Proc. of the 8th Intl. Symp. on Reliability Engineering and Risk Management (ISRERM 2022)
Edited by Michael Beer, Enrico Zio, Kok-Kwang Phoon, and Bilal M. Ayyub
 Published by Research Publishing, Singapore.
doi: 10.3850/978-981-18-5184-1_MS-08-062-cd

Maintenance Decision-making for Infrastructure Systems Using Clustering-based Cooperative Multi-Agent Deep Q-Network

D. Lee[1], J. Song*,[1]

[1]*Department of Civil and Environmental Engineering, Seoul National University, Republic of Korea*

Abstract

As infrastructure systems such as transportation and water distribution networks deteriorate due to aging and corrosion, decision-makers need to assess the system-level risk to devise an appropriate operation plan to minimize the losses caused by system failure over the lifecycle. Recently, Markov Decision Processes (MDP) has been utilized to identify optimal decision-making policies efficiently. However, in complex systems consisting of many components, it is often intractable to find the best solutions because the numbers of state and action spaces increase exponentially. To overcome the curse of dimensionality, this study develops a multi-agent deep reinforcement learning framework termed Clustering-based Cooperative Multi-Agent (CCMA) Deep Q-Network. CCMA takes a divide-and-conquer strategy, which identifies multiple subsystems by clustering and assigns an agent to each subsystem. Each agent observes states of the structures within the corresponding cluster to pursue appropriate actions and share information about the cluster with other agents. A numerical example demonstrates that the proposed method outperforms conventional maintenance schemes and subsystem-level optimal policies.

Keywords: Deep reinforcement learning, Infrastructure system, Life-cycle cost, Markov decision process, Operation and maintenance

1. Introduction

Components in civil infrastructure systems, such as pipelines and electric wires deteriorate over their life-cycle due to corrosion of materials and other environmental factors, thereby affecting the usability and serviceability of the systems. As the performance of a structure is in decline, not only does the cost of maintenance and repair increase, but also structure failures can cause a system-level failure event (e.g., power outage or disruption). As these infrastructure system failures can cause enormous socio-economic damage, apposite operation and maintenance (O&M) policies are required.

For O&M of infrastructures, various maintenance strategies have been developed: from time-based maintenance (TBM), in which maintenance is undertaken based on predetermined repair time-intervals, to condition-based maintenance (CBM), in which preventive maintenance is performed before the failure occurrences based on the information acquired or interpreted in various forms [1]. Since TBM does not require monitoring or criteria evaluation, the cost for observation and analysis can be saved. However, TBM cannot cope with the events that affect the lifespan of structures, such as natural disasters. In contrast, CBM shows superior performance in the long term despite some additional costs for monitoring.

As these conventional schemes are considered at a single-component level, it is difficult to extend them to decision-making procedures to handle system failures. No matter how well defined, the component-level optimality is not sufficient to ensure the system-level optimality [2]. Therefore, decision-makers should establish an O&M policy considering both the component- and system-level conditions. A Markov decision process (MDP) is a preferred option for system O&M owing to its capabilities of stochastic decision optimization [3]. However, there is a fundamental limit in direct application of the MDP to system-level policy establishment because of the *curse of dimensionality*, in which the state and action spaces increase exponentially in proportion to components within systems.

Recent studies investigated and employed a decision-making framework using deep reinforcement learning (DRL), termed Deep Q-Network (DQN), to tackle MDP environments with large numbers of states and actions [4], [5]. Recently, the field of application of DRL has been expanded and now includes life-cycle optimization of infrastructure systems [2], [6], [7]. However, these studies still cannot overcome the curse of dimensionality in real-scale infrastructure systems.

To address this issue, in this study, we propose a Clustering-based Cooperative Multi-Agent (CCMA) Deep Q-Network, in which each agent independently provides an optimal life-cycle policy in the assigned subsystem and communicates system information with other agents to avoid local optima. To this end, some of the penalty for system-level events (e.g., disconnection or flow reduction) is assigned to individual subsystems, which help the independently operating agents to find the system-level optimal policy.

This study is organized as follows. Section 2 explains the MDP concept as a background for deep reinforcement learning and introduces DQN. Section 3 introduces multi-agent reinforcement learning and proposes the CCMA algorithm. The algorithm is demonstrated and tested by a numerical example in Section 4. Finally, Section 5 provides a summary and concluding remarks.

2. Deep reinforcement learning

2.1 Markov decision process

A Markov decision process (MDP) is a mathematical framework to represent and solve stochastic decision-making problems. A Markov decision process is a 5-tuple $(\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{T}, \boldsymbol{R}, \gamma)$, where $\boldsymbol{S}$ is a set of states, $\boldsymbol{A}$ is a set of actions, $\boldsymbol{T}: \boldsymbol{S} \times \boldsymbol{A} \rightarrow \mathbb{R}^{|\boldsymbol{S}|}$ is the transition probability,

* E-Mail: junhosong@snu.ac.kr

$\boldsymbol{R}$: $\boldsymbol{S} \times \boldsymbol{A} \rightarrow \mathbb{R}$ is the reward function, and $\gamma \in [0,1)$ is the discount factor. An MDP is essentially an extension of Markov chains, and has the Markov property, i.e., once an agent adopts a possible action a_t in state s_t, the transition probability $T(s_{t+1}|s_t, a_t)$ depends only on the state s_t and action a_t, not on the history of the previous states or actions. Furthermore, the agent receives a reward $r(s_t, a_t)$ that is a function of the state s_t and the action a_t. The agent's goal is to maximize the future discounted returns to the time horizon T.

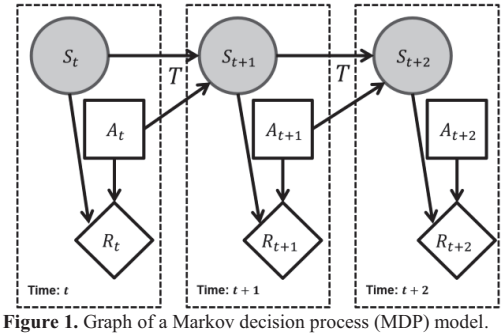

Figure 1. Graph of a Markov decision process (MDP) model.

In a stochastic environment, however, assessment of the adequacy of actions in each state using a single decision-making sequence may introduce a bias. To address this issue, the future discounted returns under a specific policy π are quantified by the following conditional expectation given the state s_t and the action a_t, termed Q-value:

$$Q_\pi(s_t, a_t) = E_{a\sim\pi}[\sum_{k=t}^{T} \gamma^{k-t} r(s_k, a_k) \,|s_t, a_t], \quad (1)$$

where a policy $\pi(a_t|s_t)$ is a mapping from the state s_t to the action a_t (or a vector representing the probability distribution for actions under state s_t).

For policy π, the existence and uniqueness of Q-value, Q_π, is guaranteed in all states if $\gamma < 1$, and dynamic programming (DP) is generally considered as the only feasible method to evaluate the exact Q_π in stochastic optimal control problems [8]. DP algorithms iteratively update the Q-value by the Bellman equation [9], which is recursively derived from Eq. 1 as

$$Q_\pi(s_t, a_t) = r(s_t, a_t) + \gamma \sum_{s\in\boldsymbol{S}} T(s|s_t, a_t) V_\pi(s), \quad (2)$$

where $V_\pi(s_t) = E_\pi[G_t|s_t] = \sum_{a_t\in\boldsymbol{A}} \pi(a_t|s_t) Q_\pi(s_t, a_t)$ is the value function, which is the expected future discounted returns on the state s_t under the policy π.

2.2 Q-learning algorithm

Many MDP algorithms have been developed to find the optimal policy π^* maximizing Q-values for all $s \in \boldsymbol{S}$. For this purpose, Watkins and Dayan [5] suggested the Q-learning algorithm, in which an agent keeps choosing the action having the highest Q-value for all $s_t \in \boldsymbol{S}$.

With sufficient exploration and exploitation, the Q-learning algorithm is indeed guaranteed to find the optimal policy for any finite MDP [8]. For efficient exploration and exploitation, the ε-greedy algorithm is often implemented: after initializing Q-value to zero, an agent selects a random action (i.e., exploration) with probability ε, or the action maximizing Q-value (i.e., exploitation) with probability $1 - \varepsilon$. It is common to start ε at a high value close to 1, and gradually decrease it as information about the environment accumulates. Then, Q-value is iteratively updated as

$$Q_{k+1}(s_t, a_t) = Q_k(s_t, a_t) + \alpha \cdot [y_t - Q_k(s_t, a_t)], \quad (3)$$

where $Q_k(s_t, a_t)$ is the Q-value on state s_t and action a_t estimated at the k^{th} iteration; $y_t = r(s_t, a_t) + \gamma \max_{a\in\boldsymbol{A}} Q_k(s_{t+1}, a)$ is the target value; and $\alpha \in (0,1]$ is the learning rate.

2.3 Deep Q-Network

In an environment with large sets of states and actions, however, the Q-learning algorithm may suffer because the computational cost grows exponentially with the number of possible states and actions. In the worst case, the Q-learning algorithm is intrinsically incapable in certain environments, e.g., an environment with continuous states and actions. Due to the low scalability, it is infeasible to evaluate the exact action-value function for every state-action pair in complicated environments.

To address this issue, Mnih et al. [10] introduced a powerful deep reinforcement learning (DRL) framework, where a deep neural network called deep Q-Network (DQN) approximates Q-value. In DQN, each layer outputs a linear combination of inputs and parameters, and activation functions between layers, including Sigmoid or rectified linear unit (ReLU), model complex nonlinear relationships. At the k^{th} iteration, the loss function $L(\theta_k)$ of DQN is given as

$$L(\theta_k) = [y_t - Q(s_t, a_t; \theta_k)]^2, \quad (4)$$

where θ_k are the parameters of DQN; $y_t = r(s_t, a_t) + \gamma \max_{a\in\boldsymbol{A}} Q(s_{t+1}, a; \theta_k)$ is the target value based on DQN; and $Q(s_t, a_t; \theta_k)$ is a parameterized Q-value on state s_t and action a_t. Then, θ_k are updated to minimize the loss function $L(\theta_k)$ in Eq. 4 by gradient descent as

$$\theta_{k+1} = \theta_k + \alpha[y_t - Q(s_t, a_t; \theta_k)] \nabla_{\theta_k} Q(s_t, a_t; \theta_k), \quad (5)$$

To stabilize the training procedure, Mnih et al. [11] utilized two innovative ideas: (1) experience replay [12], and (2) periodic updates of a target network. For experience replay, agents store experiences as tuples $e_t = (s_t, a_t, r_t, s_{t+1})$ in a replay buffer. At each training iteration, DQN is updated based on a batch of uniformly sampled tuples from the replay buffer instead of the latest experiences, so that the correlations between the experiences greatly decrease. In addition, a target network with parameters θ_k^- is introduced to compute the target value y_t, and the parameters θ_k^- are periodically updated to θ_k every N steps. Combining these two techniques, Eqs. 4 and 5 can be modified respectively as

$$L(\theta_k) = E_{e_t\sim U}\left[\left(y_t - Q(s_t, a_t; \theta_k)\right)^2\right], \quad (6)$$

$$\theta_{k+1} = \theta_k + \alpha E_{e_t\sim U}\left[\left(y_t - Q(s_t, a_t; \theta_k)\right) \nabla_{\theta_k} Q(s_t, a_t; \theta_k)\right], \quad (7)$$

where $y_t = r(s_t, a_t) + \gamma \max_{a\in A} Q(s_{t+1}, a; \theta_k^-)$ is the target value based on the target network with θ_k^-.

2.4 Multi-agent reinforcement learning

DRL can learn the complex environment and draw near-optimal policies in large spaces of states and actions. Although DRL could handle problems more efficiently, the curse of dimensionality is not fundamentally overcome and still hinders finding the optimal maintenance policies. To overcome this challenge, a divide-and-conquer strategy based on multi-agent reinforcement learning (MARL), in which agents decide independently or cooperatively to obtain their respective maximum rewards, can be a breakthrough in terms of scalability.

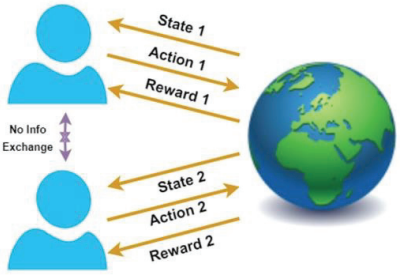

Figure 2. Independent Q-learning (IQL) architecture.

The simplest way to implement MARL is independent Q-learning (IQL) [13], in which each agent observes its local state and improves the solution independently as illustrated in Fig. 2. Since IQL is a fully decentralized MARL algorithm, the size of the state and action space is shrunk from $\prod_{i=1}^{n}|S_i|$ to $\sum_{i=1}^{n}|S_i|$ and from $\prod_{i=1}^{n}|A_i|$ to $\sum_{i=1}^{n}|A_i|$, respectively, where n is the number of agents; $|S_i|$ is the size of the state space observed by agent i; and $|A_i|$ is the size of the action space for agent i. As a result, the algorithm is scalable for environments with large spaces of states and actions.

In general, IQL shows superior performance compared to Q-learning algorithms with single-agent. However, there are some fatal issues in applying IQL to complex environments. The environment becomes non-stationary from each agent's point of view due to other learning agents [14] and does not satisfy the Markovian assumption required for the stability of Q-learning algorithms. This non-stationarity results in a gap between existing experiences and the current state, thereby rendering experience replay useless and inhibiting DQN learning.

Moreover, even if IQL overcomes the challenges and finds the optimal actions for each component, the obtained reward has a significant difference from the sum of the expected Q-values because of system-level rewards and interaction among components. In other words, the optimality of the policies detected by IQL is not guaranteed.

3. Clustering-based cooperative multi-agent Deep Q-Network

In O&M for large-scale civil infrastructure systems, decision-makers should consider both the damage and destruction of individual structures throughout systems. Even if the performance degradation appears insignificant at the level of individual structures, the accumulated damage can cause incomparably mortal socio-economic loss at the system level.

For the integrated management of infrastructure systems to prevent both component-level and system-level damage, it is necessary to consider all state and action combinations of components. As mentioned above, it is the main reason the existing DRL algorithms cannot handle large-scale systems. Therefore, a multi-agent-based algorithm is required to make up for the limitation of IQL while preserving its scalability. Depending on the amount of information shared, there is a trade-off between the optimality and the curse of dimensionality: convergence for system-level optimal solutions versus the loss of computational efficiency, which is the greatest advantage of MARL.

To this end, we introduce a Clustering-based Cooperative Multi-Agent (CCMA) Deep Q-Network for efficient O&M policy decision-making within large-scale systems in this study. In CCMA, decision-makers identify multiple subsystems from the initial system through clustering and assign an agent to each of the identified subsystems. All the agents make decisions to maximize the Q-values of the subsystem allocated to each, but the system penalty is additionally granted at the subsystem level. To determine whether each cluster is at fault for the imputed penalty, agents use the modified reward term as

$$r_{C_i}(\boldsymbol{S}_t, \boldsymbol{A}_{C_i,t}) = \sum_{j \in C_i} r_j(s_{j,t}, a_{j,t}) + \alpha \tilde{r}_i(\boldsymbol{S}_t), \qquad (8)$$

where C_i represents the i^{th} cluster; r_{C_i} is the modified reward for the i^{th} cluster; and $\boldsymbol{S}_t$ is the state set of all components in the system; $\boldsymbol{A}_{C_i,t} = \{a_{j,t}\}_{j=1}^{m}$ is the action set for the components in the i^{th} cluster with m denoting the number of components in the cluster; $r_j(s_{j,t}, a_{j,t})$ is the reward for state $s_{j,t}$ and action $a_{j,t}$ of the j^{th} component; α is the hyperparameter that determines how much weight is given to the imputed system penalty; and $\tilde{r}_i(\boldsymbol{S}_t)$ is the system penalty imputed to the i^{th} cluster, which is denoted as follow in this study:

$$\tilde{r}_i(\boldsymbol{S}_t) = \max\left(c\left(\Delta F_{C_i}\right), c\left(\Delta F_{sys}\right)\right), \qquad (9)$$

where $c(\cdot)$ is a cost function of the flow loss; ΔF_{C_i} is the flow loss in the i^{th} cluster; and ΔF_{sys} is the flow loss in the system.

Each agent is trained by DQN, which accompanies experience replay and periodic updates of the target network introduced in Section 2.4 to maximize their respective rewards.

4. Numerical example

In this section, a numerical example is presented to demonstrate the performance of CCMA: the 15-component distribution system with an origin/destination (O/D) pair in Fig. 3. All homogeneous components have five damage states: no damage, slight damage, moderate damage, extensive damage, and collapse. The flow capacity of each component for each state is 1, 0.95, 0.5, 0.25, and 0. We consider two types of available maintenance actions per component: do nothing or repair. The stationary transition probabilities of each component for the deterioration (i.e., 'do nothing') and repair are given as Eqs. 10 and 11, respectively.

$$T_{DN} = \begin{bmatrix} 0.8 & 0.2 & & & & \\ & 0.8 & 0.2 & & & \\ & & 0.8 & 0.2 & & \\ & & & 0.8 & 0.2 & \\ & & & & 1.0 \end{bmatrix}, \quad (10)$$

$$T_{RE} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}. \quad (11)$$

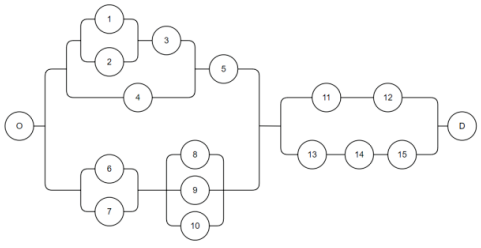

Figure 3. 15-component distribution system.

Although there is no limit to the number of components that can be repaired per time step, a reward of "–1" is given for each repair. In addition, when the maximum flow between the O/D pair decreases, the system is penalized by 5 times the amount of flow loss, i.e., $c(x) = 5x$. The system's life-cycle period is set to 50 steps (years) with a discount factor of $\gamma = 0.95$.

In this example, it is impossible to find the optimal solution due to the curse of dimensionality; the number of states $|S|$ and the number of actions $|A|$ are 5^{15} and 2^{15}, respectively. To confirm the superiority of CCMA, some maintenance schemes are implemented as baseline policies instead of the optimal solution: CBM scheme (i.e., component-level optimal policy) and the subsystem-level IQL (S-IQL). Fig. 4 shows the three subsystems identified through clustering, which are commonly used in both S-IQL and CCMA.

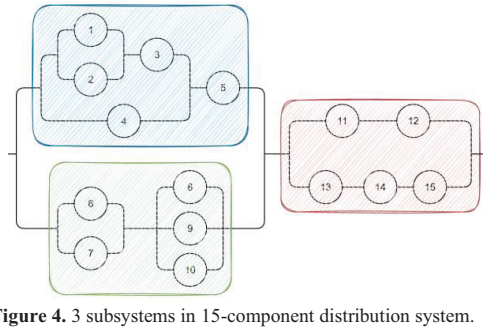

Figure 4. 3 subsystems in 15-component distribution system.

Fig. 5 shows the expected life-cycle costs obtained using CCMA, S-IQL, and CBM, respectively, by training over 2,000 episodes with 95% confidence intervals. Under the optimal CBM strategy, the expected life-cycle cost is constant regardless of the number of episodes because the agent repairs components according to the predetermined conditions. On the other hand, the expected life-cycle costs for S-IQL and CCMA consistently decrease as training progresses. Compared to CBM, these two methods based on reinforcement learning show much better performance.

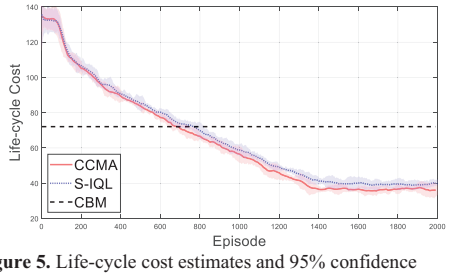

Figure 5. Life-cycle cost estimates and 95% confidence intervals during training for the distribution system.

Table 1 provides the results of O&M policies found through each methodology: the repair cost, the cost due to the flow loss, and the total life-cycle cost. For the optimal CBM, the agent ignores the loss of O/D flow and repairs components exceeding extensive damage, thereby causing minimal repair costs but excessive flow loss between the O/D pair. On the other hand, as each agent at the subsystem-level IQL makes decisions to maximize its reward (or minimize the cost) including the penalty for the flow loss in the subsystem, a low level of the O/D flow loss is maintained with frequent repairs. Finally, CCMA successfully achieves a suitable balance point between these two baselines in terms of repair cost and the cost due to the flow loss, respectively. Compared to the S-IQL solution, the CCMA policy reduces the number of repairs slightly, but inhibits the increase in flow loss, which in turn lowers total life-cycle cost.

Table 1. Comparison of CCMA with two baselines in terms of expected life-cycle costs.

Algorithms	Repair	Flow loss	Total cost
CBM	12.73	59.35	72.08
S-IQL	28.30	12.80	41.10
CCMA	24.27	13.81	38.08

5. Conclusions

In this study, an optimal decision-making framework based on deep reinforcement learning termed Clustering-based Cooperative Multi-Agent (CCMA) Deep Q-Network was proposed for O&M planning of large-scale civil infrastructure systems. To overcome the curse of dimensionality due to exponentially increasing spaces of states and actions, CCMA identifies subsystems through clustering and assigns agents to the identified groups. Each agent observes the state of each subsystem and shares the system penalty, thereby cooperating for the economical and reliable operation of the system and converging to the optimal policy at system level instead of the component level. The proposed algorithm and two baseline policies were demonstrated by a 15-component network example. In the numerical example, CCMA achieved the best policies superior to those by the baseline policies by balancing two negatively correlated repair costs and cost

due to the flow loss. Further research is underway to achieve the scalability of the proposed algorithm to large-size infrastructure systems and reallocate O&M policies effectively under budget constraints.

Acknowledgment

This work is supported by the Korea Agency for Infrastructure Technology Advancement (KAIA) grant funded by the Ministry of Land, Infrastructure and Transport (Grant 22RMPP-C163162-02)

References

[1] B. de Jonge, R. Teunter and T. Tinga, "The influence of practical factors on the benefits of condition-based maintenance over time-based maintenance," *Reliability engineering & system safety,* vol. 158, pp. 21-30, 2017.

[2] C. Andriotis and K. Papakonstantinou, "Managing engineering systems with large state and action spaces through deep reinforcement learning," *Reliability Engineering & System Safety,* vol. 191, p. 106483, 2019.

[3] J. Wang, S. Hou, Y. Su, J. Du and W. Wang, "Markov Decision Process Based Multi-agent System Applied to Aeroengine Maintenance Policy Optimization," in *2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery*, 2008.

[4] K. Arulkumaran, M. P. Deisenroth, M. Brundage and A. A. Bharath, "Deep Reinforcement Learning: A Brief Survey," *IEEE Signal Processing Magazine,* vol. 34, no. 6, pp. 26-38, 2017.

[5] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning,* vol. 8, no. 3-4, pp. 279-292, 1992.

[6] C. P. Andriotis and K. G. Papakonstantinou, "Deep reinforcement learning driven inspection and maintenance planning under incomplete information and constraints," *Reliability Engineering & System Safety,* vol. 212, p. 107551, 2021.

[7] M. Memarzadeh, M. Pozzi and J. Z. Kolter, "Optimal planning and learning in uncertain environments for the management of wind farms," *Journal of Computing in Civil Engineering,* vol. 29, no. 5, p. 04014076, 2015.

[8] R. S. Sutton and A. G. Barto, Reinforcement learning: An introduction, Cambridge: MIT press, 2018.

[9] R. Bellman, "Dynamic programming," *Science,* vol. 153, no. 3731, pp. 34-37, 1966.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602,* 2013.

[11] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King and D. Ku, "Human-level control through deep reinforcement learning," *nature,* vol. 518, no. 7540, pp. 529-533, 2015.

[12] L. J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Machine learning,* vol. 8, no. 3-4, pp. 293-321, 1992.

[13] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *Proceedings of the tenth international conference on machine learning*, 1993.

[14] J. Foerster, N. Nardelli, G. Farquhar, T. Afouras, P. H. S. Torr, P. Kohli and S. Whiteson, "Stabilising experience replay for deep multi-agent reinforcement learning," in *International conference on machine learning*, 2017.