Int. J. Nonlinear Anal. Appl. In Press, 1–34
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2024.33578.5012

A research on mobile crowd sensing with regard to heterogeneous task allocation and deep reinforcement learning based on Internet of things based on Stackelberg game theory

Zohreh Vahedi[a], Seyyed Javad Seyyed Mahdavi Chabok[b,*], Gelareh Veisi[a]

[a]*Department of Computer Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran*

[b]*Department of Electrical Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran*

(Communicated by Seyed Hossein Siadati)

Abstract

Today, with the rapid growth of Internet-based service delivery services, the realization of numerous applications, including mobile mass surveillance, has become possible. In mobile crowd sensing, equipment located at the edges of the network can be used to provide computing services, storage and execution of functions that have time priorities. Despite the many studies that have been done in the past on the application of the mobile crowd sensing approach, the management of handling heterogeneous requests by considering the quality of service has not been comprehensively investigated yet. Therefore, the main goal of this paper is to provide an approach to allocate heterogeneous tasks in the form of implementing mobile crowd sensing in such a way that both the period for the completion of the activity is reduced and the quality of coverage and service level are observed at an optimal level. Since the participating groups in such an approach have conflicts of interest, therefore, the Stackelberg inverse game theory has been used as a tool to manage the level of user participation and consider the benefit of all players. One of the features of this game model is the possibility of implementing it without having complete information about all players. In order to reach the equilibrium point of the game, the optimal strategy of the applicants is determined by using the deep reinforcement learning algorithm, because this method can be useful in finding the appropriate proposed strategy by using the history of interactions. One of the important challenges when applying learning algorithms is the lack of stability during the execution of the learning process. In this regard, an approximate policy has been used to approximate the values of the reward function, which prevents divergence during the implementation of the learning process. Another important challenge is knowing the density of user participation in mobile mass monitoring programs. The higher the number of monitoring nodes in an area, the better the coverage quality can be. For this purpose, the fuzzy system has been used, which can estimate the level of participation density by having the time range of users' presence in the study area and the level of geographic density. In this paper, three characteristics of activity completion time frame, service quality and coverage level have been evaluated. According to the obtained results, the use of such an approach increases the coverage level by more than 17% compared to the average of common methods.

Keywords: mobile crowd sensing, heterogeneous task allocation, deep reinforcement learning, internet of things, reverse Stackelberg game
2020 MSC: 68T07, 91A80

*Corresponding author
Email addresses: zohrehvahedi@mshdiau.ac.ir (Zohreh Vahedi), mahdavi@mshdiau.ac.ir (Seyyed Javad Seyyed Mahdavi Chabok), gveisi@gmail.com (Gelareh Veisi)

Received: February 2024 *Accepted:* April 2024

1 Introduction

With the rapid expansion of the sensing, computing and communication capabilities of mobile devices, mobile mass monitoring (MCS), in which a large number of participants equipped with mobile devices monitor the activities of an area in real time, has emerged as a new measurement paradigm and has attracted much attention. has attracted [13, 56, 84, 106]. In recent years, several platforms have taken advantage of MCS, such as Waze [114], Million Agents [66], Medusa [78], and Prism [25], and have been widely used in various fields such as intelligent transportation [131], healthcare [65], environmental monitoring [123] and urban public management [38] have been used.

In the MCS structure, the mobile device carried by the participants is used as one of the basic measurement units. In the first step, the structure of the system must assign functions to the appropriate participants. Then the tasks are executed, and the data measured by the participants is loaded into the system. Finally, the obtained data is evaluated and processed to implement the functions. In other words, mobile mass monitoring is an emerging paradigm in which citizens share the data generated on their mobile devices and developers use this data to extract collective information and provide people-oriented services. In MCS, task assignment strategy is a fundamental issue, and its study has gone through several stages of development. The first research focused on single-task allocation [128]. However, with more use of MCS applications, the generated functions gradually increased. Due to the limited sensing resources (for example, the number of mobile devices, types of sensors, and energy resources), multitasking under limited resources has been proposed and has become a research focus [39]. In addition, since the observation activities of the participants are affected by time and place, the study of spatio-temporal characteristics is also of great importance for allocating tasks. In this situation, various sensing devices generate a large amount of data and consume a lot of resources, including bandwidth, memory capacity, and a huge amount of energy. As a result, the quality of service (QoS) required for applications may not be achieved [51]. Therefore, service quality is one of the important challenges in this field. Even with significant advances in technologies, mobile devices are still not able to run software with high processing power (along with suitable quality for users), such as audio and image processing and data mining in social networks. In this situation, mass mobile crowd sensing requires a high volume of participants in the process of monitoring the surrounding environment using sensing devices such as smartphones. In other words, this is an emerging phenomenon that is the result of combining the distributed model of the Internet of Things (IoT) and crowd-based projects. Program Mobile crowd-sensing applications are deployed on participating nodes such as mobile phones and personal devices that can monitor the physical environment and provide monitored data to the service application. In such a large structure, monitoring devices continuously produce a large amount of data, which consumes a lot of resources, including bandwidth and energy [33]. With the studies conducted on the results of the implementation of the initial plans, it is clear that the participation of users and their adaptation to the programs provided in the platform of mobile mass monitoring plays a very important role in the proper implementation of requests [7, 49, 119].

One of the important existing challenges is the energy consumption to complete the MCS functions, which is caused by collecting sensor data, checking the data locally and transferring the data to the corresponding servers. If the implementation of the requested monitoring functions requires high energy, it is no longer possible to complete the processing of received requests with the help of resources located in the nodes located at the edge of the network, but it is also necessary to use cloud space in this regard. Of course, it should be noted that although cloud resources have high computing capacity and huge memory, their use has a relatively high cost, and on the other hand, in some situations, they are not available to subscribers. In addition to these cases, due to the need to transfer information to the cloud space and the requirement to perform processing operations on the data and then send the final information to the subscriber, considerable time is spent, so using this cloud space is not suitable for time-sensitive functions and On the other hand, for effective communication with the resources located in the fog space, it is necessary to install all the necessary infrastructures [17, 59].

1.1 The structure of the internet of things

Today, Internet of Things (IoT) technology is used in various fields, including traffic control, managing the health status of patients, and creating security in urban spaces. One of the reasons for the realization of this is the growth of modern technologies and the development of smart equipment that is able to realize two-way communication. In this structure, billions of different devices are connected with each other and can issue or receive orders. This type of communication consists of protocols such as 6LoWPAN and IEEE 802.15.4, in which smart objects alone and without the physical presence of humans are able to report their performance or influence other objects [6]. In the structure

of the Internet of Things, a wide variety of services can be defined in which wide challenges can be solved instantly by establishing a connection between equipment equipped with modern technologies. Different elements can provide multiple functions. For example, a virtual machine can also request an activity from the network or, in exchange for a specific income, allow the network to use its potential capacities [98].

Due to the large volume of production data, the management of information exchange and how to process it is very important. In this network, which consists of a lot of equipment with very diverse requests, a huge flow of requests is presented at every moment, and their accurate implementation and proper management are one of the basic elements in expanding their user area. For this purpose, a triple-layering consisting of an information receiving layer, a network layer and an application layer is used for IoT-based networks. In this structure, first the data is collected by the sensors and then sent to the processing centers in the form of the information receiving layer. In this situation, pre-processed data and redundant information are removed. Further, the transfer of information and its exchange among different machines is realized through the network layer. When the data is fully processed, the extracted information should be used in the considered equipment. This is done through the application layer, which is the final layer responsible for executing the exported signals. As a result of expanding the application scope of the Internet of Things, virtual workspaces will change a lot. In this situation, a huge amount of data is produced that needs to be stored, processed and transferred. The management of this huge amount of information in an efficient way in fifth-generation (5G) and sixth-generation (6G) networks has become doubly important. In this regard, one of the most useful tools is the use of cloud space. Cloud memories are able to store a huge amount of information and transfer it between virtual machines. When data is collected by sensors and from highly distributed locations, it is often transferred to the cloud and distributed to different systems for processing. In this situation, it is necessary to observe several criteria such as service quality, maintaining data security, and processing within the allowed time frame [53].

1.2 Using the capacity of fog nodes

One of the basic challenges in using IoT-based networks is considering the time delay in sending, processing and receiving results from users. Response delay consists of three parts:

- Information sending time frame
- Processing time
- The time frame for receiving results by users

The sending period includes the time when the request is registered in the system and sent to the network for processing. The processing time interval depends on the type of the issued request and the amount of memory and volume of the processing capacity, and the receiving time interval also indicates the time when the results of the processed request reach the subscriber. Submitted requests are entered into the system as a continuous flow. If there is a delay in the processing of one of the activities, it will cause the program to crash, which will cause other activities after that and cause a wide disturbance in the system. The management of the sent requests should be done in such a way that the processing functions are executed with the least possible delay and maximum use of the available capacities. Although the use of cloud space is very common, sending all the information to a central point and processing a huge amount of data that increases every day has made it face serious challenges. In this regard, Cisco has taken advantage of an interesting initiative. By defining the fog space as an intermediate space between the equipment located at the end of the network and the cloud space, a new processing system has been introduced. The cloud space is a distributed structure that first evaluates the requests of users and, if they require high computing capacities or huge storage capacity, they send them to the cloud space; otherwise, they are sent to nodes called cloud nodes. By using the nodes located on the edge, most of the activities can be implemented, and the results can be sent to the users. This not only significantly reduces the delay created in the field of servicing, but also greatly improves the overall accurate performance of the network by reducing the amount of processing done in the cloud [16]. Fog space is a new structure in computer networks that allows the use of capacities located at the edge for processing activities. The equipment used in this space provides processing and storage capacities, which are known as fog nodes (FN), and are used to realize the following:

- Effective allocation of available resources for appropriate activities
- Removal of generated redundant data
- Reducing the time period of performing activities

The processing capacity of mobile users consists of several physical machines (PM). Each physical machine can have one or more virtual machines (VM). Based on the location of the requested task, the type of the desired task, the amount of processing capacity and the amount of memory required, these tasks are assigned to virtual machines in the IoT network. In the system in question, requests are sent to fog nodes through the IoT network. Fog nodes are responsible for controlling, managing and sending/receiving data to virtual machines. The communication method of users located in the fog space with the server located at the edge of the network is shown in Figure 1. In this situation, it is very important to use an effective strategy in this field [67]. The purpose of mobile crowd sensing is to allocate requested heterogeneous tasks to virtual machines located at the edges of the network so that the desired services are implemented in the desired time frame and the service quality is maintained at an optimal level. On the other hand, the range of requested activities in this model of networks is very diverse. From a simple text file to the processing of videos obtained from closed-circuit cameras and identification of people and objects through sent photos, they can be in the set of requested activities. In this situation, due to the limitations of resources and capacities placed in the cloud nodes and their dynamic changes, the purpose of resource allocation planning is to manage the processing of the requested activities by using the capacities placed in the cloud and cloud nodes in such a way that the data service subscribers are satisfied. (DSS) which includes not delaying the set allowed period and sending accurate answers, and achieving the defined indicators in the field of service quality.

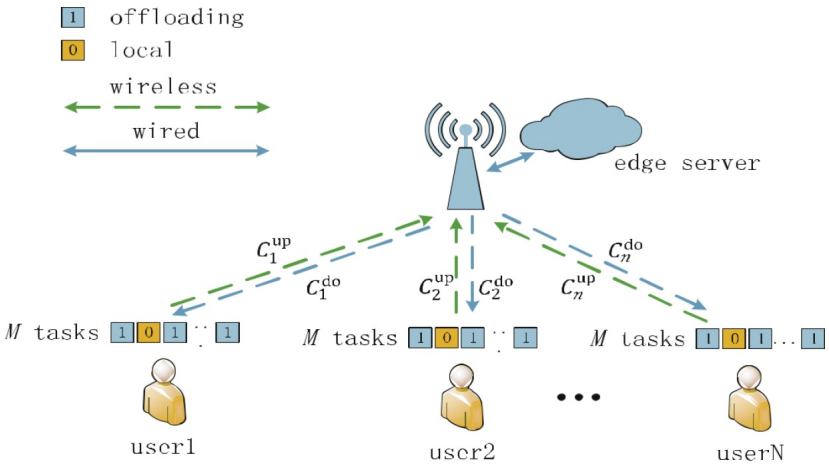

Figure 1: Schematic of how the fog layer is connected with the cloud layers and subscribers [31]

For the optimal management of requests, the following should be considered:

- Increasing the access rate of users to mass monitoring services
- Improving service quality for a wide range of heterogeneous functions
- Improve information transfer capacity
- Using as much resources as possible in the edge nodes
- Reducing delay in response
- Minimizing the amount of energy consumed by improving the amount of productivity
- Reducing the possibility of losing sent requests

In summary of the above, it can be concluded that the use of the cloud system for structures that use IoT technology is not possible due to the high volume of requests and high sensitivity to delay. Therefore, in order to prevent disruptions in the processing of requests, it is necessary to use a new approach, such as using fog. In this situation, the resources placed on the subscriber's side will be used to process and store information.

By using such capacities, it is not only possible to reduce the response delay, but also to increase the amount of people's participation in such approaches by taking advantage of distributed processing resources. In this way, with the increase in the density of participation, the amount of available resources has also increased, and mobile crowd sensing with higher quality can be implemented. On the other hand, the resources in fog nodes have a limited capacity, and

their distributability and availability level change drastically. Therefore, to manage the amount of available resources, it is necessary to present an approach that takes into account both the capacity of the resources located in the fog nodes and the conditions of the sent requests in terms of the computing capacity and the allowed time frame of the performance, and finally the allocation of virtual machines to the requests following the profit of all players should be planned.

1.3 The importance of research

A mobile mass surveillance system consists of multiple servers and many mobile users. In recent years, MCS has been used in fields such as service delivery, intelligent transportation, and climate monitoring. In this structure, the applicants send their requests and the control and assignment center uses the capacity of mobile users' equipment located in the surrounding environment who are eligible to participate, for the required processing and implementations. Communication between applicants and mobile users is also realized with the help of networks equipped with the Internet of Things.

The level of realization of MCS is significantly dependent on the amount of user participation and the amount of service capacity of each user. In fact, mobile users have very limited storage and processing resources, and for optimal use of their available capacities, it is necessary to design a comprehensive plan. With the development of the MCS platform, the number and type of requests sent have increased day by day. In this situation, requests no longer follow a specific range and include a diverse range. This causes that for the implementation of several requests, the use of a virtual machine is enough and the obtained results are of the required quality, but for a number of requested functions, due to the high volume of the required processor system or the required memory, their implementation is not desirable. It needs a large number of virtual machines to work together in parallel. In this situation, the heterogeneity of the jobs causes competition among the applicants to obtain the limited resources available in the nodes located at the edge of the network.

Although several approaches have been presented in the articles for the allocation of activities [19, 57, 108, 109, 110], most of them have focused on the allocation of similar and single activities and on one goal, therefore the allocation of tasks for a set of multiple and heterogeneous tasks and considering Several objectives at the same time, including minimizing the activity completion time, maximum coverage level, and improving the service quality level in the Internet of Things (IoT) platform have not yet been comprehensively investigated. For this purpose, an attempt is made to provide a new method for allocating multiple tasks in mobile crowd sensing based on fog computing in IoT using inverse Stackelberg game theory and with the help of fuzzy logic and a deep reinforcement learning algorithm. Solving the discussed allocation problem requires an online and adaptable method due to the dynamic conditions governing today's computer networks and the difficulty of modelling them. The proposed method will be able to achieve an effective allocation strategy automatically based on its previous experience.

1.4 Research objectives and innovation

In reviewing the literature on the subject, it was found that improving the service quality, minimizing the activity completion time and increasing the coverage level in MCS systems are important challenges in this field. In this situation, the fog can be used to transfer data wirelessly between objects defined in the network more easily. Since the resources available at the edge of the fog have limited processing and storage capacity, therefore, the management of the tasks requested for implementation and the way of using the available resources significantly affect the efficiency of the mobile mass monitoring approach. The main goal in this treatise is to reduce the average time period of allocating jobs, along with improving the quality of coverage and service to the requests submitted in the cloud platform. For this purpose, an attempt is made to provide a novel method for assigning heterogeneous functions by using a deep reinforcement learning (DRL) method. Since requests are sent to the system sequentially, their allocation should be done in a very short period of time to avoid long queues for activities. be blocked pending processing. On the other hand, the range of sent requests has a high variety, and this heterogeneity must be taken into account in planning. In general, due to the continuous changes in the amount and quality of available resources and the wide variety of sent requests, the use of law-based approaches is not very efficient, and it is necessary to update the structure used continuously and according to the past information of the system. In this situation, two of the most important innovations applied in this treatise are as follows:

- Improvement of agent training with the help of approximate policy: In general, one of the major challenges that arise when training agents with policy gradient algorithms is the possibility of losing the target due to the weak and sudden performance of agents. Recovering from the situation in this case is very difficult because the agent

starts weak traces and uses it to train policies. In such a situation, the algorithms with policies are no longer able to use the data and these examples are not efficient for them. One of the ways to overcome this challenge is to use approximate policy optimization (PPO) algorithms. The main idea of introducing PPO is to introduce an alternative goal to prevent the weakening of efficiency by guaranteeing the uniformity of policy improvement and helping to achieve the optimal strategy.

- Use of fuzzy logic: In the real world, many times there are situations where it becomes very difficult to determine the exact state of the situations. In these situations, fuzzy logic can be used in order to design a flexible approach to make appropriate decisions. In other words, uncertainties can be modeled for any situation by applying this logic.

One of the most important uncertainties in such issues is the level of active participation of users in mobile mass surveillance. This amount of participation is highly non-linear and has a lot of complexity that cannot be easily calculated at any moment. In this paper, the fuzzy system is used to determine the density of participation in each time step. The time period of users' presence and the density of geographic location are defined as input membership functions for the fuzzy system, and the level of participation density is defined as an output variable so that by calculating its amount, allocation can be implemented with better quality.

In general, the main goal of the paper is to provide a comprehensive approach to manage the allocation of heterogeneous tasks in mobile crowd sensing in such a way that both the time period for the completion of the activity is minimized and the quality of coverage and service level are observed at an optimal level. For this purpose, game theory has been used to determine the level of user participation by considering the profit of all players. For this purpose, the inverse Stackelberg game theory is used, in which there is no need to have complete information of all players (job applicants (TI) as leading players and mobile users (MU) as following players). In the proposed approach, the deep reinforcement learning (DRL) method has been used in order to determine the strategies of the applicants in order to choose the optimal equilibrium point. One of the characteristics of DRL is finding the right strategy by using the history of interactions. In this category of problems, the goal is defined as maximizing the total rewards received by the agent. In this regard, the agent uses reward signals to determine the quality of his activities.

One of the important challenges when applying learning algorithms is the lack of stability during the execution of the learning process. In this regard, an approximate policy has been used to approximate the values of the reward function in the reinforcement learning method to prevent divergence during the execution of the learning process. Another serious challenge is the density of user participation in mobile mass surveillance programs. In fact, the higher the number of monitoring nodes in an area, the better coverage quality can be created. For this purpose, a fuzzy system was used to determine the participation density with the help of the time period of users' presence in the area under study and the level of geographic density. In the following, the most important achievements of the paper can be mentioned as follows:

- Applying the inverse Stackelberg game theory to allocate heterogeneous tasks
- Using Deep Reinforcement Learning (DRL) to determine the amount of reward for each agent in order to find the appropriate strategy to implement game theory
- Combining the approach of the approximate policy optimizer (PPO) with the reinforcement learning algorithm in order to improve the performance speed and prevent the divergence of agents in finding the appropriate policy
- Using fuzzy logic to determine the density of participation in order to improve the quality of game performance
- Studying and investigating characteristics such as energy consumption and quality of service in the processing infrastructure in fog

Currently, cloud computing provides the necessary infrastructure and software services to provide the requested services needed by users on the Internet. Due to the spectacular growth of cloud computing, the number of users and the number of demands are increasing rapidly, which creates a high volume of work on servers and computing resources and, as a result, the use of an efficient approach for optimal allocation of resources becomes more obvious. For this purpose, Stackelberg's game theory is used in [35]. The structure of the game is designed in such a way that the leader owns a large number of resources and plans the allocation of resources based on the request received from mobile users. The goal from the leader's point of view is to minimize the cost of using the resources located in the fog nodes, on the other hand, the goal is from the point of view of mobile users, it is intended to minimize the cost of responding and transmitting messages to the desired cloud node. For this purpose, the entire region is divided into

regions and a fog node is considered for each region (Figure 2). The network planner must plan the operation states of fog nodes in such a way that both the costs are minimized and the service quality is maintained at the desired level. Therefore, the objective function is determined as the optimal allocation of resources in each round of the game in order to minimize the cost.

$$(U_L^S, U_F^S) \in \left\{ \min \tau_L(U_L, U_F) : U_F \in argmin_{U_P \in \Omega_P} \tau_F(U_L, \widetilde{U}_F \right\} \tag{1.1}$$

$$\tau_L(U_L^s, U_F^{sub}) > tau_L(U_L^s, U_F^s) \tag{1.2}$$

$$\tau_p : \Omega_L \times \Omega_F \to \mathbb{R}, \quad \rho \in \{L, F\} \tag{1.3}$$

Leading optimal cost indicator: $\tau_L(U_L^s, U_F^s)$

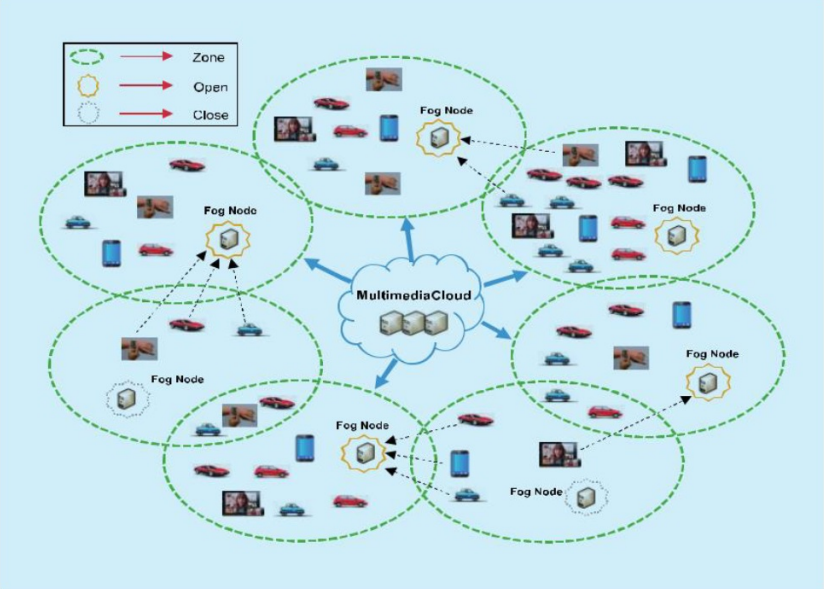

Figure 2: The resource allocation model used in [48]

2 Mobile mass monitoring

Mobile mass monitoring has emerged as a new monitoring model that effectively uses collective intelligence and user mobility in combination with advanced capabilities of smart technologies. The increase of mobile devices and the progress of the communication industry, computing power, storage space and multi-model monitoring capabilities have created this new monitoring model as mobile mass monitoring or human-centered monitoring [79, 111]. This new monitoring model can be used to implement applications designed based on the IoT infrastructure and take advantage of various sensors embedded in several points of the network to extract useful data. The MCS approach is an IoT service based on human stimulation that enables citizens to monitor individual or social phenomena by sharing data obtained from the environment by sensors [63]. In other words, MCS is a modern human-powered monitoring model that uses millions of individual mobile devices to monitor, collect and analyze urban data without using a large number of static sensors. This makes the considered monitoring operation to be realized at a low cost and with a high spatial-temporal coverage level [32].

MCS applications are focused on social activities on a large scale that cannot be implemented by using one sensor or one person alone, and when these data are valuable and valid, they are obtained in a wide time-space range with the help of many sensors [117]. Compared to networks equipped with industrial sensors, the structure of MCS by having manpower and mobile aspects guarantees better coverage and more accurate awareness of environmental conditions. In other words, MCS is a new monitoring model that uses sensors embedded in mobile devices and their use in the direction of monitoring and analyzing heterogeneous data and in large volume for different phenomena [77]. With the increasing number of mobile phones, which has now increased to more than 4.5 billion devices, as well as the diversity of users' movement patterns and daily activities, the possibility of people's participation in MCS-related activities

has increased. Currently, mobile phone devices are equipped with powerful sensors that are able to participate as an effective tool in the fields of comprehensive urban monitoring. In collective monitoring approaches used until today, the system operator, who is also known as the group owner (GO), specifies the type of expected results and the amount of incentives they receive by concluding contracts with the owners of smart equipment [14]. In this case, monitoring functions are assigned to virtual machines located in mobile nodes, which eliminates the costs of using and maintaining a large number of sensors [3, 73]. This has advantages such as better scalability, wider coverage, accurate measurements, data analysis with the help of various processors, improving flexibility and improving the level of service quality compared to traditional methods, but on the other hand, planning and managing the use of existing mobile resources also has complications. More and many challenges. In other words, the implementation of requests on a huge amount of data has a considerable cost, so that every person or company does not have the necessary financial power or computing resources to realize it, therefore, using cloud-based platforms that can The field of mobile crowd sensing is considered as an essential need in this regard.

3 Game theory

3.1 Characteristics of game theory

Complex decision problems have two common characteristics:

1. They include different factors with mutual benefits, that is, the action of one factor affects the rewards of other players.
2. They are dynamic in nature, that is, agents repeatedly compete or cooperate with each other over time, and their actions affect the evolution of the system state [9, 12, 71, 88].

In resource management issues, it is necessary to respect both the interests of subscribers and the interests of service providers. In such a situation, sometimes the profitability of one side is in contradiction with the profitability of another sector. One of the solutions that can consider the profitability of both groups is to use approaches based on game theory. Of course, it should be noted that this is only one of the practical aspects of using game theories. To better understand such a situation, consider a driver. He can influence the travel time of other drivers by choosing a specific route and a specific speed change process. When a large number of drivers take a common route at the same time, it causes traffic congestion. Or, for example, overfishing by fishermen in a short period of time poses challenges to the sustainability of the fishing period in a long period of time. These situations show how a lack of coordination between different players in a system may lead to conflicts of interest, and these examples show that where multiple parties act selfishly to optimize personal goals, it causes suboptimal use of available capacities. and there is a need to examine the mentioned challenge with the help of a holistic perspective.

For decision problems that deal with multiple individuals with conflicting goals, non-cooperative game theory proposes solutions in the form of finding an equilibrium point that meets the challenges imposed by the players' strategic behavior [88]. In [127], from the method of differential game theory to find the optimal amount of processing resources in order to maximize income Users are used. In this regard, the costs including the overflow caused by the network provider's performance delay and the creation of long queues created to provide services are considered, and Bellman's dynamic optimal planning technique has been used to find the equilibrium point. In [70], Stackelberg's game theory is used for resource allocation. The players of this approach are defined as network planners as leaders and MMU mobile multimedia users as multiple followers. The structure of the game is such that the leader owns a large number of resources and plans the allocation of resources based on the request received from the MMUs. The goal from the perspective of the leader is to minimize the cost of using the resources located in the fog nodes, and on the other hand, the goal from the MMU perspective is to minimize the cost of responding and transmitting the message to the desired fog node. For this purpose, the entire region is divided into regions and a fog node is considered for each region.

The network planner must plan the operation modes of the fog nodes in such a way that the costs are minimized and the service quality is maintained at the desired level. Therefore, the objective function is determined as the optimal allocation of resources in each round of the game in order to minimize the cost. To evaluate the performance of the introduced method, the authors have calculated the cost paid by each scheduler in the form of the number of services provided in each game, the number of serviced MMUs and the number of opened fog nodes and the response time for MMUs requests. In general, in classical game theory, it is necessary to have complete information about the sides of the game. This cannot be realized during the implementation of mobile mass surveillance, and therefore it is not possible to use game algorithms as mentioned in mobile mass surveillance. In order to overcome such problems,

in this treatise, the inverse Stackelberg game theory is used, in which there is no need to define all the characteristics of the players completely.

3.2 Stackelberg game theory

The Stackelberg game is based on knowing the complete information of the players and is a two-sided game. The leading players first choose an action from the set (Ω_L) and the following players, observing this situation, take their respective actions from the set (Ω_F) [36]. In this situation, all players seek to minimize the cost defined for themselves. It should be noted that the action of the following players depends on the action of the leading players. This sequence is shown in figure 2. Because the leader acts first, the action of the follower players is defined as a function of the leader's action ($U_F = f(U_L)$). This function represents a response curve for each leading player. In the condition that τ_F is differentiable, the response function corresponds to its partial derivative.

An equilibrium point in the Stackelberg game corresponds to finding an answer to the relation (1.1). Since the follower's optimal response ($U_L^S \in \Omega_L$) may not be unique, the game may lead to a non-optimal value of the leading value-cost function (Relation (1.2)). Therefore, there is an alternative solution according to which the forwarder makes another decision ($U_L \neq U_L^S$) which may lead to achieving a better objective function value.

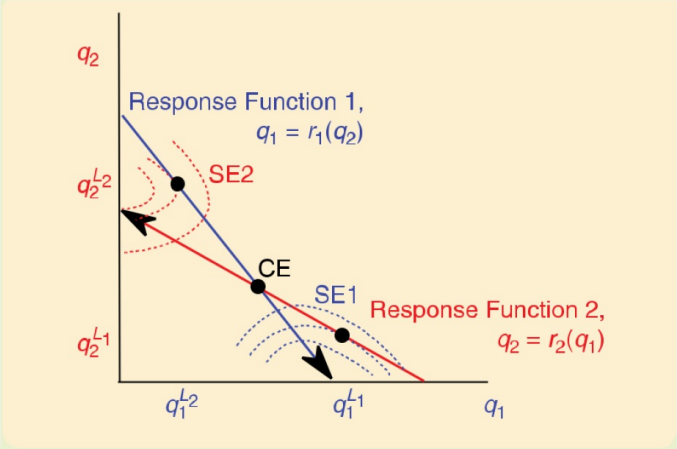

Figure 3: Finding the equilibrium point in the Stackelberg game [36]

The two-player Stackelberg game model was designed in 1934, and since then, more developed models than the two-player Stackelberg game have been presented, for example, it is possible to consider additional players with a flexible leader-follower role in the absence of He pointed out that there is enough information about them [89].

4 Reinforcement learning

One of the types of machine learning algorithms is the reinforcement learning method, which helps the agent to choose the action that will have the highest possible reward in the long term, according to the defined range of allowed actions and according to the amount of reward considered for each action. In this approach, prior knowledge is not needed at all, and based on the amount of rewards defined for different actions, the agent discovers the best path. In this method, various elements such as policy, reward function, and value function affect the agent and with the help of a sequence of discrete steps, interaction with the desired environment is established. Deep learning can also be used for online learning. Deep learning is a suitable option for machine learning systems that want to learn online and in a completely systematic way. Among the results of successful implementations of such an approach, we can mention the management of the movement of mobile robots, the learning of board games, and the automatic control systems of factories.

Positive and negative reward (penalty) system is used to manage the agent's activities. The agent has the duty to learn the optimal method of performing the action based on the rewards received in line with the actions he performs. For example, learning robots have a set of sensors and cameras that can perform a set of movements and receive a reward for performing each movement. In this situation, with the help of past actions and received rewards, the optimal policy is designed so that it can achieve its main goal with the maximum possible reward. Learning the policy helps the agent manage how it acts. For example, the considered program may be in the form of planning taxi lines

in such a way that all the services are done with the least fuel consumption and delay for the subscribers. In such a situation, adopting the best policy can have a great impact on factors such as drivers' income and air pollution.

In order to gain a better understanding of how to implement machine learning. Suppose that when the agent performs an action such as at from the set of actions A in state s_t, he gets the reward amount r_t. The value of r_t is the value of the momentary reward obtained by performing the action. The subscript t also refers to the time steps of the agent's interaction with the environment. The goal of performing actions by the agent is to maximize the sum of the obtained rewards. In this situation, the objective function is defined as a mapping that is able to map the action (at) from the set (A) to the state (s_t) from the set (S). to give There are many factors influencing the choice of strategy. For example, the certainty of actions or the indeterminacy of their results, the ability to estimate the state after performing each action or the inability to predict the future situation, the training of the agent by the trainer to perform optimal actions or learning only based on performing actions, can be effective in these conditions. Reinforcement learning has the following characteristics:

State space: At each time step t, the state s_t represents the current schedule for executing activities in VMs, and each VM is described based on the amount of available resources (CPU, RAM, bandwidth, and disk space).

Action space: action s_t represents the scheduled action at time step t for all considered activities on the available VMs. For each task, the action can be 0 or 1, which means whether or not the task is attached to the virtual machine. For example, when a task is represented as a vector (0,1,0,0,...,0), it means that a second virtual machine needs to execute it.

Receiving a reward: In classical learning, the policy (π) is defined so that if it is applied to the desired state (s_t), it leads to the action (at) ($\pi(s) = a$), but in reinforcement learning, a A series of rewards are considered for performing the action, which are temporary, because information from the system's past is not available.

Exploring the whole environment: The choice of strategy can change the type of training. The agent can explore new states by applying new actions or use his past actions to achieve the optimal reward. In this way, by choosing the type of strategy, other parts of the work environment can be evaluated and a better understanding of the conditions can be created by better searching.

Uncertain states: In many cases, all possible states are not known in advance. For example, in the case of moving cars that have front-facing cameras, the information around and behind the car is not clear. Usually, for such cases, it is appropriate to choose actions that increase the amount of observations and improve the level of awareness of the environment.

Continuous learning: In reinforcement learning, it is possible to continuously improve the learning process. In this situation, by gaining new experiences, the amount of information obtained from the environment is improved and the system will be able to show new abilities by combining the current results with previous tests.

Reinforcement learning (RL) is a branch of machine learning that is based on acquiring knowledge from the environment, adapting to the environment by improving behavioral strategies and making sequential decisions. In RL, it is assumed that an agent is embedded in the environment to implement actions. By exploring the environment and receiving feedback, the RL system creates an adaptive model without the need for Receive a large amount of data. Intuitively, RL includes a process in which agents continuously interact with the environment to make a sequence of decisions and improve decision-making capacities. In the mentioned structure, through continuous interaction with the environment, agents can perform operations and receive rewards corresponding to it, and the final goal is defined as maximizing the cumulative reward. As shown in Figure 4, in each episode in the t^{th} time step, the agent obtains the necessary information from observing the state of the environment, which is called the state of St. Then the agent acts according to the specified policy and receives the reward r_t and enters the S_{t+1} state based on it. In this situation, the policy is also updated based on the amount of reward received from the environment. The function generated by the agent is called a policy, which is responsible for mapping states to functions.

Deep Reinforcement Learning (DRL) is actually an integrator of deep learning and reinforcement learning methods, which integrates the ability to understand the deep learning method with the ability to make decisions in the reinforcement learning method. The most common DRL approach is the Deep Q Network (DQN) method. In traditional learning methods, the Q-table is used to store action values, while for problems with large dimensions, such as mobile mass monitoring, the possibility of storing all actions in tables and repeatedly searching to select the appropriate action in each state is time-consuming. And it is useless. Therefore, the DQN network uses the neural network to approximate the Q function and generate actions.

The policy scheduler ($\pi(a|s)$), which is responsible for mapping states to actions, assigns each task to a virtual machine. The momentary reward of such action is calculated as r_t. In fact, with every reward that the agent receives,

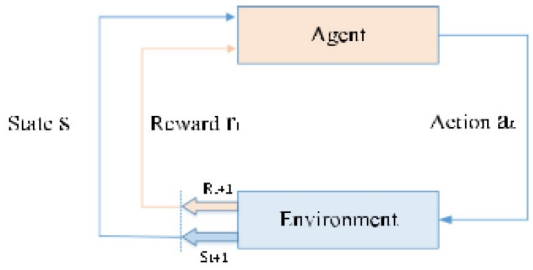

Figure 4: Schematic of the reinforcement learning structure [84]

the future state of the agent may change. The RL algorithm is implemented in such a way that it achieves the maximum possible reward in a wide time frame by choosing appropriate actions. If it is assumed that the environment under study has a Markov characteristic, then the RL framework can be considered as a Markov decision process (MDP) [99]. One of the most common approaches used in situations where the problem is included in the form of a Markov process is the Markov decision process (MDP), which is defined as equation (4.1) [75].

$$V^{\pi}(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+1} + \ldots = \sum_{i=0}^{\infty} \gamma^i \cdot r_{t+i} \tag{4.1}$$

$V^{\pi}(s_t)$: Discount cumulative reward for policy π in state s

s_t: state in step t

r_t: reward at the t^{th} step

γ: A coefficient to apply discount in rewarding

In a Markov decision process (MDP), the agent having a set of states (S) and actions (A), at each time step t, determines the state of the system (s_t) and causes the action at to occur. To perform this action, the state is changed to a new state (s_{t+1}) and the agent gets a reward (r_t). In Markov processes, the reward value and state depend only on the current state and are not a function of the system's past. In this situation, one of the best solutions to determine the optimal policy is the path that can achieve the highest reward by the agent. Of course, it should be noted that the longer the reward is received with a delay, the lower its amount will be due to the influence of the discount factor (γ). If the value of coefficient $\gamma = 0$, it means that the factor is not foresight and that moment is important for him. By increasing the amount of this discount factor, the share of bonuses also improves. This coefficient is used for the purpose of obtaining the reward as soon as possible. Of course, in some situations, the time horizon is limited.

In problems with a limited time horizon, rewards up to a certain time horizon are only considered, and in this situation, the average reward can be used according to the equation (4.2) [75]. The policy should be determined in such a way that the value of the function expressed in relation (4.1) is maximized. Such a policy is called the optimal policy, which is expressed as equation (4.3). In this regard, an example is given. Consider figure 5. Six squares represent six states and each arrow represents an action. The amount of reward considered for each action is also indicated numerically on each arrow. Only for the action that reaches the G house, the reward is considered, and the rest of the actions do not have any reward. State G is in the term an absorbing state because if it reaches these operating conditions, it will remain in these conditions. If in a problem like the aforementioned problem, all states, rewards and actions are known, the optimal policy for the function expressed in relation (1.1) can be obtained by determining the value of γ.

$$\lim_{h \to \infty} \frac{1}{h} \sum_{i=0}^{h} r_{t+i} \tag{4.2}$$

$$\pi^* = argmax V^{\pi}(s) \qquad \forall s \tag{4.3}$$

Next, consider the case where $\gamma = 0.9$. The figure placed in the lower part of figure 5 shows the optimal policy obtained in these conditions. Only one optimal policy can be defined for all states in the mentioned problem. The value of V^* is shown in the middle right figure. For example, consider the square in the lower and right corner of the figure 5. The value of the optimal value (V^*) for this square is equal to 100 because the policy has chosen an upward path and the reward for this action is equal to the value of 100. When the agent reaches state G, it remains in that unit according to the designed rewards. The value of (V^*) for the middle and bottom square is equal to 90. To get

from this point to the absorbing point, it is necessary to move to the right and then up. This amount of reward can be calculated through equation (4.4) [35]. The reward system no longer works after reaching the absorbing point.

$$0 + \gamma 100 + \gamma^2 0 + ... = 90 \tag{4.4}$$

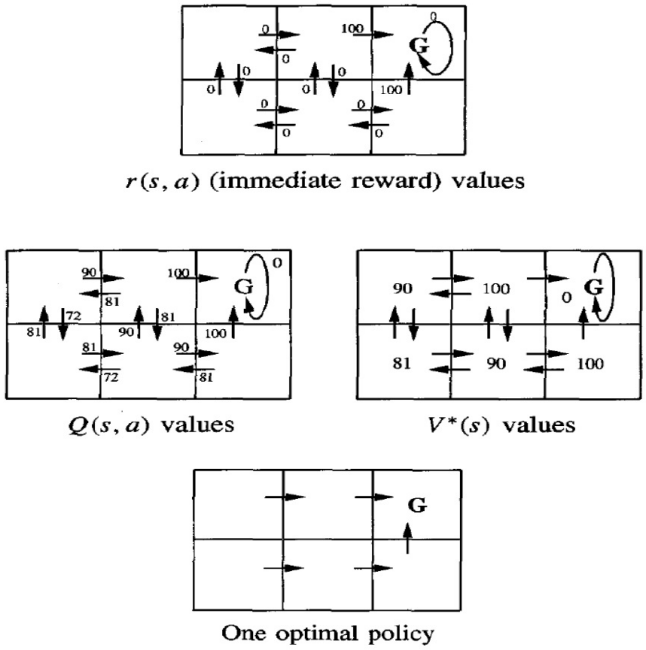

Figure 5: The structure of policy strategy learning [75]

5 Characteristics of reinforcement learning

At first, it is assumed that the conditions of the problem are in the form of a Markov process. Therefore, for RL settings, the Markov decision process (MDP) method is used, which contains the tuples shown in relation (5.1). The values of the actions of the RL agent are based on the policy determined for it ($\pi : s \rightarrow A$). This policy actually maps each state to a specific action. For each state, the state function - factor value is defined as the equation (5.2). The value function of state-action (Q) is also calculated according to equation (5.3). In this situation, the main goal of the agent is to find the optimal policy (π^*) based on which the expected amount of reward is optimized. This issue is shown in relations (5.4) and (5.5) [96].

$$tuple(S, A, P, r, \gamma) \tag{5.1}$$

$$V^{\pi}(s) = E\left[\sum_{k=0}^{\infty} \gamma^k \cdot r_{t+k} | s_t = s, \pi\right] \tag{5.2}$$

$$Q^{\pi}(s, a) = E\left[\sum_{k=0}^{\infty} \gamma^k \cdot r_{t+k} | s_t = s, a_t = a, \pi\right] \tag{5.3}$$

$$V^*(s) = \max_{\pi} V^{\pi}(s) \tag{5.4}$$

$$Q^*(s, a) = \max_{\pi} Q^{\pi}(s, a) \tag{5.5}$$

S: state space

A: Action space

P: state transition distribution matrix

r: reward signal

γ: discount factor

Learning these value functions is defined as a challenge in the field of RL, which has been studied by many and several approaches have been presented for it. The most common technique used in this field is the Q-learning method, which keeps track of the estimated values of the state-action function and is updated by the greedy objective function. In this situation, the discount factor (γ) creates control between momentary rewards and long-term rewards. While over many years and in numerous projects, the value of this parameter was considered fixed and as close to one as possible, but recent research shows that this approach cannot necessarily be chosen as the best solution [103]. In fact, a fixed value for the discount factor (γ) can lead to the occurrence of inconsistent behaviors in time, causing errors in the modeling of agent preferences and non-optimal search [76].

6 The heterogeneity of jobs

Mobile mass monitoring applications rely on a large number of participants who are willing to perform some tasks of their interest. In this situation, not only the functions have a great diversity and include from functions only for storage to functions that require complex and bulky processing, but also the range of participating users and virtual machines is also very wide. In this situation, the amount of priorities set for the requested functions, the amount of required memory capacity, the amount of necessary processing capacity and other characteristics defined for their processing are very different from each other, and this dynamic is due to the changes in the characteristics of the requests and on the other hand, the continuous changes in the defined resources. in the virtual machines participating in the mobile mass monitoring project is known as one of the main challenges raised in this direction.

One of the important features of most complex systems is the heterogeneity of their operating conditions. In this situation, the agents are equipped with special features, because the tasks sent to them require different processing capacities. In addition, sometimes an introduced task includes a number of tasks and requires multiple virtual machines for implementation [24, 85, 104]. For example, in a target tracking scenario, it is necessary to have several sensors connected together to cover the investigated area well. Another thing that can be mentioned is the implementation of relief operations during natural disasters. In such functions, it is necessary to achieve coordination between sensors and various components of the mobile crowd sensing network in such a way that the survivors receive the necessary assistance as soon as possible. Therefore, it is necessary to allocate jobs in such a way that the profit of all players is maximized.

In many cases, multiple simultaneous tasks have heterogeneous conditions, which is caused by the characteristics of diverse tasks and different fields of application of sensors in mobile mass monitoring. For example, the noise level in adjacent areas and successive time steps have significant differences, but the temperature distribution has a much smoother and more uniform trend. In fact, measuring the temperature of geographical areas requires a lower density of sensors than measuring the noise level. Even to perform a specific task such as measuring population density, the number of extracted data required for different areas and different time cycles also varies with each other because the movement of users and their geographic distribution are highly dynamic. Therefore, there is a need to design a flexible MCS platform that can support such heterogeneous multiple simultaneous activities.

With the advent of Internet of Things (IoT) technology, other traditional structures such as the use of cloud space have not been able to meet the needs of subscribers in an appropriate and high-speed manner because of many challenges such as bandwidth limitations, long service delays, lack of memory and system capacity. have been processed. According to Cisco's estimate, by 2022, more than 50 billion devices will be able to connect with each other with the IoT infrastructure [83]. In this situation, in order to meet needs such as knowing the geographic location, benefiting from real-time services and supporting the mobile mode of the equipment, it is necessary to use an interface between the IoT equipment located at the edge of the network and the cloud space. This interface, which is known as the Fog Computing space, is defined as follows according to reference [30]:

"The cloud space is a virtual platform that is able to provide processing services, storage and network functions between the equipment located on the edge of the network and cloud data centers."

Despite the potential benefits of the fog computing space structure, there are still challenges related to the dynamism, heterogeneity, and high complexity of requests sent by subscribers. In this regard, one of the most important existing problems is the management of limited resources defined in this space. In order to have a good level of quality of service (QoS), it is necessary to manage the capacity in the resources well so that it carries all the necessary criteria to improve the satisfaction of the subscribers. Based on this, the function of resource allocation approaches can be categorized into the following:

1. Map of available resources
2. Appropriate use of resources

3. Monitoring the status of resources [91, 105]

6.1 An overview of job assignment methods

In [107], deployment of nodes as static nodes has been used in order to improve the quality of data transmission. In this regard, the Archimedes curve was used to determine the required stationary nodes and the goal It is defined as minimizing the energy required for data transfer. For this purpose, one platform has been used to implement the proposed model, and messages have been sent between different equipment using the multi-version routing approach. In [92], the aware buffer routing protocol (EBRP) and effective in terms of energy consumption has been used to implement mobile mass monitoring. This protocol implements a dynamic clustering algorithm that can not only reduce costs by dynamically selecting appropriate cluster heads, but also improve the process of using available nodes by improving the rate of sending and receiving data. In [2], it has been tried to maximize the functions processed by users in mobile mass monitoring. In this regard, the mobile mass monitoring problem is first proved to be Hard-NP, and then the combination of greedy competition algorithms with linear optimization techniques has been used to solve it. However, in most of the studies conducted in the field of job allocation, some limitations such as not taking into account the time limit of users have not been taken into account.

One of the solutions for implementing multitasking mobile crowd monitoring is to use the active population approach, which has been used by the authors [87] for two modes of user movement (intentional movement and unintentional movement). For this purpose, the greedy genetic algorithm was used to ask the users to move to the location of the request in order to minimize the distance in the case of delay-sensitive functions. The plan presented in [59] used the capabilities of fog computing for the hierarchical scheduling strategy, but the problem that is noticeable in this plan is the inability of the approach to deal with the fluctuations in user requests in the fog space and the unstable situation created in the job processing queue. It is intended for cloud space. In the proposed approach, by changing the amount of requests, a decision is made only about the location of the task (cloud data set, fog nodes or edge). In [47], deep reinforcement structure based on Q learning (DQN) is used for optimal resource allocation in mobile edge processing (MEC). The designed optimizer system is defined in the form of minimization of energy consumption cost, computing cost and time delay cost. In this reference, the activities are assigned offline and due to the long-time delay, the presented approach cannot be used well in online problems. In [125], calculation in fog processing space has been used to allocate functions. The investigated criteria included quality of service (QoS), bandwidth and time delay reduction. In this regard, the deep learning model has been used to define the reward system. The main challenge raised in this reference has been the heterogeneity of activities.

In general, fog computing faces two major challenges of bandwidth limitation and energy supply resources [26]. In [132], a multi-task allocation problem is considered which includes the heterogeneity of participants in system decisions for resource allocation. In this method, a version of greedy particle swarm optimization in combination with genetic algorithm is presented to solve the challenges of the problem. The aim of this plan is to maximize the number of completed jobs by considering the set constraints. The simulation performed on the real data set shows that the proposed algorithm shows good performance under the conditions and precise settings of the parameters. In [42], researchers have presented a model for assigning tasks to mobile users, which is based on estimating the repetitive behavior of users and the use of their smartphones in collecting information from the environment. Based on the research, it can be concluded that the location of the user plays an important role in the costs. Therefore, in this reference, a solution is presented that manages the assignment functions according to the potential paths of the participating users. In this way, in addition to reducing costs, the quality of the extracted data is also significantly improved. Another point is that by increasing the number of jobs, the fee paid per user can also be reduced. In [133], a new task assignment framework is presented that predicts the movement model of users as much as possible and maps it to the presented task assignment approach. Based on the iterative movement model, the job assignment problem has become a pattern layout problem, and the results of extensive tests on real world data and simulated data show the effectiveness of the proposed framework in mobile job assignment. In general, one of the major challenges that arise during the training of agents is the possibility of losing the defined goal due to the poor performance of the agents. It is very difficult to recover from the situation in this case because the agent starts weak traces and uses it to train policies. In such a situation, the algorithms with policies are no longer able to use the data and these examples are not efficient for them. One of the ways to overcome this challenge is to use the approximate policy optimization (PPO) algorithms introduced in [82]. The main idea of introducing PPO is to introduce an alternative objective to avoid weakening the effectiveness of the policy by ensuring the uniformity of policy improvement. This objective function definition model also has the advantage that it can take advantage of unused data in policy design for the training process. In summary, the reviewed works in the field of resource allocation in fog space are collected in table 1.

Table 1: Comparison of the methods used to allocate jobs

The reference criteria were evaluated	The planning model of the approach used
[59]	The response time, the amount of cost and the number of user losses, loading jobs using the model of creating a queue, the Lyapunov function optimization method
[87]	Response time limit of bird optimization algorithm
[133]	Total execution time and resource cost of multi-objective task scheduling algorithm with adaptive neighborhood strategy
[42]	Latency weighted sum deep reinforcement learning technique
[92]	The service quality of the model of assigning jobs separately, clustering jobs based on fuzzy logic and linear optimization method
[59]	The amount of profit considering the limitation of convex optimization resources for proper allocation of power and genetic algorithm for its planning.
[83]	Response time, fog space energy consumption and lifetime of the DAG task loading network and the planning model of the innovative task loading method assigned to
[120]	The limitation of the execution time of the method
	Greedy job allocation method
	Improved MILP
[62]	Time - the cost of responding

6.2 A multi-objective heuristic method based on fuzzy logic

In many references, the independent task model has been used to model subscriber requests [46]. The method of exploiting quality-aware applications (QoE) presented by [52] works by clustering activities based on fuzzy logic. In addition to this, in order to model the priority priorities defined for the functions, the direct non-rotating graph (DAG) approach has also been used in this reference.

Heterogeneous earliest finish time (HEFT) can also control scheduling patterns in the environment with heterogeneous processors to shorten the completion time of tasks [101]. In this approach, first the activities are sorted and intelligently placed in ideal time slots so that the time to complete the activities is minimized. The HEFT method is considered among the best innovative techniques for job planning in terms of stability and performance. On the other hand, a mixed-integer linear programming (MILP) approach for modeling the loading of independent tasks with a limited deadline in the fog space has been presented by the authors [95]. In this method, IoT equipment is ignored and the capacity of the cloud computing space is assumed to be unlimited. Also, in order to solve multi-objective DAG function planning problems, the multi-objective HEFT approach has been used as a main framework to produce a set of solutions for decision makers in [28]. The provided solutions are ranked based on the distance from the population and the best solution is selected as the optimal solution set. One of the goals of such a decision is load balancing.

In [76], the interaction between minimizing the time of tasks and reducing energy consumption in cloud systems is considered, and based on this, the multi-objective optimization problem is formulated based on the dynamic frequency-voltage scaling system. In this regard, the non-dominant ordered genetic algorithm (NSGA) method was used, and then the artificial neural network method was used to predict suitable virtual machines based on the characteristics of the requested tasks. In order to improve the work scheduling process and prevent unauthorized job changes, in [37], the multi-agent cloud monitoring model based on performance has been used. In [61], a version of the modified particle swarm algorithm (PSO) is used to minimize the time required to complete tasks and improve resource utilization in cloud computing systems. In their proposed approach, the weight of particles is updated with the number of iterations and random weighting is used in the final stages. The purpose of this action is to escape from the local optimal points in order to reach the global optimal points. The aim of the authors [34] is to minimize three conflicting goals, which are:

- Construction time
- Use of resources
- Implementation cost

For these conditions, a multi-objective optimization problem is designed. After that, a discrete artificial bee colony technique based on fuzzy logic has been used to extract optimal beam solutions. In [113], the authors considered the trust levels of cloud resources in the scheduling process using a Bayesian approach, then presented a dynamic level scheduling algorithm to minimize the cost, time, and security of the task execution process.

In [129], a two-stage scheduling approach is proposed to reduce resource wastage in cloud data centers during the task allocation process. A Bayesian classifier is also proposed in the first step to cluster jobs based on timed data.

Dynamic scheduling algorithms have been used in the second step to assign tasks to suitable virtual machines. In [134], an optimal resource allocation framework is presented in which the providers can outsource their functions to external clouds when the current resources cannot meet the current demands. The main challenge in this regard is to derive an allocation strategy that also benefits the providers. to maximize and at the same time guarantee a high quality of service for users' jobs. This problem is formulated as an integer programming model and solved using a scheduling approach based on self-adaptive learning particle swarm optimization. In [10], an intelligent bio-inspired approach is proposed to derive solutions for optimal job scheduling for IoT applications in multi-process environments. In this regard, a hybrid algorithm is proposed by combining ant colony optimization techniques with genetic algorithm to select the best combination of functions in each step. In [118], using a game theory model, the authors have proposed a task scheduling algorithm that It is designed for energy management in cloud computing. By creating a mathematical model to deal with big data, they have proposed a scheduling model for load balancing for computing nodes.

The authors of [97] have introduced a user-based game theory formulated as a two-stage programmed game. In the first stage, a Stackelberg game has been used to attract the preferences of users' demands. In the second step, the authors have proposed a differential game to increase the rating of services provided by users in order to improve the service provider's position in the market and increase users' demand. In [40], the authors presented two new programmed models for the formation of cloud clusters using evolutionary game theory and genetic algorithm. To improve the profit of the whole group from generation to generation, the genetic algorithm explores the search space and finds the most suitable cloud, then the genetic model is improved by an evolutionary game that mainly considers the stability between groups. The main limitation of these methods is the scheduling of tasks, which works offline by trying to optimize a set of parameters each time a request is received. This requires high execution time and is inefficient for latency-sensitive tasks such as big data analytics in the cloud that require fast responses.

6.3 A review of mobile mass surveillance approaches

In [20], the authors have tried to exchange offline and online data and integrate their complementary features and the criteria used. In addition, as mentioned in this reference, human participation in data collection, processing and sharing will require the combination of human intelligence and machine intelligence. Appropriate planning of this combination is considered by these authors as an important design aspect for mobile crowd monitoring systems. In [59], techniques for reducing energy consumption in the process of mobile mass monitoring have been analyzed and investigated. In addition, it has been shown that the extraction of energy consumption factors in MCS and the analysis of the relevant features help in designing appropriate strategies to save energy consumption. Because citizens incur costs for participating in data collection (e.g., energy consumed from their batteries for data monitoring and reporting), designing data collection frameworks in energy-oriented designs will be a critical issue. In [133], the energy efficiency of several data collection approaches on mass monitoring in a real urban environment is investigated. Especially, these frameworks differ in terms of data reporting mechanisms and signal exchange between users and data collection units. The simulation results show that the energy consumption of different data collection methods is completely dependent on the mechanism of the data reporting system. Basically, continuous reporting for data collection can consume much more energy than contingent reporting. Although this continuous reporting approach guarantees high participation from users, it faces several challenges. After reviewing the existing studies in the field of mobile crowd sensing, the summary of the evaluations is stated below:

While distributed systems provide high computing power, but it is difficult to guarantee their reliability, in [59] a new scheduling method based on reinforcement learning is presented with the aim of providing reliable service, which focuses on improving the execution time with the computational complexity is low. The use of reinforcement learning algorithm in resource management makes the scheduler, who plays the role of agent in learning, adapt to the dynamic changes of the environment by considering the diversity of resources and functions and according to different situations. This method is significantly closer to the optimal solution by being aware of the availability of resources and receiving feedback from the environment.

Cloud service providers provide data centers to users, and their profit is estimated based on the level of users' access to services and the rental of virtual machines. In this situation, the reduction of energy consumption costs leads to an increase in their profits. Therefore, one of the main challenges for service providers is to minimize energy costs in data centers. In [120], taking into account the scalability and dependency of tasks, a new scheduling scheme based on deep reinforcement learning is presented, which is able to reduce energy costs at high scales with a large number of servers and multiple user requests. This method automatically makes the best decisions in the long term in terms of changing the environment in relation to user requests in the two stages of providing resources and scheduling jobs. This article only deals with the issue of energy cost reduction and does not consider other service quality parameters.

How to dynamically schedule resources and guarantee service quality in the long term is a key issue in the cloud environment.

In [121], a dynamic resource scheduling method based on automatic testing system using reinforcement learning method is presented in order to maximize the efficiency of the system in the long term in the cloud computing system, which leads to an increase in the demand of users, an increase in the utilization rate of resources and also a reduction in costs and Profits have increased. By making changes in the reinforcement learning method, this article has led to the improvement of its performance compared to the reinforcement learning method based on the pseudo-greedy method, which has had better performance and efficiency in providing resources in most cases. However, the proposed algorithm should be tested on a larger scale with more methods and functions. By developing it, a faster and near real-time algorithm can be provided that can also examine other parameters. In [62], a multi-objective and fair algorithm based on the branch-and-bound method for allocating functions in the cloud is presented. This algorithm obtained the set of beam solutions and used a method based on the concept of distance from positive and negative ideal solutions to select the final solution. However, meta-heuristic methods do not guarantee absolute optimality despite reaching a suitable solution, while the method presented in this article, taking into account the handover time and energy consumption, tries to distribute tasks appropriately along with simultaneously minimizing two time criteria. and energy, this algorithm does not take into account possible failures and problems during handover, as well as the unavailability of nodes to assign tasks, and does not guarantee the optimal solution for the long term. In [83], using the colonial competition algorithm, a new method for allocating available virtual machines in order to reduce energy consumption and reduce resource wastage is proposed. The results show that the presented method has achieved acceptable solutions compared to other algorithms. This algorithm does not take into account some limitations of cloud resources and the dynamics of the environment, and despite the reduction of energy consumption and waste of resources, it does not consider the two important criteria of work completion time and response time.

6.4 Machine learning approach

Because the different positions of the fog nodes and how they work directly affect the service quality of Internet of Things applications, therefore in [111], a scheduling problem in a hierarchical structure in the fog space is formulated and this problem is based on a strategy based on Deep reinforcement learning is solved. Based on the simulation results, the proposed scheme can provide better performance than the existing scheduling methods.

6.5 Movement pattern prediction

In [133], a new task allocation framework is presented which is able to predict the movement model of users as much as possible and turn the problem of assigning tasks to users into a pattern solving problem based on the repetitive movement pattern. The results of experiments on real world data and laboratory data show the effectiveness of the proposed framework in the optimal allocation of mobile mass monitoring functions.

6.6 Techniques based on game theory

Game theory is a powerful framework that is able to model interactions between multiple players based on specific goals and then suggest the most appropriate behavior in interactions with the help of the proposed model. This approach can be used in the design of distributed mechanisms where players seek to achieve their own interests. For example, in [120], an incentive structure for MCS using the Stackelberg game method is presented. In [111], a multimedia application based on quality-aware collective monitoring by the Stackelberg game is presented, and in [74], the Stackelberg game is presented to design a marginal revenue model for leading players, and in [22], a delay-sensitive approach is presented for use in theory. The game has been introduced. In [18] collective monitoring systems are modeled with a non-participative two-stage game and the behavior of leading players is evaluated under the participation of all players. In all the studied articles, the activity function model considered for the leading players is assumed to be the same, which is not consistent in the IoT space and considering the types of possible activities for users. The proposal of matching Stackelberg games and inverse Stackelberg games by considering incentives in control applications was first proposed in 1980. Examples of the use of this approach include strategies lacking complete information [44, 45], systems including high uncertainty [15], modeling the time delay in communicating [29], calculating the behavior of multiple players [102], estimating the current price in the electricity market [86], adaptive load control [60] and programming based on machine learning [80]. In table 2, the idea and innovation of different articles in the field of mobile mass monitoring are compared. When engaging intelligent systems to perform human activities, several challenges arise. Especially when due to the nature of human activities, there is a possibility of massive changes at any moment.

Table 2: Comparison of various mobile mass monitoring methods

The reference	The year of the idea of improving	The target parameters
[50]	2021	Combining the use of two different game theory techniques in combination with Q deep network techniques, agent policy optimization and gradient descent estimation of user participation level in mobile mass monitoring
[112]	2017	Management of jobs in collective monitoring with help
		Compound integer programming algorithm to increase the quality of collective monitoring and minimize the amount of energy consumption
[100]	2014	Optimization of random networks and distributed scheduling
		Network performance management with the help of distributed task scheduling to improve resource utilization capacity
[130]	2016	Combining greedy optimization methods with algorithms
		Linear programming to manage time interval minimization functions
		User participation
[81]	2018	Applying genetic algorithm and particle swarm algorithm in the direction of distribution of functions in the direction of collective monitoring of number maximization
		Processed requests
[41]	2018	distribution of non-homogeneous jobs with temporal and spatial dependencies using greedy competition algorithms to improve the quality level
		Minimizing the cost of monitoring
[116]	2018	Using the metal cooling algorithm
		In order to manage requests, allocating jobs on a real-time basis
		Minimizing the level of energy consumption
		Improving the quality of monitoring
[27]	2022	Improving the management of allocation of subscriber requests to help estimating their type of behavior, minimizing participation cost
		Increasing the level of information quality

In this situation, the defined functions can be different from one system to another. Therefore, the design of functions and their dynamic management during human-machine interaction is very important [4]. In [50], an experimental study has been conducted to explain the differences of different reinforcement learning methods and how they participate with game theory scenarios in order to create different levels of participation in artificial intelligence engineering problems. According to the review of the articles, it was found that the optimal allocation of tasks in mobile crowd sensing, taking into account the heterogeneity of activities and taking into account goals such as reducing costs and energy consumption in computing infrastructures based on fog and improving service quality indicators, has not yet been comprehensively investigated. Is. Therefore, in this treatise, the deep learning method (DRL) has been used to assign tasks in mobile mass monitoring. This technique is able to automatically and based on its previous experiences, over time, achieve an effective strategy for scheduling. In this situation, the participation rate of users is adjusted dynamically based on the quality of the requested service, and dynamic pricing of the service fee has been used to attract more participation when the participation rate is lower.

6.7 A review of the applications of the inverse Stackelberg game

The behavior of agents in multi-agent systems is often considered simultaneously. In this situation, players choose their strategies or decision values simultaneously, which are modeled as Nash equilibrium strategies. The strategy structure of Nash equilibrium has the characteristic that the strategy of each player is known as the best response against the strategies of other players [69]. However, the assumption of simultaneous decision-making is not applicable in situations where some players have more information than others and can implement their strategies before the actions of other players [93, 94]. In these cases, Stackelberg's equilibrium point can provide better performance for players [23, 90]. In this game, the calculation is done with the help of the leader-follower game strategy. In the typical Stackelberg game, the leader and the follower act sequentially based on their desired decision. Motivational strategies are used in other game models, which are called Stackelberg motivational games [43, 44, 45]. In this category of games, the leader starts the game by offering an incentive to the follower. If this game includes a mapping from the follower's decision space to the leader's decision space, an inverse Stackelberg game is created [72]. When the leader makes the desired decision, the follower's decision is directly derived from the leader's incentive function. In the original Stackelberg game model and in many of its subsequent applications, it is assumed that each player optimizes his own profit without concern for achieving a collectively optimal outcome. In order for the reverse Stackelberg game to be implemented, the leader must be able to reliably offer a favorable incentive to the follower, so that the follower believes that the leader is capable of performing the desired performance and does not violate it [8, 64]. In this sense, leading performance can be interpreted as an ultimatum strategy. The follower can still choose any decision value in his field.

However, the next leading decision associated with the following decision is already specified. If the follower does not match the desired decision value of the leader, the follower status is removed.

6.8 An overview of Q learning algorithm functions

In general, the RL algorithm provides a developed solution for complex decision-making processes. Although many methods for learning from the interactions have been presented so far, in RLs without a learning model, value functions are performed by estimating the desirability of an agent in a certain state or performing a certain action in a special state [96]. The quality of the state-action pair is usually evaluated by the amount of reward expected for future time steps. Accurate estimation of state-action values is considered as one of the basic elements in model-free RL methods and it is based on the fact that value functions define actions and allow them to react appropriately with their surrounding environment. There are many factors influencing the choice of strategy. For example, the certainty of actions or the indeterminacy of their results, the ability to estimate the state after performing each action or the inability to predict the future situation, the training of the agent by the trainer to perform optimal actions or learning only based on performing actions, can be effective in these conditions.

The main idea in the reinforcement learning method is to train the agent based on the changes that occur in the environment. One of the most common techniques used to determine the optimal policy in reinforcement learning methods is the use of Q learning algorithm. The Q learning algorithm is one of the reinforcement learning methods that does not need a model and uses the value of $Q(s_t, a_t)$ stored in the Q table to select the action. This algorithm has been used in various fields including advanced industrial processes, network control, game theory, robotics, operational search, control theory and image recognition. For example, in the cooperative stochastic game theory, the Q learning algorithm is used to maximize the profit of the whole system. By combining game theory and Q learning, effective resource distribution can be designed to improve the overall performance of the system. In addition, from learning Q to improve performance in games It has also been used with a huge scale and the presence of unsafe information [55, 115]. In the following, a brief overview of the various functions of the Q learning algorithm is presented in table 3.

Table 3: Comparing the achievements of different articles using the Q learning algorithm

The reference	The year of works carried out	Limitations
[57]	2017, A comprehensive review of augmentation algorithms applied to control approaches Using the Bellman equation to describe RL	Using offline and online methods to determine a policy to solve the control problems of non-categorization of algorithms and comparing their performance in the stated problems
[19]	2019, comprehensive review of Q learning user functions	Classification of methods based on the behavior of agents. Failure to provide details of technical and practical application of Q learning methods
		Lack of investigation of the methods of using multi-agents and their functions
[1]	2017, Multiple reinforcement learning classification with special focus on deep Q learning	Evaluation of features, challenges and deep reinforcement learning algorithms not providing a comprehensive mathematical background
		Lack of full review of introduced functions
[126]	2022, The use of Q learning algorithm in order to converge user participation and improve security conditions in mobile crowd sensing	Failure to model the movement of mobile users and changes in their dynamic conditions over time.

In order to optimally manage requests in mobile mass monitoring, it is necessary to increase the access rate of users to all types of services, improve the level of service quality, effectively use the available resources in the edge nodes, reduce the delay in response, minimize the amount of energy consumed and reduce the probability of loss. The sent requests should always be taken into consideration. In this situation, several challenges appear, including the impossibility of using cloud spaces due to the high volume of requests, limited and continuous changes of resources.

In fog nodes, he pointed out the need to respond quickly to delay-sensitive requests, consider the benefits of all participating groups, and the high variety of tasks proposed. On the other hand, although several approaches have been presented so far for the allocation of activities, they either focus on a specific model of functions, or the goals considered for them are only reducing the response time or minimizing the amount of energy consumed, or only from the user's point of view. The issue of allocation has been addressed. Therefore, the allocation of jobs for a set of multiple and heterogeneous jobs and taking into account several goals such as minimizing the time period for completing the activity, the maximum coverage level and improving the service quality level in the Internet of Things platform has not yet been comprehensively studied. In this regard, in this treatise, the combination of several methods has been used to overcome the challenges raised.

Since there are many participants in the platform of mobile crowd sensing, each group seeks to maximize the profit defined for itself, and in many cases this causes conflicts of interest, game theory has been used to make the profit of all players as The total is considered and because complete information of the players is not available in all time steps, therefore, the reverse Stackelberg game theory is used, which does not require complete information of all players to reach the equilibrium point. On the other hand, to determine the appropriate strategy for the applicants, the deep learning algorithm was used, which is able to design suitable solutions for the job providers by using the interactions that took place in the past. In order to improve the convergence in the agent training process in the reinforcement learning algorithm, the approximate policy technique is also used, which prevents the divergence and undesirable performance of the agents during policy training. On the other hand, due to the fact that the level of user participation is highly non-linear and has many complexities that cannot be easily calculated at any moment, therefore, fuzzy logic is used to estimate the time period of users' presence and the density of geographical location. They should estimate the level of participation density in order to improve the service quality. The main goal of this treatise is to allocate requested functions to virtual machines that are available and available to mobile users, taking into account features such as minimizing the time frame for assigning requests to virtual machines, maximizing the level of mass monitoring coverage by users, and improving the level of service quality to requests. To implement the proposed approach, the following assumptions are considered:

- In each virtual machine and for each time step, only one task can be executed (in other words, executing tasks in parallel is not possible).
- The scheduling method is such that it is not possible to change the assigned resource until the task is completed (in fact, if a task is assigned to a specific virtual machine, it cannot be used to perform another task until after the completion of that task).
- If the resource is assigned to a task, it needs to be queued when the next task arrives.
- The approach used in queuing is first-in-first-out. The sooner a task is requested, the more likely it is to allocate a virtual machine for it (sent tasks have a set priority, and if two tasks have the same priority, the execution priority is assigned to the task that was received earlier).
- The momentary reward and the next state after completing the task are calculated and stored along with the values of the initial state in the memory of the planner system, and this recorded information is used to determine the right strategy for the players and choose the right policy for the agents. In other words, the learning approach used lacks a model and the information recorded in the past is used for the training process of the deep reinforcement learning algorithm.

In order to properly manage the submitted requests, it is necessary to include things such as the time frame of users' presence and the participation of virtual machines in the monitoring plan, the density of available resources in terms of geography, the timely processing of requests in a suitable time frame, the minimization of the delay time for job allocation, and etc. to be considered. On the other hand, by reviewing the articles, it became clear to the author that the processing of requests with high diversity is done by using the capacities placed in the fog and in the context of mobile crowd sensing, taking into account characteristics such as the duration of execution, checking the level of participation and evaluating the level of service quality so far. It has not been comprehensively investigated. Therefore, to achieve these goals, a solution based on the inverse Stackelberg game theory, as well as the application of deep reinforcement learning algorithm, fuzzy logic and approximate policy, is used. The details of the proposed approach are described in the rest of this chapter.

In the proposed approach shown in Figure 6, two different groups of players participate. The first category is users or virtual machines that send their requests to the job manager for processing. In this situation, the request recognition unit is used to determine characteristics such as the type and priority of the sent request, the amount of time allowed for operation, the minimum required CPU capacity and the amount of capacity required for data storage. On the other hand, there are users or virtual machines that are ready to provide the capacity of the resources under their control in exchange for obtaining the minimum profit determined. In fact, this group of players has submitted their equipment specifications along with how to participate in the mobile mass monitoring program to the job manager's unit, and if deemed necessary, their participation in the processing of requests will be used.

In this paper, the requestors (TI) or the leading players in the reverse Stackelberg game are actually the same users or virtual machines that announce their required jobs to the job manager unit. For each sent task, specifications such as the amount of CPU capacity required, the priority of the assigned work, the maximum time allowed for

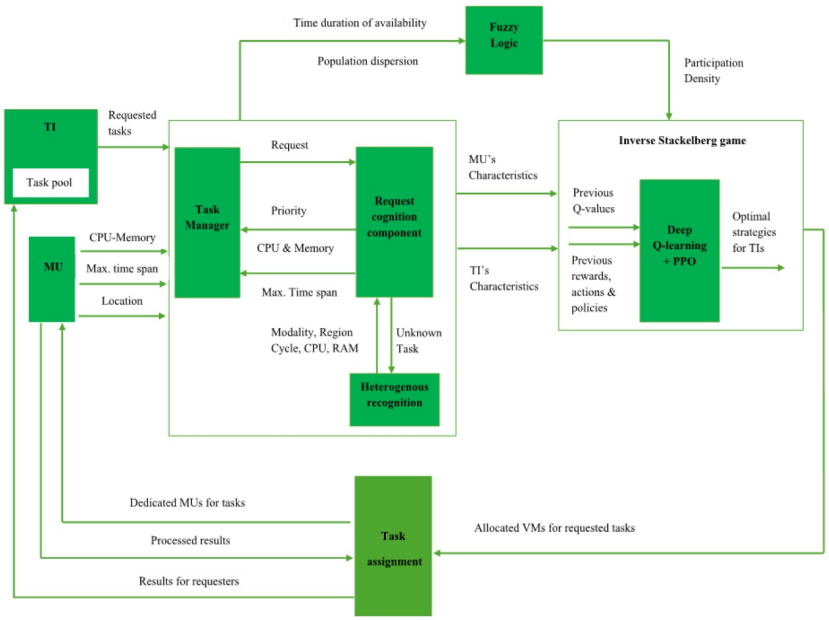

Figure 6: Structure of the proposed model

the operation, the amount of the maximum memory capacity required, the amount of cache memory, the number of cycles required to execute the instructions, etc. are determined and required. suitable virtual machines should be allocated for its execution. On the other hand, mobile users (MU) or players follow users or virtual machines that share resources with limited storage and computing capacities to perform the requested functions and earn money from this method. It should be noted that the processing capacities of mobile users are also made up of a number of physical machines, which also have one or more virtual machines. Allocation of tasks requested by leading players to virtual machines provided by following players is done by a unit called task manager. This is done with the aim of maximizing the profit of all players in total. In this situation, the ultimate goal of each leading player is to finish the request with the lowest possible cost, and the main goal of the following player is to maximize the profit and complete the tasks assigned to their virtual machines as quickly as possible. In such a structure, all requests are sent to the cloud computing department through the access point named as the task manager unit for task assignment.

Since the players' interests are in conflict with each other in many situations, Stackelberg's game theory has been used to reach the equilibrium point. Each applicant pays attention to his personal interests and tries to process the submitted application at the lowest possible cost. Therefore, in a non-cooperative game, it is necessary to find an equilibrium point according to which the pricing strategy of the applicants reaches the maximum value in total. Such a point is known as the Nash equilibrium point and the convergence process of reaching such a working point has been proven in several articles [124].

In the following problem, complete information about the players is not available, and therefore, in order to reach the optimal point, the reverse Stackelberg game was used in order to allocate resources appropriately in the absence of access to the full game conditions. In other words, it has been assumed in several articles that complete information is available from the participating players who want to use the resources located in the virtual machines in the fog space, which can accelerate the process of game theory implementation. However, in real conditions, full specifications of how players participate are not available, and to determine their presence strategy, it is necessary to use past information and how they interact in the implementation of policies designed in previous time steps. For this purpose, in this treatise, the inverse Stackelberg game theory is used, which is able to advance the game process and estimate the equilibrium point based on the designed strategies, despite the deficiency in the characteristics of the participating actors. On the other hand, one of the main points in the implementation of programs based on mobile mass monitoring is to know the level of participation of users in the implementation of specified requests. Since the presence of users and virtual machines is changing drastically and there is a strong dynamic in the essence of such issues, therefore, it is very challenging to accurately measure the amount of participation from moment to moment. For this purpose, fuzzy logic has been used to estimate the density level of users' participation in mobile mass monitoring by fuzzifying the geographical location of users' presence and estimating the time period of their presence and with the help of appropriate rules. In addition, determining the appropriate strategies for the participation of job applicants also plays

a decisive role in the implementation of game theory. For this purpose, the deep reinforcement learning algorithm is used to design appropriate strategies for the players' participation in the reverse Stackelberg game by having the history of the executed games, the amount of rewards and the type of reactions of the agents, so that both the convergence to the Nash equilibrium point is realized and the speed Resource allocation should be improved.

In order to prevent the divergence of the agent training process during the implementation of the deep reinforcement learning algorithm, the approximate policy optimizer (PPO) approach was used to estimate the values of the Q table and to determine the optimal policy, not only from the divergence of the agent in the execution of consecutive operations. prevent, but also improve the speed of convergence in identifying the desired policy. The steps of doing work in the proposed system are as follows (Table 4):

Table 4: Characteristics of participating units in the proposed structure

Specification unit type
Inputs Tasks Manager: Receive requests needed for processing
Get the specifications of virtual machines participating in collective monitoring, including the duration of participation, amount of memory and processing capacity
Outputs Sending approved requests to the request recognition unit
Final assignment of requests to virtual machines
Input application recognition unit: receiving approved applications from the job manager unit
Outputs: Determination This priority is the allowed time interval for processing requests, the allowed start and end time, the minimum required CPU capacity, the amount of capacity required for data storage and the number of cycles required to execute instructions.
Inverse Stackelberg Game Theory Players Player 1: Job Applicants (TI) (Leading Players) Player 2: Virtual Machines of Mobile Users (MU) (follower players)
The main goal: assigning requested functions to participating virtual machines
Desired characteristics: Minimization of allocation time Maximize coverage
Improving the level of service quality
Input: profile of two groups of players including TIs and MUs
Output: Allocation of available resources in virtual machines of mobile users to jobs received from mobile users
Input PPO: past values of Q table, history of implemented policies and rewards received
Output: determine the updated values of the Q table
Input DRL: history of played games, amount of reward and type of agent's reactions
Output: The right strategies for IT providers.
Fuzzy input: time interval of users to participate
The density of users in different geographical locations
Output: the density of user participation in collective monitoring

The information of the participating virtual machines is sent separately to the job manager unit on behalf of users or virtual machines for the implementation of mobile mass monitoring programs. In order to determine the density of user participation in monitoring programs, fuzzy algorithm was used to estimate the level of participation density by using the geographic location and availability time period of the applicant's virtual machines. At this stage, all the information of the players along with the estimated participation density level are entered into the game theory so that based on them, the allocation of the available resources for the submitted requests is implemented. In order to design suitable strategies for players, the combination of deep reinforcement learning algorithm and approximate policy is used to increase the speed of convergence towards the Nash equilibrium point and to avoid agent divergence in order to follow the designed policies. After the completion of allocation, characteristics such as coverage quality level, request processing time frame and service quality are examined on the obtained results in order to determine the efficiency level of the introduced approach better.

6.9 Game theory modeling

Game theory is a powerful framework that can model interactions between multiple players based on self-interest. This structure can be used to design a decentralized mechanism that prevents one-sided game deviation by some players. So far, game theory has been used in the design of incentive mechanisms in MCS, but in most references only task management for a task provider (TI) has been investigated and it has been assumed that complete information about the conditions of mobile users (MU) is available. But in practical environments, such assumptions have not been established and therefore it is necessary to conduct a comprehensive review of the designed structures.

In order to explain the structure of the used game theory, suppose that a mobile crowd sensing system consists of $M \triangleq \{m = 1, 2, ..., M\}$ number of TI and $N \triangleq \{n = 1, 2, ..., N\}$ number of MU be Each TI tries to use the capacities of a number of MUs for its desired functions with the lowest possible cost. In order to create the interaction of all

players and maximize the collective profit, it is necessary to consider the profit of all players. In this treatise, the players include the leading players and the following players, whose objective functions are defined in relations (6.1) and (6.2), respectively. In this case, the goal of each leading player is to complete the desired tasks with the lowest possible cost. The amount of profit earned by each following player and each leading player is calculated respectively through relations (6.3) to (6.4) [124]. It should be noted that the mentioned game is played between leading players on one side and between following players on the other side.

$$O.F(1) = \max \phi_m(t_m, p_m) \quad s.t. \sum_n p_m^n t_m^n \leq \delta_m \tag{6.1}$$

$$O.F(2) = \max \psi_n(t^n, p^n) \quad s.t. \sum_m t_m^n \leq k_n \tag{6.2}$$

$$\psi_n(t^n, p^n) = \sum_m p_m^n t_m^n - \sum_m C_m(t^n) \tag{6.3}$$

$$\phi_m(t_m, p_m) = \varphi_m(t_m) - \sum_n p_m^n t_m^n \tag{6.4}$$

Since all players seek to achieve their goals, it is necessary to determine a pricing strategy (p_m) for each TI and an activity time allocation strategy (t_n) for each MU in such a way that the profit of all players is guaranteed. Because the leading player takes action first, the action of the following players is also dependent on their performance ($U_F = f(U_L)$). In this situation, according to the prices proposed by the TIs, the MUs calculate the best conditions to achieve their maximum profit according to the equation (6.2).

The main challenges in this situation originate from the fact that, in the context of IoT, the type of activities provided by the leading players are both high in number and spread in a completely heterogeneous manner in the space under study, which causes disruption in reaching the point. The optimal equilibrium (Nash equilibrium point) becomes. For this purpose, in this treatise, the inverse Stackelberg game theory is used, which no longer requires full knowledge of the status of all players, and through its implementation, the equilibrium point can be achieved.

The main advantage of the reverse Stackelberg game compared to the classic Stackelberg game is the possibility of its implementation in situations where players have different reactions based on the opinions of the requestors. Compared to the basic Stackelberg game, in the reverse Stackelberg game, the leading player starts the game by sending the leading function to the follower. Since the task of the leader function is to map the follower's decision space to the leader's space ($\gamma_L : \Omega_F \to \Omega_L$), the leader's decision directly follows the follower's decision and therefore it is called the inverse method [46]. Of course, it should be noted that the inverse Stackelberg game is a special mode of the Stackelberg game. The main advantage of the inverse Stackelberg game approach compared to its original version is that it can be applied in situations where the follower has a non-unique response to the leader's decision request (U_F^d) that changes continuously. In equilibrium conditions, the global optimal point is obtained according to equation (6.5).

$$(U_L^d, U_F^d) \in argmin_{\top}((U_L, U_F) \in \Omega_L \times \Omega_F) f_0 \tau_L(U_L, U_F) \tag{6.5}$$

Without the leading player having a clear strategy, other leading players will not be able to convince the following players to follow their decisions. In this situation, several methods have been proposed to form the strategy of the leading players. One of the most practical approaches introduced is the use of virtual neural networks. In this situation, the following steps must be taken to approximate the relationship between the coefficients of the leading function and the value of the target function associated with it:

1. For each combination of coefficients that describe the strategy of the leading players $\gamma_L^c(U_F.)$, the optimal response of the follower related to it is calculated according to equation (6.6) and the target value of the leader is also calculated through equation (6.7) is evaluated.
2. The final set of sampling points (according to equation (6.8)), is divided into a training set and a test set and entered into the neural network algorithm [54]. In the following, the neural network is trained to approximate the relationship between coefficient values and forward objective function.
3. The neural network is trained using a solver for non-convex optimization problems, the result of which is the production of coefficients that maximize the reward value of the leading players' strategy.

$$U_F^c \in argmin_{\top}((U_L, U_F) \in \Omega_L \times \Omega_F) f_0 \tau_F(\gamma_L^c(U_F), U_F) \tag{6.6}$$

$$\tau_L(\gamma_L^c(U_F^c), U_F^c) \tag{6.7}$$

$$\{\vdash (C, Z_L(\gamma_L^c(U_F), U_F)) \dashv | \ c \in C_{NN}\} \tag{6.8}$$

6.10 Deep reinforcement learning

Since the optimization problem is presented in the form of a completely non-linear problem, and considering the profit from TIs, which have a large variety, makes it more complicated. On the other hand, MUs are reluctant to provide their personal information in different game situations. Therefore, in this situation, using the DRL approach to design optimal pricing strategies for TIs directly from the history of interactions can play an effective role. In the following, the process of implementing the DRL approach to determine the optimal strategy for TIs is presented. In this situation, each TI acts as an agent and the environment consists of N to MU. The DRL structure consists of the following components:

State space $S_m \triangleq s_m$

Action space $A_m \triangleq a_m$

State transition matrix $P_m \triangleq p_m$

Reward $R_m \triangleq r_m$

State space S_m: The pricing strategy of TI m in game k and its monitoring time vector belonging to all MUs is defined by the parameters $p_m(k)$ and $t_m(k)$. The state of the TI state is obtained through the equation (6.9). This relationship actually expresses the state of TI mam by considering the state of all MUs, which is known as the game history matrix.

$$S_m(k) = [P_m(k-L), t_m(k-L), ..., P_m(k-1), t_m(k-1)] \tag{6.9}$$

A_m action space: The mth TI action in the kth game is expressed as $a_m(k)$, which expresses the type of its pricing strategy. State transition probability function P_m: This function expresses the probability distribution of TI mth state transition. Its probability value is equal to $P((\vdash\ s\ \dashv|)\acute{s}, , p)$. The state-action space defined for the project has high dimensions. In other words, based on the state of the input states, the values of the Q table are estimated to determine the corresponding action. The details of the connection of different components of DQN are shown in Figure 7. The used neural network consists of a forward neural network with one input layer, two hidden layers and one output layer, all hidden layers are used by the modified linear unit for activation. In order to use the stored data from the state-action pair, a replay memory is used, which both prevents the deviation of the learning process and increases the speed of network training. The reward function R_m: The TI m reward function is obtained through the equation (6.10).

$$r_m(k) = \xi\varphi_m(t_m(k), , a_m(k)). \tag{6.10}$$

6.11 Scaling factor

In this regard, deep reinforcement learning (DQN) has been used to obtain the reward values of reinforcement learning (RL), which uses the features of the approximation function of the values of the Q table with the help of a neural network.

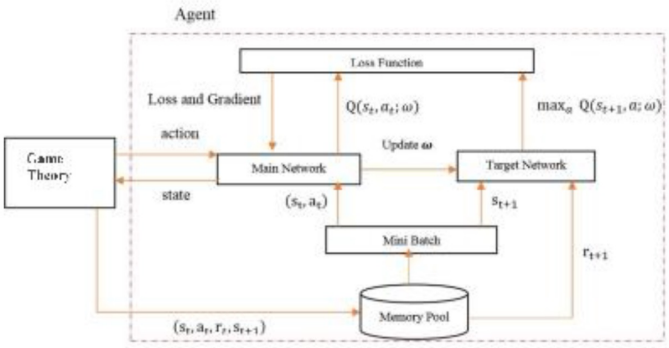

Figure 7: Learning agent design

In this structure, the agent uses the Deep Q network to determine actions in the network in order to achieve the desired goal. This network uses the input data to produce the determined output and to implement the desired neural network, it uses a fully connected neural network with two hidden layers, the performance of which is shown in Figure 7. The input layer uses the modified linear unit (ReLU) as the activation operator and the number of neurons in

the hidden layer is considered 255. It should be noted that all hidden layers also use the modified error function to activate the neurons designed for them.

6.12 Experimental replay memory

In the Deep Q network, a technique called experimental replay memory is used to prevent deviations in data distribution. The reason for this is because of learning based on small samples, because due to the correlation between the sequence of states, the learning process in the network faces serious challenges. In the Deep Q network, Q learning is not applied directly on state-action pairs during simulation, but instead the data extracted in the simulation process is stored in the desired buffer. In this situation, the agent starts with an empty buffer and then fills it during training and execution. In each time step, a quadruple tuple (s_t, a_t, r, s_{t+1}) containing the current state, current action, reward and next state is stored in the replay memory. If a state already exists in the replay memory, no more calculation is needed to obtain the reward and the next action - the target network.

Because in reinforcement learning, the data is not fixed, so a target network is used to calculate the correct value of Q for each state-action pair. In this regard, using a neural network to calculate target values and estimated values may cause divergence. For this purpose, initially, the parameters of the target network are selected to be equal to the parameters of the main network, but later on, the target network is not updated at each stage and is updated only by taking the parameters of the main network. This structure helps to break the correlation and prevents fluctuations.

6.13 Learning process

In the proposed approach, two neural networks have been used to implement the training process. In the first step, the weight of the main network (ω) is randomly assigned, and then the same weights (ω') as the main network are used in the target network, and an empty execution memory is defined to store samples. Using two networks simultaneously leads to stability in the learning process and helps to improve the effectiveness of the algorithm. During the execution of the learning process, the target network is used to retrieve the tested values, while in the original network, all updates are performed in the training phase. As the learning process progresses, the parameters of the neural network are coordinated with the parameters of the target network.

6.14 Q network training

Q learning algorithm is known as an efficient method in solving reinforcement learning problems. For the mathematical modeling of this algorithm, let π represent the policy. The function Q as a function acting on the state-action pair determines the cumulative reward of the agent through the equation (6.11).

$$Q^{\pi}(s_t, a) = (1 - \alpha)Q(s_t, a) + \alpha.(r(s_t, a) + \gamma \max f_0 Q(s_{t+1}, a; m\pi)) \tag{6.11}$$

$r(s_t, a)$: momentary reward of choosing action a in state s_t

γ: discount factor

α: Training rate

The complexity of implementing (6.11) grows exponentially with the increase in the size of the state space and operational space. In practical situations, due to the high number of action-state pairs, it is not possible to solve (6.11) directly. One of the solutions to overcome this challenge is to approximate Q values using a number of variables. In this structure, approximation plays an important role [58, 68, 122]. For this purpose, a special structure in the form of replay memory has been considered for estimation through training for DRL. The purpose of using open memory is to store a number of records, each record containing (s_t, a_t, r_t, s_{t+1}). In this case, a part of the memory is used as a predictor part. The mentioned system stores the record (s_t, a_t, r_t, s_{t+1}) obtained for each step. In the following, the amount of the reward is determined by considering the current states and the action taken. The rewarding function is defined according to the relation (6.12). The γ coefficient is present in the equations as a discount or reduction coefficient and helps to improve the convergence [122]. During network training, the same part of memory is extracted from the total available memory so that Q network learns from its past experiences. Due to the use of replay memory, enough training data is available to fit the Q value for the state-action pair, in this case, equation (6.13) is used. According to [68], parameter w is defined as a vector that is used in deep neural network for fitting.

$$R_t = r_{(t+1)} + \gamma r_{(t+2)} + \gamma^2 r_{(t+3)} + ... + \gamma^T r_{(t+T+1)} = \sum_{k=0}^{T} \gamma^k r_{(t+k+1)} \tag{6.12}$$

$$Q^{\pi}(s_t, a, w) \approx Q^{*}(s_t, a) \tag{6.13}$$

In this situation, the target network, whose output is updated by rewarding, is used as a criterion in evaluating the performance of the said network (Relation (6.14)). In the following, the gradient descent algorithm is used to minimize the difference between the output of the target network and the output of the estimation network (Relation (6.15)). In practice, the update of the w vector is used to minimize the relationship (6.15) to minimize the distance between the estimated values and the actual values.

$$Q^{lab}(s_t, a) = r(s_t, a) + \gamma . \max Q^{tar}(s_{(t+1)}, a', w') \tag{6.14}$$

$$Loss = (Q^{lab}(s_t, a) - Q^{pre}(s_t, a, w))^2 \tag{6.15}$$

6.15 Approximate policy

Real-world problems have huge dimensions that make the application of Q-learning algorithms to their realization face serious challenges. In this case, a phenomenon is created which is called the curse of dimensions (CoD). On the other hand, the DRL approach has 2 other important challenges as follows:

- Lack of stability in the learning process
- Complex design of reward system

In order to overcome the complexity of the system when the dimensions increase, neural networks can be used for approximation. In addition, the use of approximation functions creates stability in the learning process and ensures convergence to a suitable extent. In other words, using the approximate policy (PPO) in reinforcement learning structures reduces the change of parameters and increases the quality of convergence. In the mentioned structure, the values of the Q table are estimated by the PPO approach in order to design a suitable policy considering the current state of the system. In general, using a policy of approximation (PPO) has several advantages, some of which are stated below:

- High effectiveness in huge operating spaces
- More complete convergence features
- Ability to automatically learn stochastic policies

7 Evaluation of job heterogeneity

In the job assignment process, one of the main challenges is the quality quantification process of the data obtained from the monitoring of different requested jobs. In addition, calculating the amount of incentives necessary for allocating them on the available virtual machines also has complications that must be observed in the modeling.

To perform any requested monitoring task, it is necessary to have a number of virtual machines in the desired area and collect the received information in each time cycle. In fact, if a certain virtual machine in the designated area and in a certain period of time, performs the considered monitoring task, then the coverage has been achieved. In this situation, in order to monitor some functions, it is necessary to use a larger number of virtual machines to cover the area/time cycle in order to ensure the quality of the coverage. For example, if the number of virtual machines is equal to 2000, the number of designated areas is equal to 100, and the considered time cycles are equal to 24, and for the monitoring of each designated task, it is necessary to allocate 3 virtual machines for each time cycle area, therefore, assigning tasks Using greedy optimization methods requires more than 3 hours of time, and in this situation, it is essential to use effective methods [112].

In most of the studies, gradient-based methods have been introduced, in which it is assumed that the jobs are distributed homogeneously. This means that the data is taken from a single dataset or a combination of different domains that are composed of the same input feature space (for example, all inputs contain images that have the same dimensions). Considering such assumptions, it limits the generalization ability of the mentioned methods in implementing the distribution of complex and heterogeneous functions. These functions can be unimodal or multimodal, which is identified through their input data sources. These functions are sometimes in such a way that the same interpretations of a concept need to be expressed with the help of different aspects. For example, the image of an

animal and the title chosen for it can distinguish it from other animals. In this situation, the tasks of learning image classification and text classification share the same conceptual domain that can help in creating a level of knowledge on the set of animals in question. On the other hand, integrating multiple aspects of a subject or using auxiliary aspects can provide better efficiency than single-faceted functions. In such a situation, in many cases, the amount of input data for various functions is not enough or the received data is associated with many disturbances. In these cases, it is necessary to define different categories of multimodal functions in the system. An example of input spaces of heterogeneous functions is shown in figure 8.

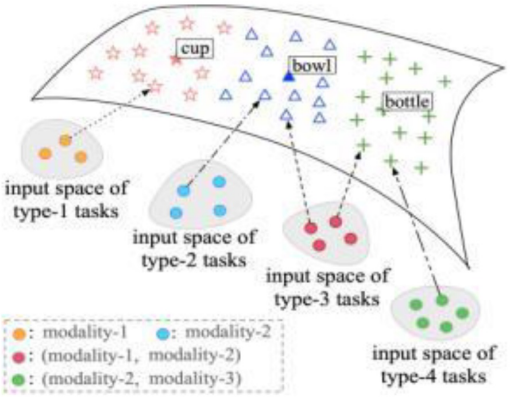

Figure 8: An example of multimodal heterogeneous functions presented in mobile mass monitoring [21]

Therefore, the main challenge when assigning heterogeneous tasks is how to divide a given task into tasks so that the specifications of all constituent tasks are fully determined or estimated to enable processing by appropriate virtual machines.

According to the definition considered in this treatise, heterogeneity is due to having multiple aspects in the requested jobs. Each of the aspects has a special status in the sequence forming the set of functions, which is analyzed and evaluated by the request analysis unit. In the request analysis unit, a grid function g_φ is used to define a task placement vector (τ) for each unique task, so that based on it, the necessary knowledge to identify the sent tasks can be extracted (Relation (7.1)). The τ vector reflects the characteristics of each task and differentiates the tasks from each other. In this situation, τ is used as a practical tool in identifying parameters in the feature aggregation network. Finally, the information of the features of functions is extracted by the relation (7.2) [5].

$$\tau = g_\varphi(c_\tau; \varphi) \in R^{F_2} \tag{7.1}$$

$$h^* = \sum_{m=1}^{M} A_m h^{(m)} \in R^{F_2} \tag{7.2}$$

$$A_m = \frac{exp(v^T . \tanh(\omega_h[h^m \bigoplus \tau]))}{\sum_l exp(v^T . \tanh(\omega_h[h^l \bigoplus \tau]))} \tag{7.3}$$

F_2: The dimensions of the embedded functions

c_τ: vector of features of functions

A_m: Probability of coefficients determining the role of each aspect

v, ω_h: learner parameters

φ: embedded task

7.1 The objective function

In this paper, after determining the pricing strategy for job applicants and determining the specifications of virtual machines participating in mobile mass monitoring, optimal allocation is done based on considering the priority of the tasks presented, the amount of processing capacity required, the required memory and the allowed time frame. However, the efficiency of the computational approach is disturbed by increasing the dimensions. To solve this challenge, the proposed solution is to use the approximate policy optimization (PPO) algorithm that was introduced

in 2017 [11]. In this situation, without the need to discretize the operational space, the outputs are considered as a probabilistic distribution, and by sampling the distribution function, the search space is tried to be improved. In other words, the PPO method is considered among the algorithms that can properly analyze the constrained policy optimization problem.

The modification process is done in such a way that a search is performed in a set of possible policies, which is called the policy space. For this purpose, it is first necessary to calculate the performance identifier of the relative policy as the difference between the value of the objective function in the current step and its value in the next step according to equation (7.4). In this situation, the step size control is updated with the help of relation (7.5) and using the learning rate. Of course, it should be noted that choosing the optimal step size (α) is not an easy task. In other words, if (α) is meant to be small, the number of iterations increases and the training becomes time-consuming and expensive, and the policy may get trapped in the local optimal point. If (α) is assumed to be large, the corresponding step in the policy space passes through the neighbourhood of suitable policies and may reduce the efficiency, and if this trend continues, the possibility of recovering the operating line becomes very weak. For this purpose, it is first necessary to calculate the performance identifier of the relative policy as the difference between the value of the objective function in the current step and its value in the next step according to equation (7.4). In this regard, the value of the adaptive penalty is calculated, and then the objective function is obtained according to equation (7.5) [35].

$$TG_{ti} = \left\{cy_{ti}^{1}, cy_{ti}^{2}, ..., cy_{ti}^{q}\right\} \tag{7.4}$$

$$SG_{ti} = \left\{reg_{ti}^{1}, reg_{ti}^{2}, ..., reg_{ti}^{p}\right\} \tag{7.5}$$

TG : Monitoring cycles

SG : Segmented areas

References

[1] A. Al Buhussain, E. Robson, and A. Boukerche, *Performance analysis of bio-inspired scheduling algorithms for cloud environments*, IEEE Int. Parallel Distrib. Process. Symp. Workshops, IEEE, 2016, pp. 776–785.

[2] T. Ali, U. Draz, S. Yasin, J. Noureen, A. Shaf, and M. Ali, *An efficient participant's selection algorithm for crowdsensing*, Int. J. Adv. Comput. Sci. Appl. **9** (2018), 399–404.

[3] C.M. Angelopoulos, O. Evangelatos, S. Nikoletseas, T.P. Raptis, J. DP Rolim, and K. Veroutis, *A user-enabled testbed architecture with mobile crowdsensing support for smart, green buildings*, IEEE Int. Conf. Commun., IEEE, 2015, pp. 573–578.

[4] K. Arulkumaran, M.P. Deisenroth, M. Brundage, and A.A. Bharath, *Deep reinforcement learning: A brief survey*, IEEE Signal Process. Mag. **34** (2017), no. 6, 26–38.

[5] D. Bahdanau, *Neural machine translation by jointly learning to align and translate*, arXiv preprint arXiv:1409.0473, (2014).

[6] F. Bao, R. Chen, and J. Guo, *Scalable, adaptive and survivable trust management for community of interest based internet of things systems*, IEEE 11th Int. Symp. Autonom. Decent. Syst., IEEE, 2013, pp. 1–7.

[7] E. Barrett, E. Howley, and J. Duggan, *Applying reinforcement learning towards automating resource allocation and application scalability in the cloud*, Concurr. Comput.: Practice Exper. **25** (2013), no. 12, 1656–1674.

[8] T. Basar, *On the relative leadership property of Stackelberg strategies*, J. Optim. Theory Appl. **11** (1973), no. 6, 655–661.

[9] T. Başar and G.J. Olsder, *Dynamic Noncooperative Game Theory*, Society for Industrial and Applied Mathematics, 1998.

[10] S. Basu, M. Karuppiah, K. Selvakumar, K.-C. Li, S. Hafizul Islam, M. Mehedi Hassan, and M.Z.A. Bhuiyan, *An intelligent/cognitive model of task scheduling for IoT applications in cloud computing environment*, Future Gen. Comput. Syst. **88** (2018), 254–261.

[11] E. Bøhn, E.M. Coates, S. Moe, and T.A. Johansen, *Deep reinforcement learning attitude control of fixed-wing UAVs using proximal policy optimization*, Int. Conf. Unmanned Aircraft Systems, IEEE, 2019, pp. 523–533.

[12] S. Bradai, S. Khemakhem, and M. Jmaiel, *Real-time and energy aware opportunistic mobile crowdsensing framework based on people's connectivity habits*, Comput. Networks **142** (2018), 179–193.

[13] S. Bresciani, S. Ur Rehman, G.M. Alam, K. Ashfaq, and M. Usman, *Environmental MCS package, perceived environmental uncertainty and green performance: In green dynamic capabilities and investment in environmental management perspectives*, Rev. Int. Bus. Strat. **33** (2023), no. 1, 105–126.

[14] H. Cai, Y. Zhu, Z. Feng, H. Zhu, J. Yu, and J. Cao, *Truthful incentive mechanisms for mobile crowd sensing with dynamic smartphones*, Comput. Networks **141** (2018), 1–16.

[15] D.H. Cansever and T. Başar, *On stochastic incentive control problems with partial dynamic information*, Syst. Control Lett. **6** (1985), no. 1, 69–75.

[16] G.F.H. Cánepa, *A context-aware application offloading scheme for a mobile peer-to-peer environment*, Ph.D. Thesis, Korea Advanced Institute of Science and Technology (KAIST), 2013.

[17] A. Capponi, C. Fiandrino, B. Kantarci, L. Foschini, D. Kliazovich, and P. Bouvry, *A survey on mobile crowdsensing systems: Challenges, solutions, and opportunities*, IEEE Commun. Surv. Tutor. **21** (2019), no. 3, 2419–2465.

[18] Y. Chen, B. Li, and Q. Zhang, *Incentivizing crowdsourcing systems with network effects*, IEEE INFOCOM 2016-The 35th Annual IEEE Int. Conf. Comput. Commun., IEEE **35** (2016), 1–9.

[19] J. Chen, H. Ma, D. Zhao, and D.S. Wei, *Participant density-independent location privacy protection for data aggregation in mobile crowd-sensing*, Wireless Person. Commun. **98** (2018), no. 1, 699–723.

[20] M. Chen, T. Wang, K. Ota, M. Dong, M. Zhao, and A. Liu, *Intelligent resource allocation management for vehicles network: An A3C learning approach*, Comput. Commun. **151** (2020), 485–494.

[21] J. Chen and A. Zhang, *Hetmaml: Task-heterogeneous model-agnostic meta-learning for few-shot learning across modalities*, Proc. 30th ACM Int. Conf. Inf. Knowledge Manag. **30** (2021), 191–200.

[22] M.H. Cheung, F. Hou, and J. Huang, *Delay-sensitive mobile crowdsensing: Algorithm design and economics*, IEEE Trans. Mobile Comput. **17** (2018), no. 12, 2761–2774.

[23] J. Cruz, *Leader-follower strategies for multilevel systems*, IEEE Trans. Autom. Control **23** (2003), no. 2, 244–255.

[24] W. Dai, H. Lu, J. Xiao, and Z. Zheng, *Task allocation without communication based on incomplete information game theory for multi-robot systems*, J. Intell. Robotic Syst. **94** (2019), no. 3, 841–856.

[25] T. Das, P. Mohan, V.N. Padmanabhan, R. Ramjee, and A. Sharma, *PRISM: platform for remote sensing using smartphones*, Proc. 8th Int. Conf. Mobile Syst. Appl. Serv. **8** (2010), 63–76.

[26] R. Deng, R. Lu, C. Lai, T.H. Luan and H. Liang, *Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption*, IEEE Internet Things J. **3** (2016), no. 6, 1171–1181.

[27] X. Ding, R. Lv, X. Pang, J. Hu, Z. Wang, X. Yang, and X. Li, *Privacy-preserving task allocation for edge computing-based mobile crowdsensing*, Comput. Electr. Eng. **97** (2022), 107528.

[28] J.J. Durillo, H. Mohammadi Fard, and R. Prodan, *MOHEFT: A multi-objective list-based method for workflow scheduling*, IEEE Int. Conf. Cloud Comput. Technol. Sci. Proc.eedings, IEEE, 2012, pp.185–192.

[29] H. Ehtamo and R.P. Hämäläinen, *Incentive strategies and equilibria for dynamic games with delayed information*, J. Optim. Theory Appl. **63** (1989), no. 3, 355–369.

[30] J. Espadas, A. Molina, G. Jiménez, M. Molina, R. Ramírez, and D. Concha, *A tenant-based resource allocation model for scaling Software-as-a-Service applications over cloud computing infrastructures*, Future Gen. Comput. Syst. **29** (2013), no. 1, 273–286.

[31] C. Fang, H. Xu, Y. Bai, T. Zhang, Y. Yang, and Z. Hu, *Deep reinforcement learning-based joint task offloading in cloud-edge-end cooperation environments*, 2nd Int. Conf. Front. Electron. Inf. Comput. Technol., IEEE, **2** (2022), 524–530.

[32] J. Feng, T. Li, Y. Zhai, S. Lv, and F. Zhao, *Ensuring honest data collection against collusive CSDF attack with binary-minmaxs clustering analysis in mobile crowd sensing*, IEEE Access **7** (2019), 124491–124501.

[33] S. Ghasemi Falavarjani, M.A. Nematbakhsh, and B. Shahgholi Ghahfarokhi, *A multi-criteria resource alloca-*

tion mechanism for mobile clouds, Int. Symp. Comput. Networks Distrib. Syst., Cham: Springer International Publishing, 2013, pp.145–154.

[34] B. Gomathi, K. Krishnasamy, and B.S. Balaji, *Epsilon-fuzzy dominance sort-based composite discrete artificial bee colony optimisation for multi-objective cloud task scheduling problem*, Int. J. Bus. Intell. Data Min. **13** (2018), no. 1–3, 247–266.

[35] L. Graesser and W.L. Keng, *Foundations of dDeep Reinforcement Learning: Theory and Practice in Python*, Addison-Wesley Professional, 2019.

[36] N. Groot, G. Zaccour, and B. De Schutter, *Hierarchical game theory for system-optimal control: Applications of reverse Stackelberg games in regulating marketing channels and traffic routing*, IEEE Control Syst Mag. **37** (2017), no. 2, 129–152.

[37] D. Grzonka, A. Jakóbik, J. Kołodziej, and S. Pllana, *Using a multi-agent system and artificial intelligence for monitoring and improving the cloud performance and security*, Future Gen. Comput. Syst. **86** (2018), 1106–1117.

[38] B. Guo, H. Chen, Z. Yu, X. Xie, S. Huangfu, and D. Zhang, *FlierMeet: a mobile crowdsensing system for cross-space public information reposting, tagging, and sharing*, IEEE Trans. Mobile Comput. **14** (2014), no. 10, 2020–2033.

[39] B. Guo, Y. Liu, W. Wu, Z. Yu, and Q. Han, *ActiveCrowd: A framework for optimized multitask allocation in mobile crowdsensing systems*, IEEE Trans. Human-Machine Syst. **47** (2016), no. 3, 392–403.

[40] A. Hammoud, A. Mourad, H. Otrok, O. Abdel Wahab, and H. Harmanani, *Cloud federation formation using genetic and evolutionary game theoretical models*, Future Gen. Comput. Syst. **104** (2020), 92–104.

[41] Y. Han, Y. Zhu, and J. Yu, *A distributed utility-maximizing algorithm for data collection in mobile crowd sensing*, IEEE Glob. Commun. Conf., 2014, pp. 277–282.

[42] S. He, D.-H. Shin, J. Zhang, and J. Chen, *Toward optimal allocation of location dependent tasks in crowdsensing*, IEEE INFOCOM Conf. Comput. Commun., IEEE, 2014, pp. 745–753.

[43] Y.C. Ho, *On incentive problems*, Syst. Control Lett. **3** (1983), no. 2, 63–68.

[44] Y.-C. Ho, P. Luh, and R. Muralidharan, *Information structure, Stackelberg games, and incentive controllability*, IEEE Trans. Autom. Control **26** (1981), no. 2, 454–460.

[45] Y.-C. Ho, P.B. Luh, and G.J. Olsder, *A control-theoretic view on incentives*, Automatica **18** (1982), no. 2, 167–179.

[46] S. Hu and G. Li, *Dynamic request scheduling optimization in mobile edge computing for IoT applications*, IEEE Internet Things J. **7** (2019), no. 2, 1426–1437.

[47] L. Huang, X. Feng, C. Zhang, L. Qian, and Y. Wu, *Deep reinforcement learning-based joint task offloading and bandwidth allocation for multi-user mobile edge computing*, Digital Commun. Networks **5** (2019), no. 1, 10–17.

[48] M. Hussin, N.A.W. Abdul Hamid, and K.A. Kasmiran, *Improving reliability in resource management through adaptive reinforcement learning for distributed systems*, J. Parallel Distrib. Comput. **75** (2015), 93–100.

[49] M. Hussin, Y.C. Lee, and A.Y. Zomaya, *Efficient energy management using adaptive reinforcement learning-based scheduling in large-scale distributed systems*, Int. Conf. Parallel Process., IEEE, 2011, pp. 385–393.

[50] B. Jang, M. Kim, G. Harerimana, and J.W. Kim, *Q-learning algorithms: A comprehensive classification and applications*, IEEE Access **7** (2019), 133653–133667.

[51] G. Javadzadeh and A.M. Rahmani, *Fog computing applications in smart cities: A systematic survey*, Wireless Networks **26** (2020), no. 2, 1433–1457.

[52] S.-K. Kim and C.K. Ahn, *Auto-tuner-based controller for quadcopter attitude tracking applications*, IEEE Trans. Circ. Syst. II: Express Briefs **66** (2019), no. 12, 2012–2016.

[53] M.D. Kristensen, *Scavenger: Transparent development of efficient cyber foraging applications*, IEEE Int. Conf. Pervasive Comput. Commun., IEEE, 2010, pp. 217–226.

[54] C. Lau, *Neural Networks: Theoretical Foundations and Analysis*, IEEE Press, 1991.

[55] J.Z. Leibo, V. Zambaldi, M. Lanctot, J. Marecki and T. Graepel, *Multi-agent reinforcement learning in sequential*

social dilemmas, arXiv preprint arXiv:1702.03037, (2017).

[56] P. Li and Y. Wang, *An active learning reliability analysis method using adaptive Bayesian compressive sensing and Monte Carlo simulation (ABCS-MCS)*, Reliab. Engin. Syst. Safety **221** (2022), 108377.

[57] Y. Liu, B. Guo, Y. Wang, W. Wu, Z. Yu, and D. Zhang, *TaskMe: Multi-task allocation in mobile crowd sensing*, Proc. ACM Int. Joint Conf. Pervasive Ubiq. Compu., 2016, pp. 403–414.

[58] R. Liu and J. Zou, *The effects of memory replay in reinforcement learning*, 56th Ann. Allerton Conf. Commun. Control Comput. (Allerton), IEEE, **56** (2018), 478–485.

[59] A. Lu and J.-H. Zhu, *Worker recruitment with cost and time constraints in mobile crowd sensing*, Future Gen. Comput. Syst. **112** (2020), 819–831.

[60] P. Luh, Y. Ho, and R. Muralidharan, *Load adaptive pricing: An emerging tool for electric utilities*, IEEE Trans. Autom. Control **27** (2003), no. 2, 320–329.

[61] F. Luo, Y. Yuan, W. Ding, and H. Lu, *An improved particle swarm optimization algorithm based on adaptive weight for task scheduling in cloud computing*, Proc. 2nd Int. Conf. Comput. Sci. Appl. Engine. **2** (2018), 1–5.

[62] S. Maharjan, Y. Zhang, and S. Gjessing, *Optimal incentive design for cloud-enabled multimedia crowdsourcing*, IEEE Trans. Multimedia **18** (2016), no. 12, 2470–2481.

[63] M. Marjanović, A. Antonić, and I.P. Žarko, *Edge computing architecture for mobile crowdsensing*, IEEE Access **6** (2018), 10662–10674.

[64] G. Martín-Herrán and G. Zaccour, *Credible linear-incentive equilibrium strategies in linear-quadratic differential games*, Advances in dynamic games and their applications: analytical and numerical developments, Boston: Birkhäuser Boston, 2009, pp. 1–31.

[65] M. Mehdi, G. Mühlmeier, K. Agrawal, R. Pryss, M. Reichert, and F.J. Hauck, *Referenceable mobile crowdsensing architecture: A healthcare use case*, Proc. Comput. Sci. **134** (2018), 445–451.

[66] MILLIONAGENTS, *Expert in data collection and processing since 2012*, http://www.millionagents.com/. Accessed 10 Nov 2019.

[67] A. Mtibaa,A. Fahim, K.A. Harras, and M.H. Ammar, *Towards resource sharing in mobile device clouds: Power balancing across mobile devices*, ACM SIGCOMM Comput. Commun. Rev. **43** (2013), no. 4, 51–56.

[68] A. Nair, P. Srinivasan, S. Blackwell, C. Alcicek, R. Fearon, A. De Maria, V. Panneershelvam, M. Suleyman, C. Beattie, S. Petersen, S. Legg, V. Mnih, K. Kavukcuoglu, and D. Silver, *Massively parallel methods for deep reinforcement learning*, arXiv preprint arXiv:1507.04296, (2015).

[69] J.F. Nash Jr, *Equilibrium points in n-person games*, Proc. Nat. Acad. Sci. **36** (1950), no. 1, 48–49.

[70] L. Ni, J. Zhang, C. Jiang, C. Yan, and K. Yu, *Resource allocation strategy in fog computing based on priced timed petri nets*, IEEE Internet Things J. **4** (2017), no. 5, 1216–1228.

[71] N. Nisan, T. Roughgarden, E. Tardos, and V.V. Vazirani, *Algorithmic Game Theory*, Cambridge University Press, 2007.

[72] M. Pachter, *Linear-quadratic reversed Stackelberg differential games with incentives*, IEEE Trans. Autom. Control **29** (2003), no. 7, 644–647.

[73] Z. Peng, X. Gui, J. An, T. Wu, and R. Gui, *Multi-task oriented data diffusion and transmission paradigm in crowdsensing based on city public traffic*, Comput. Networks **156** (2019), 41–51.

[74] S. Phoha, N. Jacobson, D. Friedlander, and R. Brooks, *Sensor network based localization and target tracking through hybridization in the operational domains of beamforming and dynamic space-time clustering*, In GLOBECOM'03. IEEE Glob. Telecommun. Conf., (IEEE Cat.), **5** (2003), no. 03CH37489, 2952–2956.

[75] PiedPiper, *Live crypto prices and market cap charts*, https://slideplayer.com/slide/13011299/, 2025.

[76] S. Pitis, *Rethinking the discount factor in reinforcement learning: A decision theoretic approach*, Proc. AAAI Conf. Artific. Intell. **33** (2019), no. 1, 7949–7956.

[77] M. Pouryazdan, C. Fiandrino, B. Kantarci, T. Soyata, D. Kliazovich, and P. Bouvry, *Intelligent gaming for*

mobile crowd-sensing participants to acquire trustworthy big data in the internet of things, IEEE Access **5** (2017), 22209–22223.

[78] M.-R. Ra, B. Liu, T.F. La Porta, and R. Govindan, *Medusa: A programming framework for crowd-sensing applications*, Proc. 10th Int. Conf. Mobile Syst. Appl. Serv. **10** (2012), 337–350.

[79] G. Rjoub, J. Bentahar, O. Abdel Wahab, and A.S. Bataineh, *Deep and reinforcement learning for automated task scheduling in large-scale cloud computing systems*, Concurr. Comput.: Practice Exper. **33** (2021), no. 23, p. e5919.

[80] T. Roughgarden, *Stackelberg scheduling strategies*, Proc. 33th Ann. ACM Symp. Theory Comput., 2001, pp. 104–113.

[81] A. Schrijver, *Theory of Linear and Integer Programming*, John Wiley & Sons, 1998.

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

[83] E.M. Shakshuki, N. Kang, and T.R. Sheltami, *EAACK—a secure intrusion-detection system for MANETs*, IEEE Trans. Ind. Electron. **60** (2012), no. 3, 1089–1098.

[84] N. Shan, X. Cui, and Z. Gao, *"DRL+ FL": An intelligent resource allocation model based on deep reinforcement learning for mobile edge computing*, Comput. Commun. **160** (2020), 14–24.

[85] O. Shehory and S. Kraus, *Methods for task allocation via agent coalition formation*, Artific. Intell. **101** (1998), no. 1–2, 165–200.

[86] H. Shen and T. Başar, *Incentive-based pricing for network games with complete and incomplete information*, Advances in dynamic game theory: numerical methods, algorithms, and applications to ecology and economics, Boston, MA: Birkhäuser Boston, 2007, pp. 431–458.

[87] Z. Shi, H. Huang, Y.-E. Sun, X. Wu, F. Li, and M. Tian, *An efficient task assignment mechanism for crowdsensing systems*, Int. Conf. Cloud Comput. Secur., Cham: Springer International Publishing, 2016, pp. 14–24.

[88] Y. Shoham and K. Leyton-Brown, *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*, Cambridge University Press, 2008.

[89] M. Simaan and J.B. Cruz Jr, *On the Stackelberg strategy in nonzero-sum games*, J. Optim. Theory Appl. **11** (1973), no. 5, 533–555.

[90] M. Simaan and J.B. Cruz Jr, *A Stackelberg solution for games with many players*, IEEE Trans. Autom. Control 18 (2003), no. 3, 322–324.

[91] P. Singh and N.K. Walia, *A review: Cloud computing using various task scheduling algorithms*, Int. J. Comput. Appl. **142** (2016), no. 7, 30–32.

[92] C.C. Sobin, V. Raychoudhury, and S. Saha, *An energy-efficient and buffer-aware routing protocol for opportunistic smart traffic management*, Proc. 18th Int. Conf. Distrib. Comput. Network, 2017, pp. 1–8.

[93] H.V. Stackelberg, *Marktform und gleichgewicht*, J. Springer, 1934.

[94] H.V. Stackelberg, *The Theory of Market Economy*, Translated by: A.T. Peacock, William Hodge, London, 1952.

[95] S. Sundar and B. Liang, *Offloading dependent tasks with communication delay and deadline constraint*, IEEE INFOCOM Conf. Comput. Commun., IEEE, 2018, pp. 37–45.

[96] R.S. Sutton and A.G. Barto, *Reinforcement learning: An introduction*, Cambridge: MIT Press, 1998.

[97] M. Taghavi, J. Bentahar, and H. Otrok, *Two-stage game theoretical framework for IaaS market share dynamics*, Future Gen. Comput. Syst. **102** (2020), 173–189.

[98] S. Tayeb, S. Latifi, and Y. Kim, *A survey on IoT communication and computation frameworks: An industrial perspective*, IEEE 7th Ann. Comput. Commun. Workshop Conf., IEEE, 2017, pp. 1–6.

[99] M. Tirmazi, A. Barker, N. Deng, M.E. Haque, Z.G. Qin, S. Hand, M. Harchol-Balter, and J. Wilkes, *Borg: the next generation*, Proc. Fifteenth Eur. Conf. Comput. Syst., 2020, pp. 1–14.

[100] M. Tomasoni, A. Capponi, C. Fiandrino, D. Kliazovich, F. Granelli, and P. Bouvry, *Profiling energy efficiency of mobile crowdsensing data collection frameworks for smart city applications*, 6th IEEE Int. Conf. Mobile Cloud Comput. Serv. Engin. (MobileCloud), IEEE, **6** (2018), 1–8.

[101] H. Topcuoglu, S. Hariri, and M.-Y. Wu, *Performance-effective and low-complexity task scheduling for heterogeneous computing*, IEEE Trans. Parallel Distrib. Syst. **13** (2002), no. 3, 260–274.

[102] T. Vallée, T.C. Deissenberg, and Q. Basar, *Optimal open loop cheating in dynamic reversed LQ Stackelberg games*, Ann. Oper. Res. **8** (1999), no. 0, 217–232.

[103] H. Van Seijen, M. Fatemi, and A. Tavakoli, *Using a logarithmic mapping to enable lower discount factors in reinforcement learning*, Adv. Neural Inf. Process. Syst. **32** (2019).

[104] L. Vig and J.A. Adams, *Multi-robot coalition formation*, IEEE Trans. Robotics **22** (2006), no. 4, 637–649.

[105] W.Y.C. Wang, A. Rashid, and H.-M. Chuang, *Toward the trend of cloud computing*, J. Electr. Commerce Res. **12** (2011), no. 4, 238.

[106] T. Wang, K. Ismail, and K.S. Abas Azmi, *The rise of MCS and EMA in the sustainable field: a systematic literature analysis*, Sustainability **14** (2022), no. 24, 16532.

[107] J. Wang, J. Tang, G. Xue, and D. Yang, *Towards energy-efficient task scheduling on smartphones in mobile crowd sensing systems*, Comput. Networks **115** (2017), 100–109.

[108] J. Wang, Y. Wang, D. Zhang, F. Wang, Y. He, and L. Ma, *PSAllocator: Multi-task allocation for participatory sensing with sensing capability constraints*, Proc. ACM Conf. Comput. Supported Coop. Work Soc. Comput., 2017, pp. 1139–1151.

[109] J. Wang, Y. Wang, D. Zhang, F. Wang, H. Xiong, C. Chen, Q. Lv, and Z. Qiu, *Multi-task allocation in mobile crowd sensing with individual task quality assurance*, IEEE Trans. Mobile Comput. **17** (2018), no. 9, 2101–2113.

[110] J. Wang, Y. Wang, D. Zhang, L. Wang, H. Xiong, A. Helal, Y. He, and F. Wang, *Fine-grained multitask allocation for participatory sensing with a shared budget*, IEEE Internet Things J. **3** (2016), no. 6, 1395–1405.

[111] L. Wang, Z. Yu, B. Guo, F. Yi, and F. Xiong, *Mobile crowd sensing task optimal allocation: A mobility pattern matching perspective*, Front. Comput. Sci. **12** (2018), no. 2, 231–244.

[112] L. Wang, Z. Yu, D. Zhang, B. Guo, and C.H. Liu, *Heterogeneous multi-task assignment in mobile crowdsensing using spatiotemporal correlation*, IEEE Trans. Mobile Comput. **18** (2018), no. 1, 84–97.

[113] W. Wang, G. Zeng, D. Tang, and J. Yao, *Cloud-DLS: Dynamic trusted scheduling for Cloud computing*, Expert Syst. Appl. **39** (2012), no. 3, 2321–2329.

[114] Waze, *Driving directions and traffic reports by Waze*, https://www.waze.com/, Accessed 8 Nov 2019.

[115] C.-Y. Wei, Y.-T. Hong and C.-J. Lu, *Online reinforcement learning in stochastic games*, Adv. Neural Inf. Process. Syst. **30** (2017).

[116] C.G. Wu, W. Li, L. Wang, and A.Y. Zomaya, *Hybrid evolutionary scheduling for energy-efficient fog-enhanced internet of things*, IEEE Trans. Cloud Comput. **9** (2018), no. 2, 641–653.

[117] S. Yang, J. Bian, L. Wang, H. Zhu, Y. Fu, and H. Xiong, *EdgeSense: Edge-mediated spatial-temporal crowdsensing*, IEEE Access **7** (2018), 95122–95131.

[118] J. Yang, B. Jiang, Z. Lv, and K.-K. Raymond Choo, *A task scheduling algorithm considering game theory designed for energy management in cloud computing*, Future Gen. Comput. Syst. **105** (2020), 985–992.

[119] B. Yang, X. Xu, F. Tan, and D.H. Park, *An utility-based job scheduling algorithm for cloud computing considering reliability factor*, Int. Conf. Cloud Serv. Comput., IEEE, 2011, pp. 95–102.

[120] D. Yang, G. Xue, X. Fang, and J. Tang, *Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing*, Proc. 18th Ann. Int. Conf. Mobile Comput. Network. **18** (2012), 173–184.

[121] D. Yang, G. Xue, X. Fang, and J. Tang, *Incentive mechanisms for crowdsensing: Crowdsourcing with smartphones*, IEEE/ACM Trans. Network. **24** (2015), no. 3, 1732–1744.

[122] Y. Yu, Q. Shi and H.-K. Lam, *Fuzzy sliding mode control of a continuum manipulator*, IEEE Int. Conf. Robot.

Biomim., IEEE, 2018, pp. 2057–2062.

[123] M. Zappatore, A. Longo, and M.A. Bochicchio, *Using mobile crowd sensing for noise monitoring in smart cities*, Int. Multidiscip. Conf. Comput. Energy Sci. (Splitech), IEEE, 2016, pp. 1–6.

[124] Y. Zhan, C.H. Liu, Y. Zhao, J. Zhang, and J. Tang, *Free market of multi-leader multi-follower mobile crowd-sensing: An incentive mechanism design by deep reinforcement learning*, IEEE Trans. Mobile Comput. **19** (2019), no. 10, 2316–2329.

[125] F. Zhang, J. Leitner, M. Milford, B. Upcroft, and P. Corke, T*owards vision-based deep reinforcement learning for robotic motion control*, arXiv preprint arXiv:1511.03791, **2015** (2015).

[126] J. Zhang, X. Li, Z. Shi, and C. Zhu, *A reputation-based and privacy-preserving incentive scheme for mobile crowd sensing: A deep reinforcement learning approach*, Wireless Networks **30** (2024), no. 6, 4685–4698.

[127] H. Zhang, Y. Xiao, S. Bu, D. Niyato, F.R. Yu, and Z. Han, *Computing resource allocation in three-tier IoT fog networks: A joint optimization approach combining Stackelberg game and matching*, IEEE Internet Things J. **4** (2017), no. 5, 1204–1215.

[128] D. Zhang, H. Xiong, L. Wang, and G. Chen, *CrowdRecruiter: Selecting participants for piggyback crowdsensing under probabilistic coverage constraint*, Proc. ACM Int. Joint Conf. Perv. Ubiq. Comput., 2014, pp. 703–714.

[129] P. Zhang and M. Zhou, *Dynamic cloud task scheduling based on a two-stage strategy*, IEEE Trans. Aut. Sci. Engin. **15** (2017), no. 2, 772–783.

[130] X. Zhou, G. Zhang, J. Sun, J. Zhou, T. Wei, and S. Hu, *Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based HEFT*, Future Gen. Comput. Syst. **93** (2019), 278–289.

[131] P. Zhou, Y. Zheng, and M. Li, *How long to wait? Predicting bus arrival time with mobile phone based participatory sensing*, Proc. 10th Int. Conf. Mobile Syst. Appl. Serv., **10** (2012), 379–392.

[132] W. Zhu, W. Guo, Z. Yu, and H. Xiong, *Multitask allocation to heterogeneous participants in mobile crowd sensing*, Wireless Commun. Mobile Comput. **2018** (2018), no. 1, 7218061.

[133] X. Zhu, Y. Luo, A. Liu, W. Tang, and M.Z.A. Bhuiyan, *A deep learning-based mobile crowdsensing scheme by predicting vehicle mobility*, IEEE Trans. Intell. Transport. Syst. **22** (2020), no. 7, 4648–4659.

[134] X. Zuo, G. Zhang, and W. Tan, *Self-adaptive learning PSO-based deadline constrained task scheduling for hybrid IaaS cloud*, IEEE Trans. Autom. Sci. Engin. **11** (2013), no. 2, 564–573.