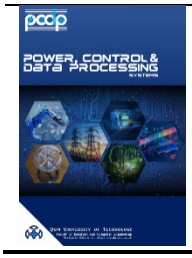
DOI: 10.30511/pcdp.2026.2088015.1070 - ISSN: 3060-8376 - Vol 3(4), 2026



POWER, CONTROL AND DATA PROCESSING SYSTEMS

Available Online at: https://pcdp.qut.ac.ir/

Intelligent and Self-Organized Resource Allocation in Sensor-Cloud Infrastructure Using Game Theory

ARTICLE INFO

Article Type
Original Research

Authors
Moslem Mohammadi Kharaji
Marzieh Varposhti*
Abbas Horri

Department of computer engineering, Shahrekord University, Shahrekord, Iran

*** Correspondence***
Address: Department of computer engineering, Shahrekord University, Rahbar Boulevard, Shahrekord, Iran. Postal Code: 8818634141.

mvarposhti@sku.ac.ir

Article History
Received: May 4, 2026
Accepted: July 22, 2026
ePublished: December 01, 2026

ABSTRACT

With the rapid advancement of sensing technologies, cloud computing, and communication systems, interaction between physical sensors and cloud-based applications has become more seamless and intelligent. This technological convergence has given rise to the concept of the Sensor-Cloud, a hybrid architecture that tightly integrates wireless sensor networks with scalable cloud infrastructures. One of the core challenges in Sensor-Cloud systems lies in efficiently managing resource allocation and task scheduling between sensors and cloud-hosted applications. Ensuring that sensors are assigned to appropriate tasks while maintaining network efficiency, fairness in resource distribution, and minimizing communication overhead remains a complex optimization problem. Traditional methods for resource management, such as heuristic algorithms and mathematical optimization techniques, often face significant trade-offs: they may offer good solutions but at the cost of high computational complexity, or achieve faster execution while compromising accuracy and fairness. To address these limitations, this paper introduces a novel game theory-based approach for resource allocation within Sensor-Cloud infrastructures. In the proposed model, each sensor acts as an autonomous rational agent that strategically selects an application to serve, aiming to maximize its own utility while contributing to overall system efficiency. By formulating this interaction as a non-cooperative game, the system dynamics naturally evolve toward a Nash equilibrium, where no sensor has an incentive to unilaterally change its allocation decision. This equilibrium represents an optimal and stable allocation of resources that balances efficiency, fairness, and computational cost. The simulation results demonstrate that the proposed method is near optimal.

Keywords: Resource Allocation; Sensor-Cloud Infrastructure; Game Theory; Distributed Learning

1 Introduction

The Internet of Things (IoT) refers to a network of physical objects equipped with sensors, actuators, software, and communication technologies that enable them to collect, exchange, and analyze data. From smart homes to industries and urban infrastructure, the IoT provides the foundation for smart services and data-driven decision-making [1, 2]. Among its components, Wireless Sensor Networks (WSNs), as key elements of the IoT, are responsible for gathering environmental data such as temperature, humidity, pressure, light, and vibration [3, 4]. However, the increasing number of sensors and the resulting volume of data lead to limitations in processing power, storage, and energy, which can reduce the efficiency of these networks [5]. To address these limitations, the concept of Sensor-Cloud Infrastructure emerged, connecting sensors and sensor networks to the cloud layer [6]. In this model, sensors collect data and transmit it to the cloud for scalable storage, analysis, and processing. By leveraging sensor virtualization, Sensor-Cloud enables the sharing, dynamic, and automatic allocation of sensing resources without users needing to worry about the physical location or hardware constraints of the sensors. This capability reduces resource wastage, enhances network flexibility and stability, and facilitates advanced applications such as environmental monitoring, smart healthcare, industrial control, and energy management [6, 7].

As this infrastructure expands and the number of applications relying on sensor data increases, the issue of efficient resource allocation to multiple applications has become increasingly important. Inadequate allocation can lead to network congestion, reduced Quality of Service (QoS), and increased energy consumption. While classic optimization approaches (like linear programming) may provide optimal solutions, they become computationally inefficient as the network size grows. On the other hand, although heuristic methods are fast, they do not guarantee optimality, and are insufficient to meet the multi-objective optimization requirements of existing cloud infrastructures [8]. In recent years, the use of game theory-based and machine learning-based methods for resource allocation have received significant attention from researchers [9-13]. In [14], a hybrid method based on Monte Carlo tree search and deep reinforcement learning is proposed for intelligent resource management. In this approach, deep reinforcement learning is employed to provide initial knowledge, while Monte Carlo tree search is used for efficient exploration of physical machines and for making better decisions in scheduling. In [15], a multi-dimensional resource allocation problem in sensing-communication-computation is considered, and a two-stage solution is proposed for it. In the first stage, an Adversarial Deep Q-network (DQN)-based algorithm is proposed for scheduling, channel allocation, and power control. In the second stage, a computational resource allocation method is proposed to jointly ensure low-latency data processing and high-security blockchain consensus.

Chitharanjan, et.al conducted a study on the state of resource allocation within sensor-cloud infrastructure. Their work elucidated the different types of resources requiring allocation, the multifaceted challenges inherent in this process, and a review of proposed methodologies designed to overcome these obstacles [16]. Nguyen et al. introduced a hierarchical resource allocation framework tailored for sensor-cloud environments. This framework was specifically engineered to address critical challenges such as the inherent dynamism of workloads, the constrained resource availability in sensor nodes, and the heterogeneity among these nodes [17]. Mahmood et al. proposed a distributed resource allocation algorithm for sensor-cloud infrastructure, emphasizing efficiency and scalability. Their algorithm is mindful of the fluctuating nature of workloads and the resource limitations of sensor nodes [18]. Mondal et al. presented a Quality of Service (QoS)-aware resource allocation framework for sensor-cloud infrastructure. The primary objective of this framework is to guarantee the fulfillment of QoS requirements for sensor applications while efficiently managing the finite resources of the sensor nodes [19]. Haseeb et al developed a secure resource allocation framework for sensor-cloud infrastructure. This framework aims to safeguard the confidentiality, integrity, and availability of sensor data, concurrently addressing the resource constraints of sensor nodes [20].

Game theory has garnered significant attention from researchers as a powerful tool for addressing resource allocation problems in computational environments such as fog computing and cloud computing [11, 12, 21-24]. The authors in [23] addressed the resource allocation problem in a fog-based Industrial IoT environment and modeled the problem as a two-stage Stackelberg game. In [24], the authors proposed a resource allocation method for fog computing based on a combination of non-cooperative game theory and an auction-based algorithm. The primary reasons for employing game theory in this domain include Modeling Autonomous Behavior, Finding Equilibrium, Optimal Resource Allocation, Managing Uncertainty and Competition, and Coalition Formation. Fog nodes or cloud servers are often conceptualized as autonomous agents pursuing specific objectives (e.g., cost minimization, profit maximization, or Quality of Service (QoS) optimization). Game theory provides a framework for modeling the interactions among these autonomous agents. A key objective in many such problems is to achieve a stable state, or “equilibrium,” where no agent has an incentive to unilaterally alter its strategy. Concepts like “Nash Equilibrium” enable researchers to identify such equilibrium points. By utilizing utility functions that represent the benefits or costs associated with resource allocation, game theory can facilitate the discovery of resource allocations that are optimal for the overall system or for specific agents. Fog and cloud networks encounter uncertainties regarding traffic volume, resource availability, and task requirements. Furthermore, competition exists among nodes or service providers for resources. Game theory can incorporate these uncertainties and competitive dynamics into the modeling

process. As previously mentioned, coalition formation methods within game theory can be leveraged to foster cooperation among nodes for enhanced workload and resource management. In summary, game theory offers a robust mathematical approach for analyzing and resolving the complex challenges of resource allocation in distributed and decentralized systems like fog and cloud computing, particularly when agents possess divergent or competing incentives.

Despite the considerable progress in resource allocation for sensor-cloud and fog computing environments, two important research gaps remain. First, many existing studies formulate the resource allocation problem as a centralized optimization task, which limits their scalability and adaptability in large-scale distributed sensor-cloud infrastructures. Second, although game-theoretic approaches have been widely applied to cloud and fog computing, relatively few studies investigate resource allocation in sensor-cloud environments through the framework of exact potential games, which provide desirable convergence and stability guarantees. These limitations motivate the development of a distributed game-theoretic resource allocation framework to ensures scalability, and convergence.

In this paper, we address the problem of resource allocation in a sensor-cloud infrastructure. We consider a scenario with multiple applications, each requiring data from various sensors. The core challenge is to achieve a fair allocation of resources to these applications. To tackle this, we propose a novel approach grounded in game theory. We design a potential game for this problem and subsequently present a learning algorithm to determine its Nash equilibrium. By exploiting the convergence properties of potential games, the system reaches a stable and efficient Nash equilibrium, where resource allocation between sensors and applications is achieved fairly and efficiently. The simulation results demonstrate that the proposed method exhibits suitable performance and can serve as an effective framework for resource management in Sensor-Cloud infrastructures. The primary contributions of this work are twofold:

1. We formulate the sensor resource allocation problem in a sensor-cloud infrastructure using game theory and establish that the devised game is an exact potential game. Within this exact potential game, each sensor acts as a player, competing with others to select the requesting application that yields the optimal outcome for itself, subject to the existing constraints.
2. A significant challenge in employing potential games is the design of a distributed learning algorithm capable of converging to the Nash equilibrium. This paper introduces a distributed learning algorithm specifically proposed to reach this equilibrium point.

This article is organized as follows: An overview of game theory is presented in Section 2. Section 3 contains our proposed algorithm. Simulation results and performance evaluation are reported in Section. 4. Finally, we conclude the paper and present future directions in Section 5.

2 Overview of Game Theory

Game theory is a branch of mathematics and decision sciences that studies the strategic behavior of agents in situations where the outcome of each player's decision depends on the decisions of other players. In the context of resource allocation in Sensor-Cloud infrastructure, sensors can be considered as players, and their choice of applications can be regarded as their actions. This model allows us to analyze how the interaction and competition among sensors can lead to stable states (equilibria).

A strategic game is defined as $\Gamma = \langle V, A, U \rangle$, comprising three main components:

-Set of Players: Players are denoted by $V = \{1,2, \dots, N\}$. In our problem, each player represents a sensor.

-Set of Actions: The set of possible actions for the players is defined as $A := \prod_{i=1}^{N} A_i$, where $A_i$is the set of actions for player i, and $a_i$is their chosen action. The joint action of all players is denoted by $a = (a_1, a_2, \dots, a_N) \in A_1 \times \cdots \times A_N$. The actions of other players, excluding player i, are denoted by a_{-i}, and their corresponding set by $A_{-i} = \prod_{j \neq i} A_j$.

-Utility Function: The utility function for each player i is defined as $u_i: A \to \mathbb{R}$, representing the player's benefit or efficiency for choosing a particular action.

Definition 1. Nash Equilibrium (NE): In a game $\Gamma = \langle V, A, U \rangle$, an action profile $a^* = (a_i^*, a_{-i}^*)$ is a Nash Equilibrium (NE) if for every player $i \in V$and for every alternative strategy $a_i \in A_i$, the relation $u_i(a^*) \geq u_i(a_i, a_{-i}^*)$ holds.

In other words, in a Nash equilibrium, no player can achieve a higher payoff by unilaterally changing their strategy. This state represents strategic stability in the system. In the Sensor-Cloud model, it signifies a resource allocation where no sensor has an incentive to change its service application.

Definition 2. Exact Potential Game: A strategic game Γ is an exact potential game with a potential function $\phi: A \to \mathbb{R}$, if for every player $i \in V$, for every $a_{-i} \in A_{-i}$, and for any two actions $a_i, a_i' \in A_i$, the following condition holds:

$$\phi(a_i, a_{-i}) - \phi(a_i', a_{-i}) = u_i(a_i, a_{-i}) - u_i(a_i', a_{-i}) \quad (1)$$

The potential function ϕ serves as a general indicator of the system's overall efficiency level. Changes in each player's utility correspond directly to changes in the potential function's value. This characteristic drives the players' decision-making process towards maximizing the system's overall potential. It has been proven that potential games always possess at least one Nash equilibrium, and the maxima of the potential function correspond to the Nash equilibria. Therefore, optimizing the potential function leads to finding stable states for resource allocation.

In the problem addressed by this paper, by modeling sensors and applications within the framework of a potential game, optimal resource allocation can be found, leading the system to a stable, convergent state. This approach not only reduces computational complexity compared to conventional optimization algorithms but also enables the analysis of the

collective behavior of sensors and an understanding of their decision-making dynamics.

3 Proposed Method

In this section, we first model the resource allocation problem in Sensor-Cloud infrastructure as an optimization problem, then we design a potential game to solve it, and finally introduce a distributed learning algorithm to achieve a Nash equilibrium.

3.1 Problem Definition

Suppose n applications $\{ap_1, ap_2, \dots, ap_n\}$ send their requests to the cloud layer to utilize the m available sensors in the infrastructure, denoted by the set $R = \{r_1, r_2, \dots, r_m\}$. In this structure, each application might require multiple resources, but each resource can only serve one application at any given time. The goal is to achieve fair and optimal allocation such that the system's efficiency rate is maximized.

To formulate the resource allocation problem as an optimization problem, the binary variables y_{ij} and x_{ij} are defined as follows:

$$y_{ij} = \begin{cases} 1 & ap_i \; requires \; r_j \\ 0 & otherwise \end{cases} \quad (2)$$

$$x_{ij} = \begin{cases} 1 & r_j \; is \; assigned \; to \; ap_i \\ 0 & otherwise \end{cases} \quad (3)$$

Accordingly, the resource allocation problem can be formulated as:

$$\begin{aligned} &\max_x \textstyle\sum_{i=1}^{n} \log\left(\sum_{j=1}^{m} x_{ij}\right) \\ &s.t. \textstyle\sum_{i=1}^{n} x_{ij} \le 1 \quad \forall j \\ &\quad x_{ij} \le y_{ij} \quad \forall i,j \\ &\quad x_{ij} \in \{0,1\} \end{aligned} \quad (4)$$

Optimization problem (4) is equivalent to the considered sensor allocation problem under the proportional fairness criterion. The objective function is concave and monotonically increasing, thereby promoting balanced resource allocation among competing applications. In particular, the marginal utility of assigning an additional sensor to an application decreases as the number of already assigned sensors increases. Consequently, the logarithmic objective naturally encourages proportional fairness across applications. The constraint $\sum_i x_{ij} \le 1$ ensures that each sensor can serve at most one application, while the constraint $x_{ij} \le y_{ij}$ restricts assignments to feasible sensor–application pairs only. Therefore, maximizing the logarithmic objective yields a proportionally fair allocation while simultaneously improving overall system efficiency.

The optimization problem formulated in (4) can be solved using centralized optimization techniques, such as Capacity Scaling or Successive Shortest Path algorithms. However, these centralized approaches require a coordinator with complete global knowledge of application requirements and sensor assignments, which may become impractical in large-scale and dynamic environments such as IoT and Sensor-Cloud systems. As a baseline solution, a sequential greedy algorithm can be employed using a shared memory structure that stores the number of sensors currently assigned to each application. Based on this shared information, each sensor selects the feasible application with the minimum number of assigned sensors at that time. The pseudocode of the greedy resource allocation algorithm is illustrated in Fig. 1. However, the greedy approach requires sensors to operate sequentially, which limits its scalability and adaptability in distributed environments.

Greedy Algorithm

Initialize global counter $q[n] = 0$
For each sensor r_j:
  $S = \{i \mid y_{ij} = 1\}$
  $i^* = \underset{i \in S}{\operatorname{argmin}}\, q$
  $q[i^*] = q[i^*] + 1$
  Assign r_j to ap_{i^*}

Figure 1: Greedy algorithm for the resource allocation problem

To overcome the limitations of centralized and sequential approaches, we propose a distributed game-theoretic framework. In the proposed model, each sensor is considered as a rational player that autonomously selects an application to serve, while a utility function is designed to encourage balanced and proportionally fair allocations. Furthermore, we show that the formulated interaction among sensors constitutes an exact potential game, which guarantees the existence of at least one Nash Equilibrium (NE) and enables convergence through simple distributed learning dynamics. Using the proposed decentralized learning mechanism, sensors iteratively update their decisions based solely on local observations and previously experienced utilities, without requiring any central coordinator or global system information. Consequently, the proposed approach enables the system to converge toward a near-optimal and proportionally fair resource allocation while remaining scalable, adaptive, and suitable for real-time dynamic Sensor-Cloud environments.

3.2 Designing a Potential Game for Resource Allocation

To formulate the resource allocation problem as a potential game $\Gamma = \langle V, A, U \rangle$, the set of players, action sets, utility functions, and the potential function must be defined appropriately.

- Set of Players (V): The set of players consists of all sensors participating in the resource allocation process; $V = \{r_1, r_2, \dots, r_m\}$.

- Set of Actions (A): The action of each player i is to select one of the requesting applications that require its service.

- Utility Functions (U): To achieve fairness in resource allocation, the utility function is designed such that sensors are encouraged to select applications with fewer allocated resources. Specifically, if player i selects application j, its utility is defined as:

$$u_i = \frac{1}{q_j} \quad (5)$$

where q_j denotes the total number of sensors currently assigned to application j under the current strategy profile. The value of q_j is computed as:

$$q_j = \sum_{i=1}^{m} \mathbb{I}\{a_i = ap_j\} \quad (6)$$

Where a_i represents the action selected by sensor r_i, $\mathbb{I}$ denotes the indicator function, which equals 1 if the condition $a_i = ap_j$ holds and 0 otherwise. According to the utility definition in (5), each sensor attempts to maximize its own utility by selecting applications with fewer assigned sensors. Consequently, the proposed utility function naturally promotes balanced and proportionally fair resource allocation among competing applications.

To demonstrate that the system converges to a stable operating point, we show that the formulated game is an exact potential game. According to Definition 2, it is sufficient to define a potential function φ such that Equation (1) is satisfied.

Lemma 1: The game Γ with the utility function defined in (5) is an exact potential game with the following potential function:

$$\varphi(a) = \sum_{i=1}^{n} \sum_{j=1}^{q_i} \frac{1}{j} \quad (7)$$

where n is the number of applications and q_i is the number of players that have chosen application i.

Proof: According to the definition, we must show that the change in each player's utility function due to a unilateral change in strategy is exactly equal to the change in the overall potential function. Assume player i changes its action from selecting application a_i to application a_i'. Keeping the strategies of other players (a_{-i}) constant, only the parts of the potential function related to applications a_i and a_i' change.

If, in the initial state, the number of sensors allocated to application a_i is q_i, and for application a_i', it is q_i', then after player i's action change, these values change to $q_i - 1$ and $q_i' + 1$, respectively. The difference in the potential function is calculated as follows:

$$\phi(a_i, a_{-i}) - \phi(a_i', a_{-i}) = \sum_{j=1}^{q_i} \frac{1}{j} + \sum_{j=1}^{q_i'} \frac{1}{j} - \sum_{j=1}^{q_i - 1} \frac{1}{j} - \sum_{j=1}^{q_i' + 1} \frac{1}{j}$$
$$= \frac{1}{q_i} - \frac{1}{q_i' + 1}$$

On the other hand, the difference in player i's utility is $u_i(a_i, a_{-i}) - u_i(a_i', a_{-i}) = \frac{1}{q_i} - \frac{1}{q_i'+1}$. Since the changes in the utility function and the potential function are identical, the game is an exact potential game.

3.3 Proposed Learning Algorithm

To achieve Nash equilibrium in this game, a distributed iterative learning algorithm is proposed that runs in discrete time steps. In the proposed approach, all sensors update their strategies in parallel and cooperatively without requiring centralized coordination. The proposed algorithm guides the sensors toward a near-optimal resource allocation by combining two complementary mechanisms: exploration and exploitation. Exploration enables sensors to investigate alternative actions and avoid poor local solutions, while exploitation allows them to reuse previously successful actions. The pseudocode of the proposed resource allocation algorithm is presented in Fig. 2.

Proposed Resource Allocation Algorithm

Input: global shared memory $q[n]$, exploration rate ε
(Run at each player r_i)

$t = 0$
Each player $r_i \in V$ selects an application ap_j randomly from its action set.
Compute initial utility $u_i(a_i(0))$ according to (5) and (6)
Set the best previous action $a_i^* = a_i(0)$, and $BU_i = u_i(a_i(0))$
Repeat:
  $t = t + 1$
  Generate a random value $p \in [0,1]$.
  If $p < \varepsilon$ (Exploration Phase):
    Select a new action a_i randomly from the action set.
  Else (Exploitation Phase):
    Select the action a_i^* that yielded the highest utility in previous steps.
  Execute the selected actions and observe the new state of the system.
  Each player computes its current utility $u_i(a_i(t), a_{-i}(t))$ according to (5) and (6)
  If $u_i(a_i(t), a_{-i}(t)) > BU_i$:
    $BU_i = u_i(a_i(t), a_{-i}(t))$
    $a_i^* = a_i(t)$
Until the action profile converges and no player can further improve its utility.

Figure 2: Proposed algorithm

At $t = 0$, each sensor r_i randomly selects an application $a_i(0)$ from its action set, which contains all applications requiring the service of sensor r_i. The sensor then computes its initial utility $u_i(a_i(0))$. Each sensor maintains a record of its best previously observed action, denoted by a_i^*, together with the corresponding best utility value BU_i. Initially, a_i^* and BU_i are set to $a_i(0)$ and $u_i(a_i(0))$, respectively.

For $t \geq 1$, each sensor selects a random action with probability ε (exploration phase) or chooses its previously best-performing action a_i^* with probability $1 - \varepsilon$ (exploitation phase). After executing the selected action, the sensor computes its utility $u_i(a_i(t), a_{-i}(t))$ and updates a_i^* and BU_i whenever a better utility value is obtained.

This iterative learning process continues until the system converges to a Nash Equilibrium. The combination of exploration and exploitation enables the algorithm to avoid poor local solutions while gradually converging toward a stable and near-optimal allocation corresponding to a maximized potential function. The flowchart of the proposed algorithm is illustrated in Fig. 3.

3.4 Time Complexity Analysis of the Proposed Algorithm

The proposed learning algorithm is executed independently and in parallel by each sensor r_i. At each iteration t, each sensor performs only a constant number of operations, including random number generation, action selection (exploration or exploitation), utility computation according to

Equations (5) and (6), and updating its best previously observed action. Since the utility computation relies on shared counters that maintain the number of sensors assigned to each application, the utility evaluation can be performed in constant time, i.e., $O(1)$. Therefore, the computational complexity per sensor per iteration is $O(1)$. Assuming that the algorithm converges after T iterations, the overall computational complexity per sensor becomes $O(T)$. Because all sensors operate simultaneously and independently, the proposed method is highly scalable and suitable for distributed Sensor-Cloud environments. Moreover, the exact potential game structure guarantees convergence to a Nash Equilibrium under the proposed learning dynamics. In practice, the combination of the potential game framework and the ε-greedy exploration mechanism enables the algorithm to converge efficiently within a reasonable number of iterations for moderately sized systems.

4 Simulation Results

To evaluate the performance of the proposed algorithm, a series of simulation experiments were conducted in a Python programming environment. In all experiments, it is assumed that applications independently and randomly request access to sensors, and resource allocation is performed based on the mechanism designed in the previous section.

In the simulation environment, the number of sensors and applications were both set to 100, and the experiments were run for 1000 iterations. The exploration rate of the algorithm (parameter ε) was set to 0.01. In each iteration, sensors update their actions based on the rules presented in the proposed algorithm (combining exploration and exploitation), and the values of the potential function and utilities are calculated.

Figure 4 shows the trend of the average potential function value $\varphi(a)$ (according to (7)) throughout the algorithm's iterations. As observed, the potential function value increases monotonically and converges to a constant value after a limited number of iterations. This behavior indicates the algorithm's convergence to a stable Nash equilibrium, which confirms the potential property of the designed game.

Figure 5 displays the changes in the average player utility values over the iterations. Comparing Figures 4 and 5 reveals that any increase in the local utility of the players is accompanied by an increase in the system's overall potential function. This correlation suggests that the individual behavior of sensors leads to an improvement in the overall system efficiency which is a desirable property of game theory-based algorithms in distributed environments.

The exploration rate ε is a key parameter affecting the performance of the proposed algorithm. To investigate its impact, experiments were conducted with different values of ε, and the results are presented in Figures 6 and 7. The results indicate that for larger values of ε, the range of fluctuation in the overall utility function is greater within shorter time intervals, as players try new strategies more frequently, leading to more dynamic system behavior. Conversely, with very small values of ε, the learning process becomes more conservative, and the convergence speed to the optimal response decreases. These findings suggest that there is an optimal point for the parameter ε that balances the speed and stability of equilibrium convergence. In the simulations, ε = 0.01 yielded the best results in terms of both speed and stability.

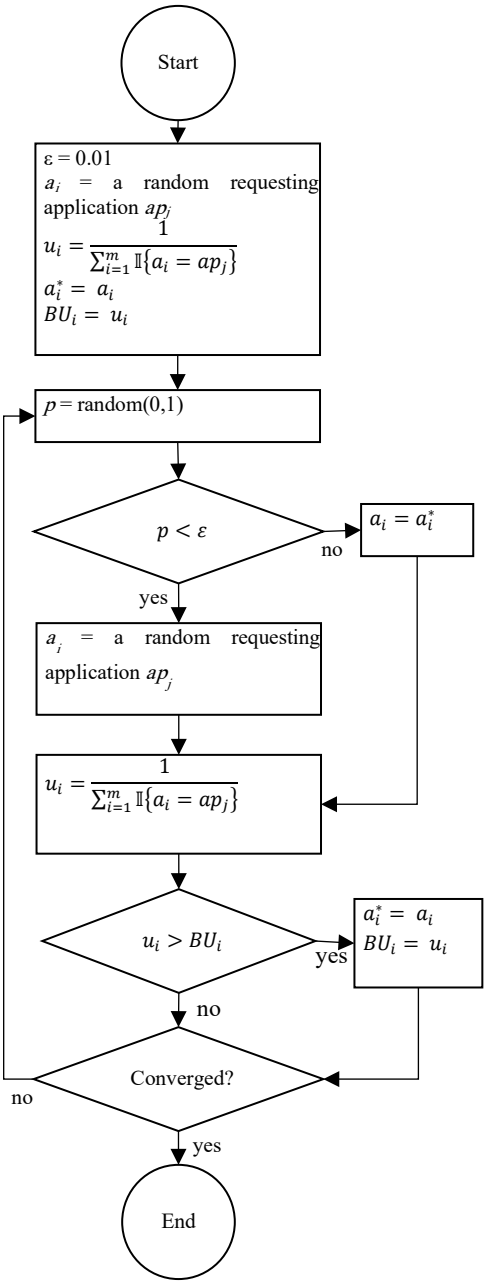


Figure 3: The flowchart of the proposed algorithm

To investigate the impact of increasing the number of players on the average utility, an experiment was conducted with a fixed number of requesting applications (five applications), while gradually increasing the number of players. The corresponding results are presented in Fig. 8. The results demonstrate that, with a fixed number of applications, the

average utility of players decreases as the number of players increases and asymptotically approaches zero. The reason is that, for each player i selecting application ap_j, the number of sensors assigned to that application increases as the total number of players grows: $\lim_{m\to\infty} q_j = \infty$. Consequently, according to the utility function defined in Equation (5), the utility of each individual player approaches zero: $\lim_{m\to\infty} u_i = 0$. This behavior reflects the increasing competition among players for a limited number of applications and resources.

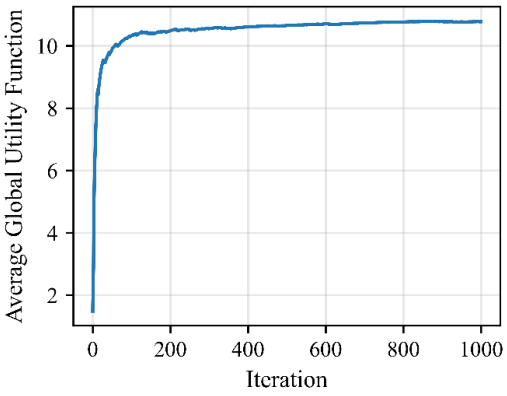


Figure 4: Average global utility function

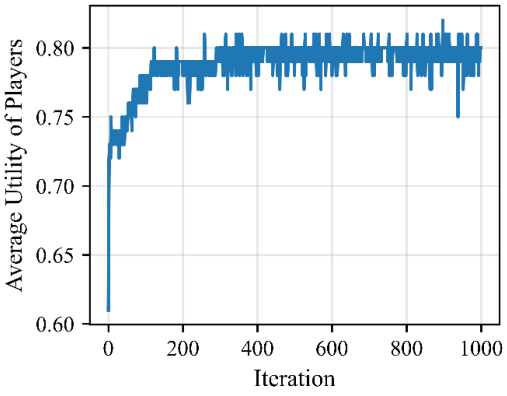


Figure 5: Average utility of players

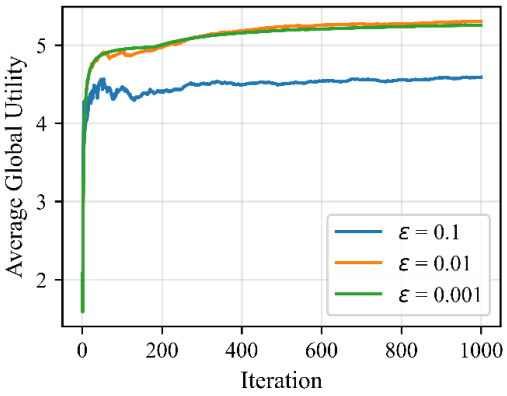


Figure 6: Average global utility function with different exploration rate

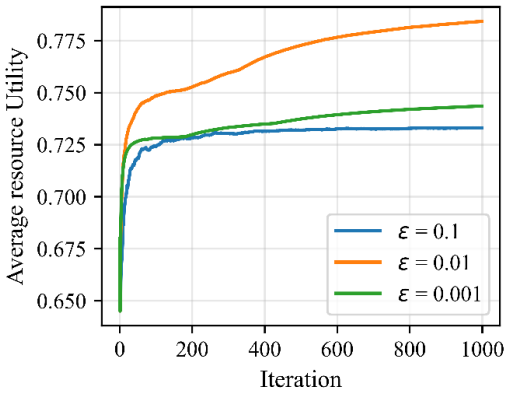


Figure 7: Average players' utility with different exploration rate

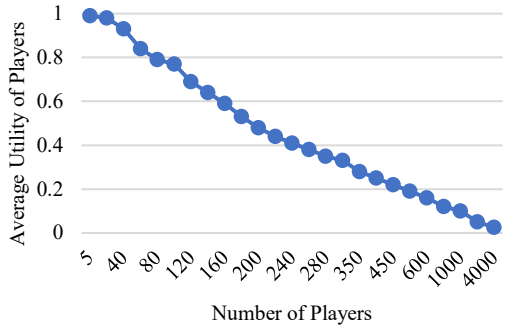


Figure 8: Average utility of players versus the number of players

To evaluate the effectiveness of the proposed method, its performance was compared with the Stackelberg game-based approach presented in [23], the non-cooperative game approach proposed in [24], the greedy algorithm, and the optimal centralized method introduced in Section 3.1. Figure 9 compares the number of successful resource assignments achieved by the considered algorithms. The results indicate that both the proposed algorithm and the Stackelberg game-based method achieve performances very close to the optimal centralized solution. Moreover, their performances are significantly better than those of the greedy algorithm and the non-cooperative game approach. The superior performance of the proposed method and the Stackelberg game approach can be attributed to their convergence to a Nash Equilibrium (NE), which leads to a near-optimal and proportionally fair allocation of resources.

One important advantage of the proposed method over the Stackelberg game-based approach is its significantly lower computational complexity. The time complexity of the Stackelberg game method is $O(nm\log m\log(1/\delta) + Tmn^2\log n)$, where δ is the precision parameter of the Stackelberg algorithm and T denotes the number of learning iterations. In contrast, the proposed algorithm has a substantially lower time complexity of $O(T)$.

Another important advantage is that the proposed algorithm operates in a fully distributed manner, whereas the Stackelberg game method requires centralized coordination and global

system information. Therefore, the proposed method is particularly suitable for large-scale, dynamic, and decentralized fog, edge, and Sensor-Cloud environments where scalability, fairness, and stability are more critical than exact global optimality. A comparison of the considered algorithms in terms of computational complexity, convergence to Nash Equilibrium, optimality, and coordination requirements is presented in Table 1.

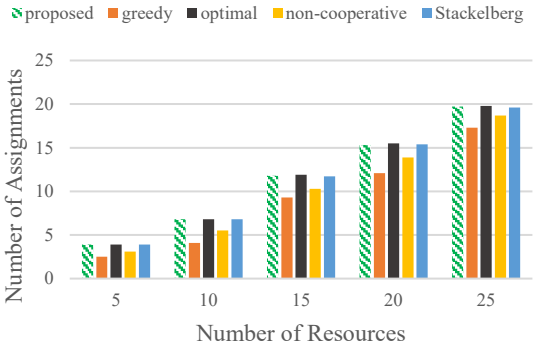


Figure 9: Number of resource assignments versus number of resources

The simulation results demonstrate that the proposed algorithm rapidly converges to a stable operating point corresponding to a Nash Equilibrium. Furthermore, the value of the potential function increases consistently with the local utilities of the players, indicating a positive correlation between individual incentives and overall system efficiency. The results also show that the exploration rate ε significantly affects both the convergence speed and the stability of the learning process. In addition, the proposed method achieves performance significantly better than the greedy and non-cooperative approaches while remaining very close to the optimal centralized solution. Therefore, the proposed algorithm provides an effective trade-off between computational efficiency, scalability, fairness, and allocation performance, making it a suitable distributed optimization framework for resource allocation in Sensor-Cloud infrastructures.

5 Conclusion and Future Work

In this paper, we introduced a game theory-based learning algorithm for intelligent and optimal resource allocation in cloud and Sensor-Cloud environments. Aiming to address the challenges associated with the dynamic allocation of resources to applications with varying demands, we designed a framework where each sensor or computational resource acts as an intelligent agent. Through collaboration and competition with other agents, it achieves a stable equilibrium in resource allocation. Simulation results demonstrated that the proposed method achieves performance very close to the optimal centralized solution while significantly outperforming greedy and conventional non-cooperative approaches. In addition, the proposed algorithm exhibits low computational complexity, scalability, and fully distributed operation, making it suitable for large-scale and dynamic environments such as the Internet of Things (IoT), Wireless Sensor Networks (WSNs), fog computing, edge computing, and Sensor-Cloud infrastructures.

Despite the achievements of this research, several opportunities exist for future investigations:

-Integration with Deep Reinforcement Learning (DRL): Combining the proposed algorithm with Deep Reinforcement Learning techniques for optimal resource management at much larger scales and with more complex demand patterns.

-Energy Consumption Optimization: Incorporating energy consumption as a criterion into the game's objective function, such that the algorithm optimizes not only resource allocation but also the overall system's energy consumption.

-Reliability and Fault Tolerance: Investigating how the proposed algorithm can maintain its performance in the presence of potential faults in sensors or computational resources and developing fault-tolerant mechanisms.

Table 1: Comparison of algorithms in terms of time complexity, convergence to Nash equilibrium, and optimality, and coordination requirements

Algorithm	Time complexity	Reach NE	Reach optimal solution	Central Coordination
proposed	$O(T)$	Yes	No	No (fully distributed)
greedy	$O(mn)$	-	No	No
optimal	$O(m^2 n \log(m+n))$	-	Yes	Yes
Non-cooperative	$O(TnP)$	No	No	Yes
Stackelberg	$(O(nm \log m \log(1/\varepsilon) + Tmn^2 \log n))$	Yes	No	Yes

Disclosure of Potential Conflicts of Interest

The Authors declare that there is no conflict of interest

Reference

[1] W. Fan, X. Chun, Z. Fan, R. Zhang, S. Liu, and Y. a. Liu, "Dual-agent DRL-based service placement, task scheduling, and resource allocation for multi-sensor and multi-user edge computing networks," *IEEE Transactions on Network Science and Engineering,* 2025. DOI: 10.1109/TNSE.2025.3560402

[2] B. Dwivedy, M. Moharana, and M. Nayak, "Internet of Things over the last decade: a review of frameworks, architecture, services, and platforms," *Smart Science,* pp. 1–36, 2026. DOI: 10.1080/23080477.2026.2640359

[3] C. Saadouni, El Jaouhari, S., Tamani, N., Ziti, S., Mroueh, L. and El Bouchti, K., "Identification techniques in the internet of things: Survey, taxonomy and research frontier," *IEEE*

Communications Surveys & Tutorials, 2025. DOI: 10.1109/COMST.2025.3541165

[4] K. Ramamurthy, S. Kulanthaivelu, and S. A. Jenish, "Future directions and emerging trends of adaptive artificial intelligence in wireless sensor networks, Internet of Things, and Industrial Internet of Things," *Adaptive AI in Sensor Informatics,* pp. 291–308, 2026. DOI: 10.1016/B978-0-443-36412-9.00003-0

[5] F. Zijie, M. A. Al-Shareeda, M. A. Saare, S. Manickam, and S. Karuppayah, "Wireless sensor networks in the internet of things: review, techniques, challenges, and future directions," *Indonesian Journal of Electrical Engineering and Computer Science,* vol. 31, no. 2, pp. 1190–1200, 2023. DOI: 10.11591/ijeecs.v31.i2

[6] S. Bose and N. Mukherjee, "SenSchedule: Scheduling heterogeneous resources in sensor-cloud infrastructure," *IEEE Transactions on Services Computing,* vol. 15, no. 4, pp. 1825–1840, 2020. DOI: 10.1109/TSC.2020.3022679

[7] K. Gadouh and H. Koubaa, "A Literature Survey on Resource Allocation for Network Function Virtualization With and Without Machine Learning in Cloud Computing," *International Journal of Network Management,* vol. 36, no. 1, p. e70034, 2026. DOI: 10.1002/nem.70034

[8] D. Bodra and S. Khairnar, "Machine learning-based cloud resource allocation algorithms: a comprehensive comparative review," *Frontiers in Computer Science,* vol. 7, p. 1678976, 2025. DOI: 10.3389/fcomp.2025.1678976

[9] G. Sripathi and D. A. Khan, "Proactive Resource Allocation Framework with SVM Regression for Cloud Environments," *SN Computer Science,* vol. 7, no. 1, p. 90, 2026. DOI: 10.1007/s42979-025-04541-w

[10] W. K. Awad, K. A. Z. Ariffin, M. Z. A. Nazri, and E. T. Yassen, "Resource allocation strategies and task scheduling algorithms for cloud computing: A systematic literature review," *Journal of Intelligent Systems,* vol. 34, no. 1, 2025. DOI: 10.1515/jisys-2024-0441

[11] Z. Xu, F. Ban, and P. Fotia, "Efficient QoS processing for internet of medical things using non-cooperative game theory: resource allocation in cloud framework," *Annals of Operations Research,* vol. 356, no. 1, pp. 427–442, 2026. DOI: 10.1007/s10479-024-05857-1

[12] M. Wang, J. Liu, Y. Li, E. Tian, and C. Peng, "A comprehensive survey of game-theoretic approaches for networked systems," *Systems Science & Control Engineering,* vol. 14, no. 1, p. 2622131, 2026. DOI: 10.1080/21642583.2026.2622131

[13] P. William, S. Nirkhi, M. Almakki, S. Pushpa, S. Kumaraguru, and B. A. Tingare, "AI-Powered Framework for Real-Time Dynamic Resource Allocation in Cloud Computing Platforms," in *2026 5th International Conference on Innovative Practices in Technology and Management (ICIPTM)*, 2026: IEEE, pp. 1–6. DOI: 10.1109/ICIPTM69057.2026.11465550

[14] Y. Huang, H. Xu, and H. Gao, "RMVS: Energy-Efficient VM Scheduling for Flexible Manufacturing in Sensor-Cloud Industrial IoT Environments," *IEEE Network,* 2026. DOI: 10.1109/MNET.2025.3650549

[15] B. Huang, J. Feng, and X. Jin, "Sensing-communication-computing integrated resource allocation for AI-empowered trustworthy IoT," *IEEE Transactions on Consumer Electronics,* 2025. DOI: 10.1109/TCE.2025.3546948

[16] K. Chitharanjan and SenthilKumar, "A study of resource allocation techniques in cloud computing," *International Journal of Business Information Systems,* vol. 36, no. 2, pp. 254–269, 2021. DOI: 10.1504/IJBIS.2021.112830

[17] T. A. Nguyen, D. Min, and E. Choi, "A hierarchical modeling and analysis framework for availability and security quantification of IoT infrastructures," *Electronics,* vol. 9, no. 1, 2020. DOI: 10.3390/electronics9010155

[18] O. A. Mahmood, A. R. Abdellah, A. Muthanna, and A. Koucheryavy, "Distributed Edge Computing for Resource Allocation in Smart Cities Based on the IoT," *Information,* vol. 13, no. 7, 2022. DOI: doi.org/10.3390/info13070328

[19] A. Mondal, S. Misra, G. Das, and A. Chakraborty, "QoS-Aware Resource Allocation for Green Sensor-as-a-Service Provisioning in Vehicular Multi-Sensor-Cloud," *IEEE Transactions on Green Communications and Networking,* vol. 7, no. 1, pp. 224–233, 2022. DOI: 10.1109/TGCN.2022.3216374

[20] K. Haseeb, A. Almogren, I. U. Din, N. Islam, and A. Altameem, "SASC: Secure and authentication-based sensor cloud architecture for intelligent internet of things," *Sensors,* vol. 20, no. 9, 2020. DOI: 10.3390/s20092468

[21] S. Sheela and S. M. D. Kumar, "Collaborative Model for Task Scheduling and Resource Allocation in Fog–Cloud Network Using Game Theory," *International Game Theory Review,* vol. 27, no. 2, p. 2450022, 2025. DOI: 10.1142/S0219198924500221

[22] A. Bandyopadhyay *et al.*, "Quantum game theory-based cloud resource allocation: A novel approach," *Mathematics,* vol. 13, no. 9, p. 1392, 2025. DOI: 10.3390/math13091392

[23] Y. Jie, C. Guo, K.-K. R. Choo, C. Z. Liu, and M. Li, "Game-theoretic resource allocation for fog-based industrial internet of things environment," *IEEE Internet of Things Journal,* vol. 7, no. 4, pp. 3041–3052, 2020. DOI: 10.1109/JIOT.2020.2964590

[24] H. Mohammady Talvar, H. Haj Seyyed Javadi, H. Navidi, and A. Rezakhani, "A new resource allocation method in fog computing via non-cooperative game theory," *Journal of Intelligent & Fuzzy Systems,* vol. 41, no. 2, pp. 3921–3932, 2021. DOI: https://doi.org/10.3233/JIFS-202122