Deep Reinforcement Learning for Data Freshness-oriented Scheduling in Industrial IoT

Jiaping Li*, Jianhua Tang*† and Zilong Liu‡
* Shien-Ming Wu School of Intelligent Engineering, South China University of Technology, China
† Pazhou Lab, Guangzhou, China
‡ School of Computer Science and Electronics Engineering, University of Essex, UK
wijiapingli@mail.scut.edu.cn, jtang4@e.ntu.edu, sgzilong.liu@essex.ac.uk

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10001430

***Abstract*—Making a timely and precise scheduling in Industrial Internet of Things (IIoT) is fundamental and critical. Recently, Age of Incorrect Information (AoII) is proposed and utilized to measure the timeliness and accuracy of monitoring. In this work, we investigate a multi-sensor update system and leverage AoII to quantify the information freshness. Our goal is to obtain an optimal scheduling policy to minimize the system-wide cost. We first model the source statuses monitored by sensors as Markov chains and the scheduling problem as a Markov decision process (MDP). Due to the heterogeneity of source statuses in IIoT, it is prohibitive to solve the formulated MDP problem by conventional methods. To this end, we make use of a deep reinforcement learning (DRL) algorithm to solve this scheduling problem. Extensive numerical results verify the effectiveness of the adopted DRL algorithm. In addition, comparing to the conventional Age of Information (AoI) oriented method, we find that the AoII oriented method is much more effective, from the perspective of system-wide cost.**

***Index Terms*—Age of Incorrect Information, Deep Reinforcement Learning, Markov Decision Process, Industrial Internet of Things**

I. Introduction

The Industrial Internet of Things (IIoT) has emerged as a critical component of "Industrial 4.0" in recent years. In IIoT, thousands of communication devices (e.g., robots, sensors, machines) spread over the factories for remote monitoring and controlling [1], [2]. For instance, the status information of manufacturing can assist the remote monitor to reduce the defect rate of processed products; an alert for disastrous event (such as fire, earthquake, flood, etc) can avoid the loss of life and property. In these scenarios, there is a strict requirement for the freshness of status information as the timeliness of information can effect the correctness of decisions made by remote controllers. In the past decade, a new metric, called age-of-information (AoI), was proposed to describe the timeliness/freshness of status data, which reflects the interval from the generating time of the recently received packet by destination to the current moment [3]–[5]. Whereas, the AoI captures only the timeliness of transmitted packets but ignores the information content, which results in that the AoI keeps increasing even if the remote monitor already has perfect knowledge of the local source status.

To overcome the limitations of AoI, many research works incorporate the content of information on top of the freshness. [6] proposes a metric called age of synchronization (AoS) which is defined as the time interval between now and when the status of local source became desynchronized with remote estimation at central controller/server. Besides, [7] observes the variation in the Wiener process of local source and studies the optimal sampling strategy which can minimize the mean square remote estimation error. Different from [6] (which disregards the precision of distant estimate) and [7] (which ignores the timeliness of remote estimation), [8] introduces a new metric, age of incorrect information (AoII), to involve both the precision and the freshness of remote estimation. The AoII is defined as a penalty function whose value is zero when the monitor estimates the status of local source accurately, otherwise the value grows with time. Nevertheless, [8] just consider a single source system with homogeneous status. For a real IIoT system with multiple sources, it always encompasses status-heterogeneous transitions [9].

In this work, we consider a multi-sensor monitoring system where the sources confront heterogeneous statuses, and adopt AoII metric to quantify the precision and freshness of remote estimation. We aim to find the optimal scheduling policy to minimize the system-wide cost. Our contributions are:

- We start the system description from a single sensor that samples status-heterogeneous Markov source, and then extend to the multi-sensor system. We model the system-wide cost minimization problem as Markov decision process (MDP).
- Due to the complexity of the MDP, we adopt a deep reinforcement learning (DRL) algorithm, i.e. asynchronous advantage actor-critic (A3C), to solve the problem.
- Numerical results show that the A3C algorithm can achieve good performance in both status-homogeneous and status-heterogeneous systems, and also outperforms the baseline algorithm.

II. System model and problem formulation

We consider an update and control system as depicted in Fig. 1, where N sensors monitor the objects/sources and transmit

This work was supported in part by the National Nature Science Foundation of China under Grant 62001168 and in part by the Foundation and Application Research Grant of Guangzhou under Grant 202102020515.

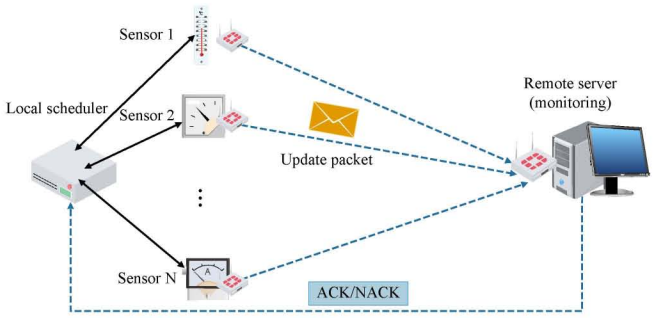


Fig. 1. System model.

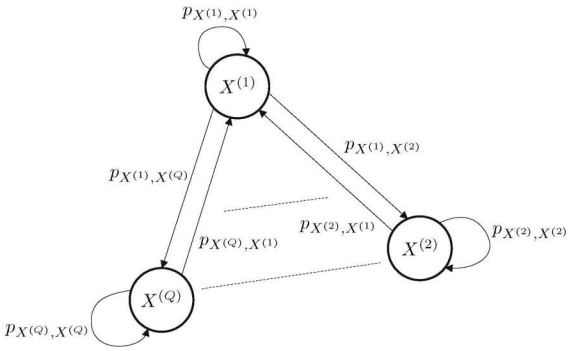


Fig. 2. The transition probability of status of source.

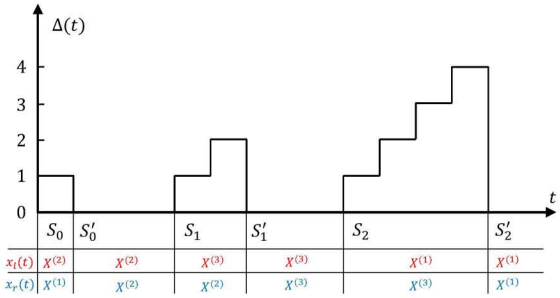


Fig. 3. The evolution of age of incorrect information.

update packets to a local scheduler simultaneously. To keep the sensing system lightweight and effectiveness, the local scheduler decides which sensors need to forward update packets to remote server via wireless link. We ignore the communication delay between sensors and local scheduler since they are very close and connected by wired link. We assume that the local scheduler has adequate capacity. Given the error-prone nature of wireless link, if remote server does not receive update packets successfully, an NACK feedback is sent back to local scheduler; otherwise, remote server sends an ACK feedback.

A. Single sensor scenario

In order to illustrate our model, we first consider the scenario with only one sensor. For simplicity, the time is slotted and the index of time-slot is denoted by $t \in \{1, 2, ..., T\}$. At the beginning of each time-slot, each sensor samples the source and local scheduler acquire the status of source, which is known as the generate-at-will model [10]. Moreover, we assume the changing of local source status only happens at the beginning of each time-slot. We classify the information process sampled by sensors into Q discrete statuses/values, i.e., $X^{(1)}, X^{(2)}, ..., X^{(Q)}$, and denote the status indicator at time-slot t by $x_l(t) \in \{X^{(1)}, X^{(2)}, ..., X^{(Q)}\}$. Besides, as illustrated in Fig. 2, we assume $x_l(t)$ as a Q statuses discrete Markov chain with one-step transition probability $p_{X^{(i)},X^{(j)}} = \mathbf{Pr}\left(x_l(t+1) = X^{(j)} | x_l(t) = X^{(i)}\right)$. Let the estimation of remote server at time-slot t be $x_r(t)$.

In this work, we investigate the age of incorrect information (AoII), which is different from age of information (AoI). As illustrated in Fig. 3, at time-slot S_0, the remote estimation is different with source status, so the value of AoII goes up to 1; At time-slot S_0', the remote estimation is the same as source status and then the value of AoII drops down to zero; At time-slot S_1, the remote estimation is incorrect again and the AoII increases over time. Without loss of generality, at time-slot S_k, $k \in \{0, 1, 2, ...\}$, if the remote estimation is incorrect and then the value of AoII is increased by one [1]. And the value of AoII resets to zero immediately when the remote estimation is correct, i.e., $x_l(t) = x_r(t)$.

We assume that 1) the sensor forwards freshest update packet only at the beginning of time-slot, 2) each transmission (of update packet) spends one time-slot similar to [11] and the transmission time of ACK/NACK can be ignored, 3) then the remote server receives update packet at the end of time-slot. Once the remote server receives the status packet correctly, it estimates the source status based on the freshest packet. We denote the probability that the update packet is received successfully by p_s, and then failed transmission with probability $p_f = 1 - p_s$. Suppose that the remote estimation is not accurate until time-slot S_k', which could be due to the transmission success or the change of source status. Therefore, between the beginning of slot S_k and the beginning of slot S_k', the remote server is not estimating the status of source correctly and the AoII denoted by $\Delta(t)$ increases over time. Besides, AoII drops down to zero due to accurate remote estimation at slot S_k'.

Considering that the local scheduler receives a recent ACK feedback at the end of slot $t_a(t)$. Hence, the local scheduler knows the remote estimation at time-slot t is the source status at the beginning of time-slot $t_a(t)$, i.e., $x_r(t) = x_l(t_a(t))$. By comparing $x_l(t)$ and $x_l(t_a(t))$, the local scheduler can judge whether the remote estimation is accurate. Let $m(t) = \max\{k | S_k' \leq t\}$ and $c(t) = \max\{k | S_k \leq t\}$ be the indices of the latest correct and the latest incorrect remote estimations at time-slot t, respectively. For instances, when time-slot t is between S_1 and S_1', $m(t) = 0$, $c(t) = 1$ and the remote estimation is incorrect; when time-slot t is between S_1' and S_2, $m(t) = 1$, $c(t) = 1$ and remote estimation is accurate. Therefore, the AoII at time-slot t can be expressed as

$$\Delta(t) = (t - S_{c(t)} + 1)\mathbb{1}_{\{m(t) \neq c(t)\}},$$

[1] In this work, we consider the step-wise AoII model other than the conventional linearly-increasing model.

where $\mathbb{1}$ is the indicator function. Let $a_t = 1$ to represent that the sensor is chosen to transmit status updates; and $a_t = 0$ otherwise. Although each transmission attempt takes one time-slot, it may take more than one slot for one successful transmission due to the unstable wireless channel.

The sensor state consists of the AoII, the source status and the remote estimation, which can be described by vector $\boldsymbol{s}_t = (\Delta(t), x_l(t), x_r(t))$. At each time-slot, the sensor can choose an action of transmitting or not transmitting, i.e., $a_t = 1$ or 0. The system state transition probability is defined as $\mathbf{P}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, a_t) = \mathbf{Pr}(\boldsymbol{s}_{t+1} = \boldsymbol{s}'|\boldsymbol{s}_t = \boldsymbol{s}, a_t = 0 \text{ or } 1)$, which means that an action $\in \{0, 1\}$ in state $\boldsymbol{s}$ at time t will lead to a new state $\boldsymbol{s}'$ at time $t+1$. We denote the source status at slot $t+1$ be x_l' and divide the transition probability into three cases as follows (we omit (t) in $\Delta(t)$, $x_l(t)$ and $x_r(t)$):

1) Case 1: At the slot t, the sensor is not scheduled to transmit update packet, i.e., $a_t = 0$. At slot $t+1$, the remote estimation does not change.

- If $x_l' = x_r$, the AoII at slot $t+1$ will be 0 and $\mathbf{Pr}((0, x_r, x_r)|(\Delta, x_l, x_r), 0) = p_{x_l, x_r}$.
- If $x_l' \neq x_r$, the AoII at slot $t+1$ will be $\Delta + 1$ and $\mathbf{Pr}((\Delta+1, x_l', x_r)|(\Delta, x_l, x_r), 0) = p_{x_l, x_l'}$.

2) Case 2: At the slot t, the sensor is scheduled to transmit update packet, i.e., $a_t = 1$. But the packet is failed to be received by destination with probability p_f.

- If $x_l' = x_r$, the AoII at slot $t+1$ will be 0 and $\mathbf{Pr}((0, x_r, x_r)|(\Delta, x_l, x_r), 1) = p_f p_{x_l, x_r}$.
- If $x_l' \neq x_r$, the AoII at slot $t+1$ will be $\Delta + 1$ and $\mathbf{Pr}((\Delta+1, x_l', x_r)|(\Delta, x_l, x_r), 1) = p_f p_{x_l, x_l'}$.

3) Case 3: At the slot t, the sensor is scheduled to transmit update packet, i.e., $a_t = 1$. And the packet is received by destination successfully with probability p_s, such that the remote estimation at slot $t+1$ is x_l.

- If $x_l' = x_l$, the AoII at slot $t+1$ will be 0 and $\mathbf{Pr}((0, x_l, x_l)|(\Delta, x_l, x_r), 1) = p_s p_{x_l, x_l}$.
- If $x_l' \neq x_l$, the AoII at slot $t+1$ will be $\Delta + 1$ and $\mathbf{Pr}((\Delta+1, x_l', x_l)|(\Delta, x_l, x_r), 1) = p_s p_{x_l, x_l'}$.

B. Multi-sensor scenario and problem formulation

In this subsection, we generalize the system to N sensors. Let $\boldsymbol{w} = \rho[w_1, w_2, ..., w_N]^T$ be the weighted transmission cost vector, where ρ is the weight and $w_n \geq 0$ is the transmission cost of sensor n. We denote the scheduling decision as $\boldsymbol{a}_t = [\boldsymbol{a}_t^{(1)}, \boldsymbol{a}_t^{(2)}, ..., \boldsymbol{a}_t^{(N)}]^T$ where $\boldsymbol{a}_t^{(n)} \in \{0, 1\}$ decides whether sensor n forwards a status packet at slot t . In this work, we aim to minimize the system-wide cost, i.e., long-term weighted sum of AoII and transmission cost, by searching an optimal schedule policy. The problem can be formulated as

$$\boldsymbol{Q1}: \min_{\pi \in \Pi} \lim_{T \to \infty} \frac{1}{T} \mathbb{E}_\pi \left(\sum_{t=1}^{T} \left(\|\boldsymbol{\Delta}(t)\|_1 + \boldsymbol{w}^T \boldsymbol{a}_t \right) \right), \quad (1)$$

$$s.t. \;\; \boldsymbol{a}_t^{(n)} \in \{0, 1\}, \; \forall t, \; \forall n, \quad (2)$$

where Π is the class of non-anticipated policies, namely scheduling decisions are made based on past and current information, $\mathbb{E}_\pi[\cdot]$ represents the expectation with respect to policy π and $\|\cdot\|_1$ is ℓ^1 norm of vector. The vector $\boldsymbol{\Delta}(t) = [\Delta_1(t), \Delta_2(t), ..., \Delta_N(t)]^T \in \mathbb{N}^N$ denotes the AoII of all sensors at time-slot t. Without loss of generality, we assume that all the sensors and receiver are synchronized initially, i.e, $\boldsymbol{\Delta}(0) = \mathbf{0}$.

Since there are 2^N possible scheduling actions at each time-slot, it is not possible to find out the optimal decision through exhausted search method in a mass of sensor network. Besides, due to the status-heterogeneous transitions for each sources [12], it is difficult to deal with the scheduling problem by conventional approaches, such as dynamic programming [13]. In section III, we propose the machine learning-based approach to solve problem $\boldsymbol{Q1}$.

III. A3C SCHEDULING ALGORITHM

Thanks to the successes of deep learning, the complex value function in reinforcement learning can be approximated by neural network [14]. Among which, an advanced algorithm, asynchronous advantage actor-critic (A3C) algorithm, can solve complex sequential decision-making problems efficiently. A3C is an actor-critic method which consists of a global agent and multiple independent agents with same network architecture and weights [15]. These independent agents (we call them local agents for distinction) interact with a different copy of the environment in parallel and send their own interaction information to the global network. In return, these independent local agents reset their network parameters to those of the global agent after global agent updates its network by using the interaction information from local agents. In this section, we reformulate our problem $\boldsymbol{Q1}$ as a Markov decision process (MDP). Then introduce the updates of actor and critic and clarify the scheduling decision based on the actor.

A. Markov Decision Process

In the standard reinforcement learning model, an agent interacts with an environment at each time step. At time step t, the agent feels the environment state $\boldsymbol{S}_t$, then chooses an action $\boldsymbol{a}_t$ from the set of possible actions $\mathcal{A}$ according to decision policy π which is mapping from states to actions. In return, since the environment is affected by action $\boldsymbol{a}_t$, the environment state changes to $\boldsymbol{S}_{t+1}$ at next time step, and the agent receives a reward r_t. The interaction process continuous until the environment reaches to a terminal state or the step t exceeds the given upper bound. The total accumulated reward at time step t is $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$, where discount factor $\gamma \in (0, 1]$ indicates the importance attached to the next step reward. The goal of reinforcement learning is to maximize the expectation of total accumulated reward from each time step, by making a proper policy.

At time-slot t, the environment/system state includes N sensor states, which can be formulated as $\boldsymbol{S}_t = [\boldsymbol{s}_t^{(1)}; \boldsymbol{s}_t^{(2)}; ...; \boldsymbol{s}_t^{(N)}]^T$ [2], where $\boldsymbol{s}_t^{(n)}$ is the state of sensor n consisting of AoII, source status and remote estimation. The

[2]For the sake of differentiation, we denote source status by x_l, remote estimation by x_r, sensor state by $\boldsymbol{s}$ and system state by $\boldsymbol{S}$.

state space $\mathcal{S} = \{\boldsymbol{S}\}$, which is the set of all possible system states, is countably infinite since the AoII can be arbitrarily large if the remote estimation is always incorrect. The scheduling action space at time-slot t is $\mathcal{A} = \{\boldsymbol{a}_t^{(1)}, \boldsymbol{a}_t^{(2)}, ..., \boldsymbol{a}_t^{(N)}\}$, whose cardinality $|\mathcal{A}|$ increases exponentially with the sensors number N.

We discussed the states transition probability of single sensor in section II-A, then the states transition probability of system with N sensors can be similarly expressed as $\mathbf{P} = [\boldsymbol{P}_1, \boldsymbol{P}_2, ..., \boldsymbol{P}_N]$, where $\boldsymbol{P}_n$ is the states transition probability of sensor n. Furthermore, for given state $\boldsymbol{S}_t$ and action $\boldsymbol{a}_t$, the one-step reward is

$$r_t = -\left(\|\boldsymbol{\Delta}(t+1)\|_1 + \boldsymbol{w}^T \boldsymbol{a}_t\right),$$

where the negative sign is adopted since $\boldsymbol{Q1}$ is a minimization problem.

B. The Critic and Actor Part

In A3C algorithm, an agent includes an actor and a critic. The critic is a deep neural network (DNN) that approximates the state-value function $V^{\pi}(\boldsymbol{S}_t)$ and the estimation of state-value is written as $V(\boldsymbol{S}_t; \boldsymbol{\theta}_v)$, where $\boldsymbol{\theta_v}$ is the parameters of global critic network. Therefore, the critic network inputs the system states $\boldsymbol{S}_t$ and predicts its state-value. Besides, the loss function of critic is formulated as

$$L(\boldsymbol{\theta}_v) = \left(r_t + \gamma V(\boldsymbol{S}_{t+1}; \boldsymbol{\theta}_v) - V(\boldsymbol{S}_t; \boldsymbol{\theta}_v)\right)^2,$$

where $\boldsymbol{\theta}$ is the parameters of actor network and $(r_t + \gamma V(\boldsymbol{S}_{t+1}; \boldsymbol{\theta}_v) - V(\boldsymbol{S}_t; \boldsymbol{\theta}_v))$ is called the one-step temporal difference (TD) error (or advantage function hereinafter).

Nevertheless, the learning process is slow since one-step methods obtains the reward r_t which only directly affects the value of the state $\boldsymbol{S}_t$ but not other states. [16] proposed a fast learning way in the Q-learning, i.e., $n-$step TD method, which is also adopted in A3C algorithm [15]. Furthermore, n-step A3C algorithm operates in the forward view, rather than the more common backward view such as eligibility traces. In consequence, the n-step TD error can be calculated as

$$TD_{error} = \sum_{i=0}^{n-1} \gamma^i r_{t+i} + \gamma^n V(\boldsymbol{S}_{t+n}; \boldsymbol{\theta}_v) - V(\boldsymbol{S}_t; \boldsymbol{\theta}_v),$$

where $\sum_{i=0}^{n-1} \gamma^i r_{t+i} + \gamma^n V(\boldsymbol{S}_{t+n}; \boldsymbol{\theta}_v)$ is called the n-step target state-value. The critic network is updated after every t_m actions and we denote the beginning steps of each update by t_s. Thus the loss function can be formulated as the mix of square of n-step TD error:

$$L(\boldsymbol{\theta}_v) = \sum_{t=t_s}^{t_s+t_m-1} \Big(\sum_{i=0}^{t_s+t_m-1-t} \gamma^i r_{t+i} + \gamma^{t_s+t_m-t} V(\boldsymbol{S}_{t_s+t_m}; \boldsymbol{\theta}_v) - V(\boldsymbol{S}_t; \boldsymbol{\theta}_v) \Big)^2, \quad (3)$$

which is the sum of multiple square of n-step TD error and n varies from 1 to t_m. Hence, the update of parameters $\boldsymbol{\theta}_v$ follows:

$$\boldsymbol{\theta}_v \leftarrow \boldsymbol{\theta}_v + \alpha_v \nabla_{\boldsymbol{\theta}_v} L(\boldsymbol{\theta}_v), \quad (4)$$

where α_v is the learning rate of the critic.

The actor is also a DNN which outputs an action on the basis of the current system state, namely approximates policy $\pi(\boldsymbol{a}|\boldsymbol{S}; \boldsymbol{\theta})$. Nevertheless, the action space is very large (the cardinality of $\mathcal{A}$ grows exponentially with N) as mentioned above, it is infeasible to set a large number of neurons corresponding to each scheduling action in the output layer. We tackle this issue by using sigmoid activation function with N neurons in the output layer. In sigmoid function, the output y and input x are related by $y = \frac{1}{1+\exp(-x)}$. Therefore, the result of each neuron in the output layer are located in the interval $(0, 1)$, which is the probability that the corresponding sensor is scheduled to forward source status packet. We denote the result of actor by proto-action $\hat{\boldsymbol{a}}_t$ and then sensors can be scheduled to forward status packets (after sampling the local source) according to probability $\hat{\boldsymbol{a}}_t$. We formulate the actual scheduling action at time-slot t as $\tilde{\boldsymbol{a}}_t$:

$$g: \ \hat{\boldsymbol{a}}_t \rightarrow \{\tilde{\boldsymbol{a}}_t | \tilde{\boldsymbol{a}}_t \in \{0, 1\}^N\}, \quad (5)$$

where g is a mapping from set $\mathbb{R}^N$ to set $\{0, 1\}^N$.

Actor updates network parameters according to the state-value estimated by critic. As elaboration in [15], the updating gradient is related to advantage function, i.e., $A(\boldsymbol{S}_t, \boldsymbol{a}_t; \boldsymbol{\theta}, \boldsymbol{\theta}_v) = \sum_{i=0}^{n-1} \gamma^i r_{t+i} + \gamma^n V(\boldsymbol{S}_{t+n}; \boldsymbol{\theta}_v) - V(\boldsymbol{S}_t; \boldsymbol{\theta}_v)$. For notation brevity, we rewrite $A(\boldsymbol{S}_t, \boldsymbol{a}_t; \boldsymbol{\theta}, \boldsymbol{\theta}_v) = \sum_{i=0}^{t_s+t_m-1-t} \gamma^i r_{t+i} + \gamma^{t_s+t_m-t} V(\boldsymbol{S}_{t_s+t_m}; \boldsymbol{\theta}_v) - V(\boldsymbol{S}_t; \boldsymbol{\theta}_v)$, where t varies from t_s to t_s+t_m-1 for each parameters update. Similar to critic network, the updating gradient is compose of mix of multiple n-step TD error, which is expressed as:

$$\sum_{t=t_s}^{t_s+t_m-1} \Big(\nabla_{\boldsymbol{\theta}} \mathbb{E}(R_t) \approx A(\boldsymbol{S}_t, \boldsymbol{a}_t; \boldsymbol{\theta}, \boldsymbol{\theta}_v) \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{a}_t | \boldsymbol{S}_t; \boldsymbol{\theta}) \Big). \quad (6)$$

In order to explore, the entropy can be added to the objective function in A3C algorithm and the update of actor network is performed as:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \underbrace{\sum_{t=t_s}^{t_s+t_m-1} \Big(\alpha \nabla_{\boldsymbol{\theta}} \mathbb{E}(R_t) + \beta \nabla_{\boldsymbol{\theta}} H(\pi(\boldsymbol{a}_t | \boldsymbol{S}_t; \boldsymbol{\theta})) \Big)}_{U(\boldsymbol{\theta})}, \quad (7)$$

where $H(\pi(\boldsymbol{a}_t|\boldsymbol{S}_t; \boldsymbol{\theta}))$ is the entropy of the policy π. This can avoid suboptimal solutions by improving the exploration ability. Besides, α is the learning rate of the actor and β is the weight of entropy regulation term.

IV. Numerical Analysis and Simulation

In this section, we first verify the effectiveness of A3C algorithm by comparing it with the optimal theoretical result in a status-homogeneous system, and then investigate the performance of A3C algorithm by comparing it against benchmark in a status-heterogeneous system which conforms more to the practical IIoT scenarios.

We use the conventional AoI minimization as the benchmark where the status transitions of sources monitored by sensors

are irrelevant, and denote the AoI of sensor n at time-slot t by $\delta_n(t)$. Therefore, the AoI minimization problem of N sensors can be decoupled into the scheduling problem of single sensor. If $a_t^{(n)} = 1$, the AoI at next time is 1 with probability $p_s^{(n)}$ and $\delta_n(t)+1$ with probability $1-p_s^{(n)}$, where $p_s^{(n)}$ is the probability that the data of sensor n is received successfully by remote server, and the one-step cost is $C_1 = p_s^{(n)} + (\delta_n(t)+1)(1-p_s^{(n)}) + w_n$; If $a_t^{(n)} = 0$, the one-step cost is $C_0 = \delta_n(t)+1$. By comparing the one-step cost under being scheduled and not scheduled, we can know

$$C_1 - C_0 = w_n - \delta_n(t)p_s^{(n)}, \tag{8}$$

which is a non-increasing function of $\delta_n(t)$. Hence, from the AoI perspective, action $a_t^{(n)} = 1$ is better than action $a_t^{(n)} = 0$ when $\delta_n(t) \geq \frac{w_n}{p_s^{(n)}}$, which is consistent with our intuition that the sensor should forward its update packets when AoI is too large. We call this scheme as *AoI-optimal* method and regard it as the baseline.

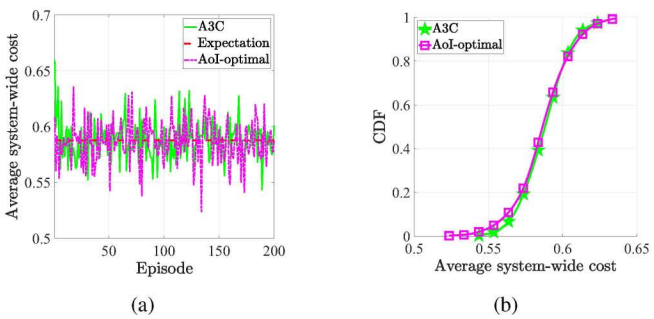
In A3C Algorithm, the actor network has one input layer that uses $3N$ neurons containing the source statuses, remote estimations and AoII of N sensors, two hidden layers that use 64 neurons with ReLU activation function, one hidden layer that uses 32 neurons with ReLU activation function, and one output layer that uses N neurons with sigmoid activation function. The critic network has one input layer that uses $3N$ neurons, three hidden layers that use ReLU activation function with 64, 32, 16 neurons respectively, and one output layer that uses one neurons with linear activation function. Moreover, the learning rates of actor and critic are both set as 0.0005 and the discount factor $\gamma = 0.9$. The agent is updated after every $t_m = 5$ actions and the agent interacts $E_{max} = 200$ episodes. Since the time-slot in problem $\boldsymbol{Q1}$ can be infinite, it is needed to truncate the time-slot to t_{max} and we set that each episode experiences $t_{max} = 10^4$ time-slots. In the following, we first consider the system has homogeneous source statuses (with single sensor or multiple sensors), such that theoretical results can be deduced [8]. Then we investigate the heterogeneous statuses multi-sensor system (no theoretical results).

A. Status-homogeneous single sensor system

In this subsection, we present the numerical analysis to validate the effectiveness of A3C algorithm by comparing with the minimal expectation of AoII [8]. We consider the status-homogeneous single sensor system with transmission cost $\boldsymbol{w} = [0]$ and 4 statuses ($\{0, 1, 2, 3\}$). Two different status transition probability matrices R_1 and R_2 are investigated respectively, which are

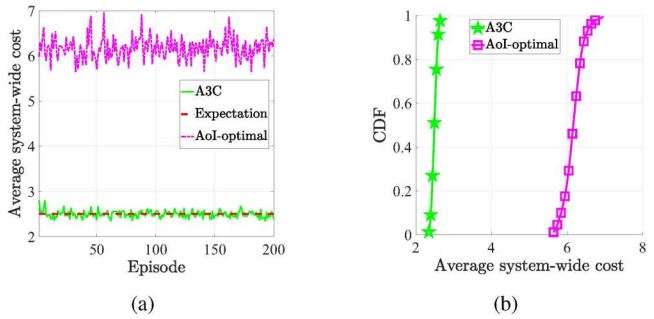
$$R_1 = \begin{bmatrix} 0.7 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.7 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.7 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.7 \end{bmatrix} \text{ and } R_2 = \begin{bmatrix} 0.1 & 0.3 & 0.3 & 0.3 \\ 0.3 & 0.1 & 0.3 & 0.3 \\ 0.3 & 0.3 & 0.1 & 0.3 \\ 0.3 & 0.3 & 0.3 & 0.1 \end{bmatrix},$$

where the transition probability in R_1 from each status to itself is larger than the one from each status to other status (i.e., $0.7 > 0.1$), and the transition probability in R_2 are opposite (i.e., $0.1 \leq 0.3$). The homogeneous statuses means that the transition probability matrix is symmetric, such as R_1 and R_2. Besides, the probability of successful transmission is set to 0.8. The initial source status and initial remote estimation are both set as status 0 and hence the initial AoII is 0. [8, Theorem 1] points that the optimal transmission policy is that the transmitter should send update packet at each time-slot or when the remote estimation is incorrect if the transition probability is R_1, and the transmitter never transmit any packet if the transition probability is R_2. Besides, the expectation of long-term average AoII with transition probability R_1 is calculated as 0.58777 [8, eq. (15)] and the one with transition probability R_2 is evaluated as 2.5 [8, eq. (16)].



(a) (b)

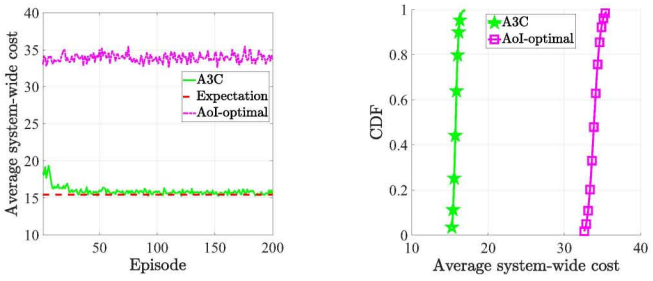
Fig. 4. $N = 1$ and transition probability is R_1. (a) The average system-wide cost versus episode. (b) The CDF of average system-wide cost.



(a) (b)

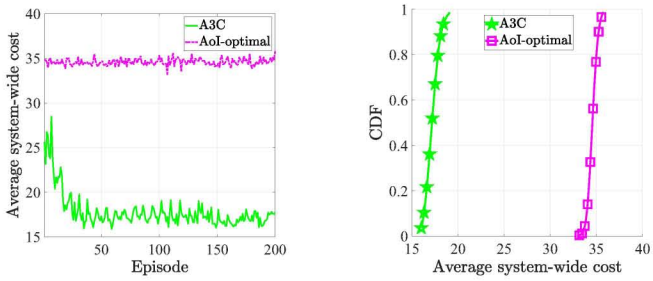
Fig. 5. $N = 1$ and transition probability is R_2. (a) The average system-wide cost versus episode. (b) The CDF of average system-wide cost.

As for systems with transition probability R_1 and R_2, the results of A3C algorithm are illustrated in Fig. 4 and Fig. 5, respectively. Fig. 4(a) and 5(a) show the system-wide cost versus episode. The system-wide cost solved by A3C algorithm are approximately equal to the theoretical expectation of average AoII. Besides, we select the convergent results to generate the cumulative distribution function (CDF). As depicted in Fig. 4(b) and Fig. 5(b), the values of CDF being 0.5 correspond to average system-wide cost at 0.58777 and 2.5, respectively. Furthermore, as for system with transition probability R_1 in Fig. 4(b), the AoI-optimal method can acquire the average system-wide cost whose expectation is also 0.58777, since the best policy is keeping updating. Nevertheless, the results of AoI-optimal are much larger than A3C algorithm in Fig. 5(b).



(a) (b)

Fig. 6. $N = 10$ and each source statuses are homogeneous. (a) The average system-wide cost versus episode. (b) The CDF of average system-wide cost.



(a) (b)

Fig. 7. $N = 10$ and each source statuses are heterogeneous. (a) The average system-wide cost versus episode. (b) The CDF of average system-wide cost.

B. Status-homogeneous multi-sensor system

Furthermore, we investigate a multi-sensor system where there are 5 sensors with transition probability R_1 and 5 sensors with transition probability R_2. The probabilities of success transmission of all sensors are 0.8 and the transmission costs of all sensors are 0. As mentioned in section IV-A, the optimal policy is that the transmitter always forwards the update packets of sensors with source transition probability R_1 and never transmits the packets of other sensors. Then the expectation of system-wide cost is calculated as 15.4389. Fig. 6 shows that A3C algorithm almost achieves the Expectation and the AoI-optimal method is far worst than Expectation.

C. Status-heterogeneous multi-sensor system

Since the most source statuses monitored by sensors are heterogeneous in the real IIoT, the more general system should be studied. We consider two heterogeneous transition probability matrices:

$$R_3 = \begin{bmatrix} 0.3 & 0.1 & 0.2 & 0.4 \\ 0.5 & 0.2 & 0.2 & 0.1 \\ 0.2 & 0.2 & 0.2 & 0.4 \\ 0.6 & 0.1 & 0.2 & 0.1 \end{bmatrix} \text{ and } R_4 = \begin{bmatrix} 0.4 & 0.4 & 0.1 & 0.1 \\ 0.2 & 0.2 & 0.3 & 0.3 \\ 0.3 & 0.4 & 0.2 & 0.1 \\ 0.7 & 0.1 & 0.1 & 0.1 \end{bmatrix}.$$

The system includes one sensor with transition probability R_1, one sensor with transition probability R_2, 4 sensors with transition probability R_3 and 4 sensors with transition probability R_4. Besides, the transmitting cost is $\boldsymbol{w} = [0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5]$, the probability of success communication is $\boldsymbol{p}_s = [0.8, 0.8, 0.8, 0.8, 0.8, 0.7, 0.7, 0.7, 0.7, 0.7]$. As depicted in Fig. 7(b), 50% average system-wide cost are less than 17.1773 and 90% system-wide cost are less than 18.1941 in A3C algorithm, which is much less than the result from AoI-optimal method.

V. Conclusions

In this paper, we studied the problem of AoII minimization in IIoT by leveraging DRL algorithm. We first modeled the multi-sensor scheduling problem as an MDP and then apply A3C algorithm to tackle the problem with large state and large action space. The simulation results show that A3C algorithm can achieve almost the optimal system-wide cost. Furthermore, from the AoII perspective, the conventional AoI-optimal method is not effective in most cases since it ignores the source statuses and remote estimation, which causes unnecessary updates. As a future work, we will extend this work to more realistic scenarios where the system is subject to physical resource constraints.

References

[1] L. D. Xu, W. He, and S. Li, "Internet of things in industries: A survey," *IEEE Trans. Ind. Informat.*, vol. 10, no. 4, pp. 2233–2243, Nov. 2014.
[2] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial internet of things: Challenges, opportunities, and directions," *IEEE Trans. Ind. Informat.*, vol. 14, no. 11, pp. 4724–4734, Nov. 2018.
[3] S. Kaul, M. Gruteser, V. Rai, and J. Kenney, "Minimizing age of information in vehicular networks," in *IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, Jun. 2011, pp. 350–358.
[4] S. K. Kaul, R. D. Yates, and M. Gruteser, "Real-time status: How often should one update?" in *Proc. IEEE INFOCOM*, Orlando, FL, USA, Mar. 2012, pp. 2731–2735.
[5] J. Li, J. Tang, and Z. Liu, "On the data freshness for industrial Internet of Things with mobile edge computing," *IEEE Internet Things J.*, vol. 9, no. 15, pp. 13 542–13 554, Aug. 2022.
[6] J. Zhong, R. Yates, and E. Soljanin, "Two freshness metrics for local cache refresh," in *Proc. IEEE ISIT*, Vail, Colorado, USA, Jun. 2018, pp. 1924–1928.
[7] Y. Sun, Y. Polyanskiy, and E. Uysal-Biyikoglu, "Remote estimation of the wiener process over a channel with random delay," in *Proc. IEEE ISIT*, Aachen, Germany, Jun. 2017, pp. 321–325.
[8] A. Maatouk, S. Kriouile, M. Assaad, and A. Ephremides, "The age of incorrect information: A new performance metric for status updates," *IEEE/ACM Trans. Netw.*, vol. 28, no. 5, pp. 2215–2228, Oct. 2020.
[9] F. Peng, Z. Jiang, S. Zhou, Z. Niu, and S. Zhang, "Sensing and communication co-design for status update in multiaccess wireless networks," *IEEE Trans. Mobile Comput.*, pp. 1–1, 2021.
[10] I. Kadota, E. Uysal-Biyikoglu, R. Singh, and E. Modiano, "Minimizing the age of information in broadcast wireless networks," in *Proc. Annu. Allerton Conf. Commun. Control. Comput.*, Sep. 2016, pp. 844–851.
[11] H. Tang, J. Wang, L. Song, and J. Song, "Minimizing age of information with power constraints: Multi-user opportunistic scheduling in multi-state time-varying channels," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 854–868, May 2020.
[12] F. Peng, Z. Jiang, S. Zhou, Z. Niu, and S. Zhang, "Sensing and communication co-design for status update in multiaccess wireless networks," *IEEE Trans. Mobile Comput.*, pp. 1–1, 2021.
[13] D. P. Bertsekas, *Dynamic Programming and Optimal Control, 2nd ed.* Belmont, MA, USA: Athena Sci., 2000.
[14] S. Marsland, *Machine Learning: An Algorithmic Perspective.* Boca Raton, FL, USA: CRC Press, 2015.
[15] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," 2016. [Online]. Available: arXiv:1602.01783
[16] J. Peng and R. J. Williams, "Incremental multi-step q-learning," *Mach. Learn.*, vol. 22, no. 1-3, pp. 283–290, 1996.