STPer: Task Scheduling and Traffic Routing with Spatiotemporal Dynamic Perception in Green Computing Power Networks

Xiaoyao Huang*, Remington R. Liu† and Jie Wu*
*Cloud Computing Research Institute, China Telecom, P.R.China
†China Telecom Research Institute, P.R.China
Email: huangxy32@chinatelecom.cn, remington.liu@outlook.com, wujie@chinatelecom.cn

***Abstract*—The growing demands of compute-intensive applications have led to the emergence of Computing Power Networks (CPNs), which integrate diverse computing resources across cloud, edge, and devices for efficient task scheduling and resource allocation in a network environment. This paper addresses the critical challenge posed by the dual spatiotemporal dynamics of tasks and resources, which complicates the optimization process of task scheduling in CPNs. Our work is the first to systematically investigate the joint optimization of task scheduling and traffic routing in green CPNs involving the dual dynamics. We formulate the profit maximization problem as an integer nonlinear programming problem that is NP-hard. To tackle this issue, we propose a novel deep reinforcement learning based algorithm capable of SpatioTemporal Dynamic Perception (STPer), which employs Graph Neural Networks(GNN) and Long Short-Term Memory(LSTM) networks. The STPer algorithm effectively captures the spatiotemporal dynamics of both resources and tasks, maximizing platform profits. Extensive simulations demonstrate that STPer significantly outperforms existing benchmark algorithms, highlighting its superior performance in optimizing resource utilization and enhancing profitability in the green CPN.**

I. Introduction

Driven by the growing demands of compute-intensive applications such as large-scale model and video rendering [1], [2], the need for massive computing power has become a critical challenge. To address the increasing demands and performance requirements, the concept of the Computing Power Network (CPN) has been proposed as a key technology for the next generation of networks [3], [4]. CPN aims to integrate and orchestrate diverse computing resources across the cloud, edge, and devices, enabling efficient, on-demand scheduling and allocation of computing power.

Energy consumption has become a critical concern in CPNs. To address this issue, energy-saving resources, such as hydropower, have been integrated into CPNs (referred to as green CPNs) which are cheaper and eco-friendly. Computing service platforms in the CPN manage multiple resource nodes and serve various kinds of users' tasks with different Quality of Service (QoS) requirements by scheduling the tasks to the computing nodes for processing. Different tasks result in varying levels of satisfaction upon completion, leading to differences in task charges [5], [6]. Moreover, computing nodes have diverse processing capabilities and resource costs, influenced by factors such as energy supply types and node performance. The profit of the platform consists of the charges from the users for task processing and the cost of resource operating. In this paper, we study the problem of joint optimization of task scheduling and traffic routing in green CPNs to maximize the platform profit.

We formulate the platform profit maximization problem as an integer nonlinear programming (INLP) problem which is NP-hard. However, applying traditional INLP methods to our problem becomes impractical due to the following challenges: 1) *High Dimensionality*: The dual spatiotemporal dynamics of user tasks and resources introduce a massive number of variables and constraints, making the solution space combinatorially large. 2) *Dynamic Dependencies*: The resource availability and user demands exhibit complex temporal and spatial dependencies that cannot be easily captured by static INLP solutions. 3) *Real-Time Requirements*: CPNs require near-instantaneous decision-making to maintain performance and efficiency, which is beyond the computational feasibility of solving NP-hard INLP problems iteratively at runtime. To tackle the above challenges, we propose a GNN and LSTM-based Reinforcement Learning (RL) algorithm for task scheduling and traffic routing with SpatioTemporal Dynamic Perception (STPer). The combination of GNN, LSTM, and RL allows our system to overcome the limitations of traditional INLP by scalability and dynamic adaptation.

II. Related Work

Task Scheduling: The task scheduling problem is a hot research topic in CPNs, aiming to meet certain performance requirements and system benefits through the optimization of task scheduling and resource allocation. In [7], the authors proposed a service intent-aware task scheduling framework for CPNs, leveraging intent-based networking principles to enhance task scheduling performance by accounting for application service intent and optimizing the integration of resource orchestration and network control. In [8], the authors addressed task scheduling for edge inference in CPNs to minimize long-term costs and proposed adaptive online algorithms with competitive ratio guarantees to handle stochastic inputs and dynamic resource provisioning. In [9], the authors proposed a scheduling scheme for ring all-reduce-based distributed model training requests that optimizes the allocation

of computing and wavelength resources while ensuring reliability and improving training efficiency.

RL in CPN: In dynamically changing environments, reinforcement learning can adapt to environmental changes and adjust strategies to achieve long-term goals. Therefore, in CPNs, part of the research works employ reinforcement learning for task scheduling and resource allocation. In [10], the authors introduced an optimization method based on the multi-agent soft actor-critic algorithm within a software-defined networking framework, aimed at enhancing data transmission efficiency within the CPN by leveraging collaboration among multiple intelligent agents and employing deep RL to optimize routing paths. In [11], the authors proposed a multi-dimensional resource matching algorithm based on deep RL, framing the resource matching process as a Markov decision process to optimize the allocation of tasks to nodes in CPNs.

In conclusion, current studies on task scheduling optimization and RL have certain limitations. From the perspective of scheduling, existing works only consider partial system characteristics, lacking a holistic approach to address the spatiotemporal dynamics and the joint optimization of task scheduling and traffic routing. From the RL perspective, the challenge of dual spatiotemporal dynamics remains unresolved, which undermines the effectiveness of RL in practical task scheduling scenarios. To address these gaps, an innovative RL framework is needed to tackle the joint optimization problem in CPNs with dual spatiotemporal dynamics.

III. System Model and Problem Formulation

A. Network Model

Fig. 1 shows the green CPN system under study in this paper, which consists of a computing service platform, heterogeneous computing nodes with different capabilities and users with different QoS requirement geographically distributed. Users subscribe computing services from the platform on a paid basis and offload tasks to the servers according to the platform's scheduling decisions. The computing nodes include cloud computing nodes and edge computing nodes, some of which are located near hydropower stations and can be powered by a mix of hydropower and grid electricity. Both user tasks and computing resources exhibit spatiotemporal dynamics: user tasks vary with social effects such as work and rest patterns, while the cost and utilization density of computing resources change with factors like green energy supply and user consumption habits. Moreover, these dual spatiotemporal dynamics are coupled to some extent. The platform works to schedule the tasks to computing nodes and route the task in the network to maximize its own profit by improving the QoS of all tasks and reducing system cost.

The green CPN is modeled as an undirected graph $G = (\mathcal{E}, \mathcal{N})$, where $\mathcal{E}$ represents link set and $\mathcal{N} = N \cup U$ represents the total set of computing node N and end user U. We use f_n to denote the computing power of node $n \in \mathcal{N}$ and b_e to denote the bandwidth capacity of $e \in \mathcal{E}$. The system operates in a time-slotted manner. At each time slot t, the system works to schedule the tasks generated from each end user $u \in U$ to the computing nodes $n \in N$. A task generated by user u can be represented by a tuple $\xi_u(t) \triangleq \langle m_u(t), r_u(t), l_u(t) \rangle$ where $m_u(t)$ is the workload, $r_u(t)$ and $l_u(t)$ are the packet size in bytes and maximum latency, respectively. Accordingly, the task set at time slot t can be denoted by $\xi(t) = \{\xi_1(t), \xi_2(t), \ldots, \xi_{|u|}(t)\}$.

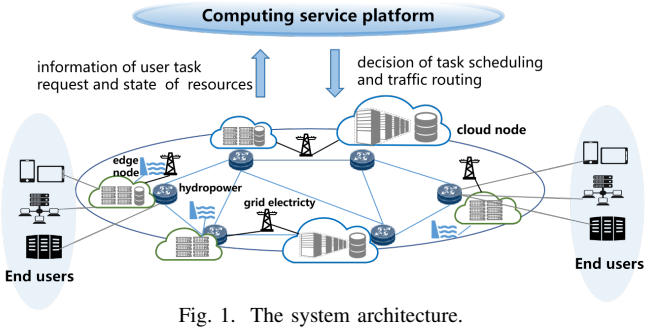

Fig. 1. The system architecture.

B. Delay Model

The delay experienced by a task $\xi_u(t)$ consists of three parts: transmission delay, traffic dispatching offset, and computation delay. Let $\mathcal{E}_{\xi_u(t)}$ and $N_{\xi_u(t)}$ represent the edges and nodes that the task $\xi_u(t)$ passes through during transmission, respectively, then the transmission delay is

$$t_u^{tr}(t) = \sum_{e \in \mathcal{E}_{\xi_u(t)}} \frac{r_u(t)}{b_e} + \sum_{n \in N_{\xi_u(t)}} t_n^{sw}, \tag{1}$$

where t_n^{sw} is the forwarding delay of node n. The traffic dispatching offset refers to the waiting time incurred when a task $\xi_u(t)$ is delayed to avoid conflicts with other tasks that are already utilizing the same transmission edge. We use $t_u^{off}(t)$ to represent the traffic dispatching offset, which will be detailed in the next subsection D. The computation delay can be calculated as $t_{u,n}^{comp}(t) = \frac{m_u(t)}{f_n}$.

Therefore, the total delay experienced by the task $\xi_u(t)$ is

$$t_u^{sum}(t) = t_u^{tr}(t) + t_u^{off}(t) + t_{u,n}^{comp}(t). \tag{2}$$

C. System Profit Model

System profit can be defined as the revenue from completing the task minus the cost of computing services [12]. Specifically, if the delay of task $\xi_u(t)$ is $t_u^{sum}(t)$, then the revenue $R_u(t)$ of completing the task $\xi_u(t)$ is

$$R_u(t) = p\left(1 + \frac{\eta}{t_u^{sum}(t)}\right) m_u(t), \tag{3}$$

where p is the unit price for computing resource and η is a coefficient. According to Ref. [13], the cost of computing for task $\xi_u(t)$ can be calculated as

$$E_{u,n}(t) = i_e(t) m_u(t) f_n^2 + i_t \frac{m_u(t)}{f_n}, \tag{4}$$

where $i_e(t)$ is the electricity price. The first item on the right hand side represents the computing energy consumption, and the second item represents the computing power usage cost. We consider a more realistic situation, that is, some computing

nodes are near green energy supply (such as hydropower considered in this article), so when calculating the computing cost, the calculation formula can be supplemented as

$$E_{u,n}(t) = i_e(t)\alpha_n(t)\, m_u(t) f_n^2 + i_t \frac{m_u(t)}{f_n}, \tag{5}$$

where $\alpha_n(t)$ is the price discount of node n due to the supply of hydropower in time frame t of time cycle T. If $\alpha_n(t) = 1$, it means there is no hydropower supply, while $\alpha_n^i(t) = 0$, it means that the power supply is entirely hydropower.

D. Problem Formulation

Let $y_{u,n}(t) \in \{0,1\}$ indicate whether task $\xi_u(t)$ is scheduled to the node $n \in N$, where $y_{u,n}(t) = 1$ means the node n is chosen as the destination node. Each task can only be scheduled to one node, thus we have $\sum_{n\in N} y_{u,n} = 1, \forall u \in U$.

The chosen node for task $\xi_u(t)$ is denoted by $n_u^*(t) = \sum_{i\in N} y_{u,i}(t) i$. Let $x_{\{i,j\},u}(t) \in \{0,1\}$ denote whether task $\xi_u(t)$ uses the edge $\{i,j\} \in \mathcal{E}$, where $x_{\{i,j\},u}(t) = 1$ means the edge $\{i,j\}$ is used. Since the task $\xi_u(t)$ is generated by node u and ended by node $n_u^*(t)$, we have

$$\sum_{i\in N,\{u,i\}\in\mathcal{E}} x_{\{u,i\},u}(t) - \sum_{i\in N,\{i,u\}\in\mathcal{E}} x_{\{i,u\},u}(t) = 1. \tag{6}$$

$$\sum_{i\in N,\{n_u^*(t),i\}\in\mathcal{E}} x_{\{n_u^*(t),i\},u}(t) - \sum_{i\in N,\{i,n_u^*(t)\}\in\mathcal{E}} x_{\{i,n_u^*(t)\},u}(t) = 1. \tag{7}$$

For all intermediate nodes forwarding data, we have

$$\sum_{j\in N,\{i,j\}\in\mathcal{E}} x_{\{i,j\},u}(t) - \sum_{j\in N,\{j,i\}\in\mathcal{E}} x_{\{j,i\},u}(t) = 0, \tag{8}$$

where $\forall u \in U, \forall i \in N \backslash \{n, n_u^*(t)\}$. To avoid loops, we have

$$\sum_{j\in N,\{i,j\}\in\mathcal{E}} x_{\{i,j\},u}(t) \leqslant 1, \forall u \in U, \forall i \in N. \tag{9}$$

Let $b_{\{i,j\},u}(t) \in \{x | x \in \mathbb{Z}, 0 \leqslant x \leqslant \Delta t - 1\}$ denote the beginning time slot of task $\xi_u(t)$ on edge $\{i,j\}$ and $o_{\{i,j\},u}(t) \in \mathbb{Z}^+$ state an offset of a time slot as a number of full length of time cycle Δt. Thus, a time slot position for task $\xi_u(t)$ on edge $\{i,j\}$ is given by $b_{\{i,j\},u}(t) + o_{\{i,j\},u}(t)\Delta t$. The offset variable is used when $b_{\{i,j\},u}(t)$ is very close to Δt, and an offset variable is needed to schedule this task to the next time cycle for transmission. First, for paths that are not selected as routes, variables $b_{\{i,j\},u}(t)$ and $o_{\{i,j\},u}(t)$ should be 0. So we have

$$b_{\{i,j\},u}(t) + o_{\{i,j\},u}(t) \leqslant M_s \cdot x_{\{i,j\},u}(t), \tag{10}$$

where M_s is an arbitrary and sufficiently large constant ($M_s \gg b_{\{i,j\},u}(t) + o_{\{i,j\},u}(t)$). Furthermore, the time slot on the outgoing link must be scheduled later than the time slot on the incoming link by at least the switching delay t_n^{sw} of node n. So for $\forall u \in U, \forall i \in N \backslash \{n_u^*(t)\}$ we have

$$\sum_{j\in N,\{i,j\}\in\mathcal{E}} \left(b_{\{i,j\},u}(t) + o_{\{i,j\},u}(t)\Delta t\right) - \tag{11}$$
$$\sum_{j\in N,\{j,i\}\in\mathcal{E}} \left(b_{\{j,i\},u}(t) + o_{\{j,i\},u}(t)\Delta t\right) \geqslant t_n^{sw} \sum_{j\in N,\{i,j\}\in\mathcal{E}} x_{\{i,j\},u}(t),$$

As mentioned above, tasks routed on the same link must not overlap. Therefore, for two tasks scheduled on the same link at different time slots, k_a and k_b, one task must end before the other starts. Let $\alpha_{i,j,p,q,w,v}(t) \in \{0,1\}$ be an auxiliary variable, where $\forall (w,v) \in \{0,1\} \times \{0,1\}$. For $\forall (p,q) \in \mathcal{N} \times \mathcal{N}, \forall \{i,j\} \in \mathcal{E}$, we have

$$\begin{aligned}
&\left(b_{\{i,j\},q}(t) + v\Delta t\right) - \left(b_{\{i,j\},p}(t) + w\Delta t\right) \\
&\geqslant \frac{r_p(t)}{b_{\{i,j\}}} - M_r\left(3 - \alpha_{i,j,p,q,w,v}(t) - x_{\{i,j\},p}(t) - x_{\{i,j\},q}(t)\right), \quad (12)\\
&\left(b_{\{i,j\},p}(t) + w\Delta t\right) - \left(b_{\{i,j\},q}(t) + v\Delta t\right) \\
&\geqslant \frac{r_q(t)}{b_{\{i,j\}}} - M_r\left(2 + \alpha_{i,j,p,q,w,v}(t) - x_{\{i,j\},p}(t) - x_{\{i,j\},q}(t)\right), \quad (13)
\end{aligned}$$

where M_r is an arbitrary and sufficiently large constant.

For $\xi_u(t) \in \xi_{|U|}(t)$, the delay requirement should be satisfied, we have

$$\begin{aligned}
&\sum_{j\in N,\{j,n_u^*(t)\}\in\mathcal{E}} \left(b_{\{j,n_u^*(t)\},u}(t) + o_{\{j,n_u^*(t)\},u}(t)\Delta t\right) - \\
&\qquad \sum_{j\in N,\{u,j\}\in\mathcal{E}} \left(b_{\{u,j\},u}(t) + o_{\{u,j\},u}(t)\Delta t\right) \\
&\leqslant l_u(t) - t_{u,n_u^*}^{comp}(t) - \sum_{j\in N,\{j,n_u^*(t)\}\in\mathcal{E}} \frac{r_u(t)}{b_{\{j,n_u^*(t)\}}} x_{\{j,n_u^*(t)\},u}(t), \quad (14)
\end{aligned}$$

where the transmission delay plus the computation delay should be less than the maximum delay of the task.

Finally, the optimization objective for our problem is to maximize the system benefit, that is, to maximize the sum of the task offloading benefits minus the task offloading costs in all time cycles. It can be formally expressed as

$$\begin{aligned}
&\max \sum_t \sum_{n\in\mathcal{N}} \sum_{u\in U} y_{u,n}(t)(R_u(t) - E_{u,n}(t)). \quad (15)\\
&\text{s.t. Eqs. } (6) - (14)
\end{aligned}$$

Since the decision variables of this problem contain integers and the variables are nonlinear with respect to the objective function, this is an integer nonlinear programming problem, which is an NP-hard problem and difficult to solve. The problem is further made more difficult by the dynamic nature of the environment including the spatiotemporal dynamics of both user tasks and heterogeneous resources, which is beyond the ability of the traditional optimization algorithms. To effectively tackle the problem, we propose a novel deep reinforcement learning based algorithm capable of STPer, which employs GNN and LSTM networks.

IV. Proposed Framework

In this section, we propose the STPer algorithm to maximize the platform profit by jointly optimizing the decision of task scheduling and traffic routing. The overview of the STPer is shown in Fig. 2. To address the dual spatiotemporal dynamics, we design a perception network based on GNN and LSTM to extract spatiotemporal features as input for the reinforcement learning framework. Additionally, we develop an iterative optimization algorithm to solve the traffic routing optimization problem in the form of cumulative fractional expressions.

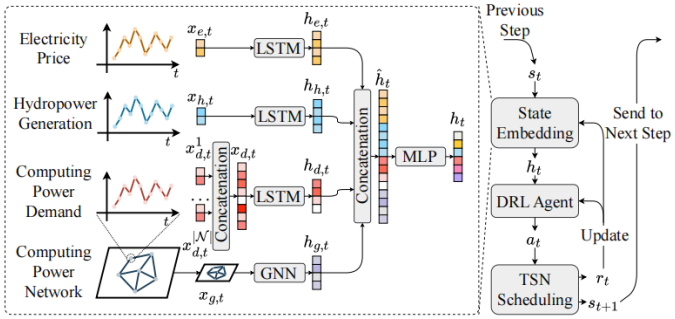

Fig. 2. Overview of the proposed STPer.

A. Modeling of Markov Decision Process

Before introducing the details of deep reinforcement learning, it is necessary to model the problem using a Markov Decision Process (MDP). The MDP can be represented by a 5-tuple $\langle \mathcal{A}, \mathcal{S}, R, P, \gamma \rangle$, where $\mathcal{A}$ is the action space. $\mathcal{S}$ is the state space. R is the reward function. P is the state transition probability function and γ is the discount factor.

1) Action Space $\mathcal{A}$: At time frame t, The agent needs to assign each task $\xi_u(t)$ generated by node n to its destination $n_u^*(t)$ according to the observation. So, the action space can be defined as $\mathcal{A} \triangleq \times_{u \in U}\{x | x \in \mathbb{Z}, 1 \leq x \leq |N|\}$.

2) State Space $\mathcal{S}$: As mentioned before, electricity prices, hydropower generation, computing power requirements of each node and their spatial distribution will play a decisive role in decision making, so, the state space can be defined as $\mathcal{S} \triangleq \{i_e(t), \alpha_n(t), m_n(t), n \in \mathcal{N}, t \in T\}$.

3) Reward Function $R(s_t, a_t)$: The reward function is similar to Eq. (15), which is the benefit of each time frame minus the cost, so that the agent is updated in the direction of maximizing the objective function, that is, $R(s_t, a_t) = \sum_{x \in \mathcal{A}} \sum_{u \in U} (R_u(t) - E_{u,x}(t))$.

4) State Transfer Function $P(s_{t+1}|s_t, a_t)$: Since we are studying a model-free reinforcement learning environment, there is typically no explicit state transition function.

5) Discount Factor γ: A lower γ prioritizes short-term rewards, while a higher γ emphasizes the long-term rewards. Here, $\gamma = 0.99$ is adpoted.

B. Feature Extraction Network

1) Spatial Feature Extraction with GNN: To capture the spatial correlation features of task demands at each node in the CPN, a GCN with a graph learning module is proposed to extract these features. First, the adjacency matrix of the graph G is denoted as A, and the task demand of node n is mapped to a vector x_n od length d using an MLP. The features of these n nodes are then aggregated into an $n \times d$ matrix $H^{(0)}$. The graph convolution operation at one layer is given by

$$H^{(l+1)} = \sigma(D^{-\frac{1}{2}} A D^{-\frac{1}{2}} H^{(l)} W), \tag{16}$$

where $\sigma(\cdot)$ is the activation function such as $\mathrm{ReLU}(\cdot)$, D is the degree matrix of A, W represents the trainable parameter matrix of the current layer convolution transformation.

Here, we input the computing power demands of each node into the GNN to obtain the spatial correlation features of the computing power demand.

$$h_g(t) = \mathrm{GNN}(m_1(t), \cdots, m_{|\mathcal{N}|}(t), A). \tag{17}$$

2) Temporal Feature Extraction with LSTM: Considering that electricity prices, hydropower generation, and computing power demand are directly influenced by recent historical values, this section adopts an LSTM-based information fusion framework to handle their long-term temporal dependencies.

We input the electricity price $i_e(t)$, hydropower generation $\alpha_n(t)$ and user demand $m_u(t)$ of each node in each time cycle into LSTM networks to obtain their hidden representations.

$$h_i(t) = \mathrm{LSTM}(i_e(t), h_i(t-1)), \tag{18}$$

$$h_\alpha(t) = \mathrm{LSTM}(\alpha(t), h_\alpha(t-1)), \tag{19}$$

$$h_r(t) = \mathrm{LSTM}(m_1(t) \parallel \cdots \parallel m_{|\mathcal{N}|}(t), h_r(t-1)), \tag{20}$$

where $\parallel$ is the concatenation operation.

3) Feature Fusion: In this section, we apply a "concatenation" method to fuse the spatial and temporal features, enhancing the model's ability to capture the task demands and electricity market dynamics. Specifically, the spatial feature $h_g(t)$ and the temporal features $h_i(t)$, $h_\alpha(t)$, and $h_r(t)$ are concatenated or aggregated to produce a fused feature representation $h_f(t)$, i.e., $h_f(t) = h_i(t) \parallel h_\alpha(t) \parallel h_r(t) \parallel h_g(t)$. This fusion method allows the model to simultaneously leverage information from both the spatial and temporal dimensions. After the feature fusion, the resulting fused representation is further processed by a MLP to capture higher-level abstract features. Therefore, the final features are given by $h(t) = \mathrm{MLP}(h_f(t))$.

C. Deep Reinforcement Learning

The reinforcement learning framework implemented in STPer is Proximal Policy Optimization (PPO) [14]. The learning process of STPer, based on PPO, operates as follows: First, the current task offloading policy is applied to determine the destination for each node's tasks, and a routing scheduling algorithm is employed to manage task assignments. Subsequently, the advantage function for the task offloading policy is computed. To address sampling bias caused by policy updates, importance sampling is used to adjust the weights during the training process. Finally, a loss function derived from the preceding components is utilized to update the policy. After multiple training iterations, once the offloading policy stabilizes and shows minimal changes, a convergent task offloading policy is obtained. Below, we detail the key functions involved in this process.

The advantage function is defined as $A_\pi(s_t, a_t) = Q_\pi(s_t, a_t) - V_\pi(s_t)$, where $Q_\pi(s_t, a_t) = E_{s_{t+1}, a_{t+1}, \cdots}[\sum_{l=0}^{\infty} \gamma^t r_{t+l}]$ represents the expected profit of action a_t in the state of s_t. $V_\pi(s_t) = E_{a_t, s_{t+1}, \cdots}[\sum_{l=0}^{\infty} \gamma^t r_{t+l}]$ represents the expected reward of being in the state of s_t.

To balance bias and variance, STPer employs Generalized Advantage Estimation (GAE) to compute the advantage function. GAE is calculated as $A_t = \sum_{l=0}^{\infty} (\gamma\phi)^l \kappa_{t+l}^V$, where ϕ is

a parameter that balances bias and variance. κ_{t+l}^{V} is computed using $\kappa_{t}^{V}=r_{t}+\gamma V_{\pi'}\left(s_{t+1}\right)-V_{\pi'}\left(s_{t}\right)$, where r_t represents the reward obtained by the system for the decision made at time t. To correct sampling bias introduced by policy updates, importance sampling weights are used. These weights are defined as $\eta_{\theta}=\frac{\pi_{\theta}\left(a_{t}|s_{t}\right)}{\pi_{\theta'}\left(a_{t}|s_{t}\right)}$, where $\pi_{\theta}\left(a_{t}|s_{t}\right)$ is the updated policy and $\pi_{\theta'}\left(a_{t}|s_{t}\right)$ is the policy before the update.

The PPO loss function is defined as $J_{ppo}\left(\theta\right)=E\left[\min\left(\eta_{\theta}A_{\pi'}\left(s_{t},a_{t}\right),\mathrm{clip}\left(\eta_{\theta},1-\delta,1+\delta\right)A_{\pi'}\left(s_{t},a_{t}\right)\right)\right]$, where $\mathrm{clip}(\cdot)$ restricts η_θ to the range $1-\delta$ and $1+\delta$.

D. Traffic Routing

Since $n_{u}^{*}(t)$ of each task for each node is given in the reinforcement learning in the previous subsection, so the objective function at time cycle t of Eq. (15) can be reformulated as

$$\max\sum_{n\in\mathcal{N}}\sum_{u\in U}p\eta\frac{m_{u}\left(t\right)}{t_{u}^{sum}\left(t\right)}. \tag{21}$$

Note that this is an objective function in the form of a sum fraction, so we first use a quadratic transformation to make it easier to handle. The quadratic transformation of formula $\sum_{n\in\mathcal{N}}p\eta\frac{m_{n}(t)}{t_{u}^{sum}(t)}$ is

$$\max\sum_{n\in\mathcal{N}}\sum_{u\in U}2y_{u,n}\sqrt{p\eta m_{u}\left(t\right)}-y_{u,n}^{2}t_{u}^{sum}\left(t\right), \tag{22}$$

where $\forall y_{u,n}\in\mathbb{R}$ is the auxiliary variable. From Ref. [15], we know that Eq. (21) and Eq. (22) are equivalent. For the solution of the problem Eq. (22), we use iterative optimization to solve the problem. First, we fix $\mathbf{x},\mathbf{b},\mathbf{o},\alpha$, the auxiliary variable $y_{u,n}$ has an optimal closed-form solution $y_{u,n}^{*}=\frac{\sqrt{p\eta m_{u}(t)}}{t_{u}^{sum}(t)},\forall n\in N,u\in U$. Then fix the auxiliary variable $y_{u,n}$ and the problem is an integer linear programming problem about $\mathbf{x},\mathbf{b},\mathbf{o},\alpha$, which can be solved by existing toolboxes such as CVX [16].

V. Performance Evaluation

In this section, we use real-world electricity price [17] and hydropower generation [18] datasets to validate the effectiveness of the proposed STPer by comparing it with other baseline algorithms. To verify the spatiotemporal correlation of electricity prices, hydropower generation, and computing power demand, ablation experiments are conducted to analyze the effectiveness of different feature extraction modules.

A. Simulation Setup

We implemented the STPer framework using Python, with the reinforcement learning environment developed using the Gymnasium library [19]. In the simulation, the number of computing nodes is 10. The computing power f_n of node n is randomly chosen in $[1,10]$ TFLOPs. The bandwidth capacity d_e of link e is randomly chosen in $[10,50]$Mb/s. The packet size $r_u(t)$ of a task $\xi_u(t)$ is 5Mb and the latency requirement is 2s. According to [20], the unit price p is 0.5 and η is 2. i_t is 0.2. The workload of each user is a trigonometric function with time as the independent variable plus a Gaussian random variable. The following baseline algorithms are chosen.

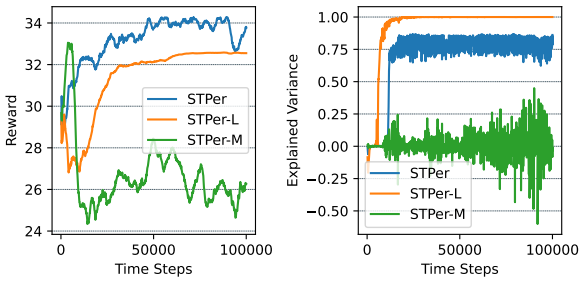

Fig. 3. The cumulative reward and explained variance (EV) of the value function across time steps for each model during training.

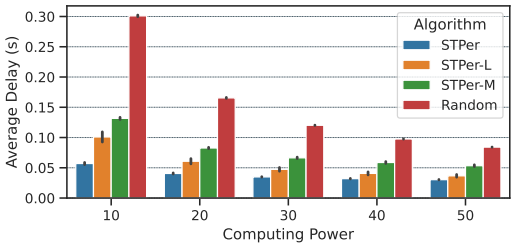

Fig. 4. Impact of computing power in average delay.

- **STPer-L:** Remove the GNN module and only use the time series information of electricity price, hydropower generation and task demand to make decisions.
- **STPer-M:** Remove the GNN and LSTM modules and make decisions without extracting the spatiotemporal correlation between electricity prices, hydropower generation and task demand.
- **Random:** Making decisions by random scheduling and routing without any optimization.

B. Analysis of Reward and Explained Variance

First, we analyze the relationship between the cumulative reward and explained variance (EV) across the training process for each algorithm. The results are as shown in Fig. 3. STPer outperforms the other algorithms in terms of cumulative reward while with a lower explained variance than STPer-L. This suggests that STPer's exploration-exploitation balance is more effective, allowing it to maximize long-term rewards despite not perfectly fitting the value function. In contrast, STPer-L with an explained variance close to 1, appears to overfit the training data, leading to lower generalization and, consequently, suboptimal rewards.

C. Varying the Computing Power of the Nodes

In Fig. 4, the average delay of all algorithms is evaluated with varying computing power ranging from 10 to 50 TFLOPs. It can be seen that as computing power increases, the average task delay decreases significantly. Specifically, at low computing power, delays are relatively high, but as computing power increases, the task delay reduces substantially. This indicates that when the computing capability is limited, the algorithm requires more time to complete tasks, but with increased computational resources, the task processing speed improves, resulting in reduced delays. The performance of different algorithms varies under the same computing power configuration. In the same configuration, STPer performs the

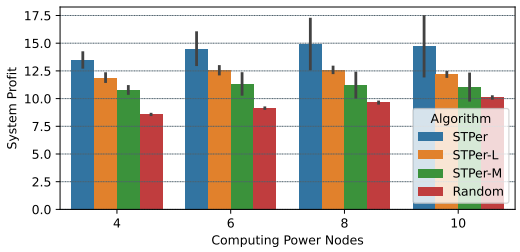

Fig. 5. Impact of computing power in system profit.

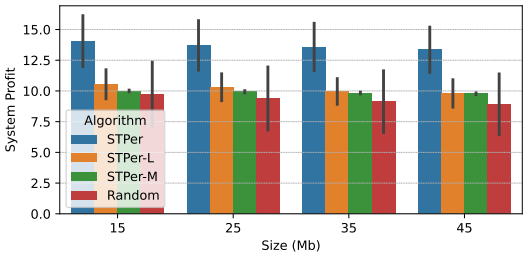

Fig. 6. Impact of data size of tasks.

best, while STPer-M only slightly outperforms the random approach. In Fig. 5, the system profit of all algorithms is further evaluated with varying computing power. It can be observed that all algorithms have the same trend that as the node computing power increases, the system profit of all algorithms also increases. This is because, with the increase in node computing power, task delays decrease, and the resulting benefits outweigh the increase in system costs due to the higher computing power, leading to an overall increase in system profit which is verified in the result trend in Fig. 4 . In the same configuration, STPer achieves the highest system profit, while STPer-M only slightly outperforms the random approach, which demonstrates the efficiency of STPer.

D. Varying the Data Size of the Tasks

The impact of varying data size of the task on system benefit is evaluated with the data size ranging in [15,45] Mb. As shown in Fig. 6, it can be seen that the system benefit resulted from all the algorithms decreases as the data size increases. This is because the increased data size incurs improved transmission delay, which results in the task value declining. Among all the algorithms, STPer achieves the highest system benefit, while Random achieves the lowest with varying data size, which shows the robustness of STPer.

VI. Conclusion

In this paper, we addressed the critical challenges posed by spatiotemporal dynamics in computing power networks. The profit maximization problem was formulated as an NP-hard integer nonlinear programming problem. To tackle this, we proposed a novel deep reinforcement learning algorithm, STPer, which leverages GNN and LSTM networks to effectively capture the spatiotemporal dynamics of both resources and tasks. Additionally, we designed an iterative optimization algorithm to solve the traffic routing problem, expressed as cumulative fractional equations. Extensive experiments demonstrate the superiority of the proposed STPer algorithm compared to baseline methods.

References

[1] E. Kasneci *et al.*, "Chatgpt for good? on opportunities and challenges of large language models for education," *Learning and individual differences*, vol. 103, p. 102274, 2023.

[2] D. Yu, R. Chen, X. Li, M. Xiao, G. Zhang, and Y. Liu, "A gpu-enabled real-time framework for compressing and rendering volumetric videos," *IEEE Transactions on Computers*, 2023.

[3] 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*, pp. 175–185, 2021.

[4] B. Lei, Q. Zhao, and J. Mei, "Computing power network: an interworking architecture of computing and network based on ip extension," in *2021 IEEE 22nd international conference on high performance switching and routing (HPSR)*. IEEE, 2021, pp. 1–6.

[5] Z. Cui, T. Zhao, L. Wu, A. K. Qin, and J. Li, "Multi-objective cloud task scheduling optimization based on evolutionary multi-factor algorithm," *IEEE Transactions on Cloud Computing*, 2023.

[6] Z. J. K. Abadi, N. Mansouri, and M. Khalouie, "Task scheduling in fog environment—challenges, tools & methodologies: A review," *Computer Science Review*, vol. 48, p. 100550, 2023.

[7] Q. Tang, R. Xie, L. Feng, F. R. Yu, T. Chen, R. Zhang, and T. Huang, "Siats: A service intent-aware task scheduling framework for computing power networks," *IEEE Network*, 2023.

[8] M. Ji, Z. Qian, and B. Ye, "When cpn meets ai: Resource provisioning for inference query upon computing power network," in *2023 IEEE 29th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2023, pp. 2261–2268.

[9] L. Chen, Y. Li, C. Natalino, Y. Li, B. Zhang, Y. Fan, W. Wang, Y. Zhao, and J. Zhang, "Reliable and efficient rar-based distributed model training in computing power network," *Journal of Optical Communications and Networking*, vol. 16, no. 5, pp. 527–540, 2024.

[10] Z. Chen, S. Zhang, Y. Tang, M. Xu, X. Wu, M. Ye, Z. Zhang, Y. Qi, H. Lu, and W. Huo, "Optimization of computing power network routing strategies based on multi-agent soft actor-critic," in *2024 9th International Conference on Intelligent Computing and Signal Processing (ICSP)*. IEEE, 2024, pp. 426–430.

[11] L. Wei, J. Li, Y. Gao, L. Shi, H. Lian, G. Cheng, and M. He, "Resource matching algorithm based on multidimensional computing resource measurement in computing power network," in *2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. IEEE, 2024, pp. 2295–2300.

[12] X. Huang, R. R. Liu, B. Lei, W. Xing, and X. Zhang, "Platform profit maximization for space-air-ground integrated computing power network supplied by green energy," in *ICC 2024 - IEEE International Conference on Communications*, 2024, pp. 2507–2512.

[13] X. Huang, R. R. Liu, B. Lei, G. Huang, and B. Zhang, "Deep reinforcement learning based multistage profit aware task scheduling algorithm for computing power network," in *GLOBECOM 2023 - 2023 IEEE Global Communications Conference*, 2023, pp. 3524–3529.

[14] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[15] K. Shen and W. Yu, "Fractional programming for communication systems—part i: Power control and beamforming," *IEEE Transactions on Signal Processing*, vol. 66, no. 10, pp. 2616–2630, 2018.

[16] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," https://cvxr.com/cvx, Mar. 2014.

[17] H. Upton, "Electricity day ahead prices," [Online], https://www.kaggle.com/datasets/henriupton/electricity-dayahead-prices-entsoe/data, 2023.

[18] V. Ball and T. Pegoraro, "Germany energy day ahead 2018 to 2022," [Online], https://www.kaggle.com/datasets/vinceball/germany-energy-day-ahead-2018-to-2022/data, 2022.

[19] M. Towers, A. Kwiatkowski, J. Terry, J. U. Balis, G. D. Cola, T. Deleu, M. Goulão, A. Kallinteris, M. Krimmel, A. KG, R. Perez-Vicente, A. Pierré, S. Schulhoff, J. J. Tai, H. Tan, and O. G. Younis, "Gymnasium: A standard interface for reinforcement learning environments," 2024. [Online]. Available: https://arxiv.org/abs/2407.17032

[20] X. Huang, G. Ji, B. Zhang, and C. Li, "Platform profit maximization in D2D collaboration based multi-access edge computing," *IEEE Transactions on Wireless Communications*, pp. 4282–4295, 2023.