Pamukkale Univ Muh Bilim Derg, 27(5), 579-588, 2021

Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi

Pamukkale University Journal of Engineering Sciences

An effective method to use centralized Q-learning in multi-robot task allocation

Çok robotlu görev atama probleminde merkezi Q-öğrenme kullanmak için etkili bir yöntem

Hatice Hilal EZERCAN KAYIR[1*]

[1]Department of Electrical and Electronics Enginering, Engineering Faculty, Pamukkale University, Denizli, Turkey.
hezercan@pau.edu.tr

Received/Geliş Tarihi: 10.01.2021 Revision/Düzeltme Tarihi: 05.02.2021 doi: 10.5505/pajes.2021.90490
Accepted/Kabul Tarihi: 07.02.2021 Research Article/Araştırma Makalesi

Abstract

The use of Q-learning methods in multi-robot systems is a challenging area. Multi-robot systems have dynamic and partially observable nature because of robot's independent decision-making and acting mechanisms. Whereas, Q-learning is defined on Markovian environments theoretically. One way to apply Q-learning in multi robot systems is centralized learning. It learns optimal Q-values for state space of overall system and joint action spaces of all agents. In this case, the system can be considered as stationary and optimal solutions can be converged. But, centralized learning requires full knowledge of the environment, perfect inter-robot communication and good computational power. Especially for large systems, the computational cost becomes huge because of exponentially growing learning space size with the number of robots. The proposed approach in this study, subG-CQL, divides the overall system into small-sized sub-groups without adversely affecting the system's task performing abilities. Each sub-group consists of less number of robots performing less tasks and learns in centralized manner for its own team. So, the learning space dimension is reduced to a reasonable level and required communication remains limited to the robots in the same the sub-group. Due the centralized learning is used, it is expected that the successful results are achieved. Experimental studies show that the proposed algorithm provides increase in the task assignment performance of the system and efficient use of system resources.

Keywords: Multi-Robot systems, Task allocation, Q-Learning, Centralized learning.

Öz

Çok robotlu sistemlerde Q-öğrenme yönteminin kullanımı oldukça problemlidir. Çok robotlu sistemlerde, robotun bağımsız karar verme ve hareket etme mekanizmaları nedeniyle dinamik ve kısmen gözlemlenebilir yapıya sahiptir. Oysa, Q-öğrenme yöntemi teorik olarak Markovian olarak nitelendirilebilecek ortamlar üzerinde tanımlanmıştır. Çok robotlu sistemlerde Q-öğrenmeyi uygulamanın bir yolu, merkezi öğrenmedir. Merkezi öğrenme, tüm sistemin durum uzayı ve tüm robotların tümleşik hareket uzayları için optimal Q-değerlerini öğrenir. Bu durumda, sistem statik olarak değerlendirilmekte ve optimal çözüm yakınsama mümkün olmaktadır. Ancak, merkezi öğrenme, çevre hakkında tam bilgi edinmeyi, robotlar arası iyi bir haberleşme sağlanmasını ve iyi hesaplama gücü gerektirir. Özellikle büyük sistemler için, robot sayısındaki artışla birlikte üstel büyüyen öğrenme uzayı boyutu nedeniyle hesaplama maliyeti çok yüksek olmaktadır. Bu çalışmada önerilen yaklaşım olan subG-CQL, sistemin görev yapma yeteneklerini olumsuz yönde etkilemeden genel sistemi küçük boyutlu alt gruplara ayırır. Her bir alt grup daha az sayıda robottan oluşur, daha az görev yapar ve kendi ekibi için merkezi bir şekilde öğrenir. Böylece öğrenme alanı boyutu makul bir düzeye indirilir ve gerekli iletişim aynı alt gruptaki robotlarla sınırlı kalır. Merkezi öğrenmenin kullanılması nedeniyle başarılı sonuçlara ulaşılması beklenmektedir. Deneysel çalışmalar, önerilen algoritmanın sistemin görev atama performansında artış ve sistem kaynaklarının verimli kullanımını sağladığını göstermektedir.

Anahtar kelimeler: Çok robotlu sistemler, Görev atama, Q-Öğrenme, Merkezi öğrenme.

1 Introduction

With the rapid growth of technology, multi-robot systems (MRS) become most popular especially for complex applications. MRS has the ability of faster task execution because team members can run simultaneously. MRS is highly fault-tolerant, when one robot gets out-of-run the others take over its role. And also, it has distributed sensing and acting facilities which provide wide working area, fast and flexible execution. [1]. An MRS environment is partially observable and dynamic in nature [2]. The robots in MRS operate with their own local sensing and each has its decision-making and acting mechanisms. Moreover, robot interaction and information sharing are complicated due to noisy and insufficient communication [3]. These explain why a precise and accurate coordination in MRS should be provided [4].

Multi-robot task allocation (MRTA) is the process of ensuring that robots do the appropriate tasks at right time in an appropriate order [5]. MRTA has a key function to get the necessary coordination and optimize system performance. One widely used approach to solve MRTA problems is auction protocols, which is a special kind of market-based approaches [6]. Auction-based task allocation approaches have advantages of implementation simplicity and distributed planning centralized decision-making ability [7]. Distributed mapping [8], multi-robot box pushing [6], multi-robot path planning [9] are some examples of auction based multi robot coordination. In auction protocols, it is considered that the tasks are items and robots are customers. Tasks are announced by auctioneer robot with base price representing their costs. Customer robots calculate the cost of announced tasks according to their own possibilities and send bids to auctioneer. The cost of a task may be travelled distance or execution time for mobile robots [10].

*Corresponding author/Yazışılan Yazar

Auction process ends up by allocating the tasks to suitable robots in a manner that maximizes the system gain [11],[12].

In MRS, it is not possible to guarantee perfect coordination by traditional ways because of environment's partially observable and ambiguous nature [13]. Generally, tasks arise at unpredictable instants in unpredictable sequence while system is running. For this reason, it is not possible to pre-plan the assignment of tasks to robots. Task allocation needs to be performed instantaneously as the tasks appear [14]. Moreover, each robot has independent sensing, decision-making and acting mechanisms. This prevents robots from predicting others' behaviors. An efficient system coordination is only possible if robots can adopt to changing environmental conditions. Thanks to the acquisition of learning skills to the system, robots would overcome unpredicted and uncertain situations [15]. As an example of learning-based coordination, robots use their past task allocation experiences for bidding future tasks [16]. In [17], a learning-based approach to reason about future task allocation. Robots learn the bid values and use them in auction process successfully for underwater exploration which is a dynamic environment with high level uncertainty [18]. Efficient solutions are obtained by using reinforcement learning for dynamic task allocation problems in fire-disaster response [19],[20].

Q-learning (QL) is a value-function based model-free reinforcement learning method [21]. It learns optimal Q values for each state-action pair in tabular form [4]. QL is suitable to apply for complex applications, e.g., robotic systems, because it does not require environment formulation. In fact, QL is defined for MDP environments theoretically. It is problematic to apply for MRS due to dynamic and partially observable characteristics of them [22]. One way to use QL in MRS is distributed learning approach in which the robots learn only for their own state-action pairs. Whereas it is easy to implement, that the environment is not stationary so far, contradicts the requirement of MDP environment. So, distributed learning does not assure to reach optimal solution. The other way is centralized learning that works on joint state and joint action spaces. This needs perfect inter-agent communication. In centralized learning, joint action space dimension increases exponentially with the number of robots. Especially for large systems, the huge learning space dimension causes computational and implementational difficulties [23].

In this study, a new method, subG-CQL algorithm, is proposed to overcome the problems encountered in the use of QL in MRS. The main goal of this approach is to divide the system into smaller sub-groups by allowing some robots to concentrate on specific tasks without adversely affecting the system. subG-CQL algorithm has a positive all-round impact on system performance. It becomes possible to use the robots more efficiently as they deal with less variety of tasks. Learning is carried out by centralized manner in these sub-groups, which are completely independent from each other. Thus, it is possible to exploit the advantage of convergence to the optimal solution of centralized learning. In addition, the scalability problem arising from the large learning dimension, which is the biggest problem of centralized learning, has been solved. The comparative results show the successful solutions of the proposed method on system performance.

The arrangement of the paper is as follows: In Section 2, Q-learning basics for single-agent and multi-agent cases are given briefly. Section 3 examines the problem handled in this study. The proposed approach is explained, and algorithm is given in Section 4. Section 5 is about application details, such as system structure and performance metrics. Experimental results and comments are included in Section 6. Conclusion part is in Section 7.

2 Q-Learning basics

Reinforcement learning is a class of machine learning techniques that do not need any mentor or system model [24]. Learning process takes place based on the feedback which is the measure of the changes in environment states as a result of agent's action. In reinforcement learning theory, this feedback is called as reward. If the agent's action causes the state to change as desired, the reward receives a value in a way that reinforces this action. In the opposite case, the reward value is in the form of penalty obstructing this action. In short, reinforcement learning techniques are methods of trial-and-error. Due it does not require any prior information about the system, reinforcement learning methods seem like a good learning approach especially for complex environments [21].

Q-learning (QL) is a reinforcement learning method based on value function approach. In QL proposed by [22], an agent learns Q values of each state-action pair by using reward received as a feedback of its actions' effect on environment states. Theoretical details of QL for both single-agent and multi-agent cases are given below.

2.1 Single agent Q-Learning

Single agent Q-learning is defined on the environment defined as Markov decision process. A Markov decision process (MDP) is a tuple of $< S, A, P, \rho >$. Here, S is the set of discrete and finite states of environment, A is the set of discrete and finite actions of agent, $P: S \times A \times S \rightarrow \Pi(S)$: [0,1] is the probabilistic state transition function and $\rho: S \times A \times S \rightarrow \mathbb{R}$ is the reward function in reel numbers [19].

At step k, agent takes action $a(k) \in A$ then environment state, $s(k) \in S$, is switched to $s(k+1) \in S$. Agent receives the reward, $r(k) = \rho(s(k), a(k), s(k+1))$, as feedback of its action's effect on environment [24]. Agent's action $a(k)$ at state $s(k)$ is determined by agent's action policy, h. In MDP, each agent has a deterministic, static, and optimal action policy [21]. For each step, action policy h leads agent selects its action in a manner that it maximizes the expected value of overall gain. Action-value function $Q^h: S\ x\ A \rightarrow R$, implies the expected total gain value of each state-action pair in according to action policy. Action-value function is the discounted sum of all future reward and it is expressed as in (1), where γ is the discount factor.

$$Q^h(s,a) = E\{\sum_{i=0}^{\infty} \gamma^i\, r(k+i) \mid\ s = s(k),\ \ a = a(k),\ \ h\} \qquad (1)$$

Optimal action-value function is defined as Q-function given in equation (2) and it satisfies Bellman optimality equation [25].

$$Q^*(s,a) = \max_h Q^h(s,a), \qquad \forall s \in S\ ve\ \forall a \in A \qquad (2)$$

Q-learning is a value-function based and model-free reinforcement learning method [26]. In Q-learning, optimal Q-values for each-state action pair are learned in an iterative manner by the equation in (3). γ is the discount factor and α is the learning rate [27].

$$\begin{aligned} Q(s(k),a(k)) = Q(s(k),a(k)) \\ + \alpha_k \left[r(k) + \gamma \max_{a' \in A} Q_k(s(k+1), a') \right. \\ \left. - Q(s(k),a(k)) \right] \end{aligned} \qquad (3)$$

This equation does not need environment model and probabilistic state transition functions. If this equation is recurred infinitely many times for each state-action pair and α is appropriately diminished at each step, the learned Q-values converge to optimal ones [22].

2.2 Multi agent Q-Learning

Stochastic game (SG) is defined as the tuple of $< S, A, P, \rho_j >$, where S is the set of finite and discrete environment states, $A = A_1 \times A_2 \times \dots \times A_m$ is the joint action set for all agents. $P: S \times A \times S \to \Pi(S): [0,1]$ is the state transition function defined for each state and joint action pair and $\rho_j: S \times A \times S \to R, j = 1 \dots m$ represents the reward of each agent [27]. With this definition, an SG can be thought as the generalized form of MDP. For an SG, the state transitions are realized by joint actions of all agents.

The Nash equilibrium states the joint action policy, that each agent's action policy ensures maximum total reward value against other agents' action policy [21]. In the Nash equilibrium, total reward cannot be improved by changing one agent's action policy in the case that all other agents' action policies are kept same. Nash-Q-learning algorithm is a multi-agent Q-learning aiming to reach Nash equilibrium [22]. For agent j, the Q-values are updated according to equation (4) by using joint actions. $Nash_j$ implies the Nash equilibrium for all agents.

$$Q^j(s, a_1, \dots, a_m) = Q^j(s, a_1, \dots, a_m) + \alpha\left[\rho_j + \gamma Nash_j\left(s, Q^1, \dots, Q^j, \dots, Q^m\right) - Q^j(s, a_1, \dots, a_m)\right] \quad (4)$$

Nash-Q-learning gives successful solution under some assumptions, e.g., fully cooperative systems [28]. A fully cooperative SG can be considered as MDP and optimal solution can be converged [29].

3 Problem statement

In most real-world MRS applications, the working environment has partially observable and dynamic nature due to noisy sensor measurements, limited communication, and unpredictable effects of agents' actions [3]. These properties contradict the theory of reinforcement learning and explain why an optimal solution in MRS coordination cannot be reached by traditional QL algorithms [22].

One approach to apply QL for MRS is to use decentralized learning structure. In decentralized learning, each robot learns Q values for only its own states and actions by directly applying QL rules defined for single-agent case [30]. Robots do not concern with the results of other robots' actions. So, decentralized learning is simple to run, and it does not need inter-robot communication. Dimension of learning space, which consists of individual state and action spaces, is small and computational cost is low [30]. On the other hand, the MRS environment is no longer stationary due to the robots' independent actions, which contradicts the QL theory. Since robots perform the learning process individually without considering the decisions of others, behavioral conflicts are inevitable [30]. This constitutes the major reason not to reach optimal solutions [22]. However, the decentralized learning is preferred in many applications because it is easy to implement and learning space dimension is small. Successful results have been achieved for small environments under some constraints [31]. Independent Q-learning (IQL) is an example having high-degree of decentralization [32]. Empirical results show that IQL works well in simple applications only [33]. Hyper Q-learning try to solve nonstationary problem by observing other agents' actions [34]. [35] uses coordination graphs to estimate global Q values.

In centralized Q-learning, which is another approach for multi-agent QL, robots learn global Q values using joint actions of whole team. Since the joint actions and joint states of all team members are considered, the MRS environment could be assumed as MDP. It is expected that the optimal solution is converged [22]. Centralized learning requires full knowledge of all robots' actions and all possible forms of states and perfect communication among robots. Deficiency in these factors results in failure to achieve the desired success [2]. Furthermore, the dimension of joint action space grows exponentially in the number of robots. This means that the learning spaces becomes huge and computational complexity increases enormously for large MRS's [28]. Whereas centralized learning promises to reach optimal solution, it is very difficult to implement.

Both decentralized and centralized learning approaches have some trade-offs. In most studies, hybrid learning schemas are proposed to combine the advantages and discord the problematic points. In these generally the learning is carried out in distributed manner, but there is an external coordination mechanism to obtain global solution. In modular Q-learning approach, robots learn their own Q values and there exist a control unit to overcome the behavior conflict [36]. Sequential Q-learning algorithm proposes the agents learn independently in a pre-determined order. Each agent observes others' actions and then learn for its own state-action pairs [37]. In CTDE algorithm, learning process is run in distributed manner with the full knowledge of environment states [38]. VDN [39] and QMIX [40] are examples which joint-action value functions are factorized into individual ones. So, learning space scalability problem is minimized. But they have constraints of being applicable to systems having at least one optimal solution [41].

In this study, subG-CQL algorithm is proposed to provide the use of QL in MRS easily. subG-CQL algorithm splits the whole system sub-groups independent with each other. Each sub-group behaves as a small-sized system and all of them consists the overall system. subG-CQL algorithm force the robots refuse some of their task types. It aims to match the tasks done by large number of robots to the robots having the ability of performing less number of task type. So, the robots carrying out large number of task types drop some of them. As a result, each robot performs less number of task type and each task type are done by less number of robots. At the end, the robots performing the same task types compose a sub-group. For each sub-group, the learning can be realized in centralized manner easily because of the small size of them. The details of the algorithm are given in Section 4.

4 Proposed approach: subG-CQL algorithm

Let a heterogenous MRS consists of m different-skilled robots, $R_i, i = 1, \dots, m$. These robots are responsible for fulfilling n different type of tasks $T_j, j = 1 \dots n$. In a heterogenous MRS, each robot is capable of some tasks, not all. And also, some robots are more likely to perform certain tasks due to their physical structure. Every robot can perform different number of tasks types.

Robot R_i has the robot task set (RTS), Γ_i, which is a list of tasks can be executed by R_i as in (5).

$$\Gamma_i = \{ \langle T_z , d_{iz} \rangle \mid d_{iz} \in [0,1] \} \quad (5)$$

d_{iz} is the willingness parameter of R_i to do T_z.

Robot-task relation matrix (RTM) carries information about R_i robots and Γ_i sets, $i = 1..m$ and it is defined as in equation (6).

$$M = [m_{ij}] = \begin{cases} m_{ij} = 1 , & T_j \in \Gamma_i \\ m_{ij} = 0 , & T_j \notin \Gamma_i \end{cases} \quad (6)$$

Row-sum of RTM, $r_i = \sum_{j=1}^{n} m_{ij}$, is the number of all tasks can be done by R_i and it is equal to the size of Γ_i. Column-sum of RTM, $c_j = \sum_{i=1}^{m} m_{ij}$, is the number of robots that can perform T_j tasks.

subG-CQL algorithm, proposed approach in this study, aims to divide the system into sub-groups with less number robots performing limited number of tasks for each. The idea behind of subG-CQL algorithm is that the tasks executed by large number of robots are matched to the robots having the responsibility of fewer tasks types. Thus, the variety of tasks that robots have to do will decrease and the robots performing the same type of tasks will form small-sized groups. So, it is possible to concentrate the robots on less number of task type.

Initially, there is no sub-group, all robots form just one team. At each iteration, the algorithm selects the task performed maximum number of robots. This corresponds to task T_j such that RTM column having the highest c_j value. This T_j task is matched to the robot R_i with the least r_i value. If task T_j or robot R_i is already present in one of the existing sub-groups, then $< R_i, T_j, w_{ij} >$ triplet is added to this sub-group. If not, a new sub-group is created with this triplet, the first element of it. w_{ij}, represents the degree of R_i-T_j pair and it is calculated as in (7).

$$w_{ij} = \frac{d_{ij}}{r_i \, c_j} \quad (7)$$

If there are more than one task having same c_j value, the algorithm consults to the column weight which is the measure of how many robots do these tasks. The task with the lowest column weight is selected. Column weight, ω_j is calculated by using equation (8).

$$\omega_j = \sum_{i=1}^{m} m_{ij} r_i \quad (8)$$

In the case of more than one row having minimum r_i value, the robot with the highest willingness parameter d_{ij} is matched to the task.

Each robot R_i is connected to a sub-group L_g with a dependency parameter η_{ig} calculated as the sum of w_{ij} for the sub-group L_g. In some cases, both R_i and T_j matched with each other exist in present sub-groups but different ones. Then the triplet $< R_i, T_j, w_{ij} >$ is added to all of them. Such robots belonging to multiple sub-groups are stored in set C. During the algorithm, robots' new task sets are constituted. Λ_i is the new task set of robot R_i and it contains the task type T_j's which are matched to R_i with the w_{ij} weights.

After the formation of sub-groups is completed, the robots in C are forced to choose one sub-group which has the highest η_{ig}. The triplets related to R_i robot are excluded from other sub-groups. New task sets are edited by deleting the dropped task types dropped by R_i. Although this task refusing procedure, subG-cQL algorithm provides each task type is performed by enough robots thanks to its matching strategy.

At the end of this step, subG-CQL algorithm has finished. Overall system is divided into sub-groups that each of them acts as an independent small-sized system. In each sub-group, learning is run in centralized manner with its own state space and joint action space of robots in this sub-group. subG-CQL algorithm is given below in detail.

subG_CQL Algorithm

Input: Robot-task relation matrix, M
Output: Sub-groups $L_f, f = 1 \dots p$
Robots' new task set $\Lambda_i, i = 1 \dots m$

$x = 0, p = 0, C = \emptyset, \Lambda_i = \emptyset$
for $\forall c_j \neq 0, j = 1 \dots . n$
  Select task T_j such that $\underset{j}{\text{argmax}}\, c_j$
  Select robot R_i such that $\underset{i}{\text{argmin}}\, r_i, r_i \neq 0$
  if multiple c_j
    Select T_j such that $\underset{j}{\text{argmin}}\, \omega_j$
  end

  if multiple r_i
    Select R_i such that $\underset{i}{\text{argmax}}\, d_{ij}$
  end
  Match $T_j - R_i$, such that $\underset{i}{\text{argmin}}\, r_i, r_i \neq 0$
  Calculate w_{ij}
  if $R_i \notin L_f \wedge T_j \notin L_g, f = 1 \dots p, g = 1 \dots p$
    $p = p + 1$
    Create new sub-group $L_p = \emptyset$
    $L_p = L_p \cup < R_i, T_j, w_{ij} >$
    Calculate η_{ip}
  end
  if $R_i \in L_f, f = 1 \dots p$
    $L_f = L_f \cup < R_i, T_j, w_{ij} >$
    Calculate η_{if}
  end
  if $T_j \in L_g, g = 1 \dots p$
    $L_g = L_g \cup < R_i, T_j, w_{ij} >$
    Calculate η_{ig}
  end
  if there exists f and g such that $f \neq g$
    $C = C \cup R_i$
  end
  $\Lambda_i = \Lambda_i \cup T_j$
  $m_{ij} = 0$
  $r_i = r_i - 1, c_j = c_j - 1$
  Update ω_j
end

if $C \neq \emptyset$
  for $\forall R_q \in C$
    Select L_t such that $\underset{v}{\text{argmax}}\, \eta_{qf}, f = 1 \dots p$
    for $\forall (R_q \in L_h)$
      if $h \neq v$
        $L_h = L_h \setminus < R_q, T_j, w_{qj} >$
        $\Lambda_i = \Lambda_i \setminus T_j$
      end
    end
  end
end

5 Application

5.1 System structure

For experimental studies, two different MRS, that one is a small-sized system, and the other is a great and much more complex system, are prepared.

In System-I, there exist six robots having the ability to do five different type of tasks. System-I is fully heterogenous nature due to the robots' different physical properties and different skills. Robots and related tasks in System-I are given in Table 1.

Table 1. Robots and related tasks for system-I.

Robots	Related Task Type
$\boldsymbol{R_1}$	T_1, T_4, T_5
$\boldsymbol{R_2}$	T_2, T_4
$\boldsymbol{R_3}$	T_1, T_3
$\boldsymbol{R_4}$	T_4, T_5
$\boldsymbol{R_5}$	T_1
$\boldsymbol{R_6}$	T_1, T_3, T_5

System-II is like System-I, but it is much bigger and has more complex structure with ten robots and eight different type of tasks should be done. System-II is also highly heterogenous. Robots and task relation for System-II are shown in Table 2.

Table 2. Robots and related tasks for system-II.

Robots	Related Task Type
$\boldsymbol{R_1}$	T_3, T_5
$\boldsymbol{R_2}$	T_1, T_2, T_5, T_7, T_8
$\boldsymbol{R_3}$	T_3, T_6, T_7
$\boldsymbol{R_4}$	T_1, T_3, T_4, T_6
$\boldsymbol{R_5}$	T_2, T_5, T_7, T_8
$\boldsymbol{R_6}$	T_1, T_2
$\boldsymbol{R_7}$	T_4, T_6, T_7
$\boldsymbol{R_8}$	T_2, T_5, T_8
$\boldsymbol{R_9}$	T_2, T_6, T_7
$\boldsymbol{R_{10}}$	T_3, T_5, T_8

For both systems, all types of tasks are equally probable. Each task type has two priority level named as high-priority and low-priority. High-priority tasks have should be done absolutely and primarily and they are more time-consuming than the low-priority tasks. High-priority tasks is 30%-35% of all tasks and remains are low-priority tasks.

Tasks appear at any time and in random sequence during system operation. Task allocation process is executed in auction-based manner. The tasks, that are announced but not assigned, auctioned once more at a certain time later. If it is not assigned, it will be dropped from the list. It is assumed that the tasks are allocated to any robot absolutely done.

5.2 Performance metrics

The learning-based task allocation methods are applied to the systems described above. The experimental results are evaluated in terms of three performance metric; completed task ratio, idle time ratio, and learning space dimension.

Completed task ratio (CTR) is the ratio of the number of tasks executed to the total number of tasks announced. The assigned task number is used instead of the completed task number. CTR is calculated in percent.

Idle time ratio (ITR) represents the ratio of robot's free time to whole execution time. ITR is determined by following equation (9).

$$ITR = \frac{t_{exe} - t_{busy}}{t_{exe}} \tag{9}$$

Here, t_{exe} is the total execution time of a robot and t_{busy} is the duration that robot is busy with auction process, performing any task or charging.

In literature, it is given that the computational cost of Q-learning is directly related to the learning space dimension calculated as the product of state space dimension and action space dimension [23],[28]. Let U be the learning space and $|\cdot|$ represent space dimension. The individual learning space dimension of robot R_i is determined by equation (10), where S_i is the state space of R_i and A_i is the action space of R_i.

$$|U_i| = |S_i| \, x \, |A_i| \tag{10}$$

In decentralized learning, the learning space dimension of overall system with m robot is equal to the sum of individual learning spaces given in equation (11).

$$|U| = \sum_{i=1}^{m} |S_i| \, x \, |A_i| \tag{11}$$

Centralized learning uses joint action and state spaces. State space of whole system is the union of individual state spaces as in equation (12) and the dimension of it is in equation (13).

$$S = \bigcup_{i=1}^{m} S_i \tag{12}$$

$$|S| \leq \sum_{i=1}^{m} |S_i| \tag{13}$$

Joint action space is the cartesian product of individual action spaces as in equation (14) and dimension of joint action space is calculated by equation (15).

$$A = \prod_{i=1}^{m} A_i \tag{14}$$

$$|A| = \prod_{i=1}^{m} |A_i| \tag{15}$$

In worst case, the joint action space is in its largest form of $|A| = |A_i|^m$ and this occurs when the system is fully cooperative. Learning space dimension in centralized learning is simply calculate by using equation (16).

$$|U| = |S| \, x \, |A| \tag{16}$$

In the proposed algorithm, the system is divided into p sub-groups. Each sub-group run the learning process in centralized manner. But these sub-groups act as if it was an individual agent of whole system. So, the state space and action space dimension of each sub-group are determined by equations (13) and (15) respectively and learning space of overall system is calculated by adding up the learning space of all sub-groups.

6 Experimental results

Experimental studies are realized on two different systems Whose details are given in Section 4. To emphasize the impact and successful results of the proposed approach, subG-CQL, algorithm, it is compared with the three other methods given in

literature, centralized Q-learning, semi-centralized Q-learning and decentralized Q-learning.

A well-known example of centralized Q-learning (CQL) method is studied in [28]. CQL method assumes the environment as an SG which is primary requirement of multi-agent Q-learning theory. Learning is carried out on the overall state space of the environment and joint action space of robots. Robots' main aim is to reach Nash equilibrium.

In semi-centralized Q-learning (semi-CQL) method, robots gather information about states of whole system and use it to determine their own behavior. But each robot learns the Q-values of its own state-action pairs in a similar way of [38].

In decentralized Q-learning (DQL) method, each robot acts as an independent learner in [32]. The robots run learning process individually by using their own state and action spaces. And they do not care the behavior of their teammates.

subG-CQL algorithm, proposed in this study, divides the system into small groups uncorrelated with each other. Every sub-group behaves as a different system and learning is managed in centralized manner for each of them. Because the system size is diminished to small ones, the scalability problem of in CQL can be handled. As a result of subG-CQL algorithm, two sub-groups are formed in System-I. R_1 robot drops T_1 task from its RTS. Similarly, R_6 robot renounces T_5 as it can do this type of task. The sub-group structure of System-I is in Table 3.

Table 3. Sub-groups formed after subG-CQL for system-I.

Sub-Group-1:	
Robots	Related Task Type
$\boldsymbol{R_1}$	T_4, T_5
$\boldsymbol{R_2}$	T_2, T_4
$\boldsymbol{R_4}$	T_4, T_5
Sub-Group-2:	
Robots	Related Task Type
$\boldsymbol{R_3}$	T_1, T_3
$\boldsymbol{R_5}$	T_1
$\boldsymbol{R_6}$	T_1, T_3

subG-CQL algorithm creates three sub-groups in System-II. Many robots have stopped performing multiple tasks as shown in Table 4. For example, R_2 robot drops T_5 and T_7 tasks, it performs only three types of task after subG-CQL whereas it has five types of task in its RTS originally.

Table 4. Sub-groups formed after subG-CQL for system-II.

Sub-Group-1:	
Robots	Related Task Type
$\boldsymbol{R_1}$	T_3, T_5
$\boldsymbol{R_{10}}$	T_3, T_5
Sub-Group-2:	
Robots	Related Task Type
$\boldsymbol{R_2}$	T_1, T_2, T_8
$\boldsymbol{R_5}$	T_2, T_8
$\boldsymbol{R_6}$	T_1, T_2
$\boldsymbol{R_8}$	T_2, T_8
Sub-Group-3:	
Robots	Related Task Type
$\boldsymbol{R_3}$	T_6, T_7
$\boldsymbol{R_4}$	T_4, T_6
$\boldsymbol{R_7}$	T_4, T_6, T_7
$\boldsymbol{R_9}$	T_6, T_7

The experimental results for four different method are analyzed below separately.

6.1 Completed task ratio (CTR)

CTR gives the ratio of completed tasks and it can be considered as the success of task allocation process. CTR values of all task types for both systems are also written in Table 5 and Table 6 for both systems. CTR values are in percent and they are rounded to the nearest integer for simplicity.

CTR values are shown by graphs in for all approaches comparatively in the figures below. CTR for low-priority and high priority tasks separately in Figure 1 and Figure 2 for System-I and in Figure 3 and Figure 4 for System-II respectively. In the graphs, TjL represents the low priority tasks of task type T_j. Similarly TjH shows the high-priority tasks.

When the graphs are examined, the behavior of all methods is nearly similar for both systems. Some variations are caused from the differences in the systems. For every task, the number of robots that perform it is not same. In particular, the tasks that can be done by more robots have greater CTR values than others.

Table 5. CTR values for system-I.

	T_{1L}	T_{1H}	T_{2L}	T_{2H}	T_{3L}	T_{3H}	T_{4L}	T_{4H}	T_{5L}	T_{5H}
CQL	74	93	52	78	57	87	65	94	61	88
Semi-CQL	62	86	41	59	47	65	52	82	49	74
DQL	57	81	32	63	37	79	40	83	42	75
subG-CQL	76	94	55	68	53	89	62	92	58	82

Table 6. CTR values for system-II.

	T_{1L}	T_{1H}	T_{2L}	T_{2H}	T_{3L}	T_{3H}	T_{4L}	T_{4H}	T_{5L}	T_{5H}	T_{6L}	T_{6H}	T_{7L}	T_{7H}	T_{8L}	T_{8H}
CQL	42	87	62	92	55	87	53	84	61	91	59	86	66	94	60	92
Semi-CQL	38	73	54	82	49	72	46	68	53	82	54	72	59	81	52	78
DQL	35	77	50	85	45	77	42	72	50	85	46	77	56	84	49	81
subG-CQL	46	85	68	96	51	84	50	87	60	92	64	94	65	93	54	89

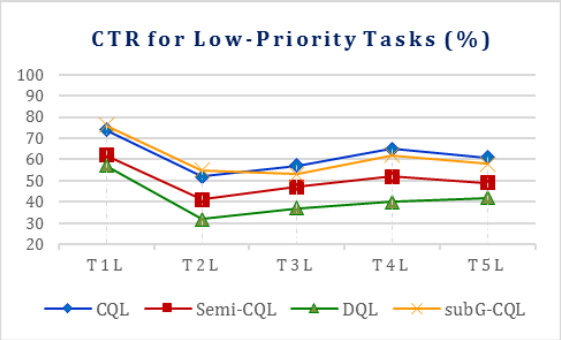

Figure 1. CTR for low-priority tasks for System-I.

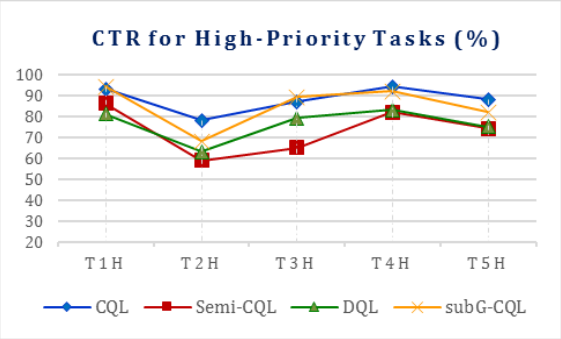

Figure 2. CTR for high-priority tasks for System-I.

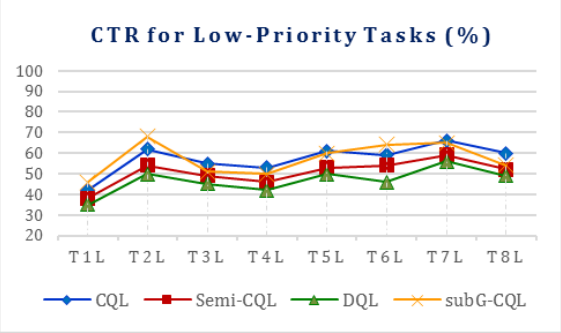

Figure 3. CTR for low-priority tasks for System-II.

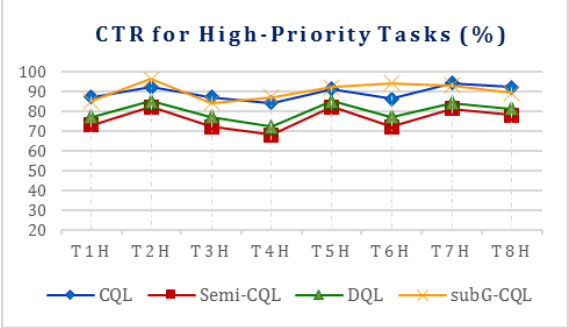

Figure 4. CTR for high-priority tasks for System-II.

The aim of the learning based MRTA is to increase the number of completed tasks, especially for the tasks with high-priority. Experimental results indicate that CTR values of high-priority tasks are higher than that of low-priority tasks thanks to auction-based task allocation.

As expected, CQL algorithm has the highest CTR values for each type of task. In CQL, it can be thought that the environment is nearly stationary and contains no uncertainty because overall state space of the environment and joint action spaces of all agents are considered. On the contrary, DQL algorithm results in the lowest CTR values for low-priority tasks. CTR of high-priority tasks are reasonably high. In DQL, robots learn individually for their state and action spaces. As a result of this, robots force themselves to do high-priority tasks rather than low-priority ones. semi-CQL algorithm's CTR values for low-priority tasks are a bit higher than DQL and for high-priority tasks are lower than DQL, but not so poor. semi-CQL gather information about overall system states but it uses decentralized learning structure. This explains why its results are so similar to DQL.

subQ-CQL algorithm, proposed in this study, has satisfactorily good results of CTR for both high-priority and low-priority tasks although these values are a bit lower than the results of CQL. subG-CQL learns in centralized manner but in smaller size. Whole system is divided into sub-groups which have no common task or robot with each other. One advantage of this algorithm is that it provides easy-to-apply centralized learning for each small-sized sub-group. The other advantage is to concentrate the robots on less variety of task type. To deal with the same task type makes much more tasks completed because every task type has different features such that duration, difficulties etc.…

6.2 Idle Time Ratio (ITR)

Robots and their abilities are considered as system resources. In most cases, all tasks could not be done due to the scarcity of resources in MRS applications. Insufficiencies in the number of robots causes that a lot of tasks announced cannot be assigned to the robots because all robots are busy with another task at auction duration. ITR values are meaningful for effective use of resources. The reduction in the free time of the robots means that robots do right tasks at the right time. ITR values of all robots for System-I and System-II are given in Table 7 and Table 8 respectively in percent.

Table 7. ITR values for system-I.

	R_1	R_2	R_3	R_4	R_5	R_6
CQL	21	17	19	18	15	22
Semi-CQL	27	20	23	27	37	28
DQL	32	21	29	34	44	31
subG-CQL	12	14	17	15	13	18

Table 8. ITR values for system-II.

	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9	R_{10}
CQL	19	21	23	14	17	16	13	21	18	21
semi-CQL	25	23	27	19	23	35	19	18	24	29
DQL	27	24	31	21	26	39	21	24	27	33
subG-CQL	12	14	9	11	13	10	12	15	13	14

subG-CQL algorithm provides low ITR values because it knows all existing situations of the environment and use this information in learning process. So, task allocation process is managed in balance and robots are run effectively. The highest ITR values are obtained by DQL algorithm. In DQL, robots learn independently, and they do not concern their teammates' behavior. This causes behavior conflicts, which is the major disadvantage of decentralized learning, among team members. For example, two robots learn same action for the same task and this task is assigned one of them, the other goes to idle

mode. The ITR values of semi-CQL algorithm is better than DQL and worse than CQL. This is because it knows the other robots' actions but learn in decentralized manner.

The best ITR values is achieved by subG-CQL algorithm. It forces some robots drop some of its type of task and concentrate on less task type. Thus, it becomes possible to utilize the robots effectively and idle time of robots decreases as desired. ITR values of robots are drawn graphically Figure 5 and Figure 6 for System-I and System-II respectively.

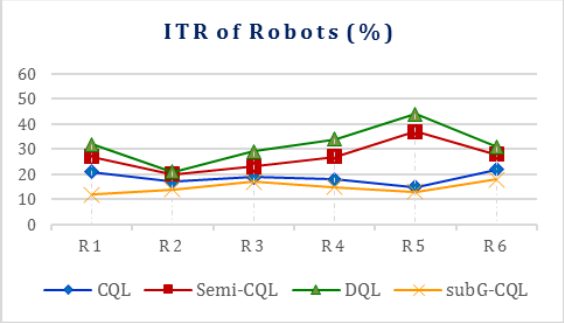

Figure 5. ITR of robots for System-I.

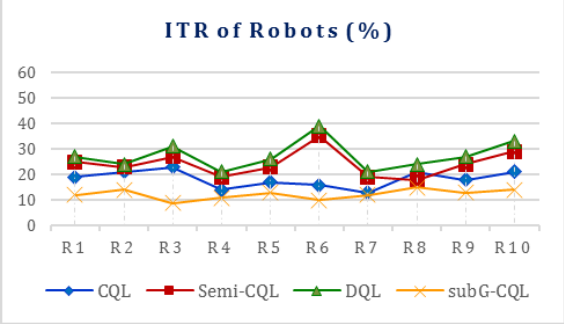

Figure 6. ITR of robots for System-II.

6.3 Learning space dimension

Learning space dimension whose details are explained in Section 5.2, is directly related to the learning schema and system size. The learning space dimensions for both systems are given in Table 9.

Table 9. Learning Space Dimensions.

	System-I	System-II
CQL	$17 * 2^{12} * 3^4$	$7 * 2^{29} * 3^8 * 5^2$
Semi-CQL	426	$3 * 2^{10}$
DQL	426	$3 * 2^{10}$
subG-CQL	2^{12}	$2^8 * (2^2 * 3^4 + 1)$

The learning space dimension of System-II is higher than that of System-I, because the size of System-II much larger than System-I in all algorithms. The smallest learning space dimension is obtained for DQL and semi-CQL which use decentralized learning. CQL algorithm has the highest learning space dimension caused by the joint action space dimension. Especially for System-II, the learning space dimension is huge. It is a big obstruction in apply centralized learning which has a great success to reach optimal solution.

subG-CQL algorithm provides a reasonable learning space dimension compared to CQL, although both learn in centralized manner. subG-CQL algorithm divides the system into small-sized sub-groups and each one uses its own joint action and state spaces. Its learning space dimension is equal to the sum of sub-group's learning space dimensions. Whereas, CQL considers joint action and state space of whole system.

The results of the experimental studies can be summarized as follows in terms of the performance metrics discussed.

- As expected, the best CTR values, which is a sign of successful MRTA process, are obtained in CQL method for both high-priority and low-priority tasks. CTR values of the proposed algorithm, subG-CQL, are a bit lower than CQL,
- The lowest ITR values are achieved by subG-CQL algorithm. Low ITR means that the robots and their abilities are effectively used and their waste time is decreased,
- semi-CQL and DQL methods, both of which learns in decentralized manner, have the lowest learning space dimension. subQ-CQL algorithm propose a learning space dimension which is higher than DQL but reasonably less than CQL method. Huge learning space dimension which is the case of CQL brings the scalability problems and results in high computational cost and application difficulties.

When all metrics are evaluated together, it is seen that subG-CQL algorithm offers the optimal solution among all approaches. It has sufficiently high CTR values, good ITR values and acceptable learning space dimension. These results indicate that division of the system into sub-groups have no adverse effect on system also.

7 Conclusions

Multi-robot system environments are dynamic and contains high-level uncertainty due to independent sensing, decision-making and acting facilities of robots. Q-learning method provides optimal solution for robotic applications, but it is problematic to use in multi-robot domains. When decentralized learning structure is processed, robots do not take care the behaviors of their teammates and they ignore the effects of dynamic environment characteristics. These are the main reasons that the optimal task allocation cannot be reached. Additionally, behavior conflicts occur because of the independent actions of team members. The biggest advantage of this structure is that the small learning space dimension brings low computational load and easy application. When centralized Q-learning is used, the multi-robot system can be considered as Markovian, because the overall state space of the environment and joint action spaces of all robots are taken into account. This means that the major requirement of Q-learning to converge the optimal solution, is ensured. However, it is quite difficult to use centralized Q-learning especially for large systems. Computational cost is so high due the huge joint action space dimension increasing exponentially in the number of robots. Also, gathering the full knowledge of environment states and agents' actions needs perfect communication capability among robots. In this study, an efficient solution has been developed to use Q-learning in multi-robot systems. The subG-CQL algorithm, proposed here, divides the whole system into sub-groups which each of them behaves as an independent small-sized system. This process is carried out in a way that does not lose the task performance of the system. The combination of these sub-groups constitutes the whole system. Each sub-group use Q-learning in centralized manner. Because they have small system size in number of robots and number of tasks performed, the computational cost and communication

requirement is reduced to a reasonable level. Thus, all the advantages of centralized learning are utilized. The experiments are realized comparatively for four different approaches: centralized Q-learning, semi-centralized Q-learning, decentralized Q-learning, and the proposed algorithm, subG-CQL. It is seen that a fairly good system performance is obtained by the proposed algorithm, subG-CQL, in terms of completed task ratio, idle time ratio of robots and learning space dimension. It successfully combines the advantages of decentralized and centralized learning schemas such as convergence to optimal solution, low learning space dimension, low computational and communication cost, and easy application. The experimental results emphasize the effectiveness of the proposed algorithm.

8 Author contribution statements

In the scope of this study, the Hatice Hilal EZERCAN KAYIR contributed for all stages which include the formation of the idea, the literature review, the construction of the theoretical background, the design and application of the study, supplying the materials used, the assessment of obtained results, the spelling and checking the article in terms of content.

9 Ethics committee approval and conflict of interest statement

There is no need to obtain permission from the ethics committee for the article prepared.

There is no conflict of interest with any person/institution in the article prepared.

10 References

[1] Arkin RC., *Behavior-Based Robotics*. Massachusetts, USA, MIT Press, 1998.

[2] Dai W, Lu H, Xiao J, Zeng Z, Zheng Z. "Multi-robot dynamic task allocation for exploration and destruction". *Journal of Intelligent & Robotics Systems*, 98, 455-479, 2020.

[3] Bernstein DS, Givan R, Immerman N, Zilberstein S. "The complexity of decentralized control of Markov decision processes". *Mathematics of Operation<s Research*, 27(4), 819-840, 2002.

[4] Mataric MJ. "Reinforcement learning in multi-robot domain", *Autonomous Robots*, 4(1), 73-83, 1997.

[5] Gerkey BP, Mataric MJ. "A formal analysis and taxonomy of task allocation in multi-robot systems". The *International Journal of Robotics Research*, 23(9), 939-954, 2004.

[6] Gerkey BP, Mataric MJ. "Sold!: auction methods for multi robot coordination". *IEEE Transactions on Robotics and Automation*, 18(5), 758-768, 2002.

[7] Dias MB, Zlot RM, Kaltra N, Stentz A. "Market-based multirobot coordination: a survey and analysis". *in Proceedings of the IEEE*, 94(7), 1257-1270, 2006.

[8] Zlot R, Stentz A, Dias MB, Thayer S. "Multi-robot exploration controlled by a market economy". *IEEE International Conference on Robotics and Automation*, Washington DC, USA, 11-15 May 2002.

[9] Lagoudakis MG, Markakis E, Kempe D, Keskinocak P, Kleywegt AJ, Koenig S, Tovey CA, Meyerson A, Jain S. "Auction-based multi-robot routing*". Robotics: Science & Systems*, Massachusetts, USA, 8-10 June 2005.

[10] Mosteo AR, Montano L. "Comparative experiments on optimization criteria and algorithms for auction based multi-robot task allocation". *IEEE International Conference on Robotics and Automation*, Roma, Italy, 10-14 April 2007.

[11] Zlot R, Stentz A. "Market-based multirobot coordination for complex tasks". *International Journal of Robotics Research Special Issue on the 4th International Conference on Field and Service Robotics*. 25(1), 73-101, 2006.

[12] Hanna H. "Decentralized approach for multi-robot task allocation problem with uncertain task execution". *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Alberta, Canada, 2-6 August 2005.

[13] Ezercan Kayir HH, Parlaktuna O. "Strategy planned Q-learning approach for multi-robot task allocation". *Proc. 11th International Conference on Informatics in Control, Automation and Robotics*, Vienna, Austria, 1-3 September 2014.

[14] Ezercan Kayir HH. "Experienced-task based multi robot task allocation". *Anadolu University of Sciences & Technology-A: Applied Sciences & Engineering*, 18(4), 864-875, 2017.

[15] Ezercan Kayir HH. "Q-Learning Based Failure Detection and Self-Self-Recovery Algorithm for Multi Multi-Robot Domains". *Elektronika Ir Elektrotechnika*, 25(1), 3-7, 2019.

[16] Busquets D, Simmons R. "Learning when to auction and when to bid". *Distributed Autonomous Robotic Systems*, 7, 21-30, 2006.

[17] Farinelli A, Iocchi L, Nardi D. "Distributed on-line dynamic task assignment for multi-robot patrolling". *Autonomous Robots*, 41(6), 1321-1345, 2017.

[18] Scheider J, Apfelbaum J, Bagnell D, Simmons R. "Learning opportunity costs in multi-robot market based planners". *International Conference on Robotics and Automation*, Barcelona, Spain, 18-22 April 2005.

[19] Jones EG, Dias MB, Stentz A. "Learning-enhanced market-based task allocation for oversubscribed domains". *IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, USA, 29 October-2 November 2007.

[20] Tian YT, Yang M, Qi XY, Yang YM. "Multi-robot task allocation for fire-disaster response based on reinforcement learning". *Eighth International Conference on Machine Learning and Cybernetics*, Baoding, China, 12-15 July 2009.

[21] Yang E, Gu D. "Multiagent reinforcement learning for multi-robot systems: a survey", Department of Computer Sciences, University of Essex, UK, Technical Report, 2004.

[22] Matignon L, Laurent GJ, Le Fort-Piat N. "Hysteretic Q-learning: an algorithm for decentralized reinforcement learning in cooperative multi-agent teams". *IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, USA, 29 October-2 November 2007.

[23] Buşoniu L, Babuška R, Schutter B. "A comprehensive survey of multiagent reinforcement learning". *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, 38(2), 156-172, 2008.

[24] Russel S, Norvig P. *Artificial Intelligence a Modern Approach*. 2nd ed. New Jersey, USA, Prentice Hall, 2003.

[25] Tuyls K, Nowè A. "Evolutionary game *theory and multi-agent reinforcement learning". The Knowledge Engineering Review*, 20(1), 63-90, 2005.

[26] Sutton RS, Barto AG. *Reinforcement Learning: An Introduction*. Massachusetts, USA, MIT Press, 1998.
[27] Watkins CJ, Dayan P. "Q-learning". *Machine Learning*, 8, 279-292, 1992.
[28] Hu J, Wellman MP. "Nash Q-learning for general sum games". *Journal of Machine Learning Research*, 4, 1039-1069, 2003.
[29] Boutlier C. "Planning learning and coordination in multiagent decision processes". *6th Conference on Theoretical Aspects of Rationality and Knowledge*, TARK'96, Renesse, The Netherlands, 17-20 March 1996.
[30] Wang Y, de Silva CW. "Extend single-agent reinforcement learning approach to a multi-robot cooperative task in an unknown dynamic environment". *IEEE International Joint Conference on Neural Networks*, Vancouver, Canada, 16-21 July 2006.
[31] Martinson E, Arkin RC. "Learning to role-switch in multi-robot systems". *IEEE International Conference on Robotics and Automation*, Taibei, Taiwan, 14-19 September 2003.
[32] Tan M. "Multi-agent reinforcement learning: Independent vs. cooperative agents". *Tenth international Conference on Machine Learning*, Massachusetts, USA, 27-29 June 1993.
[33] Matignon L, Laurent GJ, Le Fort-Piat N. "Independent reinforcement learners in cooperative markov games: a survey regarding coordination problems". *The Knowledge Engineering Review*, 27(01), 1-31, 2012.
[34] Tesauro G. "Extending q-learning to general adaptive multi-agent systems". *16th International Conference on Neural Information Processing Systems*, Vancouver, Canada, 8-13 December 2003.
[35] Kok JR, Vlassis N. "Collaborative multiagent reinforcement learning by payoff propagation". *Journal of Machine Learning Research*, 7, 1789-1828, 2006.
[36] Park KH, Kim YJ, Kim JH. "Modular Q-learning based multi-agent cooperation for robot soccer". *Robotics and Autonomous Systems*, 35, 109-122, 2001.
[37] Wang Y, de Silva CW. "A machine-learning approach to multi-robot coordination". *Engineering Applications of Artificial Intelligence*, 21(3), 470-488, 2008.
[38] Oliehoek FA, Amato C. A Concise Introduction to Decentralized POMDPs. Switzerland, Springer, 2016.
[39] Sunehag P, Lever G, Gruslys A, Czarnecki WM, Zambaldi VF, Jaderberg M, Lanctot M, Sonnerat N, Leibo JZ, Tuyls K, Graepel T. "Value decomposition networks for cooperative multi-agent learning based on team reward". *17th International Conference on Autonomous Agents and Multiagent Systems, Stockholm*, Sweeden, 10-15 July 2018.
[40] Rashid T, Samvelyan M, Schroeder C, Farquhar G, Foerster J, Whiteson S. "QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning". *Journal of Machine Learning Research*, 21, 1-51, 2020.
[41] Son K, Kim D, Kang WJ, Hostallero D, Yi Y. "QTRAN: Learning to Factorize with Transformation for Cooperative Multi-Agent Reinforcement learning". *36th International Conference on Machine Learning*, Long Beach, California, USA, 9-15 June 2019.