	120
Simulation duration (min)	60	60	60

Table 4. NS-3 Simulation configuration parameters

Parameter	Value
NS-3 version	3.36
Simulation area	5 km × 5 km
Channel model	Log-distance path loss ($\alpha = 3.5$)
Communication range R_i^{comm}	50–500 m (device-type dependent)
Carrier frequency	2.4 GHz
Transmission power P_i^{tx}	10–100 mW
Link failure probability	0.01–0.05
Simulation warm-up period	5 min
Random seed	42
Independent runs per experiment	10

Statistical significance of all reported performance differences was assessed using the Wilcoxon rank-sum test at a significant level of $\alpha = 0.01$, with results reported as mean ± standard deviation across 10 independent runs.

4.2 System response latency performance analysis

System response latency is the key performance indicator for evaluating the quality of service in metropolitan IoT systems. Figure 2 depicts the curves for the average end-to-end latency achieved by all four methods in the three testbeds, considering different task arrival rates. It is observed from Figure 2 that the proposed framework achieves optimal performance in all the scenarios in terms of latency.

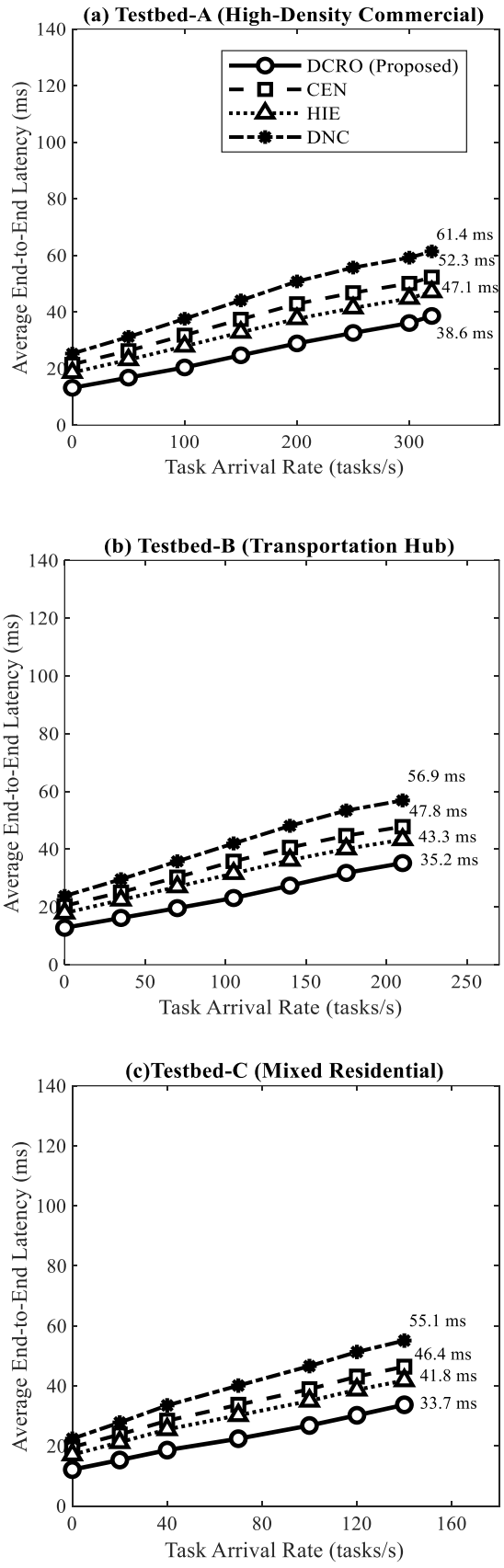

Figure 2. Average end-to-end latency curves of four methods across three testbeds as a function of task arrival rate

In the high-density commercial district scenario of Testbed-A, when the arrival rate of tasks reaches 320 tasks/s, the average latency of DCRO is 38.6 ± 1.2 ms (mean ± standard deviation over 10 independent runs), outperforming CEN (52.3 ± 3.8 ms), HIE (47.1 ± 2.9 ms), and DNC (61.4 ± 4.3 ms), achieving latency reductions of 26.2%, 18.1%, and 37.1% respectively. Wilcoxon rank-sum tests confirm that all pairwise differences between DCRO and each baseline are statistically significant ($p < 0.01$). The latency of CEN grows sharply in high-load scenarios, especially because the scheduling bottleneck of the central controller worsens considerably with the increase in device scale. Although HIE improves the bottleneck to some degree, the bottleneck of inter-level synchronization still exists. The bottleneck of DNC results from its extremely low resource utilization caused by the lack of cooperation mechanisms in coalitions, which leads to a sharp increase in the latency of queuing tasks. The proposed DCRO bypasses the bottleneck of the central controller of CEN and reduces the latency of tasks waiting to be scheduled by the controller. Regarding tail latency, DCRO achieves a P99 latency of 61.2 ms and standard deviation of 8.3 ms under Testbed-A, substantially lower than CEN (P99: 103.7 ms, std: 19.6 ms), demonstrating effective suppression of latency jitter under extreme load conditions.

4.3 Resource utilization and energy efficiency analysis

Resource utilization and energy efficiency are directly related to the economic viability and sustainability of metropolitan-scale IoT systems. Figure 3 depicts a comparison of CPU resource utilization and total system energy consumption for the four methods on the three testbeds. As illustrated in Figure 3, DCRO obtains considerable advantages on both dimensions of resource utilization and energy consumption.

In terms of resource utilization, DCRO obtains average CPU utilization rates of 82.3 ± 1.8%, 79.6 ± 1.5%, and 76.8 ± 1.3% on Testbed-A, Testbed-B, and Testbed-C respectively, while CEN obtains corresponding rates of 67.4%, 65.2%, and 63.1%. The high resource utilization rate of DCRO is derived from the intra-coalition resource pooling mechanism in which all devices contribute their idle computational resource to the shared pool for uniform resource allocation using the Shapley value mechanism of GT-CFM, thus eliminating the resource fragmentation issue in traditional resource allocation methods. In terms of energy efficiency, DCRO obtains an average energy consumption reduction of 19.7% compared to DNC, mainly because of the optimization of communication paths using the SODCA algorithm in clustering structures to reduce energy consumption in long-distance data transmission. In terms of per-task energy efficiency, DCRO achieves 124.3 ± 3.1 μJ/task on Testbed-A, representing a 31.4% reduction compared to CEN (181.2 ± 5.4 μJ/task, $p < 0.01$ by Wilcoxon rank-sum test), with consistent advantages across all three testbeds.

4.4 Fault recovery capability and system stability analysis

Fault recovery robustness was evaluated through random device fault injection and network partition simulation. Under continuous fault injection at 2%/min on Testbed-A, DCRO sustains only 11.3 ± 0.9% throughput degradation, compared to 42.6 ± 4.1%, 28.3 ± 2.7%, and 21.7 ± 2.1% for CEN, HIE, and DNC respectively, with all differences statistically significant ($p < 0.01$, Wilcoxon rank-sum test).

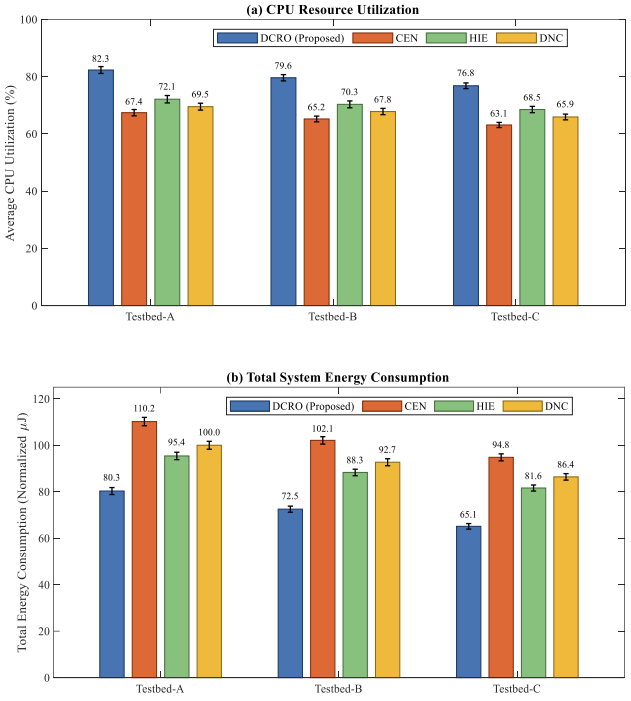

Figure 3. Comparative bar charts of resource utilization (top) and total system energy consumption (bottom) of four methods across three testbeds

This resilience stems from localized fault response via Island Autonomous Mode and coalition-level task migration, avoiding global disruption. As shown in Table 5, DCRO achieves an average recovery time of 1.83 seconds, far below CEN's 12.47 seconds, validating the LHCP re-election mechanism. These results are consistent with the theoretical complexity advantages of LHCP over PBFT summarized in Table 4: whereas PBFT's $\mathcal{O}(n^2)$ message complexity would generate approximately $N^2 = 5.76 \times 10^6$ messages at the scale of Testbed-A ($N = 2400$), LHCP's $\mathcal{O}(n\ log\ n)$ complexity reduces this to approximately $2400 \times \log_2(2400) \approx 27000$ messages, a reduction of over 99%, which directly contributes to the observed recovery time advantage.

4.5 Scalability analysis

Scalability is an important dimension for evaluating whether the framework can adapt to the continuously growing scale of metropolitan IoT deployments. Experiments measure the trends of system average latency and coalition formation convergence time as a function of device count for each method, with device scale linearly expanded from 200 to 5,000, as shown in Figure 4.

DCRO's average latency exhibits an approximately logarithmic growth trend as device scale increases, whereas CEN demonstrates pronounced super-linear growth, exceeding the maximum latency constraint threshold when device scale reaches 3,000 and losing service capability entirely. The local response mechanism of the SODCA algorithm constrains the processing complexity of topology reorganization to $\mathcal{O}(|\mathcal{N}_{\text{affected}}|^2)$, enabling the framework to maintain an average latency of 39.4 ms even when device count reaches the scale of 5,000, validating the excellent scalability of the proposed framework. Regarding coalition formation convergence speed, as shown on the right side of Figure 4, DCRO converges to Nash-stable equilibrium within 15 iterations across all device scales, with the growth in

convergence iteration count relative to device scale substantially below the theoretical upper bound, demonstrating that the distributed iterative mechanism of GT-CFM possesses good scale robustness.

Table 5. Fault recovery performance comparison of each method under network partition scenarios

Method	Avg. Recovery Time (s)	Max. Recovery Time (s)	Throughput Retention Rate During Recovery (%)	Data Integrity Verification Pass Rate (%)
DCRO (Proposed)	1.83	4.21	88.7	99.6
CEN	12.47	28.63	31.2	94.3
HIE	6.34	15.82	58.4	96.8
DNC	4.17	9.56	71.3	97.1

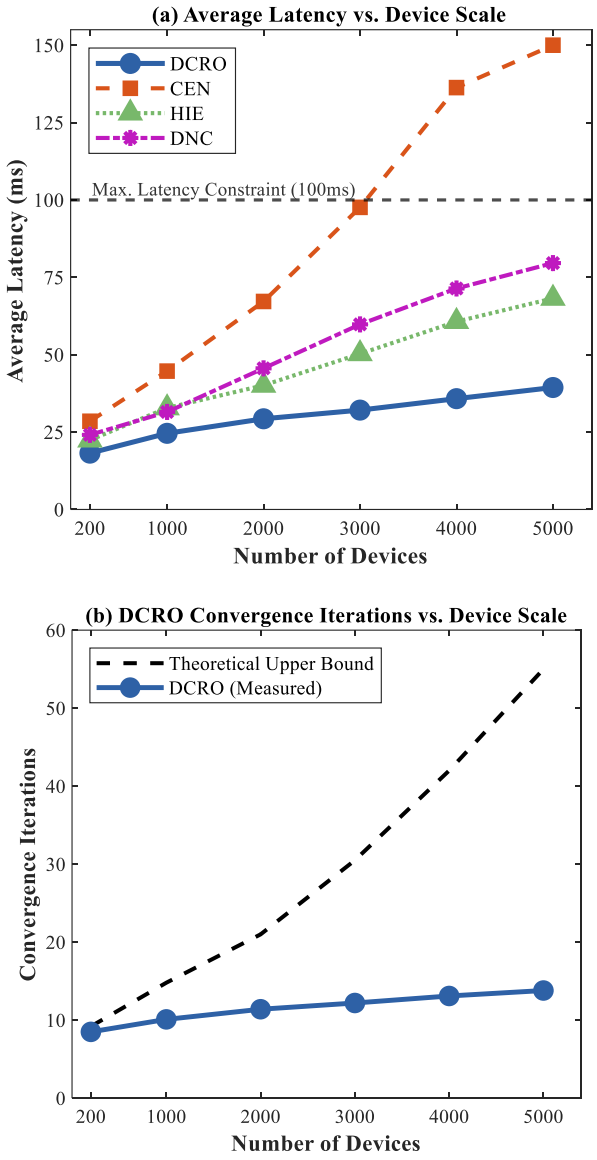

Figure 4. Average latency variation curves of each method as device scale expands from 200 to 5,000 and DCRO coalition formation convergence iteration count

5. Conclusion

This paper proposes a Distributed Coalition-based Resource Orchestration framework for heterogeneous IoT devices in smart cities, called DCRO. SODCA utilizes Nash stability in coalition formation via Fiedler value-based topology awareness. GT-CFM uses Shapley values for fair resource allocation and adaptive split-merge rules. LHCP significantly reduces consensus communication overheads without compromising fault tolerance. Finally, the use of a Merkle-DAG architecture allows for lightweight state verification. Evaluation of DCRO in three metropolitan-scale testbeds indicates a 26.2% latency reduction and 31.4% energy savings compared to a centralized solution. Additionally, 11.3% throughput degradation under continuous fault injection and stable coalition formation in 15 iterations with 5,000 devices are achieved. Future directions include heterogeneous wireless channel effects and federated learning-based privacy.

Ethical issue

The authors are aware of and comply with best practices in publication ethics, specifically regarding authorship (avoidance of guest authorship), dual submission, manipulation of figures, competing interests, and compliance with research ethics policies. The authors adhere to publication requirements that the submitted work is original and has not been published elsewhere.

Data availability statement

The manuscript contains all the data. However, additional data will be provided by the corresponding author upon reasonable request.

Conflict of interest

The authors declare no potential conflict of interest.

References

[1] M. Trigka, E. Dritsas, Edge and cloud computing in smart cities, Future Internet 17(3) (2025) 118. https://doi.org/10.3390/fi17030118

[2] S. Gokulakrishnan, J. Gnanasekar, Peer-toPeer convoluted fault recognition to conquer Single-Point stoppage in Cloud systems, International Journal of Pure and Applied Mathematics 116(21) (2017) 559-577. https://www.researchgate.net/publication/321127306_Peer-to peer_convoluted_fault_recognition_to_conquer_single-point_stoppage_in_Cloud_systems

[3] L. Zhao, J. Wang, J. Liu, N. Kato, Optimal edge resource allocation in IoT-based smart cities, IEEE Network 33(2) (2019) 30-35. DOI: 10.1109/MNET.2019.1800221

[4] Y. Zhang, F. Lyu, P. Yang, W. Wu, J. Gao, IoT intelligence empowered by end-edge-cloud orchestration, China Communications 19(7) (2022) 152-156. DOI: 10.23919/JCC.2022.9837843

[5] N. Kherraf, H.A. Alameddine, S. Sharafeddine, C.M. Assi, A. Ghrayeb, Optimized provisioning of edge computing resources with heterogeneous workload in IoT networks, IEEE Transactions on Network and Service Management 16(2) (2019) 459-474. DOI: 10.1109/TNSM.2019.2894955

[6] C. Chi, Y. Wang, X. Tong, M. Siddula, Z. Cai, Game theory in Internet of Things: A survey, IEEE Internet

of Things Journal 9(14) (2021) 12125-12146. DOI: 10.1109/JIOT.2021.3133669

[7] A. Shahraki, A. Taherkordi, Ø. Haugen, F. Eliassen, A survey and future directions on clustering: From WSNs to IoT and modern networking paradigms, IEEE Transactions on Network and Service Management 18(2) (2020) 2242-2274. DOI: 10.1109/TNSM.2020.3035315

[8] S. Shamshirband, J.H. Joloudari, S.K. Shirkharkolaie, S. Mojrian, F. Rahmani, S. Mostafavi, Z. Mansor, Game theory and evolutionary optimization approaches applied to resource allocation problems in computing environments: A survey, Mathematical Biosciences and Engineering 18(6) (2021) 9190-9232. doi: 10.3934/mbe.2021453

[9] S. Durand, K. Khawam, D. Quadri, S. Lahoud, S. Martin, Cross Device Distributed Federated Learning Coalition Formation Game for Constrained IoT, IEEE Internet of Things Journal (2025). DOI: 10.1109/JIOT.2025.3584417

[10] C.-C. Lin, Y. Chiang, H.-Y. Wei, Multi-service edge computing management with multi-stage coalition game task offloading, IEEE Transactions on Network and Service Management 21(3) (2024) 3278-3291. DOI: 10.1109/TNSM.2024.3358414

[11] R. Guo, Z. Guo, Z. Lin, W. Jiang, A hierarchical byzantine fault tolerance consensus protocol for the internet of things, High-Confidence Computing 4(3) (2024) 100196. https://doi.org/10.1016/j.hcc.2023.100196

[12] E.U. Haque, W. Abbasi, A. Almogren, J. Choi, A. Altameem, A.U. Rehman, H. Hamam, Performance enhancement in blockchain based IoT data sharing using lightweight consensus algorithm, Scientific reports 14(1) (2024) 26561. https://doi.org/10.1038/s41598-024-77706-x

[13] R. Du, Z. Wang, J. Shen, Certificateless data integrity auditing with sparse Merkle trees for the cloud-edge environment, Scientific Reports 15(1) (2025) 39202. https://doi.org/10.1038/s41598-025-14041-9

[14] L. Cui, C. Xu, S. Yang, J.Z. Huang, J. Li, X. Wang, Z. Ming, N. Lu, Joint optimization of energy consumption and latency in mobile edge computing for Internet of Things, IEEE Internet of Things Journal 6(3) (2018) 4791-4803. DOI: 10.1109/JIOT.2018.2869226

[15] W. Zhang, H. Ou, Reinforcement learning based multi objective task scheduling for energy efficient and cost effective cloud edge computing, Scientific Reports 15(1) (2025) 41716. https://doi.org/10.1038/s41598-025-25666-1

[16] N.M. Dankolo, N.H.M. Radzi, N.H. Mustaffa, N.I. Arshad, M. Nasser, D. Gabi, M.N. Yusuf, Optimizing resource allocation for IoT applications in the edge cloud continuum using hybrid metaheuristic algorithms, Scientific reports 15(1) (2025) 14409. https://doi.org/10.1038/s41598-025-97648-2

[17] S.A. Memon, D. Andriukaitis, D. Navikas, V. Markevicius, A. Valinevicius, M. Zilys, M. Prauzek, J. Konecny, P. Brida, Z. Li, Centralized and Distributed Controller Placement in Terms of Scalability and Fault Tolerance: A Review, 2025 29th International Conference on Methods and Models in Automation and Robotics (MMAR), IEEE, 2025, pp. 449-454. DOI: 10.1109/MMAR65820.2025.11150828

[18] Z. Ali, A. Mahmood, S. Khatoon, W. Alhakami, S.S. Ullah, J. Iqbal, S. Hussain, A generic Internet of Things (IoT) middleware for smart city applications, Sustainability 15(1) (2022) 743. https://doi.org/10.3390/su15010743

[19] A. Casteigts, P. Flocchini, W. Quattrociocchi, N. Santoro, Time-varying graphs and dynamic networks, International Journal of Parallel, Emergent and Distributed Systems 27(5) (2012) 387-408. https://doi.org/10.1080/17445760.2012.668546

[20] S. Mayukha, R. Vadivel, Optimizing resource allocation in intelligent communication networks: Fundamentals and challenges, Machine Learning for Radio Resource Management and Optimization in 5G and Beyond, CRC Press2025, pp. 15-39. https://www.taylorfrancis.com/chapters/edit/10.1201/9781003514336-2/optimizing-resource-allocation-intelligent-communication-networks-mayukha-vadivel

[21] S. Hudda, K. Haribabu, A review on WSN based resource constrained smart IoT systems, Discover Internet of things 5(1) (2025) 56. https://doi.org/10.1007/s43926-025-00152-2

[22] N.C. Luong, Z. Sui, D. Van Le, J. Cao, B. Ma, N.D. Hai, R. Zhang, V. Van Quang, D. Niyato, S. Feng, Incentive Mechanism Design for Resource Management in Satellite Networks: A Comprehensive Survey, IEEE Internet of Things Journal 13(3) (2025) 3938-3964. DOI: 10.1109/JIOT.2025.3637167

[23] M.H. Amini, J. Mohammadi, S. Kar, Distributed holistic framework for smart city infrastructures: Tale of interdependent electrified transportation network and power grid, Ieee Access 7 (2019) 157535-157554. DOI: 10.1109/JIOT.2025.3637167

[24] J.M. Zolezzi, H. Rudnick, Transmission cost allocation by cooperative games and coalition formation, IEEE Transactions on power systems 17(4) (2003) 1008-1015. DOI: 10.1109/TPWRS.2002.804941

[25] L. Qin, Y. Zhu, S. Liu, X. Zhang, Y. Zhao, The Shapley value in data science: advances in computation, extensions, and applications, Mathematics 13(10) (2025) 1581. https://doi.org/10.3390/math13101581

[26] R. Massin, C.J. Le Martret, P. Ciblat, A coalition formation game for distributed node clustering in mobile ad hoc networks, IEEE Transactions on Wireless Communications 16(6) (2017) 3940-3952. DOI: 10.1109/TWC.2017.2690419