Joint Communication and Synchronization Performance Optimization in Digital Twin Enabled Networks

Hanzhi Yu*, Yuchen Liu†, and Mingzhe Chen*

*Department of Electrical and Computer Engineering and Frost Institute for Data Science and Computing, University of Miami, Coral Gables, FL, 33146, USA, Emails: {hanzhiyu,mingzhe.chen}@miami.edu.

†Department of Computer Science, North Carolina State University, Raleigh, NC, 27695, USA, Email: yuchen.liu@ncsu.edu.

GLOBECOM 2024 - 2024 IEEE Global Communications Conference | 979-8-3503-5125-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/GLOBECOM52923.2024.10901087

***Abstract*—In this paper, we investigate an accurate synchronization between a physical network and its digital network twin (DNT) that is a virtual representation of the physical network. The considered network includes a physical network where a base station (BS) serves a set of users, and a DNT that evolves with the status of both DNT and the physical network. The BS must use its limited spectrum resources to serve the users, as well as transmit the physical network information to the cloud server for DNT synchronization. Since the DNT can predict the physical network status, the BS may not need to transmit physical network information to the server at each time slot thus saving spectrum resources to serve users. However, if the BS does not transmit physical information to the DNT over a long period of time, the DNT may not be able to represent the physical network accurately. To this end, the BS must determine whether to send physical network information to the server to update DNT and the spectrum resources used for physical network information transmission and serving users. We formulate this resources allocation problem as an optimization problem aiming to maximize the sum of data rates of all users, while minimizing the gap between the states of the physical network and the DNT. The formulated problem is challenging to solve by conventional optimization methods, since the BS may not be able to know the future status of the DNT. To solve this problem, we design a gate recurrent unit (GRU) and soft action-critic (SAC) based algorithm. The GRU enables the DNT to predict its future states by using historical state data, and updating the DNT when the BS does not transmit physical network information. The SAC based algorithm enables the BS to learn the relationship between the physical network information transmission and the future status estimation accuracy of the DNT thus determining whether to transmit physical network information to the cloud server, ensuring an accuracte synchronization between the physical network and the DNT. Simulation results demonstrate that our designed algorithm can promote the weighted sum of data rates and the similarity between the status of the DNT and the physical network by up to 10.31% compared to a baseline method integrating the GRU and the deep Q network.**

I. Introduction

Digital twins (DTs) that are real-time virtual representations of physical products, processes or services can be used for real-time monitoring, simulations, and optimization of physical products [1]. Currently, researchers have focused on the generation of digital network twins (DNT), which is a virtual representation of a physical network, such that one can use generated DNTs for wireless network monitoring and optimization. First, constructing a DNT requires mapping not only physical objects (i.e., devices and infrastructure of a physical network) but also several unique networking factors (i.e., network protocols, wireless channel dynamics, and the network performance metrics). Hence, it is impractical to directly map all network features for DNT generation and one must select appropriate network features for DNT creations [2]. Second, no standardized metrics exist for evaluating the DNT performance [3]. Third, privacy protection in DNT remains a key issue to be addressed since private information of physical network users may be leaked in the process of information exchange between a physical network and a DNT [4].

Recently, a number of existing works [5]–[8] have studied the generation and creations of DNTs. In particular, the authors in [5] designed a deep neural network (DNN) based method to create a DNT for a mobile edge computing system. The designed DNT is used to approximate the optimal user association scheme in the physical network. In [6], the authors created a DNT using a Bayesian model. In [7], the authors proposed a continual learning framework to build a synchronized DNT of a autonomous vehicle network so as to make vehicle driving decisions. In [8], the authors presented several use cases, the design standardization, and an implementation example of the DNT. However, most of these works [5], [6], [8] did not consider the similarity and synchronization between the DNT and a physical network and they directly assumed that the DNT can be updated in real time. Meanwhile, these works [5]–[8] did not consider how the generation of DNTs affect the physical network performance since DNT generations require the transmission of a large amount of data, which will also introduce significant communication overhead.

The main contribution of this work is to design a novel DNT framework that jointly optimizes the performance of a physical network and the synchronization between the DNT and the physical network. Our key contributions include:

- We consider a DT enabled network that consists of a physical network with a base station (BS), several users, and a DNT. The DNT is a virtual representation of the physical network and can predict physical network dynamics. The BS must use limited spectrum resources to serve the users and transmit the information of the physical network to a cloud server to generate the DNT. Since the DNT can predict the physical network status, the BS may not need to transmit the information to the server at every time slot, and thus conserving spectrum resources to better serve the users. To this end, the BS in the physical network needs to determine whether to transmit the physical network information to the cloud server for updating the DNT, and optimize spectrum resource allocation for the users and physical network information transmission. We formulate this problem as an optimization problem aiming to maximize the data

This work was supported by the U.S. National Science Foundation under Grants CNS-2312139, SaTC-2350076, CNS-2312138, and SaTC-2350075.

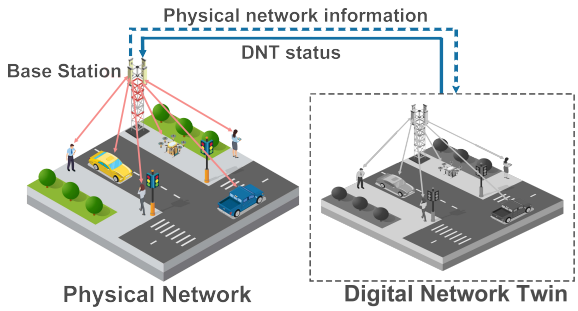

Fig. 1. The considered DNT enabled network.

rates of all users while minimizing the gap between the states of the physical network and the DNT.

- The formulated problem is challenging to solve since the BS may not be able to know the future status of the DNT. To solve this problem, we proposed a machine learning method that integrates the gate recurrent units (GRUs) and the soft actor-critic (SAC). The GRUs allow the DNT to predict its future state using historical data, ensuring DNT status updates when the information of the physical network cannot be transmitted. The SAC can learn from the GRU prediction accuracy to enable the BS to determine whether to send the physical network information to the cloud server, ensuring an accurate synchronization between the physical network and the DNT. Compared to other RL methods, the objective function of the SAC includes an entropy term, which not only stabilizes the algorithm but also promotes the exploration towards the global optimal policy.

Simulation results show that compared to a baseline method integrating the GRU and deep Q network (DQN), our proposed algorithm can improve the weighted sum of data rates and the similarity between the status of the DNT and the physical network by up to 10.31%.

II. System Model and Problem Formulation

We consider a DNT enabled network (see Fig. 1) that consists of: 1) a physical network in which a BS serve a set $\mathcal{U}$ of U mobile users, and 2) a DNT that is a mapping of the physical network and can be used for physical network monitoring and network dynamic prediction [9]. The BS must serve the users in the physical network and transmit the information of the physical network to a cloud server for DNT generation [10]. Next, we first introduce the mobility model of each user. Then, we introduce the data transmission model. Finally, we introduce our considered optimization problem.

A. Mobility Model

For simplicity, we use a random walk model to model the mobility of each user [11]. At each time slot t, each user u has 5 probable movements: 1) stay at the current location, 2) move forward, 3) move backward, 4) move left, and 5) move right. The probability of user u choosing each movement is $\boldsymbol{p}_u = [p_{u,1}, p_{u,2}, p_{u,3}, p_{u,4}, p_{u,5}]$, where $p_{u,1}$ is the probability of user u choosing to stay at the current position and $p_{u,2}$, $p_{u,3}$, $p_{u,4}$, $p_{u,5}$ are, respectively, the probability of user u moving forward, backward, left, and right. $\boldsymbol{p}_u$ is assumed to be identical for all time slots. We consider the coordinate of user u at time slot t is $\boldsymbol{l}^{\mathrm{U}}_{ut} = \left[l^{\mathrm{U}}_{ut,1}, l^{\mathrm{U}}_{ut,2}\right]$, and the distance that each user can move in a time slot is Δl. Thus, the coordinate of user u at time slot $t+1$ is

$$\boldsymbol{l}^{\mathrm{U}}_{u(t+1)} = \begin{cases} \left[l^{\mathrm{U}}_{ut,1}, l^{\mathrm{U}}_{ut,2}\right] & \text{with probability } p_{u,1}, \\ \left[l^{\mathrm{U}}_{ut,1}, l^{\mathrm{U}}_{ut,2} + \Delta l\right] & \text{with probability } p_{u,2}, \\ \left[l^{\mathrm{U}}_{ut,1}, l^{\mathrm{U}}_{ut,2} - \Delta l\right] & \text{with probability } p_{u,3}, \\ \left[l^{\mathrm{U}}_{ut,1} - \Delta l, l^{\mathrm{U}}_{ut,2}\right] & \text{with probability } p_{u,4}, \\ \left[l^{\mathrm{U}}_{ut,1} + \Delta l, l^{\mathrm{U}}_{ut,2}\right] & \text{with probability } p_{u,5}. \end{cases} \quad (1)$$

B. Transmission Model

We assume that the BS serves the users and sends the physical network information to the cloud server by employing the orthogonal frequency division multiple access (OFDMA) technique. We assume that the BS can allocate a set $\mathcal{N}$ of N resource blocks (RBs) to serve users and transmit physical network information. Each user can only occupy one RB. The transmission rate of the BS transmitting data to user u at time slot t is

$$c_{ut}\left(\boldsymbol{x}_{ut}\right) = \sum_{n=1}^{N} x_{ut,n} B \log_2\left(1 + \frac{Ph\left(\boldsymbol{l}^{\mathrm{U}}_{ut}\right)}{I_n + BN_0}\right), \quad (2)$$

where B is the bandwidth of each RB; $\boldsymbol{x}_{ut} = [x_{ut,1}, ..., x_{ut,N}]$ is the RB allocation vector, $x_{ut,n} \in \{0,1\}$ indicates whether RB n is allocated to user u at time slot t with $x_{ut,n} = 1$ indicating that RB n is allocated to user u at time slot t, otherwise, we have $x_{ut,n} = 0$; P is the transmit power, which is assumed to be equal for all users; $h\left(\boldsymbol{l}^{\mathrm{U}}_{ut}\right) = o_u d^{-2}_{ut}$ is the channel gain between user u and the BS with o_u being the Rayleigh fading parameter, $d_{ut} = \sqrt{\|\boldsymbol{l}^{\mathrm{U}}_{ut} - \boldsymbol{l}^{\mathrm{N}}\|_2}$ being the distance between user u and the BS at time slot t, $\boldsymbol{l}^{\mathrm{N}}$ being the coordinate of the BS; I_n is the interference caused by the users that are located in other service areas (e.g., other BSs that are not included in our considered physical network) and use RB n; N_0 is the noise power spectral density. Similarly, the transmission rate of the BS transmitting physical network information to the cloud server at time slot t is

$$c^{\mathrm{C}}_t\left(\boldsymbol{y}_t\right) = \sum_{n=1}^{N} y_{t,n} B \log_2\left(1 + \frac{Ph\left(\boldsymbol{l}^{\mathrm{C}}\right)}{I_n + BN_0}\right), \quad (3)$$

where $\boldsymbol{y}_t = [y_{t,1}, ..., y_{t,N}]$ is the RB allocation vector, $y_{t,n} \in \{0,1\}$ indicates whether RB n is allocated to the cloud server at time slot t with $y_{t,n} = 1$ indicating that RB n is allocated to the cloud server at time slot t and the BS will send the physical network information to the cloud server, otherwise, we have $y_{t,n} = 0$; $h_n\left(\boldsymbol{l}^{\mathrm{C}}\right)$ is the channel gain between BS n and the cloud server with $\boldsymbol{l}^{\mathrm{C}}$ being the coordinate of the cloud server. We assume the data size of the physical network information that is needed to be transmitted to the cloud server is D. Given the data rate $c^{\mathrm{C}}_t\left(\boldsymbol{y}_t\right)$, the transmission delay of

the BS transmitting physical network information to the cloud server at time slot t can be represented as $T_t^{\mathrm{C}}(\boldsymbol{y}_t) = \frac{D}{c_t^{\mathrm{C}}(\boldsymbol{y}_t)}$.

C. Digital Network Twin Model

The DNT is a mapping of the physical network and hence the DNT should have the same status and the wireless management strategy of the physical network. We define a vector $\boldsymbol{s}_t$ to represent the status of the physical network at time slot t. In our considered network, the network dynamics are introduced by user movements. Hence, $\boldsymbol{s}_t = \left[\boldsymbol{l}_{1t}^{\mathrm{U}}, ..., \boldsymbol{l}_{Ut}^{\mathrm{U}}\right]$. Given $\boldsymbol{y}_t$ and $\boldsymbol{s}_t$, the status $\boldsymbol{s}'_t(\boldsymbol{y}_t)$ of DNT at time slot t is

$$\boldsymbol{s}'_t(\boldsymbol{y}_t) = \begin{cases} \boldsymbol{s}_t, & \text{if } \sum_{n=1}^{N} y_{t,n} = 1, \\ \hat{\boldsymbol{s}}_t & \text{if } \sum_{n=1}^{N} y_{t,n} = 0, \end{cases} \tag{4}$$

where $\hat{\boldsymbol{s}}_t = \left[\hat{\boldsymbol{l}}_{1t}^{\mathrm{U}}, ..., \hat{\boldsymbol{l}}_{Ut}^{\mathrm{U}}\right]$ is the status estimated by the DNT at time slot t. From (4), we see that when the BS assigns an RB to send the physical network information to the cloud server (i.e., $\sum_{n=1}^{N} y_{t,n} = 1$), the status of the DNT at time slot t is the same as the status of the physical network $\boldsymbol{s}_t$ since DNT can directly obtain the status information of the physical network. In contrast, when the BS does not send the physical network information to the cloud server (i.e., $\sum_{n=1}^{N} y_{t,n} = 0$), the DNT must estimate the changes of the DNT (i.e., user movement) and uses the prediction as the status of the DNT. If the DNT and the physical network have the same status (i.e., $\boldsymbol{s}_t = \boldsymbol{s}'_t$), we consider the DNT synchronizes with the physical network.

D. Problem Formulation

Given the defined system model, our goal is to maximize the sum of the data rate of all users, while guaranteeing the synchronization between the physical network and the DNT over a set $\mathcal{T}$ of T time slots. This maximization problem includes optimizing the RB allocation vectors $\boldsymbol{x}_{1t}, ..., \boldsymbol{x}_{Ut}$, and physical network information transmission vector $\boldsymbol{y}_t$. The maximization problem is formulated as follows

$$\max_{\{\boldsymbol{x}_{1t},...,\boldsymbol{x}_{Ut},\boldsymbol{y}_t\}_{t\in\mathcal{T}}} \sum_{t=1}^{T}\sum_{u=1}^{U} \epsilon c_{ut}(\boldsymbol{x}_{ut}) - \frac{(1-\epsilon)}{U}\|\boldsymbol{s}_t - \boldsymbol{s}'_t(\boldsymbol{y}_t)\|_2^2, \tag{5}$$

$$\text{s.t. } x_{ut,n} \in \{0,1\}, u \in \mathcal{U}, n \in \mathcal{N}, t \in \mathcal{T}, \tag{5a}$$

$$\sum_{n=1}^{N} x_{ut,n} \leq 1, \forall u \in \mathcal{U}, t \in \mathcal{T}, \tag{5b}$$

$$y_{t,n} \in \{0,1\}, n \in \mathcal{N}, t \in \mathcal{T}, \tag{5c}$$

$$\sum_{n=1}^{N} y_{t,n} \leq 1, n \in \mathcal{N}, t \in \mathcal{T}, \tag{5d}$$

$$T_t^{\mathrm{C}}(\boldsymbol{y}_t) \leq \gamma, t \in \mathcal{T}, \tag{5e}$$

where $\epsilon \in (0,1)$ is a weight parameter, γ is the upper bound of the delay of transmitting the physical network information to the cloud server. In (5), the constraints (5a) and (5b) imply that each user can occupy only one RB, (5c) and (5d) imply that the BS must determine whether to allocate one RB to send the physical network information to the cloud server, (5e) implies that the delay of transmitting the physical network information to the cloud server has to meet a requirement γ.

The problem (5) is challenging to solve by conventional optimization methods due to the following reasons. First, the BS may not be able to know a future status $\boldsymbol{s}'_{t+1}$ of the DNT since it depends not only on $\boldsymbol{y}_t$ but also on the status estimation accuracy of the DNT. Second, the status of the considered physical network and DNT varies in real-time due to the movement of the users and the physical network information transmission. If we use conventional optimization methods to solve problem (5) so as to adapt to these dynamics, we must implement the optimization methods at each time slot. Hence, the computational complexity increases significantly. To solve problem (5), we use a reinforcement learning (RL) based method which can learn the relationship between $\boldsymbol{y}_t$ the status estimation accuracy of the DNT to determine $\boldsymbol{y}_t$ at each time slot, so as to keep an accurate synchronization between the physical network and the DNT.

III. Proposed Solution

To solve problem (5), we introduce a machine learning method that integrates the GRUs and the SAC algorithm. In our proposed method, the GRUs are used to predict the status of the DNT, and the SAC is used to determine whether to send the physical network information to the cloud server while maintaining an accurate synchronization between the physical network and the DNT. Compared to other RNN methods for user mobility predictions, the GRUs can effectively fix the gradient vanishing and the gradient exploding problems thus improving prediction accuracy and reducing computational overhead. Compared to other RL methods, the SAC is based on maximizing entropy which makes the algorithm more stable, while also encouraging exploration to find multiple near-optimal actions. Next, we first introduce the use of GRUs for DNT status prediction. Then, we introduce the components of our proposed SAC based RL framework. Finally, we explain the procedure of using our proposed method to solve problem (5).

A. GRUs for DNT Status Prediction

We first introduce the use of GRUs for predicting the status of the DNT. The GRU-based prediction model is implemented by the cloud server and the output of the model is the status of the DNT, which will be used for the update of the DNT wireless resource management strategy. A GRU-based prediction model consists of three components: 1) the input, 2) the output, and 3) the GRU model, which are introduced as follows.

1) Input: The input of the GRU model is M previous states of the DNT, which is represented as $\boldsymbol{S}'_t = \left(\boldsymbol{s}'_{t-M+1}, ..., \boldsymbol{s}'_t\right)$.

2) Output: The output of the GRU model is $\hat{\boldsymbol{s}}'_{t+1} = \left[\hat{\boldsymbol{l}}_{1(t+1)}^{\mathrm{U}}, ..., \hat{\boldsymbol{l}}_{U(t+1)}^{\mathrm{U}}\right]$.

3) GRU model: The GRU model is used to approximate the function between the input $\boldsymbol{S}'_t$ and the output $\boldsymbol{s}'_{t+1}$, thus building the relationship between the historical states and the current state of the DNT. A GRU model consists of an input layer, a hidden layer, and an output layer. The hidden states $\boldsymbol{h}_t$ of the units in the hidden layer at time slot t are used to store the information related to the previous states from time slots 1 to t. The hidden states $\boldsymbol{h}_t$ of the GRU are updated based on $\boldsymbol{S}'_t$ and $\boldsymbol{h}_{t-1}$. Next, we introduce how to update the hidden state $h_{t,j}$ of hidden unit j given $\boldsymbol{S}'_t$.

At each time slot t, the hidden state is determined by two gates: 1) the reset gate $\boldsymbol{r}_t^{\mathrm{G}}$, and 2) the update gate $\boldsymbol{z}_t^{\mathrm{G}}$. The reset gate $\boldsymbol{r}_t^{\mathrm{G}}$ determines the historical states information is retained in the candidate hidden state $\tilde{\boldsymbol{h}}_t$, which is given by

$$\boldsymbol{r}_t^{\mathrm{G}} = \sigma\left(\boldsymbol{W}^{\mathrm{r}}\boldsymbol{S}'_t + \boldsymbol{U}^{\mathrm{r}}\boldsymbol{h}_{t-1}\right), \tag{6}$$

where $\sigma(\cdot)$ is the sigmoid function, and $\boldsymbol{W}^{\mathrm{r}} \in \mathbb{R}^{N^{\mathrm{h}} \times 2U}$ and $\boldsymbol{U}^{\mathrm{r}} \in \mathbb{R}^{N^{\mathrm{h}} \times N^{\mathrm{h}}}$ are the weight matrices of the reset gate with N^{h} being the number of the units in the hidden layer. Given the reset gate $\boldsymbol{r}_t^{\mathrm{G}}$, the candidate hidden state $\tilde{\boldsymbol{h}}_t$ is

$$\tilde{\boldsymbol{h}}_t = \tanh\left(\boldsymbol{W}^{\tilde{h}}\boldsymbol{S}'_t + \boldsymbol{U}^{\tilde{h}}\left(\boldsymbol{h}_{t-1} \odot \boldsymbol{r}_t^{\mathrm{G}}\right)\right), \tag{7}$$

where $\tanh(\cdot)$ is the hyperbolic tangent function, $\boldsymbol{W}^{\tilde{h}} \in \mathbb{R}^{N^{\mathrm{h}} \times 2U}$ and $\boldsymbol{U}^{\mathrm{r}} \in \mathbb{R}^{N^{\mathrm{h}} \times N^{\mathrm{h}}}$ are the weight matrices of the hidden states, and $\odot$ is an element-wise multiplication.

The update gate $\boldsymbol{z}_t^{\mathrm{G}}$ determines the size of the information stored in the hidden state to update the hidden state $\boldsymbol{h}_t$, which is given by

$$\boldsymbol{z}_t^{\mathrm{G}} = \sigma\left(\boldsymbol{W}^{\mathrm{z}}\boldsymbol{S}'_t + \boldsymbol{U}^{\mathrm{z}}\boldsymbol{h}_{t-1}\right), \tag{8}$$

where $\boldsymbol{W}^{\mathrm{z}} \in \mathbb{R}^{N^{\mathrm{h}} \times 2U}$ and $\boldsymbol{U}^{\mathrm{z}} \in \mathbb{R}^{N^{\mathrm{h}} \times N^{\mathrm{h}}}$ are the weight matrices of the update gate. The hidden state $\boldsymbol{h}_t$ can be updated by

$$\boldsymbol{h}_t = \left(\boldsymbol{1} - \boldsymbol{z}_t^{\mathrm{G}}\right) \odot \tilde{\boldsymbol{h}}_t + \boldsymbol{z}_t^{\mathrm{G}} \odot \boldsymbol{h}_{t-1}. \tag{9}$$

The GRU model iteratively updates the hidden states to store the input $\boldsymbol{S}'_t$ until the hidden state of the current time t is computed. The output layer of the GRU model will predict the state of the DNT at time slot $t+1$ based on the hidden state $\boldsymbol{h}_t$:

$$\hat{\boldsymbol{s}}'_{t+1} = \boldsymbol{W}^{\mathrm{o}}\boldsymbol{h}_t, \tag{10}$$

where $\boldsymbol{W}^{\mathrm{o}}$ is the output weight matrix. Based on (10), given $\boldsymbol{h}_t$, we can get the output $\hat{\boldsymbol{s}}'_{t+1}$.

B. GRU Training

The GRU-based prediction model approximates the function between the output future state $\hat{\boldsymbol{s}}'_{t+1}$ and the input previous M states of the DNT $\boldsymbol{s}'_{t-M+1}, ..., \boldsymbol{s}'_t$. Thus, the loss function of our proposed GRU-based model is

$$\mathcal{L}^{\mathrm{G}} = \frac{1}{2U}\|\hat{\boldsymbol{s}}'_{t+1} - \boldsymbol{s}_{t+1}\|^2. \tag{11}$$

Given the defined loss function, next, we introduce the training process of our proposed GRU-based algorithm. We use the mini-batch stochastic gradient descent (SGD) method to update the parameters of the model to minimize (11). The update rule of the parameter matrices at time slot t is given by

$$\boldsymbol{W}^i_{t+1} = \boldsymbol{W}^i_t - \alpha\nabla_{\boldsymbol{W}^i_t}\mathcal{L}^{\mathrm{G}}, \boldsymbol{U}^j_{t+1} = \boldsymbol{U}^j_t - \alpha\nabla_{\boldsymbol{U}^j_t}\mathcal{L}^{\mathrm{G}}, \tag{12}$$

where α is the learning rate, $i \in \{\mathrm{r}, \tilde{h}, \mathrm{z}, \mathrm{o}\}$, $j \in \{\mathrm{r}, \tilde{h}, \mathrm{z}\}$, $\nabla_{\boldsymbol{W}^i}\mathcal{L}^{\mathrm{G}}$ is the gradient of the loss function with respect to $\boldsymbol{W}^i$, and $\nabla_{\boldsymbol{U}^j}\mathcal{L}^{\mathrm{G}}$ is the gradient of the loss function with respect to $\boldsymbol{U}^j$.

C. Components of the SAC Algorithm

Next, we introduce the use of SAC based RL to solve problem (5). The SAC based RL model consists of seven components: 1) agent, 2) state, 3) action, 4) policy, 5) reward function, 6) value function, and 7) Q function, which are introduced as follows.

1) Agent: The agent in our considered problem is the BS. The BS can collect information of the physical network, such as the locations of all users, the Rayleigh fading power gains, the interference caused by the users that are out of the service area.

2) State: The state of the BS is used to describe the current status of the physical network. Each state of the physical network is $\boldsymbol{s}_t$. Thus, each state of the BS can be represented as $\boldsymbol{s}_t$.

3) Action: The action of the BS is to determine whether to transmit physical network information to the cloud server and, if so, choose a proper RB for the transmission. Hence, each action of the BS at time slot t is $\boldsymbol{a}_t = \boldsymbol{y}_t$. Here, the reason that we consider only physical network information transmission indicator $\boldsymbol{y}_t$ as action without considering RB allocation vector $\boldsymbol{x}_{1t}, ..., \boldsymbol{x}_{Ut}$ as action is because when the physical network information transmission indicator $\boldsymbol{y}_t$ is determined, the optimal RB allocation vector $\boldsymbol{x}_{1t}, ..., \boldsymbol{x}_{Ut}$ can be determined by an optimization algorithm. In consequence, the action space of the RL can be significantly reduced.

4) Policy: The policy of the BS is the conditional probability of the BS choosing action $\boldsymbol{a}_t$ based on state $\boldsymbol{s}_t$. The policy is implemented by a DNN parameterized by $\boldsymbol{W}$, which describes the relationship of the positions of users, the similarity between the physical network and the DNT, and the data rates of all users. Then, the conditional probability of the BS taking action $\boldsymbol{a}_t$ based on the $\boldsymbol{s}_t$ is $\boldsymbol{\pi}_{\boldsymbol{W}}(\boldsymbol{a}_t|\boldsymbol{s}_t)$. Hereinafter, we use actor to refer the policy function.

5) Q function: The Q function estimates the expected reward of the BS taking action $\boldsymbol{a}_t$ at each state $\boldsymbol{s}_t$. The BS uses a DNN with parameter $\boldsymbol{\theta}$ to approximate the Q function $Q_{\boldsymbol{\theta}}(\boldsymbol{s}_t, \boldsymbol{a}_t)$.

6) Value function: The value function estimates the expected reward of the BS at each state $\boldsymbol{s}_t$. The value function of the BS is implicitly parameterized by the Q function parameters, which is given by

$$V_{\boldsymbol{\theta}}(\boldsymbol{s}_t) = \mathbb{E}_{\boldsymbol{a}_t}\left[Q_{\boldsymbol{\theta}}(\boldsymbol{s}_t, \boldsymbol{a}_t) - \alpha\log\boldsymbol{\pi}(\boldsymbol{a}_t|\boldsymbol{s}_t)\right], \tag{13}$$

where α is the temperature parameter.

7) Reward: The reward function $r(\boldsymbol{s}_t, \boldsymbol{a}_t)$ evaluates action $\boldsymbol{a}_t$ at state $\boldsymbol{s}_t$. The reward function of the BS is the weighted sum of the data rates of all users and the similarity between the states of the physical network and DNT. The data rate of the users depends on the physical network information transmission since the physical network information transmission must occupy one RB. Hence, if the BS does not transmit the physical network information to the cloud server, the optimization problem in (5) can be formulated as

$$\max_{\{\boldsymbol{x}_{1t},...,\boldsymbol{x}_{Ut}\}_{t\in\mathcal{T}}} \sum_{t=1}^{T}\sum_{u=1}^{U} c_{ut}(\boldsymbol{x}_{ut}) - \frac{1}{U}\|\boldsymbol{s}_t - \hat{\boldsymbol{s}}_t\|_2^2, \quad (14)$$
$$\text{s.t. } (5a), (5b).$$

Since we only need to optimize $\boldsymbol{x}_{1t}, ..., \boldsymbol{x}_{Ut}$, the Hungarian algorithm [12] can be used to find the optimal solution. Based on (14), the reward function when the BS does not transmit the physical network information to the cloud server is

$$\boldsymbol{r}(\boldsymbol{s}_t, \boldsymbol{a}_t) = -\frac{(1-\epsilon)}{U}\|\boldsymbol{s}_t - \hat{\boldsymbol{s}}_t\|_2^2 + \sum_{u=1}^{U} \epsilon c_{ut}(\boldsymbol{x}_{ut}), \quad \text{if} \sum_{n=1}^{N} y_{t,n} = 0. \quad (15)$$

If the BS allocates a RB to transmit the physical network information to the cloud server, the BS must first determine the RB that is used for physical network information transmission. The RB used for physical network information transmission can expressed as $R = \arg\max_n \{T_t^{\mathrm{C}}(\boldsymbol{y}_t), T_t^{\mathrm{C}}(\boldsymbol{y}_t) < \gamma\}$. Given the RB R used for the physical network information transmission, the optimization problem in (5) is simplified as

$$\max_{\{\boldsymbol{x}_{1t},...,\boldsymbol{x}_{Ut}\}_{t\in\mathcal{T}}} \sum_{t=1}^{T}\sum_{u=1}^{U} c_{ut}(\boldsymbol{x}_{ut}), \quad (16)$$
$$\text{s.t. } (5a), (5b),$$
$$\sum_{u=1}^{U} x_{ut,R} = 0, t \in \mathcal{T}. \quad (16a)$$

where constraint (16a) implies that the BS cannot allocate RB R to any users since it has been used for physical network information transmission. Based on (16), the reward when the BS transmits the physical network information to the server is

$$r(\boldsymbol{s}_t, \boldsymbol{a}_t) = \sum_{u=1}^{U} \epsilon c_{ut}(\boldsymbol{x}_{ut}). \quad (17)$$

Similarly, since we only need to optimize $\boldsymbol{x}_{1t}, ..., \boldsymbol{x}_{Ut}$, we can still use the Hungarian algorithm to find the solution.

D. SAC Training at the BS

Next, we introduce the training process of the SAC algorithm. We first assume that at the start of each iteration, the BS samples a set of state transitions $\{(\boldsymbol{s}_t, \boldsymbol{a}_t, r_t(\boldsymbol{s}_t, \boldsymbol{a}_t), \boldsymbol{s}_{t+1})\}$ based on the policy $\boldsymbol{\pi}_{\boldsymbol{W}}$. These transitions are then stored in a replay buffer $\mathcal{D}$. Next, we introduce the training process of the policy and the value function separately.

TABLE I
SYSTEM PARAMETERS

Param.	Val.	Param.	Val.	Param.	Val.
U	10	N	6	B	1
P	1	N_0	1×10^{-5}	o_u	1
$\boldsymbol{l}^{\mathrm{N}}$	[0, 0]	$\boldsymbol{l}^{\mathrm{C}}$	[20, 20]	ϵ	0.987
D	0.1	γ	2.5	M	5
N^{h}	128	α	0.05	$\lambda_{\boldsymbol{\pi}}$	1×10^{-4}
β	0.8	λ_Q	1×10^{-3}		

- **Training of the policy neural network**: The loss function of the policy neural network is

$$J_{\boldsymbol{\pi}}(\boldsymbol{W}) = \mathbb{E}_{\boldsymbol{s}_t\sim\mathcal{D}}[\mathbb{E}_{\boldsymbol{a}_t\sim\boldsymbol{\pi}_{\boldsymbol{W}}}[\alpha\log\boldsymbol{\pi}_{\boldsymbol{W}}(\boldsymbol{a}_t|\boldsymbol{s}_t) - Q_{\boldsymbol{\theta}}(\boldsymbol{s}_t, \boldsymbol{a}_t)]]. \quad (18)$$

  At each iteration, the policy $\boldsymbol{\pi}_{\boldsymbol{W}}$ will be updated using the stochastic gradient descent (SGD) method so as to minimize (18). The policy update rule is given as

$$\boldsymbol{W} \leftarrow \boldsymbol{W} - \lambda_{\boldsymbol{\pi}}\hat{\nabla}_{\boldsymbol{W}} J_{\boldsymbol{\pi}}(\boldsymbol{W}), \quad (19)$$

  where $\lambda_{\boldsymbol{\pi}}$ is the learning rate, and $\hat{\nabla}_{\boldsymbol{W}} J_{\boldsymbol{\pi}}(\boldsymbol{W})$ is the approximated gradient of (18).
- **Training of the neural network in Q function**: In SAC, we use two soft Q functions to reduce the positive bias in the policy improvement step. In particular, we parameterize each soft Q functions by $\boldsymbol{\theta}_k$ with $k \in \{1, 2\}$, and train them independently by minimizing the loss function $J_Q(\boldsymbol{\theta}_k)$ [13]. The loss function of the Q function neural network $J_Q(\boldsymbol{\theta}_k)$ is given by

$$J_Q(\boldsymbol{\theta}_k) = \frac{1}{2}\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t)\sim\mathcal{D}}\left[Q_{\boldsymbol{\theta}_k}(\boldsymbol{s}_t, \boldsymbol{a}_t) - \left(r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \beta V_{\overline{\boldsymbol{\theta}}_k}(\boldsymbol{s}_{t+1})\right)^2\right], k \in \{1, 2\}, \quad (20)$$

  where $\beta \in (0, 1)$ is a discount factor; and $\overline{\boldsymbol{\theta}}_k$ is the parameters of the target network, which is a slowly updated copy of the Q function neural network parameterized by $\boldsymbol{\theta}_k$. Given (20), the parameters of the Q function is updated by a SGD method as follows:

$$\boldsymbol{\theta}_k \leftarrow \boldsymbol{\theta}_k - \lambda_Q\hat{\nabla}J_Q(\boldsymbol{\theta}_k), k \in \{1, 2\}, \quad (21)$$

  where λ_Q is the learning rate, and $\hat{\nabla}J_Q(\boldsymbol{\theta}_k)$ is the approximated gradient of (20).

IV. SIMULATION RESULTS AND ANALYSIS

For simulations, we consider a 100×100 block area served by the wireless network. The BS is located in the center of the area serving $U = 10$ moving users. The starting position $\boldsymbol{l}_{u0}^{\mathrm{U}}$ of each user u is randomly selected. The BS collects 2,000 trajectories with the length of each trajectory being $T = 30$ from all U users to train a GRU model. Other parameters used in the simulations are listed in Table I. For comparison purposes, we use a GRU and deep Q network (DQN) based model as the baseline. The baseline model parameters are similar to that of the designed scheme.

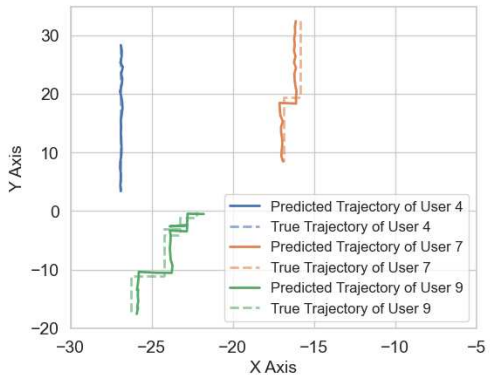

Fig. 2. The prediction of the user movement trajectories via the GRU model.

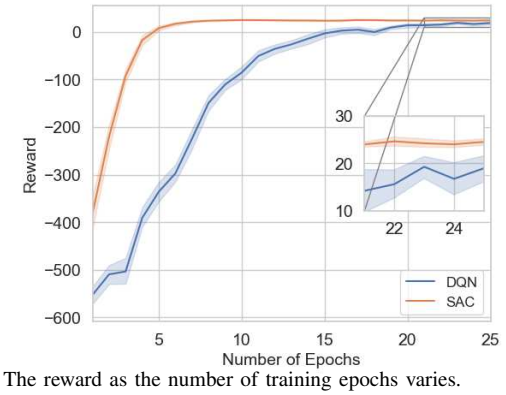

Fig. 3. The reward as the number of training epochs varies.

Fig. 2 shows the user movement trajectories predicted by the GRU model. The users in Fig. 2 are randomly selected from 10 users. From Fig. 2, we see that the positioning mean square errors of user 4, user 7, and user 9 are respectively 0.007, 0.07 and 0.24. This is because the GRU effectively captures dependencies in the historical user movement through its gating mechanisms thus enabling accurate predictions of future user movements. From this figure, we can also see that the mobility prediction accuracy of users 4 and 7 is much higher compared to that of user 9. This is because the movement dynamics of user 9 are higher compared to that of users 4 and 7.

In Fig. 3, we show how the weighted sum of the data rates and the synchronization accuracy between the DNT and the physical network changes as the number of training epochs varies. Fig. 3 shows that, as the number of training epochs increases, the average rewards of both considered algorithms increase. This is because the policy of determining the physical network information transmission is optimized by the considered RL algorithms. From Fig. 3, we also see that our designed algorithm can improve the weighted sum of data rates and the synchronization accuracy by up to 10.31% compared to the baseline. This is because SAC incorporates an entropy term into its objective function to enhance the model's exploratory capabilities towards the global optimal policy.

V. Conclusion

In this paper, we have proposed a DNT enabled network which includes a physical network and its DNT. The BS in the physical network must use its limited spectrum resources to serve a set of users and transmit the physical network information to a cloud server for DNT generation. We have formulated this resources allocation problem as an optimization problem whose goal is to maximize the sum of data rates of all users, while minimizing the gap between the state of the physical network and the DNT. The formulated problem is challenging to solve by conventional optimization methods, since the BS may not be able to know the future status of the DNT. To solve this problem, we have proposed a GRU and SAC based algorithm. The GRU enables the DNT to predict its future state to maintain updates when the physical network information is not transmitted. The SAC enables the BS to find the relationship between the physical network information transmission and the future status estimation accuracy of DNT thus determining whether to transmit physical network information to the cloud server, ensuring an accurate synchronization between the physical network and the DNT. Simulation results have shown that compared to a baseline method using the GRU and the DQN, our proposed GRU and SAC based algorithm can achieve significant promotion in terms of the weighted sum of the data rates and the similarity between the status of the DNT and the physical network.

References

[1] S. Mihai, M. Yaqoob, D. V. Hung, W. Davis, P. Towakel, M. Raza, M. Karamanoglu, B. Barn, D. Shetve, R. V. Prasad, H. Venkataraman, R. Trestian, and H. X. Nguyen, "Digital twins: A survey on enabling technologies, challenges, trends and future prospects," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 4, pp. 2255–2291, September 2022.

[2] Z. Zhang, M. Chen, Z. Yang, and Y. Liu, "Mapping wireless networks into digital reality through joint vertical and horizontal learning," *arXiv preprint arXiv:2404.14497*, April 2024.

[3] A. Sharma, E. Kosasih, J. Zhang, A. Brintrup, and A. Calinescu, "Digital twins: State of the art theory and practice, challenges, and open research questions," *Journal of Industrial Information Integration*, vol. 30, pp. 100383, September 2022.

[4] Y. Wu, K. Zhang, and Y. Zhang, "Digital twin networks: A survey," *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 13789–13804, September 2021.

[5] R. Dong, C. She, W. Hardjawana, Y. Li, and B. Vucetic, "Deep learning for hybrid 5G services in mobile edge computing systems: Learn from a digital twin," *IEEE Transactions on Wireless Communications*, vol. 18, no. 10, pp. 4692–4707, July 2019.

[6] C. Ruah, O. Simeone, and B. M. Al-Hashimi, "A Bayesian framework for digital twin-based control, monitoring, and data collection in wireless systems," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 10, pp. 3146–3160, August 2023.

[7] O. Hashash, C. Chaccour, and W. Saad, "Edge continual learning for dynamic digital twins over wireless networks," *arXiv preprint arXiv:2204.04795*, April 2022.

[8] X. Lin, L. Kundu, C. Dick, E. Obiodu, T. Mostak, and M. Flaxman, "6G digital twin networks: From theory to practice," *IEEE Communications Magazine*, vol. 61, no. 11, pp. 72–78, November 2023.

[9] Z. Yang, M. Chen, Y. Liu, and Z. Zhang, "A joint communication and computation framework for digital twin over wireless networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 18, no. 1, pp. 6–17, December 2023.

[10] M. Chen, D. Gündüz, K. Huang, W. Saad, M. Bennis, A. V. Feljan, and H. V. Poor, "Distributed learning in wireless networks: Recent progress and future challenges," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3579–3605, October 2021.

[11] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Performance optimization of federated learning over mobile wireless networks," in *Proc. IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Atlanta, GA, USA, May 2020, pp. 1–5.

[12] R. Jonker and T. Volgenant, "Improving the Hungarian assignment algorithm," *Operations research letters*, vol. 5, no. 4, pp. 171–175, October 1986.

[13] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, et al., "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, December 2018.