Federated Deep Reinforcement Learning for Task Scheduling in Heterogeneous Autonomous Robotic System

Tai Manh Ho, Kim-Khoa Nguyen, and Mohamed Cheriet,
Synchromedia Lab, École de Technologie Supérieure, Université du Québec, QC, Canada
Email: manh-tai.ho.1@ens.etsmtl.ca;kim-khoa.nguyen@etsmtl.ca;mohamed.cheriet@etsmtl.ca

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10000980

***Abstract*—In this paper, we investigate the problem of task scheduling in automated warehouses with heterogeneous autonomous robotic systems. We formulate the task scheduling for a heterogeneous autonomous robots (HAR) system in each warehouse as a queueing control optimization problem in which we aim to minimize the queue length of tasks that are waiting to be processed. We propose a deep reinforcement learning (DRL) based approach that employs the proximal policy optimization (PPO) to achieve an optimal task scheduling policy. We then propose a federated learning based algorithm to improve the performance of the PPO agents. The simulation results fully demonstrate the performance improvement of our proposed algorithm in terms of average queue length compared to the distributed learning algorithm.**

***Index Terms*—industrial automation, federated deep reinforcement learning**

I. Introduction

In the smart warehouses, the culmination of warehouse automation and smart logistics, robotics play an indispensable role in attaining efficient automation solutions. Smart logistics includes the intelligent organization, planning, control, and execution of goods/items flow in the warehouse. Furthermore, the innovation in wireless communications and battery technologies prolongs the serving time of robotics; thus, replacing human workers with the robotic system can reduce labor costs, enhance warehouse working efficiency, and improve reliability [1].

Cloud robotics has emerged as a new technology for path planning and task scheduling in robotic systems [2]. It enables robotic systems to be endowed with powerful capability whilst reducing costs through cloud technologies. For efficient utilization of robotic resources in a smart warehouse, it is necessary to develop a powerful and adaptive algorithm that can efficiently allocate and schedule the robotics' resources for task management.

The task scheduling problem in warehouse has been well studied for decades [3]–[6]. In [3] a protocol is developed to address vehicle blocking, and a semi-open queueing network model is proposed to analyze system performance and evaluate design trade-offs in an autonomous vehicle-based storage and retrieval system. In [4], the authors proposed an efficient heuristic algorithm for assigning items to pods in a robotic mobile fulfillment system (RMFS). A parametric heuristic model for order task selection process for a RMFS was proposed in [5] within a realistic simulation environment. A soft actor-critic and hierarchical deep reinforcement learning based algorithm was proposed in [6] for multi-logistic robot task allocation.

978-1-6654-3540-6/22 © 2022 IEEE

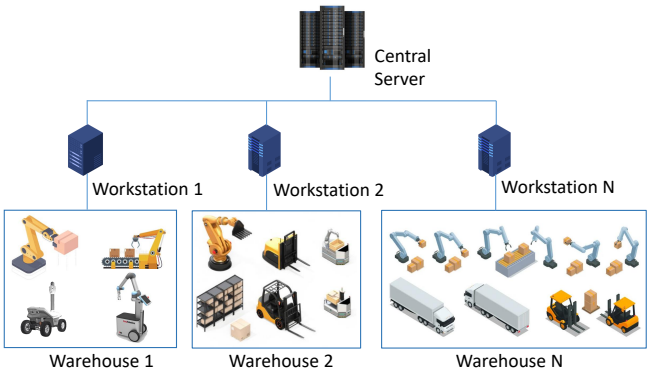

Figure 1: Illustration of a warehouse management system.

In a heterogeneous autonomous robotic system, the traditional task scheduling approaches are inefficient to tackle the stochastic nature of the goods flow and the heterogeneous characteristic of the multiple types of autonomous mobile robots. Thanks to its outstanding performance in handling stochastic decision-making problems, deep reinforcement learning (DRL) has been shown as an effective technique in developing adaptive algorithms in various domains [6], [7].

Leveraging the robustness of the federated learning (FL) technique [8], in this paper, we propose a federated DRL-based scheme that achieves an optimal task scheduling policy and improves the performance of the system compared to the distributed learning framework.

II. System Model and Problem Formulation

We consider an e-commerce company consisting of a set $\mathcal{M}$ of M warehouses which are geographically distributed as shown in Fig. 1. Each warehouse is with a set $\mathcal{K}_m$ of K_m heterogeneous autonomous robots (HAR) that perform the tasks, i.e., storage and retrieval items in the

Authorized licensed use limited to: PES University Bengaluru. Downloaded on June 15,2023 at 05:18:17 UTC from IEEE Xplore. Restrictions apply.

warehouse. The robots are heterogeneous with different mobility capabilities, i.e., automated guided vehicles (AGVs), autonomous mobile robots (AMRs), collaborative mobile robots (cobots). These mobile robots can autonomously navigate through a warehouse using sensors, cameras and safety mechanisms to build a digital map of their environment and freely move around the warehouse without external guidance. The warehouse management system (WMS) feeds tasks to the robots via wireless communication, i.e., a 5G wireless network, and the digital mapping provides all the information they need to accomplish their tasks. We assume that each warehouse is operated by a WMS implemented in a workstation. All the workstations are connected to a central server of the company.

Within each warehouse, the items are stored on the racks that are arranged back-to-back, and a set of racks forms a block of racks. To facilitate robot movements to the racks and reduce the blockage of the robots in a large-scale warehouse, each block of racks is surrounded by aisles as shown in Fig 2. There are a picking station and a delivery station at the corners of the warehouse where the items arrive and depart from the warehouse.

Each warehouse operates as follows: when an item or an order arrives, a corresponding storage or retrieval task is requested. If a storage task is scheduled for a robot, the robot has to pick up the item at the picking station and stores the item at a predefined location on the racks. If a robot is assigned a retrieval task, the robot will travel to the location where the item is stored to bring the item to the delivery station via the shortest path. A robot can be anyplace inside the warehouse and ready for a new task if it is free, i.e., in an idle state after finishing a task. We consider single-command operations in which each robot only performs a single task at every time slot.

The task scheduling process in a warehouse can be modeled as a queueing system. An arrival task (storage or retrieval task) will be entered in a queue of tasks and waiting to be serviced. The queue is emptied according to a first-come-first-serve (FCFS) discipline, and there is no preemptive priority between storage and retrieval tasks. We assume that the storage and retrieval tasks arrive according to the Poisson process. The location of each item is predefined following a discretely uniform distribution. Each robot in a warehouse acts as a server. The mobility patterns of the robots are different, hence the service times of the robots are different. The service time of a robot is the sum of the travel time of the robot to reach the location of the item and the time to pick and place the item. The service time of a robot is typically not exponentially distributed. Therefore, we consider a general distribution for the service times of the robots.

A task scheduling queueing system with Poisson arrival, general service time and multiple servers can be characterized as an $M/G/K$ queue, where M stands for the memoryless Poisson arrival, G represents general service time, and K is the number of robots employed.

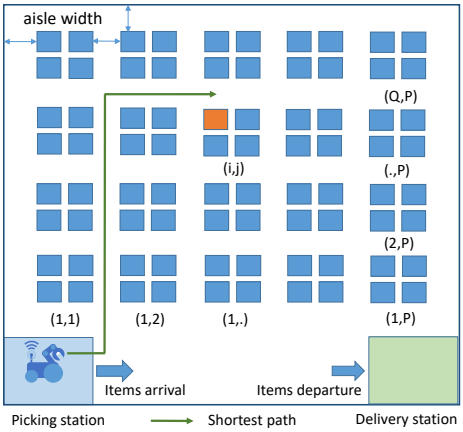

Figure 2: Illustration of a warehouses.

Unfortunately, the exact expressions for mean queue length and waiting time in an $M/G/K$ queue are not available. However, numerous approximations do exist. The first approximation for the mean waiting time for an $M/G/K$ queue was given by Lee and Longton [9] as follows:

$$\mathrm{E}[W^{M/G/K}] \approx \left(\frac{C^2+1}{2}\right)\mathrm{E}[W^{M/M/K}], \tag{1}$$

where $\mathrm{E}[W^{M/M/K}]$ is the mean waiting time with exponentially distribution with the same mean service time $\mathrm{E}[X]$ as in the $M/G/K$ system. $C^2 = var(X)/(\mathrm{E}[X])^2$ is the squared coefficient of variation (SCV) of service time variable X.

The expected service time of any robot is the sum of the expected time to travel to the picking station, to the location of the item, and/or to the delivery station. The travel time of the robots may involve some additional time due to the congestion or blocking by other robots. However, the phenomenon of the robots' congestion is very sophisticated and difficult to model mathematically. Therefore, in this paper, we only consider the expected travel time of the robots which is calculated by the expected travel distance via the shortest path. The shortest path from the picking station to the block (i, j) and the shortest path from the block (i, j) to the picking station or delivery station is calculated as follows:

$$d_{i,j} = i(Rl_s + a) + j(Sw_s + a) + a, \tag{2}$$

where a is the width of an aisle, l_s and w_s are the length and width of a rack, respectively. R and S are the number of racks in a row and in a column of a block, respectively. The expected travel distance of a robot is given as follows:

$$\begin{aligned}\mathrm{E}[d_{i,j}] &= \frac{1}{PQ}\sum_{i=1}^{P}\sum_{j=1}^{Q} d_{i,j} \\ &= \frac{1}{PQ}\sum_{i=1}^{P}\sum_{j=1}^{Q}[i(Rl_s + a) + j(Sw_s + a) + a] \\ &= \frac{P+1}{2}(Rl_s + a) + \frac{Q+1}{2}(Sw_s + a) + a,\end{aligned} \tag{3}$$

Authorized licensed use limited to: PES University Bengaluru. Downloaded on June 15,2023 at 05:18:17 UTC from IEEE Xplore. Restrictions apply.

where P and Q are the number of rows and columns of blocks in the warehouse, respectively. The shortest path from block (i,j) to block (i',j') is calculated as:

$$d_{i,j\to i',j'} = |i-i'|(Rl_s+a) + |j-j'|(Sw_s+a). \quad (4)$$

The expected travel distance from block (i,j) to block (i',j') is given as:

$$\mathrm{E}[d_{i,j\to i',j'}] = \frac{P}{2}(Rl_s+a) + \frac{Q}{2}(Sw_s+a). \quad (5)$$

The expected travel time of robot k is given as follows:

$$\mathrm{E}[X_k] = \frac{1}{\bar{v}_k}\left[\mathrm{E}[d_{i,j}] + \mathrm{E}[d_{i,j\to i',j'}]\right], \quad (6)$$

where $\bar{v}_k$ is the average velocity of the robot k. The service rate of robot k is calculated by the expected travel time $\mu_k = \frac{1}{\mathrm{E}[X_k]}$.

The upper bound for mean queue length of a heterogeneous multi-server queue $M/M/K$ is given as follows:

$$\mathrm{E}[Q] = p_0 \frac{\left(\sum_{k=1}^{K}\mu_k\right)^K}{\prod_{k=1}^{K}\sum_{j=1}^{k}\mu_j}\frac{\rho^{K+1}}{(1-\rho)^2}, \quad (7)$$

where p_0 is the probability the queue is empty

$$p_0 = \left[1 + \sum_{k=1}^{K-1}\frac{\lambda^k}{\prod_{j=1}^{k}\sum_{i=1}^{j}\mu_i} + \frac{\lambda^K}{(1-\rho)\prod_{k=1}^{K}\sum_{j=1}^{k}\mu_j}\right]^{-1}, \quad (8)$$

and $\rho = \frac{\lambda}{\mu}$, $\mu = \sum_{k=1}^{K}\mu_k$, $\lambda = \lambda_I + \lambda_O$, and λ_I and λ_O are the arrival rate of storage task and retrieval task, respectively.

The service policy π_m of the queueing system in warehouse m is the task scheduling policy that assigns arrival tasks to the robots at each decision time. The objective is to find an optimal service policy π_m^* that minimize the long-run average queue length of tasks in the system

$$\min_{\pi_m} \lim_{T\to\infty}\frac{1}{T}\sum_{t=0}^{T-1}\mathrm{E}[Q_m(t)] \quad (9a)$$

$$\text{s.t. } \lambda \le \mu \quad (9b)$$

Constraint (9b) is the stable constraint for the queueing system.

The formulated problem in (9) is a stochastic long-term optimization problem which is difficult to handle by traditional optimization approaches. In the following section we propose a deep reinforcement learning approach to obtain an optimal service policy for the formulated queueing system.

III. Proximal Policy Optimization

In this section, we formulate our task scheduling process as a Markov Decision Process (MDP). Then, we propose a Proximal Policy Optimization based algorithm which is an on-policy policy gradient DRL to obtain an optimal policy for the formulated MDP.

The queue length dynamic can be formulated as follows:

$$Q_m(t+1) = \Big[Q_m(t) - I_m(t) - O_m(t)\Big]^+ + \Lambda_m^I(t+1) + \Lambda_m^O(t+1), \quad (10)$$

where $I_m(t)$ and $O_m(t)$ are the number of storage and retrieval tasks that are successfully processed at the previous decision period. $\Lambda_m^I(t+1)$ and $\Lambda_m^O(t+1)$ are the arrival number of storage and retrieval tasks at the current decision time.

The task scheduling problem at a warehouse m can be characterized as a MDP as follows:

A. System State, Action, and Reward Design

Consider an infinite-horizon discounted MDP, defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathbf{Pr}, r, \gamma)$, where $\mathcal{S}$ is a finite set of states, $\mathcal{A}$ is a finite set of actions, $\mathbf{Pr}: \mathcal{S}\times\mathcal{A}\times\mathcal{S}\to\mathbb{R}$ is the transition probability $r: \mathcal{S}\to\mathbb{R}$ is the reward function, and $\gamma\in(0,1)$ is the discount factor.

1) **State space:** The system state at time slot $t+1$ is defined by the tuple $\mathcal{S}_m = (Q_m(t), I_m(t), O_m(t), \Lambda_m^I(t+1), \Lambda_m^O(t+1), \{\mathbf{h}_{m,k}(t)\}_{k\in\mathcal{K}_m})$ in which $h_{m,k}[t]$ is the current status of robot k in warehouse m, i.e., free or busy.
2) **Action space:** An action scheduled to a robot (not currently busy) is either a task (storage or retrieval) or a command allowing the robot to be idle until next decision even if there are still waiting jobs in the queue. The action space at time $t+1$ is defined by the tuple $\mathcal{A}_m = (\{a_{m,k}(t+1)\}_{m\in\mathcal{M},k\in\mathcal{K}_m})$, indicating the task scheduling to the robots in the set $\mathcal{K}_m$ in the warehouse m at time $t+1$.
3) **Reward function:** Designing the reward function is the most challenging part of the reinforcement learning. The immediate reward at each time step should reflect the performance of the system. In the formulated problem, we aim to minimize the queue length Q_m of the waiting storage and retrieval tasks. As a result, the reward function is proportional to the number of accomplished tasks at the decision time and inverse proportional to the queue length. The reward for each agent at time t as is designed as follow:

$$r_m(t) = \kappa_1(I_m(t) + O_m(t)) + \kappa_2/Q_m(t), \quad (11)$$

where κ_1 and κ_2 are the positive coefficients to adjust the impact of the accomplished tasks and the waiting tasks to the achievable reward.

B. Proximal Policy Optimization

Proximal policy optimization (PPO) [7] is a model-free, online, on-policy, policy gradient reinforcement learning method. This algorithm is a type of policy gradient training that alternates between sampling data through environmental interaction and optimizing a clipped surrogate objective function using stochastic gradient descent (SGD).

Authorized licensed use limited to: PES University Bengaluru. Downloaded on June 15,2023 at 05:18:17 UTC from IEEE Xplore. Restrictions apply.

$$L(s_m, a_m, \theta_m^n, \theta_m) = \min\left(\frac{\pi_{\theta_m}(a_m|s_m)}{\pi_{\theta_m^n}(a_m|s_m)} A^{\pi_{\theta_m^n}}(s_m, a_m),\ \ \text{clip}\left(\frac{\pi_{\theta_m}(a|s)}{\pi_{\theta_m^n}(a|s)}, 1-\epsilon, 1+\epsilon\right) A^{\pi_{\theta_m^n}}(s_m, a_m)\right), \tag{13}$$

$$\theta_m^{n+1} = \arg\max_{\theta_m} \frac{1}{|\mathcal{D}_m^n|T} \sum_{\tau \in \mathcal{D}_m^n} \sum_{t=0}^{T} \min\left(\frac{\pi_{\theta_m}(t|s_t)}{\pi_{\theta_m^n}(a_t|s_t)} A^{\pi_{\theta_m^n}}(s_m(t), a_m(t)), g(\epsilon, A^{\pi_{\theta_m^n}}(s_m(t), a_m(t)))\right); \tag{14}$$

$$\phi_m^{n+1} = \arg\min_{\phi_m} \frac{1}{|\mathcal{D}_m^n|T} \sum_{\tau \in \mathcal{D}_m^n} \sum_{t=0}^{T} \left[V_{\phi_m^n}(s_m(t)) - r_m(s_m(t), a_m(t))\right]^2; \tag{15}$$

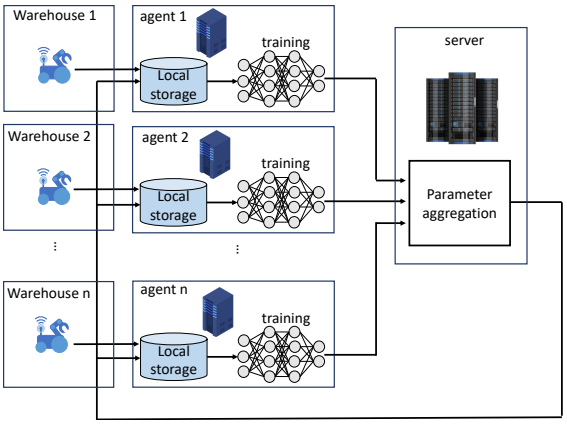

Figure 3: Federated learning framework.

PPO alternatively constructs a unconstrained surrogate objective function to remove the incentive for large policy updates. PPO updates policies of the agents by taking multiple steps of (usually minibatch) SGD to maximize the objective

$$\theta_m^{n+1} = \arg\max_{\theta_m} \mathop{\mathrm{E}}_{s,a\sim\pi_{\theta_m^n}} \left[L(s_m, a_m, \theta_m^n, \theta_m)\right], \tag{12}$$

where L is given in (13). $\pi_{\theta_m}(a_m|s_m)$ is new parameterized policy of the agent m that tries to seek the optimal parameter vector θ_m and $\pi_{\theta_m^k}(a_m|s_m)$ is the old policy. ϵ is a small hyperparameter implying how far away the new policy is allowed to go from the old one. The advantage function $A^{\pi_{\theta_m^n}}(s_m, a_m)$ can be calculated by

$$A^{\pi_{\theta_m^n}}(s_m, a_m) = Q^{\pi_{\theta_m^n}}(s_m, a_m) - V^{\pi_{\theta_m^n}}(s_m), \tag{16}$$

where $Q^{\pi_{\theta_m^n}}(s_m, a_m)$ is the action-value function estimated by samples, and $V^{\pi_{\theta_m^n}}(s_m)$ is the approximation of the state-value function.

$$Q^{\pi_{\theta_m^n}}(s_m(t), a_m(t)) = \mathrm{E}\left[\sum_{l=0}^{\infty} \gamma^l r_m(s_m(t+1))\right]. \tag{17}$$

The PPO algorithm is presented in Alg. 1.

C. Federated Learning for Model Aggregation

In this section, we adopt the Federated Averaging (FedAvg) technique [8] in which at each update round, a subset $\bar{M} \leq M$ of agents are selected and averaging at the central server for the number of epochs. Choosing the number of local epochs training for each communication round of global update is crucial in FL since it affects the convergence of the FedAvg algorithm. In our scenario, the warehouses are heterogeneous, hence, a large number of local epochs may lead each local agent towards the local optimal as opposed to the global objective, which potentially results in a divergence. However, a larger number of local epochs can reduce communication cost, which can improve the overall convergence speed in communication-constrained conditions. On the other hand, with a smaller number of local epochs, the local agent may not complete training within a given communication round and hence, could significantly reduce the global performance.

Algorithm 1 PPO algorithm for task scheduling in automated warehouse system

1: Initialize policy parameters $\{\theta_m^{(0)}\}$, initialize value function parameters $\{\phi_m^{(0)}\}$ for agent m;
2: **for** $n = 0, 1, 2, ..$ iterations **do**
3: Workstation m collects information of its warehouse including the number of items and orders in the queue, the number of the arrival items and orders at the decision time, and the status of each robot (idle or busy);
4: Workstation m schedules the tasks for the robots based on the decision of the policy network $\pi_{\theta_m}(a_m|s_m)$;
5: Collect a minibatch of D_m transitions $\mathcal{D}_m^n = \{s_i, a_i, r_i, s_{i+1}\}_{i=0:D_m-1}$;
6: Compute advantage estimates $\hat{A}(s_m(t), a_m(t))$ based on the current value function $V_{\phi_m^n}(s_m(t))$;
7: Update the policy by maximizing the PPO-clip objective in (14) where

$$g(\epsilon, A) = \begin{cases} (1+\epsilon)A & A \geq 0 \\ (1-\epsilon)A & A < 0. \end{cases}$$

8: Fit value function by regression on mean-squared error in (15)
9: **end for**

Proposition 1. *At each communication round, the global model parameters are aggregated by averaging the received*

Authorized licensed use limited to: PES University Bengaluru. Downloaded on June 15,2023 at 05:18:17 UTC from IEEE Xplore. Restrictions apply.

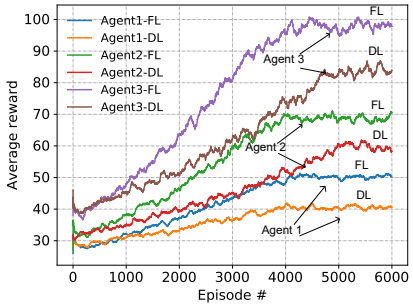

Figure 4: Reward FL vs. DL.

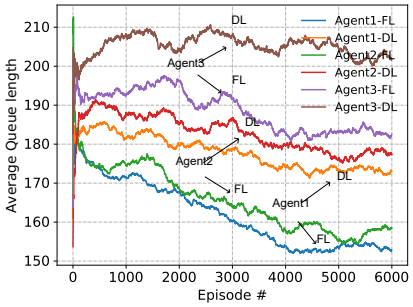

Figure 5: Queue length FL vs. DL.

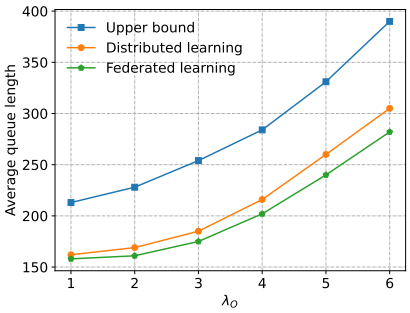

Figure 6: Upper bound

Algorithm 2 Federated Averaging for automated warehouse system

1: Initialize policy parameters $\{\theta_G^{(0)}\}$, initialize value function parameters $\{\phi_G^{(0)}\}$ for the global model;
2: Initialize number of local update epochs Ω;
3: **for** each communication round **do**
4: Central server selects a subset of $\bar{M}$ workstations at random (each workstation is chosen with probability p_m);
5: Each workstation m sends its parameters of policy network and value network $\theta_m^{(n+1)}$ and $\boldsymbol{\phi}_m^{(n+1)}$ to the central server;
6: Central server aggregates the global parameters according to (18) and sends back to all selected workstations;
7: Each workstation m updates its local parameters by the global parameters from central server, and trains the local model for Ω epochs by Algorithm 1;
8: **end for**

parameters from selected agents as follows:

$$\theta_G^{(n+1)} = \frac{1}{\bar{M}} \sum_{m \in \bar{M}} \theta_m^{(n+1)}$$
$$\boldsymbol{\phi}_G^{(n+1)} = \frac{1}{\bar{M}} \sum_{m \in \bar{M}} \boldsymbol{\phi}_m^{(n+1)}, \quad (18)$$

where $\bar{M}$ is the number of the selected agents at the communication round $n+1$

The FedAvg algorithm is presented in Alg. 2 and in Fig. 3

IV. Simulation Results

A. Simulation Setting

We consider an e-commerce company consisting of three warehouses, each with a workstation and a set of 10 robots. The system setting is given in Table II. The arrival rate of storage and retrieval task is set in the range of $[1, 6]$ tasks per minute. The location of each task is uniformly distributed within the blocks of racks in each warehouse. To train the agents, the arrival rates of the tasks (storage and retrieval task) are set to 3, 4, and 5 tasks per minute in the first, second, and third warehouse, respectively. The maximum speed of the robots in the first, second, and third warehouses are set to 1.5, 2.5, and 3 m/s, respectively.

Table I: PPO hyperparameters setting

Parameter	Value
Policy network	128, tanh, 128, tanh, 128, tanh
Value network	128, tanh, 128, tanh, 128, linear
Learning rate	$1e-4$
Batch size	20
Discount factor	0.995
ϵ clip	0.1

Table II: Simulation parameters

Parameter	Value
Number of warehouses	3
Number of robots in each warehouse	10
Width of an aisle	2 m
Length of a rack	2.4 m
Width of a rack	1.2 m
Number of racks in a row of a block	2
Number of racks in a column of a block	2
Number of rows of blocks	10
Number of column of blocks	20

To evaluate the performance of our proposed algorithm (denotes as 'FL'), a distributed learning algorithm (denotes as 'DL') is presented as the baseline for comparison. In the 'DL' algorithm, each workstation trains its PPO agent model with its own local experiences independently. There is neither interaction nor information exchange between the workstations.

B. Results Analysis

Fig. 4 plots the convergence of the accumulative rewards of our proposed 'FL' algorithm and the 'DL' baseline for three agents in the warehouses. It can be seen that the reward functions of both 'FL' and 'DL' algorithms converge gradually according to the increase of the episodes to a stable value after around 4000 to 6000 training episodes. The PPO algorithm is on-policy which has a higher variance and less sample efficient than the off-policy method, therefore it will need more samples to converge to a good solution. Moreover, we can observe that the proposed 'FL' algorithm always achieves a higher reward than that of the 'DL' baseline for three agents. This result demonstrates

Authorized licensed use limited to: PES University Bengaluru. Downloaded on June 15,2023 at 05:18:17 UTC from IEEE Xplore. Restrictions apply.

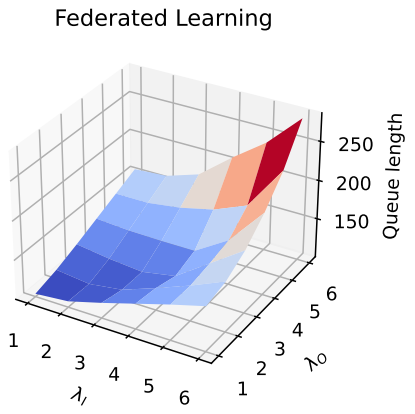

Figure 7: FL Queue vs. λ_I and λ_O

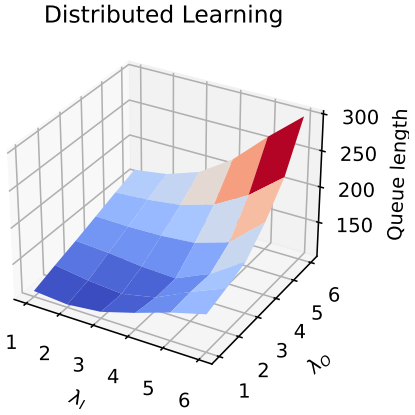

Figure 8: DL Queue vs. λ_I and λ_O

the efficiency and robustness of the proposed 'FL' algorithm in improving the performance of the traditional distributed algorithm.

Fig. 5 shows the convergence behavior of the mean queue length, i.e., our objective function. It can be observed that the average queue lengths of both 'FL' and 'DL' algorithms decrease gradually to a stable value presenting the effectiveness of the training process. This convergence suggests that the proposed 'FL' algorithm converges to an optimal policy. Moreover, the 'FL' algorithm always converges to a lower queue length than that of the 'DL' algorithm. This result again shows that our proposed algorithm significantly outperforms the 'DL' algorithm.

In Fig. 6, we compare the performance of our proposed 'FL' algorithm and the 'DL' baseline with the upper bound of the mean queue length analyzed in Section II. As shown, our proposed 'FL' algorithm can achieve an average queue length which is much smaller than the upper bound, and outperforms the 'DL' baseline.

Fig. 7 and Fig. 8 depict the average queue length with the variation of the arrival rates of the storage λ_I and retrieval task λ_O. It is obviously seen that with the fixed service rate, i.e., fixed capabilities of the robots, the queue length increases when the arrival rates of the tasks increase. The queue length increases gradually when the arrival rate is relatively small, i.e., less than 4 tasks per minute, but increases rapidly when the arrival rate exceeds 4 tasks per minute. When the arrival rate exceeds the service rate, the robots cannot keep up with the arrival tasks, and the queue length increases without bound. Furthermore, we can observe that when the arrival rate is small, the queue lengths of the 'FL' and 'DL' algorithms are identical, but when the arrival rate increases the 'FL' algorithm can achieve a lower queue length than the 'DL' algorithm, which is similar to the result depicted in Fig. 5.

V. Conclusion

In this paper, we have proposed a federated DRL-based scheme for task scheduling in a heterogeneous autonomous robotic system. The problem of task scheduling is formulated as a queueing control problem. In the proposed scheme, each warehouse in the system is operated by a PPO agent implemented in a workstation to schedule tasks for the robots in the warehouse, and a central server acts as a global model aggregator. The simulation results show that our proposed algorithm achieves a performance much lower than the upper bound in terms of average queue length and outperforms the distributed learning scheme in which each agent is trained independently with its own local experience and without information exchange with other agents.

Acknowledgment

The authors thank Mitacs, Ciena, and ENCQOR for funding this research under the grant IT13947

References

[1] J. Wen, L. He, and F. Zhu, "Swarm robotics control and communications: imminent challenges for next generation smart logistics," *IEEE Commun. Mag.*, vol. 56, no. 7, pp. 102–107, 2018.

[2] W. Chen, Y. Yaguchi, K. Naruse, Y. Watanobe, K. Nakamura, and J. Ogawa, "A study of robotic cooperation in cloud robotics: Architecture and challenges," *IEEE Access*, vol. 6, pp. 36 662–36 682, 2018.

[3] D. Roy, A. Krishnamurthy, S. S. Heragu, and C. J. Malmborg, "Blocking effects in warehouse systems with autonomous vehicles," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 2, pp. 439–451, 2013.

[4] H.-J. Kim, C. Pais, and Z.-J. M. Shen, "Item assignment problem in a robotic mobile fulfillment system," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 4, pp. 1854–1867, 2020.

[5] A. Bolu and Ö. Korçak, "Adaptive task planning for multi-robot smart warehouse," *IEEE Access*, vol. 9, pp. 27 346–27 358, 2021.

[6] H. Tang, A. Wang, F. Xue, J. Yang, and Y. Cao, "A novel hierarchical soft actor-critic algorithm for multi-logistics robots task allocation," *IEEE Access*, vol. 9, pp. 42 568–42 582, 2021.

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

[8] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[9] A. Lee and P. Longton, "Queueing processes associated with airline passenger check-in," *Journal of the Operational Research Society*, vol. 10, no. 1, pp. 56–71, 1959.

Authorized licensed use limited to: PES University Bengaluru. Downloaded on June 15,2023 at 05:18:17 UTC from IEEE Xplore. Restrictions apply.