Resource Synchronization Mechanism Design in Computing Power Networks: An Adaptive Multi-Agent Reinforcement Learning Framework

Siyi Huang, Nuocheng Yang, Sihua Wang, Xinxin He, Changchuan Yin
Beijing Laboratory of Advanced Information Network, Beijing University of Posts and Telecommunications, Beijing, China
Emails: {huangdayi,yangnuocheng,sihuawang,hxx_9000,ccyin}@bupt.edu.cn.

2025 11th International Conference on Computer and Communications (ICCC) | 979-8-3315-4558-1/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICCC68654.2025.11438141

***Abstract*—The computing power network (CPN) is an emerging architecture for efficient sharing of computing resources, which requires timely and accurate network-wide computing resource information collection. However, existing information synchronization mechanisms in CPN, which are based on fixed rules employed on a centralized controller, face significant challenges in adapting to highly dynamic fluctuations of resources. Furthermore, the decentralized nature of large-scale networks poses a serious threat to achieving globally coordinated and efficient information exchange. To address these issues, we propose a novel adaptive and intelligent distributed synchronization framework. Specifically, we design a gradient-adaptive reporting strategy at the computing devices to dynamically adjust reporting intervals based on the resource variation rate, significantly reducing communication overhead while capturing critical state changes. Then, to tackle the coordination complexity and information staleness caused by decentralized decision-making, we introduce a multi-agent reinforcement learning to learn optimal notification strategies that balance information freshness, accuracy, and transmission costs. Extensive simulations on a real-world CPU utilization dataset demonstrate that our method significantly reduces communication costs and sampling frequency while substantially improving information freshness and reconstruction accuracy, highlighting its effectiveness for scalable and dynamic computing power networks.**

***Index Terms*—AI-enabled wireless networks, Multi-agent reinforcement learning, Resource allocation.**

I. Introduction

With the rapid advancement of artificial intelligence (AI), computing power network (CPN), which provides flexible services by mobilizing computing resources across cloud and edge devices, has attracted much attention [1]–[3]. To optimize the performance of CPN, real-time collection and synchronization of dynamic computing resources information are needed [4]–[6], which causes excessive transmission overhead and delay due to the frequent data updates [7]. Therefore, the optimization of transmission overhead in information collection and synchronization mechanisms has been widely discussed.

Recently, a number of works such as [8]–[10] have studied collect information from network-wide based on centralized controllers. The authors in [8] employed in-band network telemetry to embed computing resource information into the headers of service packets, which are periodically transmitted to the centralized controller. To further minimize the information transmission overhead, the authors in [9] designed an interval control algorithm that only sends data that exceeds the change threshold. The authors in [10] applied reinforcement learning (RL) to find the optimal device scheduling strategies for traffic allocation, thus reducing the information transmission overhead. However, centralized synchronization methods typically rely on global information for optimization, which faces challenges of centralized bottlenecks and poor scalability due to the expansion of network scale. In contrast, distributed synchronization architectures exhibit advantages in fault tolerance, scalability, and adaptability to dynamic environments.

Recently, a number of works such as [11]–[13] have investigated synchronization techniques in a distributed architecture, where servers share computing power resources. The authors in [11] designed a mechanism that integrates periodic multicast and threshold-triggered synchronization in distributed scenarios to optimize resource collection overhead. To reduce synchronization latency, the authors in [12] proposed an offline knapsack-based algorithm and an online randomized greedy algorithm for optimizing the synchronization period. The authors in [13] employed single-agent reinforcement learning to select subsets of announcement objects, evaluating synchronization effectiveness through task completion latency and network overhead, and capturing the characteristics of resource information changes. Moreover, these works [11]–[13] generally consider direct end-to-end synchronization between distributed routers, overlooking the regulatory function of routers in CPN and essential processes like device-to-router data reporting.

To address these challenges, we develop a comprehensive distributed reporting-and-notification framework for CPN to balance computing resources synchronization effectiveness between device-to-router (named reporting) and router-to-router (named notification), and the overall transmission overhead. Our key contributions are as follows:

- We propose a novel reporting and notification framework in CPN, which devices adaptively schedule report intervals while routers select and notify a subset of neighbor routers. It is formulated as a long-horizon optimization problem whose aim is to maintain up-to-date computing resource information across the network under communication re-

This work was supported by the National Key Research and Development Program of China under Grant 2024YFE0200300.

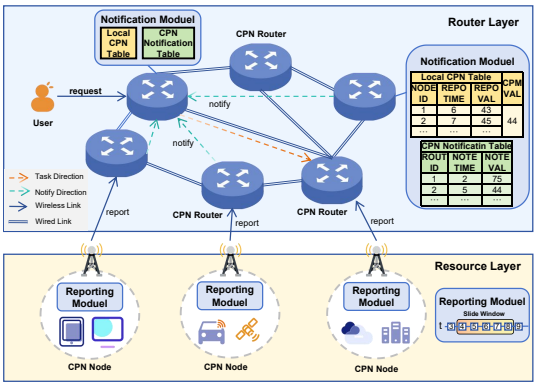


Fig. 1: Considered computing power network system.

sources and delay constraints.

- To solve the formulated problem, we propose a hybrid solution that combines a lightweight gradient-adaptive reporting mechanism with a multi-agent reinforcement learning method notification policy. Compared to traditional periodic or threshold synchronization strategies, the proposed method can achieve a superior trade-off between synchronization effectiveness and communication overhead without relying on manual rule tuning.
- Numerical evaluation results show that the proposed approach reduces device sampling by about 20% and reconstruction MSE by up to 31.9% compared to periodic or threshold baselines.

II. System Model and Problem Formulation

A. System Setup

Consider a distributed CPN, composed of a set of devices $\mathcal{N} = \{1, \ldots, N\}$ and a set of routers $\mathcal{R} = \{1, \ldots, R\}$. Each router p connects with a subset of devices $\mathcal{N}_p$ and a subset of neighbor routers $\mathcal{R}_p$, as shown in Fig. 1. The local computing resource of device n at time slot t is $a_{n,t} \in \mathbb{R}$, which needs to be synchronized to each CPN router. To autonomously monitor local computing power variations while meeting the resource limitations, a computing resource reporting model is deployed on each device to control the reporting interval of $a_{n,t}$. Based on the collected computing power information $a_{n,t}$ from $\mathcal{N}_p$, each CPN router p integrates notification modules to exchange the aggregated information $U_{p,t} = \{\boldsymbol{\tau}_{p,t}, \boldsymbol{c}_{p,t}\}$ with a subset of neighbors in $\mathcal{R}_p$ for global information synchronization, where $\boldsymbol{\tau}_{p,t} = [\tau_{1,t}, \cdots, \tau_{n,t}]$ stores the timestamps of the most recent reports from each device $n \in \mathcal{N}_p$, $\boldsymbol{c}_{p,t} = \left[a_{1,\tau_{1,t}}, \cdots, a_{n,\tau_{n,t}}\right]$ represents the resources information $a_{n,t}$ received at time $\boldsymbol{\tau}_{p,t}$. The aggregated information $U_{p,t}$ are stored in the CPN notification table which each CPN router maintains, and the table records the most recent notification information of its neighbor for notifying decision making itself, rather than for broadcasting to others. Due to the resource limitation, each router p must select a subset of neighbors to exchange its local computing power information $U_{p,t}$ in time slot t. In summary, the process of reporting and notification in the considered computing power network system is as follows:

1) Each device in CPN n dynamically reports its local computing power information $a_{n,t}$ to its connected router p based on the equipped reporting modules through a wireless connection.
2) Each router p in CPN aggregates received computing power information and generate $U_{p,t}$.
3) Each router p in CPN selects a subset of CPN routers to notify aggregated reporting information $U_{p,t}$, and updates notification status in the CPN notification table accordingly.

B. Reporting Resources Information from Devices to Routers

*1) **Reporting Module**:* To reduce the communication overhead due to repeatedly information reporting, each CPN device needs to adjust reporting interval $\delta_{n,t} \in \mathbb{Z}^+$ based on the recent computing power fluctuations stored in $w_{n,t} = [a_{n,t-W+1}, \ldots, a_{n,t}]$, where W is the length of slide windows. The reporting interval is determined by

$$\delta_{n,t} = f(w_{n,t}), \tag{1}$$

where $f(\cdot)$ is a predefined function that can generate a reporting interval to improve the information synchronization efficiency. The time slot for device n to report $a_{n,t}$ to the connected router is given by:

$$t_n^{\text{report}}(k+1) = t_n^{\text{report}}(k) + \delta_{n,t_n^{\text{report}}(k)}, \tag{2}$$

where $t_n^{\text{report}}(k)$ denotes the k-th reporting decision time of device n and $k \in \mathbb{Z}^+$. The binary reporting action variable $s_{n,t}$ can be formulated as:

$$s_{n,t} = \begin{cases} 1, & \text{if } t = t_n^{\text{report}}(k), k \in \mathbb{Z}^+, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

*2) **AoI Models for Devices in CPN**:* To capture the freshness of computer power information $a_{n,t}$ collected by its connected routers in the considered CPN system, an age of information (AoI) indicator is introduced. The AoI of the resources from CPN device n at time slot t is given by

$$\phi_{n,t}(s_{n,t}) = \begin{cases} t_{n,t}^{\text{trans}}, & \text{if } s_{n,t} = 1, \\ \phi_{n,t-1}(s_{n,t-1}) + \Delta t, & \text{otherwise,} \end{cases} \tag{4}$$

where $t_{n,t}^{trans}$ denotes the wireless transmission delay between device n and its connected router, Δt represents the slot duration. Hence, upon successful reception, the device-level AoI at the router is reset to the network transmission delay.

C. Notification Resources Information between Routers

*1) **Notification Module**:* $\boldsymbol{\tau}_{p,t}$ in $U_{p,t}$ is the time slot corresponding to the latest computation resource information reporting from $\mathcal{N}_p$ received by router p, $\tau_{n,t} \in \boldsymbol{\tau}_{p,t}$ can be represented as

$$\tau_{n,t} = \max_{n \in \mathcal{N}_p} \left\{ t_n^{\text{report}} + t_n^{\text{trans}} \right\}, \tag{5}$$

where t_n^{report} represents the timestamp that device n decides to report its data, t_n^{trans} represents the time slot required for data to transmit through the wireless network. Thus, it may deviate from the true data $a_{n,t}$ of the current time slot due to latency in data transmission. We define $t_{p,q}$ as the last time that router p notifies its local computing power information to router q, and $\tilde{\boldsymbol{c}}_{p,q,t}$ is the computing power value notified, which is calculated by

$$[\tilde{\boldsymbol{c}}_{p,q,t}]_n = \begin{cases} [\boldsymbol{c}_{p,t}]_n, & \text{if } b_{p,q,t}=1, \\ [\tilde{c}_{p,q,t_{p,q}}]_n, & \text{otherwise.} \end{cases} \tag{6}$$

We introduce the decision about information notification $b_{p,q,t} \in \{0,1\}$, where $b_{p,q,t}=1$ represents the CPN router p transmitting $\boldsymbol{c}_{p,q,t}$ to router q at time slot t, and $b_{p,q,t}=0$, otherwise. Let $\boldsymbol{D}_{p,t} \in \{0,1\}^{|\mathcal{R}|}$ denote the binary adjacency vector maintained by CPN router p at time t, where each element $[\boldsymbol{D}_{p,t}]_q = 1$ indicates that router p can directly communicate with router q, and $[\boldsymbol{D}_{p,t}]_q = 0$ otherwise. This adjacency vector serves as a lightweight representation of the router-level connectivity structure within the local domain and is updated dynamically according to link status, topology changes, or control-plane signaling.

*2) **AoI Models for Routers in CPN**:* When a router p receives and notifies the aggregated resource information $\boldsymbol{c}_{p,t}$ from its connected devices, it needs to choose a subset of neighbors $\mathcal{R}_p$ to share. The delay in the transmission of the updated information between router p and q is calculated as

$$t_{p,q}^{\text{route}} = \frac{d_{p,q}}{R_{p,q}}, \tag{7}$$

where $d_{p,q}$ is the payload size of the update message and $R_{p,q}$ is the data rate. We denote the AoI of $\boldsymbol{c}_{p,q,t}$ at router q for router p as

$$\begin{aligned} &\Phi_{p,q,t}(s_{n,t}, b_{p,q,t}) \\ &= \begin{cases} \phi_{n,t} + (t_{p,q} - [\boldsymbol{\tau}_{p,t}]_n) + t_{p,q,t}^{\text{route}}, & \text{if } b_{p,q,t}=1, \\ \Phi_{p,q,t-1}(s_{n,t-1}, b_{p,q,t-1}) + \Delta t, & \text{otherwise.} \end{cases} \end{aligned} \tag{8}$$

Then, we can denote the average AoI of all routers $\overline{\Phi}_t$ as

$$\overline{\Phi}_t = \frac{\sum_{p,q\in\mathcal{R},p\neq q} \Phi_{p,q,t}}{\sum_{p\in\mathcal{R}} \|\boldsymbol{D}_{p,t}\|_1}. \tag{9}$$

Therefore, the AoI not only depends on the notification policy between routers, but also on the AoI of the data currently held by routers.

*3) **Error between Router Pairs**:* For each pair of routers p and q, we denote the error $e_{p,q,t}$ between the current perceived value $\boldsymbol{c}_{p,q,t}$ and the last notified value $\tilde{\boldsymbol{c}}_{p,q,t}$ as follow

$$e_{p,q,t} = \|\boldsymbol{c}_{p,t} - \tilde{\boldsymbol{c}}_{p,q,t}\|_1. \tag{10}$$

The error $e_{p,q,t}$ represents an absolute difference, which is caused by untimely notification policies. Then, we capture the pairwise error for all agent pairs in the system by aggregating the delta into a matrix. To evaluate the overall error across the network, we compute the average error as

$$\bar{e}_t = \frac{1}{\sum_{p\in\mathcal{R}} \|\boldsymbol{D}_{p,t}\|_1} \sum_{p,q\in\mathcal{R},p\neq q} e_{p,q,t}. \tag{11}$$

*4) **Network Synchronization Cost**:* In our model, the information synchronization cost is induced by the routing protocols required for synchronizing information dissemination, which is given by

$$O_p(b_{p,q,t}) = \sum_{q\in\mathcal{R},p\neq q} b_{p,q,t}\gamma_{p,q}(O_0 + d_{p,q}), \tag{12}$$

where O_0 represents the fixed overhead of the routing protocol header, $\gamma_{p,q}$ is a static cost factor for each directed link from router p to router q, which is calculated as hop count. The average cost of network synchronization overhead at time slot t is given by

$$\bar{O}_t = \frac{1}{R} \sum_{p\in\mathcal{R}} \sum_{q\in\mathcal{R}} O_p(b_{p,q,t}). \tag{13}$$

D. Problem Formulation

Our goal is to minimize long-term information staleness and perceptual deviation while reducing the incurred synchronization cost, and the optimization problem is formulated as

$$\min_{s_{n,t},b_{p,q,t}} \sum_{t\in\mathcal{T}} \omega_1 \cdot \frac{\overline{\Phi}_t}{\xi} + \omega_2 \cdot \bar{e}_t + \omega_3 \cdot \overline{O}_t, \tag{14}$$

$$\text{s.t. } \phi_{n,t} \le \phi_{\max}, \forall n \in \mathcal{N}, \tag{14a}$$

$$\Phi_{p,q,t} \le \Phi_{\max}, \forall p,q \in \mathcal{R}. \tag{14b}$$

where $\xi > 0$ represents the tolerance parameter to delay, ω_1, ω_2 and ω_3 are balancing weights, $\phi_{\max}$ represents the maximum allowable AoI threshold for computation updates and $\Phi_{p,q,t}$ represents the upper bound of the end-to-end AoI .

Since the formulated optimization problem involves coupled reporting and notification decisions under dynamic computing power variations, jointly optimizing these two processes leads to a large and highly non-convex solution space, making the problem difficult to solve through conventional convex-based methods. To overcome these challenges, we decompose the optimization problem (14) into two coordinated subproblems: device-side reporting control and router-side notification optimization. The decomposed parts can be optimized on their own timescale while maintaining overall coordination due to their shared global objective that jointly minimizes the AoI and communication cost across the networks.

III. Proposed Method

A. Gradient-adaptive Reporting Strategy

To monitor the device n's computing resource information variations, we first construct a cubic spline interpolation function $g_t(\tau)$, which can fit the data in $w_{n,t}$, where $\tau \in [t-W+1, t]$. Then, we define the resource variation gradient, which can be given by

$$G_t = \left|\frac{dg_t(\tau)}{d\tau}\right|_{\tau=t} + \lambda_g \cdot \left|\frac{d^2 g_t(\tau)}{d\tau^2}\right|_{\tau=t}, \tag{15}$$

where $\frac{dg_t(\tau)}{d\tau}$ and $\frac{d^2g_t(\tau)}{d\tau^2}$ denote the first-order and second-order derivatives of $g_t(\tau)$ at time slot t, and $\lambda_g \geq 0$ is the coefficient weight. G_t can capture both abrupt changes (first derivative) and smooth but nonlinear transitions (second derivative) of $w_{n,t}$, which can be utilized for the reporting decision optimization. Based on the gradient G_t, the system dynamically determines the next reporting interval:

$$\delta_{n,t} = \frac{T_{max}}{1 + k \cdot G_t}, \tag{16}$$

where T_{max} is the maximum allowed reporting interval, $k > 0$ is a sensitivity factor. It can ensure shorter intervals when variation is high. This method can override the original reporting plan when necessary to ensure data accuracy and timeliness by considering the devices' computing power fluctuations.

B. Distributed RL-based Notification Strategy

To address the notification problem in decentralized environments, we formulate the notification problem as a decentralized partially observable Markov decision process (POMDP) and introduce a Multi-Agent Proximal Policy Optimization (MAPPO) based algorithm [14] to dynamically balance notifying accuracy and communication consumption, where routers serve as agents, jointly optimizing their notification strategies.

1) Modeling of Multi-Agent POMDP: Each router in the considered CPN is treated as an autonomous agent interacting with a local environment over time. The environment evolves in discrete time slots, and the decision-making of each agent is conditioned solely on local observations. The proposed distributed RL algorithm consists of six components: a) agent, b) state, c) policy, d) critic, e) action, and f) reward, which are specified as follows:

- **Agent:** The agents that perform the proposed RL algorithm are the distributed CPN routers.
- **State:** The state, defined as $\boldsymbol{\nu}_{p,t}$, consists of: 1) the routers' instantaneous value difference $e_{p,q,t}$ between the current computing power information calculated by CPM and the last notified value for each target router q, 2) the AoI $\Phi_{p,q,t}$, representing the time since the last notification from router p to q, 3) the previous action bit $b_{p,q,t-1}$, indicating whether router p chose to notify q in the last time step. Thus, the full local observation is given by

$$\boldsymbol{\nu}_{p,t} = \{e_{p,q,t}, \Phi_{p,q,t},\, b_{p,q,t-1}\}_{q \in \mathcal{M}_p}. \tag{17}$$

- **Policy:** The policy network is designed to generate probabilities of choosing actions based on the local state $\boldsymbol{\nu}_{p,t}$. The policy is parameterized by θ_t, and the mapping is formally expressed as:

$$\pi_{\boldsymbol{\theta}_t}\left(\boldsymbol{\nu}_{p,t}, \boldsymbol{\mu}_{p,t}\right) = \Pr\left(\boldsymbol{\mu}_{p,t} \mid \boldsymbol{\nu}_{p,t}\right), \tag{18}$$

  where the policy is optimized to balance freshness, accuracy, and communication overhead over time.
- **Critic:** To guarantee the robustness of the training process, we adopt a centralized critic architecture, denoted by $V_{\boldsymbol{\varphi}}(\boldsymbol{\nu}_{p,t})$, which estimates the state-value function under the joint observation state $\boldsymbol{\nu}_{p,t} = [\boldsymbol{\nu}_{1,t}, ..., \boldsymbol{\nu}_{R,t}]$.
- **Action:** At each time slot t , router p selects an action vector $\boldsymbol{b}_{p,t} = [b_{p,1,t}, b_{p,2,t}, \ldots, b_{p,R,t}]$. The policy network outputs a continuous vector $\boldsymbol{\mu}_{p,t}$ based on the local observation $\boldsymbol{\nu}_{p,t}$, and each element is mapped to a discrete decision $b_{p,q,t}$. The action $\boldsymbol{b}_{p,t}$ determines how computing power information is disseminated across the network, jointly affecting the estimation errors $e_{p,q,t}$ and the AoI dynamics $\Phi_{p,q,t}$, and ultimately balancing information accuracy and communication cost in dynamic CPN environments.
- **Reward:** The reward determined by actions $\boldsymbol{\mu}_{p,t}$ based on local observation $\boldsymbol{\nu}_{p,t}$ is formulated as

$$\begin{aligned} r\left(\boldsymbol{\mu}_{p,t}|\boldsymbol{\nu}_{p,t}\right) = &-\omega_1 \cdot \tanh\left(\frac{1}{\xi \cdot \sum_{p \in \mathcal{R}} \left\|\boldsymbol{D}_{p,t}\right\|_1} \sum_{p \neq q} \Phi_{p,q,t}\right) \\ &- \omega_2 \cdot \bar{e}_t - \omega_3 \cdot \bar{O}_t, \end{aligned} \tag{19}$$

  where ξ is the tolerance coefficient for AoI, and $\omega_1, \omega_2, \omega_3$ are tunable weights that reflect the importance of freshness, accuracy, and efficiency.

We use MAPPO to optimize the proposed Markov decision process model. It maintains a centralized critic to guide policy updates while each agent possesses its own actor network for independent execution. All agents share a centralized critic network $V_{\boldsymbol{\varphi}}(\boldsymbol{\nu}_{p,t})$, which estimates the state value based on the historical interactions. During each training episode, all agents interact with the environment in parallel and store their interaction histories into a centralized replay buffer. Each entry includes the local state, action, reward, estimated value, and next state. Once a sufficient number of episodes are collected, the buffer contents are used to compute advantage estimates and update the policy networks. The buffer will be cleared after each update round to ensure on-policy training, making sure that all training samples are generated exclusively by the current policy version.

To reduce the variance of policy gradients and stabilize training, we use Generalized Advantage Estimation (GAE) to compute the advantage of each sampled action. For each agent p at time t, the temporal-difference (TD) error is first computed as:

$$\psi_{p,t} = r\left(\boldsymbol{\mu}_{p,t}|\boldsymbol{\nu}_{p,t}\right) + \gamma V_{\boldsymbol{\varphi}}(\boldsymbol{\nu}_{p,t+1}) - V_{\boldsymbol{\varphi}}(\boldsymbol{\nu}_{p,t}), \tag{20}$$

where $r\left(\boldsymbol{\mu}_{p,t}|\boldsymbol{\nu}_{p,t}\right)$ represents the immediate reward, γ is the discount factor for future rewards, and $V_{\boldsymbol{\varphi}}(\boldsymbol{\nu}_{p,t})$ and $V_{\boldsymbol{\varphi}}(\boldsymbol{\nu}_{p,t+1})$ denote the estimated state value and the next-state value, respectively. The value function is parameterized by the critic with parameters $\boldsymbol{\varphi}$. Then, GAE accumulates the TD errors over multiple future steps with an exponential weighting, forming the advantage estimate $\hat{A}_{p,t}$:

$$\hat{A}_{p,t} = \sum_{l=0}^{T-t-1} (\gamma\lambda)^l \psi_{p,t+l}, \tag{21}$$

where λ is the bias-variance trade-off parameter, controlling the weight of longer-term versus short-term information. $\hat{A}_{p,t}$ represents the advantage of taking action $\boldsymbol{\mu}_{p,t}$ in state $\boldsymbol{\nu}_{p,t}$, which means how much better it is than average.

Based on the advantage estimates, the actor and critic networks are jointly optimized using the clipped-PPO surrogate to ensure stable and coordinated policy updates. Specifically, we form the probability ratio as

$$r_{p,t}(\theta_t) \;=\; \frac{\pi_{\theta_t}(\boldsymbol{\mu}_{p,t} \mid \boldsymbol{\nu}_{p,t})}{\pi_{\theta_t}^{\text{old}}(\boldsymbol{\mu}_{p,t} \mid \boldsymbol{\nu}_{p,t})}. \tag{22}$$

θ_t updates by minimizing the clipped surrogate augmented:

$$\mathcal{L}_{\text{actor}}(\theta_t) = -\mathbb{E}_t\Big[\min\Big(r_{p,t}(\theta_t)\,\hat{A}_{p,t}, \\ \text{clip}\big(r_{p,t}(\theta_t), 1-\epsilon, 1+\epsilon\big)\,\hat{A}_{p,t}\Big)\Big] - c_{\text{en}}\,\mathcal{H}\big(\pi_{\theta_t}(\cdot \mid \boldsymbol{\nu}_{p,t})\big), \tag{23}$$

where $\epsilon > 0$ is the clipping parameter, $\mathcal{H}(\pi_{\theta_t})$ denotes the policy entropy, which can be calculated by

$$\mathcal{H}(\pi_{\theta_t}(\cdot \mid \boldsymbol{\nu}_{p,t})) = \mathbb{E}_{\boldsymbol{\mu}\sim\pi_{\theta_t}}[-\log \pi_{\theta_t}(\boldsymbol{\mu}_{p,t} \mid \boldsymbol{\nu}_{p,t})]. \tag{24}$$

$c_{\text{en}} > 0$ is the entropy coefficient. The clipping operation constraints the policy ratio to $[1-\epsilon, 1+\epsilon]$, preventing excessively large policy updates and improving training stability.

The critic is updated by minimizing the mean squared error between the predicted value and the return:

$$\mathcal{L}_{\text{critic}}(\boldsymbol{\varphi}) = \mathbb{E}_t\Big[\big(y_{p,t} - V_{\boldsymbol{\varphi}}(\boldsymbol{\nu}_{p,t})\big)^2\Big], \tag{25}$$

where $y_{p,t} = r(\boldsymbol{\mu}_{p,t} \mid \boldsymbol{\nu}_{p,t}) + \gamma\, V_{\boldsymbol{\varphi}}(\boldsymbol{\nu}_{p,t+1})$ is the target value. The training is performed in mini-batches, where the actor and critic networks are iteratively updated until the policy converges.

After training, the centralized critic will be discarded. Each agent independently executes its actor network π_{θ_t} based on its local observation $\boldsymbol{\nu}_{p,t}$, without RL system communication with other agents, so that the system can make real-time notification decisions in large-scale distributed systems.

IV. Simulation Results and Analysis

For our simulations, we consider a computing power network with 5 CPN devices and 5 CPN routers. All routers form a full mesh logical connection (i.e. $[\boldsymbol{D}_{p,t}]_q = 1,\ \forall p, q \in \mathcal{R}$). The dynamic computing power resources of devices are based on the CPU utilization dataset from Alibaba [15]. Due to the separate design of the reporting module and the notification module, we perform the testing sequentially. We implement two reporting baselines for comparison:

- **Periodic Reporting:** Devices report their resource information periodically at a fixed interval, regardless of the variation magnitude in CPU usage [11].
- **Threshold-triggered Reporting:** Devices report their resource information when the relative change of CPU utilization series compared to the last reported value exceeds a preset threshold [11].

TABLE I: Comparative Sampling and Reconstruction Performance

Reporting Strategy	Avg Reporting Cost	Cost Increase (% v.s. Gradient)	Avg MSE	MSE Increase (% v.s. Gradient)
Gradient-Adaptive	60.2	0.0%	10.3369	0.0%
Threshold-Triggered	75.2	+24.9%	14.9037	+44.2%
Periodic Reporting	75.7	+25.7%	15.1896	+47.0%

We calculate the mean squared error (MSE) between the reconstructed resource data sampled from a cubic spline interpolation sequence and the true data.

In Fig. 2, we show the reconstruction results of different reporting strategies. From Fig. 2, we can observe that the proposed gradient-adaptive strategy achieves 20 % reduction in sampling and an up to 31.9% improvement in reconstruction MSE compared to the periodic reporting and threshold-triggered reporting. This is due to the fact that the compared baselines incur redundant sampling with limited accuracy gains, while the proposed method can effectively reduce MSE due to its gradient-aware design that captures both the magnitude and rate of variations. We also present the average sampling counts and reconstruction MSE of different reporting strategies across the data sets in Table I.

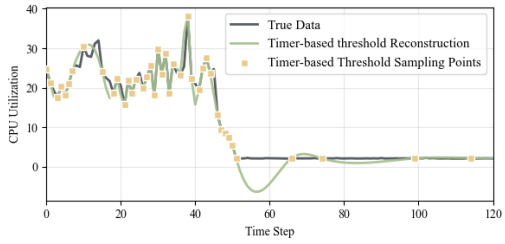
For the simulations of the notification module, we implement four notification methods for comparison:

- **Periodic Notification:** Each CPN router broadcasts its updated $\boldsymbol{c}_{p,t}$ to all available neighbors at fixed intervals.
- **Threshold-triggered Notification:** Each router transmits its updated value only when the change exceeds a predefined threshold.
- **Random Notification:** In each time slot, routers transmit their updates to neighbors with a fixed probability.
- **Multi-agent advantage actor-critic (MAA2C):** A multi-agent value-based RL method that employs a centralized critic and decentralized actors [16].

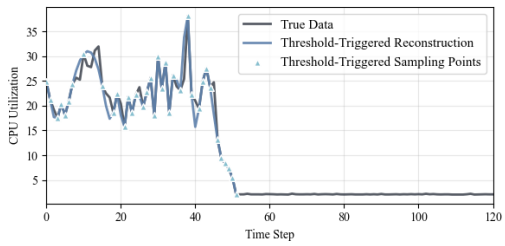
In Fig.3, we show how the convergence trend of the five considered algorithms changes as the number of iterations varies. From Fig.3, we can observe that the proposed method can achieve 0.51% higher reward and faster convergence compared with MAA2C. This is due to proposed method's ability to find more adaptive policies for dynamic environments. From Fig.3, we can also observe that proposed method demonstrates faster convergence speed and superior stability compared to MAA2C. This is due to the reward clipping mechanism of proposed method can stabilize agent training and prevent unstable reward fluctuations caused by environmental volatility. Meanwhile, in Fig.3, we observe that proposed method maintains optimized AoI and error metrics simultaneously, unlike other algorithms that only excel in minimizing either AoI or error.

V. Conclusion

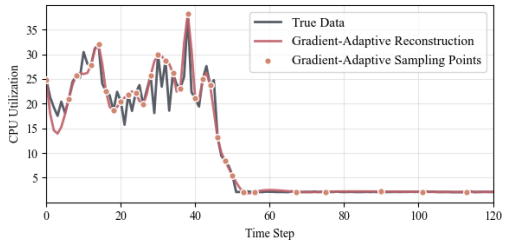
In this paper, we propose a novel distributed framework to optimize information synchronization in CPN. Our hybrid



(a) True Data vs Periodic Reporting Method



(b) True Data vs Threshold-Triggered Method



(c) True Data vs Proposed Method

Fig. 2: Illustration of the convergence trend with different baseline.

approach features a gradient-adaptive reporting strategy for devices' computing resources information reporting, which significantly reduces sampling overhead (by 20%) and reconstruction error (by 31.9%) by dynamically adjusting updates based on resource volatility. Concurrently, we developed a MAPPO-based MARL policy for router-to-router notification, which effectively balances information freshness, accuracy, and communication costs. Simulation results confirmed that our MARL approach achieves superior convergence, stability, and scalability compared to MAA2C and other baselines. The proposed framework provides an effective and scalable solution for maintaining network-wide state awareness in a dynamic, resource-constrained CPN.

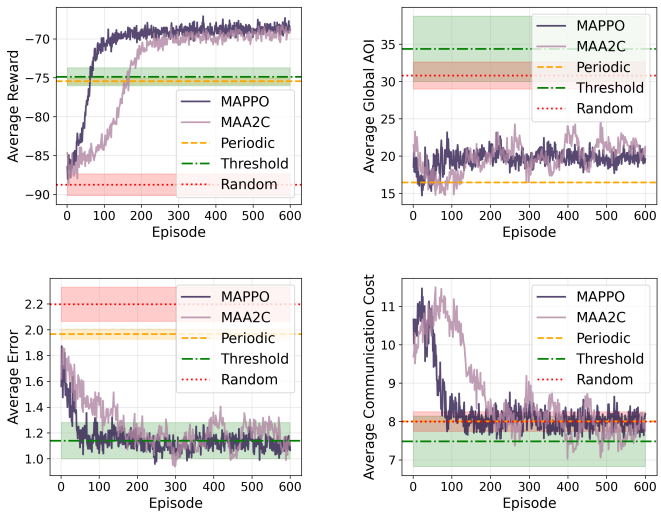


Fig. 3: Comparison of different notification strategy.

REFERENCES

[1] X. Tang, C. Cao, Y. Wang, S. Zhang, Y. Liu, M. Li, and T. He, "Computing Power Network: The Architecture of Convergence of Computing and Networking Towards 6G Requirement," *China Communications*, vol. 18, no. 2, pp. 175–185, Feb. 2021.

[2] G. Cui, Q. He, B. Li, X. Xia, F. Chen, H. Jin, Y. Xiang, and Y. Yang, "Efficient Verification of Edge Data Integrity in Edge Computing Environment," *IEEE Transactions on Services Computing*, vol. 15, no. 6, pp. 3233–3244, Dec. 2021.

[3] W. Sun, Z. Li, Q. Wang, and Y. Zhang, "FedTAR: Task and Resource-Aware Federated Learning for Wireless Computing Power Networks," *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 4257–4270, May. 2022.

[4] Q. Jia, Y. Hu, H. Zhang, K. L. Peng, P. P. Chen, R. Xie, and T. Huang, "Research on Deterministic Computing Power Network," *Journal on Communications*, vol. 43, no. 10, pp. 55–64, Oct. 2022.

[5] X. Gong, C. Bai, S. Ren, J. Wang, and C. Wang, "A Survey of Compute First Networking," in *Proc. IEEE 23rd International Conference on Communication Technology (ICCT)*, China, Nov. 2023, pp. 688–695, IEEE.

[6] J. Zhao, T. Wang, H. Tong, N. Yang, C. Yin, and D. Niyato, "A Joint Flow Scheduling Scheme for the Uplink of 5G-TSN in Industrial Internet of Things Systems," *IEEE Transactions on Cognitive Communications and Networking*, June 2025, Early access.

[7] Y. Liang, G. Xiao, Z. Hua, W. Xing, Y. He, and X. Cheng, "Survey of Computing Power Network Industry Standards," in *Proc. China Automation Congress (CAC)*, China, Oct. 2023, pp. 5583–5588, IEEE.

[8] E. Song, T. Pan, C. Jia, W. Cao, J. Zhang, T. Huang, and Y. Liu, "INT-Label: Lightweight In-Band Network-Wide Telemetry via Interval-Based Distributed Labelling," in *Proc. IEEE INFOCOM*, Vancouver, Canada, May. 2021.

[9] X. Liu, S. Zhou, J. Peng, W. Zhang, D. Tang, and K. Li, "Stopping Criteria for Distributed Data Storage in Compressive CrowdSensing Systems," *IEEE Internet of Things Journal*, vol. 11, no. 7, pp. 11767–11778, July. 2024.

[10] H. Tong, M. Chen, J. Zhao, Y. Hu, Z. Yang, Y. Liu, and C. Yin, "Continual Reinforcement Learning for Digital Twin Synchronization Optimization," *IEEE Transactions on Mobile Computing*, vol. 24, no. 8, pp. 6843–6857, Feb. 2025.

[11] Y. Luo, W. Tang, and X. Yang, "Computing Power Resource Sensing and Notification Mechanism Based on Extended IGMP in Computing-Aware Network," in *Proc. International Conference on Computer and Communication System (ICCCS)*, China, Apr. 2025, pp. 1137–1144, IEEE.

[12] K. Poularakis, Q. Qin, L. Ma, S. Kompella, K. K. Leung, and L. Tassiulas, "Learning the Optimal Synchronization Rates in Distributed SDN Control Architectures," in *Proc. IEEE INFOCOM*, Paris, France, Apr. 2019.

[13] I. Panitsas, A. Mudvari, and L. Tassiulas, "D2Q Synchronizer: Distributed SDN Synchronization for Time Sensitive Applications," in *Proc. IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*, Athens, Greece, May. 2025, pp. 1–6, IEEE.

[14] N. Yang, S. Wang, M. Chen, C. Yin, and C. G. Brinton, "A Privacy Preserving and Byzantine Robust Collaborative Federated Learning Method Design," in *IEEE International Conference on Communications*, Denver, CO, USA, June 2024.

[15] Alibaba Group, "Alibaba Cluster Trace Program (clusterdata)," https://github.com/alibaba/clusterdata, 2017, GitHub repository (dataset releases 2017–2025). [Online]. Accessed: 19-Oct-2025.

[16] X. Li, G. Chen, G. Wu, Z. Sun, and G. Chen, "Research on Multi-Agent D2D Communication Resource Allocation Algorithm Based on A2C," *Electronics*, vol. 12, no. 2, pp. 360, Feb. 2023.