Journal of Information and Organizational Sciences
Volume 49, Number 1 (2025)
Journal homepage: jios.foi.hr

JIOS

DOI: 10.31341/jios.49.1.1
Open Access

UDC 004.89:519:83:629.3
Original Scientific Paper

Optimized Offloading in Vehicular Edge Computing: A Game Theoretic Analysis

Abdelkarim Ait Temghart[1]*, Mbarek Marwan[2] and Mohamed Baslam[1]

[1]TIAD laboratory, FST, Sultan Moulay Slimane University, Beni Mellal, Morocco
[2]*ENSIAS, Mohamed V University, Rabat, Morocco*
*Correspondence: aittemghart.abdelkarim@gmail.com

PAPER INFO

Paper history:
Received 29 July 2024
Accepted 16 October 2024

Citation:
Ait Temghart, A., Marwan, M. & Baslam, M. (2025). Optimized Offloading in Vehicular Edge Computing: A Game Theoretic Analysis. In Journal of Information and Organizational Sciences, vol. 49, no. 1, pp. 1-14

ABSTRACT

This paper introduces a novel approach to Vehicle Edge Computing (VEC), addressing the need for low-latency, high-reliability applications in vehicle networks. By leveraging nearby Multi-access Edge Computing (MEC) resources, VEC enhances data processing speed and reliability for applications like autonomous driving, real-time traffic management, and infotainment systems. The proposed solution models a multi-user non-cooperative computation offloading game in vehicular MEC networks, where each vehicle adjusts its offloading probability based on factors like distance to the MEC access point, communication model, and competition for resources. Additionally, a best response-learning algorithm is designed based on the computation offloading game model. The approach focuses on maximizing each vehicle's utility while ensuring convergence to a single, stable equilibrium under defined conditions. To demonstrate the effectiveness and performance of the proposed algorithms, comprehensive experiments were performed. Numerical results demonstrate the fast convergence and improved performance achieved.

Keywords: Game theory, Non-cooperative game, Vehicle Edge Computing, Computation offloading

1. Introduction

The rapid progress in Intelligent Transportation Systems (ITS) and automated driving technology has led to the development of numerous promising applications, including driving safety, traffic optimization, and in-vehicle entertainment [1], [2]. The demand for enhanced in-vehicle computing and storage capabilities has increased due to the proliferation of computationally intensive and time-critical applications [3], [4]. However, it is challenging for individual vehicles to fully satisfy the requirements of these applications using their own local resources due to constraints such as physical space limitations and economic costs. The emergence of 5G networks has stimulated the development of MEC as a solution to reduce end-to-end transmission delays and alleviate compute and storage constraints [5], [6], [7]. However, MEC (Mobile Edge Computing) servers are constrained by limited computing and storage resources [8], [9], which can pose significant challenges when faced with the increasing computational demands from vehicles. This resource limitation may result in degraded Quality of Service (QoS) for vehicle applications, diminishing the benefits of offloading tasks. Thus, efficient task offloading decisions become essential [10]. Vehicles must carefully manage the trade-off between application execution time and the availability of MEC resources. This balance is critical to avoid resource exhaustion, maintain optimal workload distribution, and ensure seamless application performance in ITS.

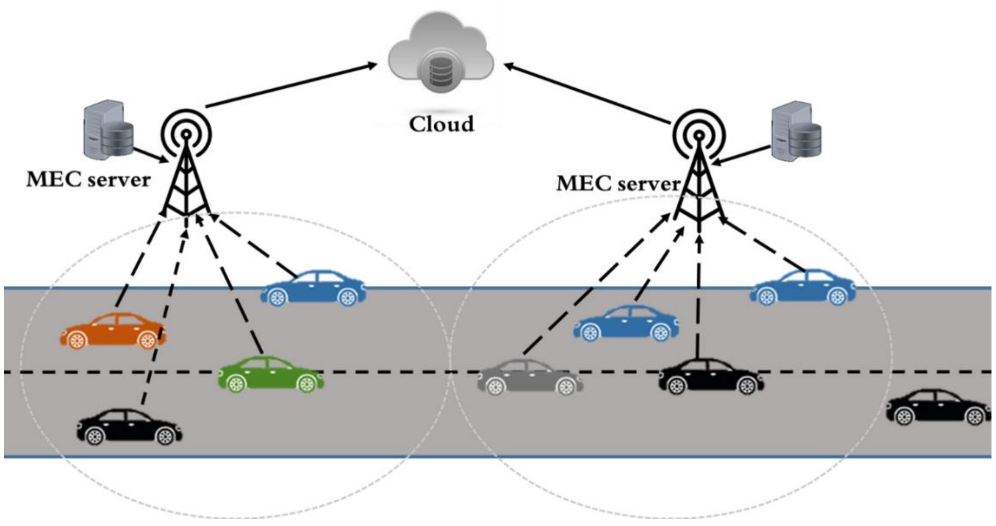

Figure 1. Competition-Based Offloading in Vehicular MEC Networks

In multi-vehicle computation offloading scenarios, as illustrated in Figure 1, both centralized and distributed approaches are employed to manage offloading strategies. Centralized methods focus on optimizing resource allocation for all vehicles through server-side planning, whereas distributed methods depend on interactions among vehicles to evaluate resource availability and maximize individual utility before offloading tasks to MEC servers [11], [12]. This paper focuses on distributed computation offloading, where each vehicle competes for MEC server resources. Vehicles independently determine their offloading strategies based on the current availability of MEC resources. The interaction among vehicles is modeled using a game-theoretic framework, which analyzes offloading strategies by assuming rational decision-making behavior. This paper's main contributions include:

1) Proposing a multi-user non-cooperative computation offloading game for MEC scenarios.

2) Designing a participant's payoff function considering factors like distance to MEC access points and competition for resources.

3) Developing a distributed algorithm ensuring convergence to a stable equilibrium.

4) Conducting experiments to analyze the proposed algorithms' effectiveness and performance.

The rest of this paper is structured as follows: Section II presents a review of related work. Section III introduces the computation offloading game model, including detailed discussions of the utility and price functions within the payoff structure. Section IV describes a distributed algorithm for solving the game. In Section V, we discuss the results, focusing on the convergence and uniqueness of the Nash equilibrium. Lastly, Section VI concludes the paper and suggests directions for future research.

2. Related Works

The challenge of computation offloading for mobile devices within MEC (Mobile Edge Computing) or MCC (Mobile Cloud Computing) architectures has been widely researched. Muoz et al. in [13] presented a framework for jointly optimizing the use of radio and computing resources using the trade-off between energy consumption and latency. In their approach, the authors focus on balancing energy consumption and latency by jointly optimizing the use of radio and computing resources. To minimize both the execution latency of all tasks and the energy consumption of the mobile device. Dinh et al. [14] proposed an optimization framework of offloading from a single mobile device to multiple edge devices. Zhang Jianchao studied multi-user computation offloading in dynamic mobile cloud computing environments [15]. Their research used a stochastic game approach to show that dynamic offloading decisions can achieve a pure-strategy Nash Equilibrium (NE). To ensure convergence to this NE, they developed the MASL algorithm, which provides a guaranteed convergence rate. The author of [16] formulates a bi-level optimization framework for efficient computation offloading in MEC systems, using game theory to achieve Nash equilibrium and optimize resource pricing and allocation. his algorithm reduces system costs and maximizes profits for both ESP and DMs. Chen introduced a game-theoretic framework for efficient computation offloading in mobile cloud computing environments [17]. The study modeled the decision-making process for decentralized offloading among mobile users as a decentralized computation offloading game. To enhance system utility, Lyu et al. proposed a semi-distributed heuristic offloading decision algorithm (HODA) that optimizes both offloading decisions and the allocation of communication and computation resources [18]. In a different approach, Bi et al. developed an offloading strategy aimed at maximizing computation rates within a multi-user MEC network powered by wireless power transfer (WPT) [19]. Their scheme uses a binary computation offloading policy, where each energy-harvesting device chooses between processing tasks locally or offloading them to an MEC server. An adaptive task offloading and resource allocation algorithm using deep reinforcement learning in mobile edge computing is proposed in [20] to optimize response time, energy consumption and system utility, taking user mobility into account. In [21], the authors introduce a Stackelberg game to optimize task offloading and pricing in mobile edge computing, proposing a differentiated pricing strategy to enhance server profit and reduce user latency. Cardellini et al. explored a three-tier architecture for mobile computing by proposing a model to assess how computation offloading impacts user-perceived performance [22]. They framed the problem as a generalized Nash equilibrium and developed a distributed algorithm to compute this equilibrium. Liu et al. tackled the task offloading challenge using a matching theory approach, introducing pricing-based matching algorithms aimed at minimizing overall network delay [23]. The authors in [24] developed the ECORA strategy to enhance resource allocation in vehicular networks, focusing on collaborative computation and storage, as well as adaptive task offloading, which significantly reduced task processing delays and improved network load balancing. The authors [25] explore online task offloading in mobile edge computing networks, focusing on time-variant task arrivals and diverse user requirements. They introduce a collaboration architecture that divides edge nodes into public and private ones, developing a system model for task latency analysis. A deep reinforcement learning-based online task scheduling algorithm is proposed, demonstrating significant latency reductions of 1.09%, 2.65%, and 4.34% compared to offline methods for varying correlation coefficients. The authors in [26] introduce a method that deploys passive RIS in a cell free network to enhance wireless capacity. Using deep reinforcement learning (DRL), the authors formulate a joint beamforming optimization problem to address complexity, achieve better latency and energy efficiency in mobile edge computing (MEC) environments. The work presented in [7] explores task scheduling strategies for MEC servers located on Roadside Units (RSUs). The primary aim is to minimize the total completion time for various applications. The authors introduce an optimized algorithm that prioritizes tasks and applications, meeting specific time constraints for task completion. Dai introduced a new offloading scheme for Vehicular Edge Computing (VEC) in [27], leveraging deep reinforcement learning to minimize processing latency. This scheme effectively reduces latency by dynamically optimizing offloading decisions between vehicles, edge servers, and the cloud, adapting to the changing conditions of the computing environment. In this paper, we emphasize the importance of incorporating vehicle locations and task deadline constraints into the MEC offloading process. Based on these factors, a best-response algorithm is utilized to guarantee that the strategy chosen by each vehicle leads to a unique Nash equilibrium.

3. Computation Offloading Model

For Vehicular Edge Computing, the development of an efficient compute offloading model is crucial for optimizing resource allocation and improving system performance. In this section, a description of the main elements and parameters of the offloading model is given, with a detailed perspective on the decision-making framework and the criteria used to evaluate system performance.

3.1. Parameters Abstraction

The application in each vehicle is abstracted using three essential parameters: input data size (L_i), computational complexity (a_i), and completion deadline ($t_{i;max}$). These parameters encapsulate the fundamental characteristics of the application, influencing its execution time and resource requirements.

3.2. Local Execution Time

The time taken for vehicle i to execute a task locally depends on the computational power of its onboard CPU ($f_{i;l}$) and the complexity of the application (α_i). This can be mathematically represented as:

$$t_{i;l} = \frac{\alpha_i L_i}{f_{i;l}} \tag{1}$$

3.3. Communication Model

The communication link between each vehicle and the MEC access point is represented by a frequency-flat block-fading Rayleigh channel. The path loss (θ_i) is influenced by the distance (d_i) between the vehicle and the access point. Additionally, the model incorporates the channel-fading coefficient (h_i) and the noise power (N_0) from white Gaussian noise. The data transmission rate (R_i) for vehicle i is then calculated based on the available channel bandwidth (W_i) and the transmission power (P_i) as follows:

$$R_i = W_i \, log_2 \left(1 + \frac{P_i \, d_i^{-\theta_i}}{N_0} h_i^2\right) \tag{2}$$

3.4. Transmission Delays

The transmission delays for vehicle i during computation offloading include uplink ($t_{i;U}$) and downlink ($t_{i;D}$) components. These delays are determined by the uplink and downlink transmission overheads ($\beta_{i,U}$ and $\beta_{i,D}$), as well as the data transmission rate (R_i). Mathematically, they are given by:

$$t_{i;U} = \frac{\beta_{i,U} \, L_i}{R_i} \tag{3}$$

$$t_{i;D} = \frac{\beta_{i,D} \, L_i}{R_i} \tag{4}$$

3.5. MEC Execution Time

The execution time ($\tau_{i;e}$) for vehicle i's application in the MEC is dependent on the size of the input data (L_i) and the processing capability of the MEC server (f_e). This can be represented as:

$$\tau_{i;e} = \frac{\alpha_i \, L_i}{f_e} \tag{5}$$

3.6. Total Execution Time

The total computation offloading time ($t_{i;e}$) for vehicle i is the combination of three factors: uplink transmission delay, processing time at the MEC server, and downlink transmission delay.

$$t_{i;e} = t_{i;U} + \tau_{i;e} + t_{i;D} \tag{6}$$

A clear understanding of these components is fundamental for designing efficient utility and pricing models in the offloading game, allowing for optimized decision-making and resource allocation.

4. System Model

In this section, we begin by presenting the fundamental model of the computation offloading game, delineating the specific utility and price functions within the payoff framework based on parameters given in table 1. Next, we model the computation offloading as a non-cooperative game, applying the concept of Nash equilibrium. We also present proofs to demonstrate the existence and uniqueness of the equilibrium.

Parameter Meaning	**Parameter Meaning**
$n = 7$	Number of vehicles
$\sigma_i = 0.7$	MEC pricing factor
$\lambda_i = 0.7$	Task arrival rate
$t_{\mathrm{i,max}}$	The maximum execution time

α_i	Task computational complexity
$\beta_{i,U}$	Overhead of uplink transmission
$\beta_{i,D}$	Downlink transmission overhead and the ratio of output
$L_i = 1MBits$	The input data size
f_{il}	Local computation capability
f_e	MEC computation capability
θ_i	Path loss exponent
h_{ij}	The channel fading coefficient
d_i	Distance to RSU
W_i	Channel bandwidth
$N_0 = 0.01$	The white Gaussian noise power
$P_i = 0.2$	Transmit power

Table 1. Parameters Table

4.1. Utility of Non-cooperative of the Computation Offloading

Task execution time plays a crucial role in determining the vehicle's utility, considering the variables $t_{i;l}$, $t_{i;e}$, and $t_{i;\max}$ discussed in the previous section. Shorter execution times, whether the task is processed locally or offloaded to the MEC server, lead to higher utility. However, if the execution time exceeds $t_{i;max}$, the vehicle gains no benefit and may incur penalties. As a result, the utility function for vehicle i in a game with computation offloading can be derived as follows:

$$U_i(\rho) = \frac{t_{i;\max} - t_{i;l}}{t_{i;\max}}(1 - \rho_i) + \frac{t_{i;\max} - t_{i;e}}{t_{i;\max}}\rho_i \tag{7}$$

MEC environments can benefit from implementing a dynamic pricing mechanism to manage the offloading of vehicles due to limited computing and storage resources [28], [29]. By adjusting the pricing factor $\sigma \in [0;1]$ according to the task arrival rate λ, the platform can ensure a balanced load, preventing both over- or under-utilization of available resources.

$$C_i(\rho) = \rho_i^2 \sigma\left[1 - \prod_{j \neq i}\left(1 - \lambda_j \rho_j\right)\right] \tag{8}$$

The payoff function for vehicle i in the computation offloading game can be formulated as follows:

$$\Pi_i(\rho) = \frac{t_{i;\max} - t_{i;l}}{t_{i;\max}}(1 - \rho_i) + \frac{t_{i;\max} - t_{i;e}}{t_{i;\max}}\rho_i - \rho_i^2 \sigma\left[1 - \prod_{j \neq i}\left(1 - \lambda_j \rho_j\right)\right] \tag{9}$$

4.2. Game Formulation

In this paper, the computation offloading game is modeled as $\Gamma = \{N; (\rho_i)_{i\in N}; (\Pi_i)_{i\in N}\}$, where $N = \{1, 2, \dots, n\}$ denotes the set of vehicles within the coverage of a MEC access point. $\rho_i \in [0, 1]$ represents the probability that vehicle i chooses to offload its computation, also known as its strategy, and Π_i is the payoff function of vehicle i. The term $\rho_{-i} = (\rho_1, \dots, \rho_{i-1}, \rho_{i+1}, \dots, \rho_n)$ refers to the offloading probabilities of all other vehicles in the system except vehicle i. The Nash equilibrium concept is then applied to characterize the equilibrium of this computation offloading game.

Definition 1. A probability vector for computation offloading, represented as ρ^* , is termed a Nash equilibrium (NE) if no vehicle can improve its payoff by unilaterally changing its strategy from the equilibrium. i.e.:

$$\Pi_i(\rho_i^*; \rho_{-i}^*) \geq \Pi_i(\rho_i; \rho_{-i}^*) \ ; \ \forall \rho_i \tag{11}$$

Theorem 1. In the game, Nash equilibrium of the computation offloading exists and is unique.

Proof
The condition established by Rosen [30], [31] for proving the concavity of the utility function is as follows:

$$\frac{\partial^2 \Pi_i}{\partial \rho_i^2} < 0$$

The first derivative of the function can be defined mathematically with the following equation:

$$\frac{\partial \Pi_i}{\partial \rho_i} = -\frac{\mathrm{t_{i;max}} - \mathrm{t_{i;l}}}{\mathrm{t_{i;max}}} + \frac{\mathrm{t_{i;max}} - \mathrm{t_{i;e}}}{\mathrm{t_{i;max}}} - 2\rho_i \sigma\left[1 - \prod_{j \neq i}\left(1 - \lambda_j \rho_j\right)\right] \tag{12}$$

In the same line, the computation offloading of utility function is:

$$\frac{\partial^2 \Pi_i}{\partial \rho_i^2} = -2\sigma\left[1 - \prod_{j \neq i}\left(1 - \lambda_j \rho_j\right)\right] < 0 \tag{13}$$

Thus, we conclude that a Nash equilibrium exists in the system.
To establish the uniqueness of the Nash equilibrium using the contraction-mapping theorem [32], [33], we demonstrate that the norm of the Jacobian matrix is less than 1, i.e., ($\|J\| < 1$).

$$J_{ij} = \frac{\partial \rho_i}{\partial \rho_j} = \begin{cases} 0\,, & i = j \\ \dfrac{-(\mathrm{t_{i;l}} - \mathrm{t_{i;e}})\prod_{k \neq i,j}(1 - \lambda_k \rho_k)}{2\sigma t_{i;max}\left[1 - \prod_{j \neq i}\left(1 - \lambda_j \rho_j\right)\right]^2}\,, & i \neq j \end{cases} \tag{14}$$

The maximum value of the off-diagonal elements of J_{ij} is found by calculating the sum of their absolute values.

$$\|J\|_\infty = \sum_{i \in \{1 \ldots N\}} \frac{-|\mathrm{t_{i;l}} - \mathrm{t_{i;e}}|\prod_{k \neq i,j}(1 - \lambda_k \rho_k)}{2\sigma t_{i;max}\left[1 - \prod_{j \neq i}\left(1 - \lambda_j \rho_j\right)\right]^2} < 1 \tag{15}$$

While $\|J\|_\infty < 1$, then equation (9) is a contraction mapping. Finally, the uniqueness of the Nash equilibrium is established.

4.3. Distributed Best-Response Algorithm for Computation Offloading Game

The Distributed Best Response algorithm 1 permits vehicles in a multi-access edge computing (MEC) environment to make decentralized offloading decisions through a trade-off between local computation and offloading to the edge server. This algorithm works iteratively, so that each vehicle can update its offload strategy according to local execution time, transmission conditions and the strategies of other vehicles.

Algorithm 1 Distributed Best-Response Algorithm for Computation Offloading Game

```
1:  Initiation: $s = 0$, the offloading probability vector $\rho_s$, the pricing vector $\sigma$, and the task arrival rate vector $\lambda$.
2:  Locally at each node $i$, iterate through $s$.
3:  Set $s \leftarrow s + 1$.
4:  **for** $i \in \{1, \ldots, n\}$ **do**
5:    Calculate the local execution time of the task without computation offloading:
          $\tau_{i;e} = \frac{\alpha_i L_i}{f_e}$
6:    Estimate the data transmission rate based on the distance between vehicle and MEC access point:
          $R_i = W_i \log_2 (1 + \frac{P_i d_i^\theta h_i^2}{N_0})$
7:    Calculate the total execution time of the task with offloading:
          $t_{i;e} = t_{i;U} + \tau_{i;e} + t_{i;D} = \frac{\beta_{i,U} L_i}{R_i} + \frac{\alpha_i L_i}{f_e} + \frac{\beta_{i,D} L_i}{R_i}$
8:    Update the best-response strategy according to offloading probabilities of other vehicles in the previous stage:
          $\rho_i^s = \left[\frac{-(\mathrm{t_{i;l}} - \mathrm{t_{i;e}})\prod_{k \neq i,j}(1 - \lambda_k \rho_k)}{2\sigma t_{i;max}\left[1 - \prod_{j \neq i}\left(1 - \lambda_j \rho_j^{-j}\right)\right]^2}\right]_0^1$
9:    **if** $\rho_i^s$ i has converged **then**
10:       Vehicle $i$ decides whether to offload its task with a probability $\rho_i^s$.
11:   **end if**
12: **end for**
13: Go to Step 3.
```

Initially, the algorithm initializes several key variables, in particular the offload probability vector (ρ^s), the pricing vector (σ) and the job arrival rate vector (λ). Each vehicle calculates the time required to process a task locally, based on the task size (L_i), the workload coefficient (α_i) and the vehicle CPU frequency ($f_{i,l}$). Then, this local execution time is compared with the time required to download the calculation to the MEC server. After calculating the local and offloading execution times, the vehicle updates its offloading probability based on the difference between the two times and the offloading strategies of other vehicles in the network. The updated offloading probability (ρ_i^s) considers the impact of other vehicles' decisions by incorporating a probability function that reflects the task arrival rates and offloading probabilities from the previous iteration.

The algorithm repeats this process, iteratively refining each vehicle's offloading probability until convergence is achieved. Convergence occurs when the offloading probability stabilizes, meaning no further significant updates are needed. At this point, each vehicle has decided whether to offload its computation to the MEC server with an optimal probability, resulting in efficient resource utilization and balanced task execution across the network.

The algorithm's structure is formally outlined below, illustrating how each vehicle independently adjusts its offloading strategy in response to network conditions and other vehicles' decisions. This distributed and iterative process ensures that the system reaches a Nash equilibrium, optimizing computation offloading across all vehicles. The algorithm 1 efficiently optimizes the computation offloading decisions, ensuring that vehicles achieve a balance between local processing and offloading, thereby enhancing system-wide performance in the MEC environment.

5. Numerical Results

5.1. Tuning Parameters

The following section aims at providing a test case demonstrating the ability of the proposed game-theoretical model to enhance resource allocation of each vehicle. In particular, it is worth noting that our analysis is confined to the computation offloading game within a single MEC platform and does not account for inter-MEC handover concerns. Additionally, we assume that each vehicle has knowledge of the offloading probabilities of other vehicles from the previous MEC stage. Based on the parameters and variables described in table 2, we have designed the algorithm 1. This algorithm is developed to obtain the results discussed in the following subsection.

Parameter	Veh-1	Veh-2	Veh-3	Veh-4	Veh-5	Veh-6	Veh-7
α_i	1.5	2.0	1.8	1.6	1.7	1.9	2.1
$\beta_{i,U}$	0.8	1.2	1.0	0.9	1.1	1.3	1.4
$\beta_{i,D}$	0.1	0.05	0.08	0.07	0.06	0.09	0.11
$f_{i,l}$	1.0	1.2	0.9	1.1	1.3	1.0	0.8
f_e	2.5	2.0	1.5	2.2	2.1	1.8	1.9
θ_i	1.8	2.5	2.0	2.3	2.1	1.9	2.2
h_{ij}	0.6	0.7	0.8	0.75	0.65	0.7	0.8
d_i	0.15	0.18	0.2	0.17	0.16	0.19	0.21
W_i	6	8	7	6.5	7.5	8.0	7.0

Table 2. Simulation Parameters for Each Vehicle

5.2. Convergence of Vehicles' Strategies towards Nash Equilibrium

The convergence of the vehicle competition towards the Nash equilibrium is clearly illustrated in Figure 2 below. Throughout 8 iterations, we observe rapid convergence, showing that the vehicles swiftly adjust their strategies to reach a stable equilibrium. This rapid convergence highlights the efficiency of the competition dynamics in achieving a balanced result within a brief period.

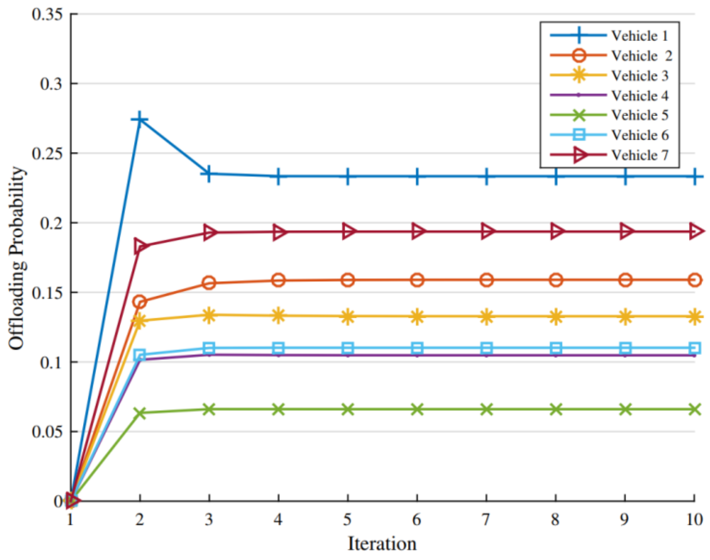

Figure 2. Offloading Probability Convergence for Each Vehicle (n = 7)

5.3. Offloading Probability as a Function of Distance: Non-Competitive vs. Competitive Behavior

As shown in figure 3. In the absence of competition among vehicles for computation offloading, each vehicle offloads to the MEC with a probability of 1 when it is near the MEC. As the vehicle moves away from the MEC, its offloading probability decreases, which is expected in isolation without competition from other vehicles.

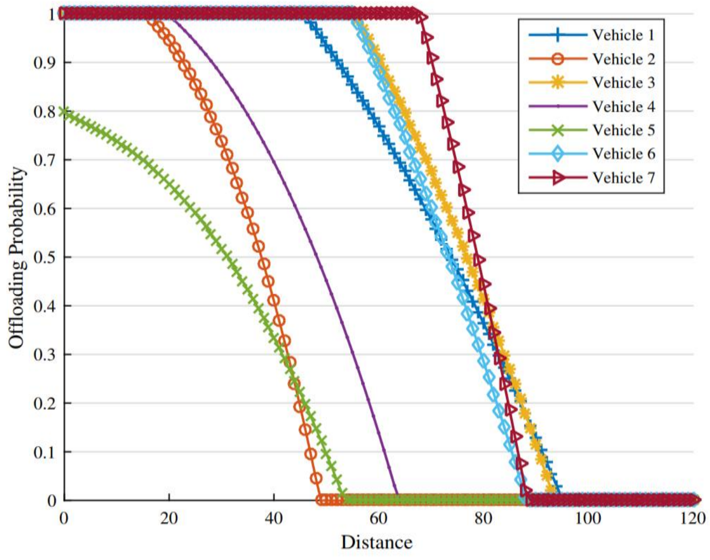

Figure 3. Offloading Probability versus Distance: No Competition

Figure 4 illustrates the competitive offloading behavior of vehicles. When they are close to the MEC, the probability of offloading is approximately the same for all vehicles. Nevertheless, as they move further away, distinct trends can be observed. The probability of offloading increases for vehicles 1, 3 and 7, while it decreases for vehicles 2, 4, 5 and 6, until it reaches zero before 75 meters. In particular, vehicle 1's offloading probability decreases to zero at 90 meters, followed by vehicle 3 at 105 meters, and finally, vehicle 7 at 120 meters. This behavioral variation underlines the strategic adjustments that each vehicle makes in response to the actions of others and their distance from the MEC.

In this way, the observed dynamics demonstrate the impact of competition on offloading strategies, with vehicles continually adapting their probabilities in order to optimize their performance within the MEC. The result confirms the idea that competitive interactions significantly influence offloading decisions, regardless of distance from the MEC.

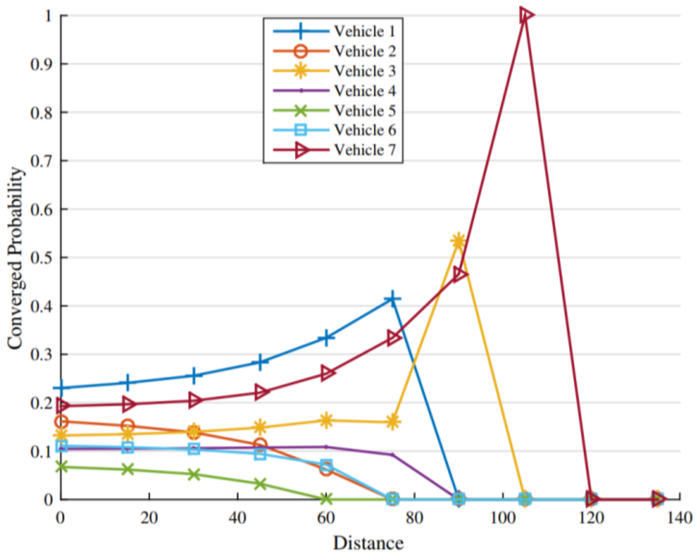

Figure 4. Offloading Probability versus Distance: With Competition

5.4. Impact of Resource Competition and Pricing on Offloading Strategies

From the results of figures 5 the observed phenomenon can be explained through resource competition dynamics in MEC networks. Initially, vehicles exhibit similar offloading probabilities, driven by utility maximization principles. As task arrival rates and pricing factor increase beyond 0.5, indicating ample resources in the MEC server, vehicles adapt their offloading strategies based on the probabilities of others. This strategic response reflects the dynamic nature of resource allocation and allows vehicles to optimize utility given changing network conditions. MEC server operators can further influence competition intensity by deploying varying server resources.

Both figures 5 and 6 highlight significant changes in the offloading probabilities when the task arrival rate and the pricing factor exceed 0.3, indicating a critical threshold. In particular, vehicles 1 and 7 increase their offloading probabilities, while the others simultaneously reduce their probabilities. The above behavior reflects the impact of competitive interactions and resource availability on strategic adjustments.

This dynamic response of vehicles to these parameters underlines the important role of strategic adaptation in optimizing performance in MEC networks. MEC server operators can influence competition by varying server resources, thus shaping vehicle strategies. Sensitivity to task arrival rate and pricing factor reinforces the need for dynamic resource management in MEC environments.

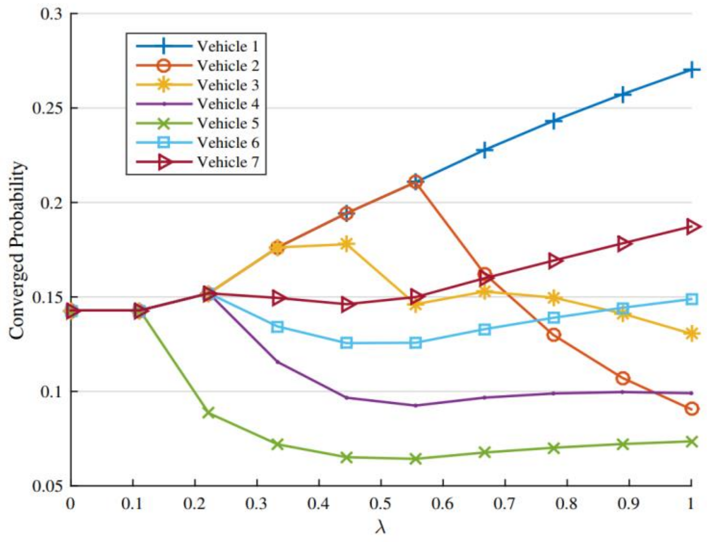

Figure 5. Offloading probability versus task arrival rate λ_i

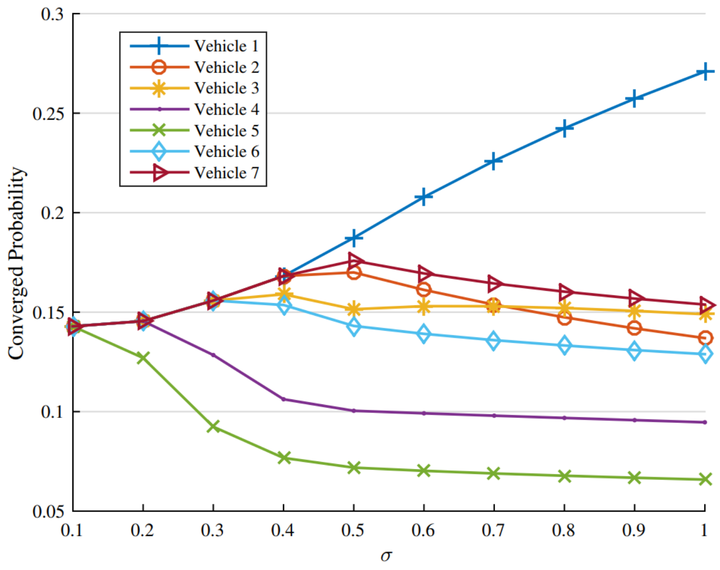

Figure 6. Offloading probability versus MEC pricing factor σ_i

5.5. Utility Analysis Across Different Offloading Probability Scenarios

In this section, we examine the implications of our results and compares the utilities obtained under different Offloading Probability scenarios. In addition, the comparison of total utilities for different numbers of vehicles is illustrated in table 3 and figure 7.

- **Utility Zero_Prob**: In the case of no offloading, most vehicles have low or negative utilities. For example, vehicle 3 has a utility of -0.1111 and vehicle 6 has a utility of -0.0555. This scenario, with a total utility of 0.0811, demonstrates the inefficiency of not using offloading.

- **Utility One_Prob**: It entirely based on offloading without any local treatment also results in poor performance, since several vehicles have negative utilities. As two examples, vehicle 2 has a utility of -0.0375 and vehicle 6 has a utility of -0.1190. Its total utility of -0.2729 is the lowest of all scenarios, and indicates that this approach is not suitable for maximizing system performance.
- **Utility Equi_Prob**: The equi-probability scenario, based on a uniform distribution of offloading among vehicles. The utility of most vehicles is positive. The total utility of 2.3044 shows that this balanced approach is more efficient than no offloading or full offloading.
- **Utility Final_Prob**: Our proposed game-theoretic approach, which iteratively optimizes offloading probabilities, achieves the highest utilities for all vehicles. This is because each vehicle has a positive utility. This brings the total utility to 2.4984, which is the highest of all the scenarios. Clearly, this proves the effectiveness of our optimization process in achieving optimal vehicle performance.

Vehicle	**Utility_Zero_Prob**	**Utility_One_Prob**	**Utility_Equi_Prob**	**Utility_Final_Prob**
1	0.166667	0.139088	0.493416	0.532627
2	0.074074	-0.037587	0.362242	0.359877
3	-0.111111	-0.097960	0.207411	0.196504
4	0.191919	0.019016	0.481546	0.499078
5	0.273504	-0.077733	0.515114	0.507702
6	-0.055556	-0.119056	0.259340	0.292444
7	-0.458333	-0.098674	-0.014628	0.110261
All-vehicles	0,081164	-0,272907	2,304442	**2,498495**

Table 3. Utility comparison for different offloading probabilities

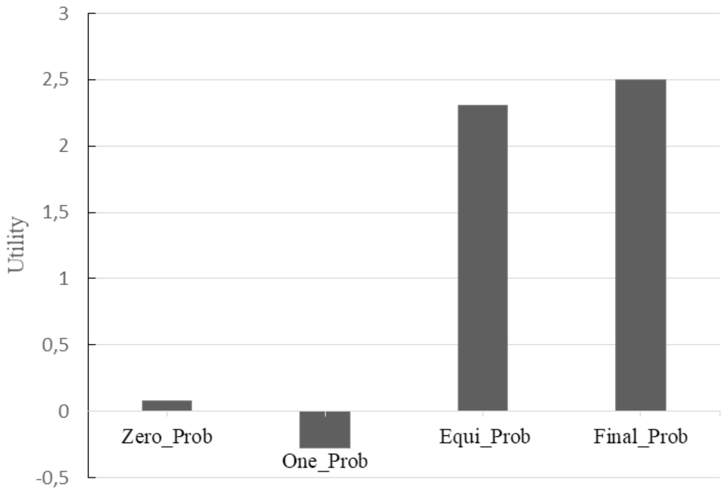

Figure 7. The total utility for different offloading probabilities ρ_i

The comparative analysis of different offloading probability scenarios underscores the importance of strategic offloading in mobile edge computing environments. The results clearly indicate that neither extreme (zero or full offloading) is optimal. Instead, a balanced and optimized approach, as achieved through our game-theoretic method, ensures the best overall performance.

The superiority of our approach is evident from the consistently higher utilities achieved across all vehicles. This highlights the role of iterative optimization in dynamically adjusting offloading probabilities to match the varying computational demands and resource availability of each vehicle. By doing so, our method not only enhances individual vehicle performance but also maximizes the collective utility of the system.

6. Conclusions

This paper proposes an innovative approach to optimize computation offloading in vehicular MEC networks. The system adapts the offloading probability based on factors such as the proximity to MEC access points and resource competition. A distributed best-response algorithm is utilized to guarantee convergence to a stable equilibrium, improving system efficiency. Experimental results demonstrate the approach's ability to achieve fast convergence and enhanced performance. Our study demonstrates that an optimized, game-theoretic approach to computation offloading is critical for maximizing utility in mobile edge computing. Furthermore, our method significantly outperforms existing dynamic offloading strategies, providing optimal utility for all participating vehicles. This underscores the potential of game theory and iterative optimization in improving the efficiency and performance of MEC systems. Future work could explore the application of this approach in more complex and heterogeneous edge computing environments to further assess its robustness and scalability.

Declarations

Funding: This research received no external funding.

Conflict of interest: The authors have no competing interests to declare that are relevant to the content of this article.

Data availability: Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Ethics approval and consent to participate: This article does not contain any studies with human **participants or animals performed by any of the authors.**

Consent for publication: Not applicable.

Materials availability: Not applicable.

Author contribution: All authors wrote the main manuscript text and reviewed and approved the manuscript.

References

[1] S. Zhou, B. Yuan, K. Xu, M. Zhang, and W. Zheng, "THE IMPACT OF PRICING SCHEMES ON CLOUD COMPUTING AND DISTRIBUTED SYSTEMS," *J. Knowl. Learn. Sci. Technol.*, vol. 3, no. 3, pp. 193–205, Sep. 2024, doi: 10.60087/jklst.v3.n3.p206-224.

[2] X. Deng *et al.*, "A review of 6G autonomous intelligent transportation systems: Mechanisms, applications and challenges," *Journal of Systems Architecture*, vol. 142, p. 102929, Sep. 2023, doi: 10.1016/j.sysarc.2023.102929.

[3] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, "Vehicular Edge Computing and Networking: A Survey," Jul. 25, 2019.

[4] A. Biswas and H.-C. Wang, "Autonomous Vehicles Enabled by the Integration of IoT, Edge Intelligence, 5G, and Blockchain," *Sensors*, vol. 23, no. 4, p. 1963, Feb. 2023, doi: 10.3390/s23041963.

[5] M. Díaz, C. Martín, and B. Rubio, "State-of-the-art, challenges, and open issues in the integration of Internet of things and cloud computing," *Journal of Network and Computer Applications*, vol. 67, pp. 99–117, May 2016, doi: 10.1016/j.jnca.2016.01.010.

[6] A. Ait Temghart, M. Marwan, and M. Baslam, "Optimization of computational offloading in the mobile edge: a game theoretic approach," *Cluster Comput*, vol. 28, no. 3, p. 203, Jun. 2025, doi: 10.1007/s10586-024-04886-6.

[7] Y. Liu *et al.*, "Dependency-Aware Task Scheduling in Vehicular Edge Computing," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4961–4971, Jun. 2020, doi: 10.1109/JIOT.2020.2972041.

[8] M. Marwan, A. A. Temghart, F. Sifou, and F. AlShahwan, “A Decentralized Blockchain-based Architecture for a Secure Cloud-Enabled IoT,” *JMM*, Nov. 2020, doi: 10.13052/jmm1550-4646.1636.

[9] A. A. Temghart, D. A. Omar, M. Baslam, and M. Marwan, “Game theoretical approach for a fair and effective pricing strategy in cloud computing,” *International Journal of High Performance Systems Architecture*, vol. 10, no. 1, p. 43, 2021, doi: 10.1504/IJHPSA.2021.115512.

[10] M. Marwan, A. Ait Temghart, S. Ouhmi, and M. Lazaar, “Security, QoS and energy aware optimization of cloud-edge data centers using game theory and homomorphic encryption: Modeling and formal verification,” *Results in Engineering*, vol. 24, p. 102902, Dec. 2024, doi: 10.1016/j.rineng.2024.102902.

[11] A. Ait Temghart, M. Marwan, and M. Baslam, “Stackelberg Security Game for Optimizing Cybersecurity Decisions in Cloud Computing,” *Security and Communication Networks*, vol. 2023, pp. 1–13, Dec. 2023, doi: 10.1155/2023/2811038.

[12] M. Marwan, F. AlShahwan, Y. Afoudi, A. A. Temghart, and M. Lazaar, “Leveraging artificial intelligence and mutual authentication to optimize content caching in edge data centers,” *Journal of King Saud University-Computer and Information Sciences*, vol. 35, no. 9, p. 101742, 2023.

[13] O. Muñoz, A. Pascual-Iserte, and J. Vidal, “Optimization of Radio and Computational Resources for Energy Efficiency in Latency-Constrained Application Offloading,” *IEEE Transactions on Vehicular Technology*, vol. 64, no. 10, pp. 4738–4755, Oct. 2015, doi: 10.1109/TVT.2014.2372852.

[14] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, “Offloading in Mobile Edge Computing: Task Allocation and Computational Frequency Scaling,” *IEEE Transactions on Communications*, vol. 65, no. 8, pp. 3571–3584, Aug. 2017, doi: 10.1109/TCOMM.2017.2699660.

[15] J. Zheng, Y. Cai, and Y. Wu, “Dynamic Computation Offloading for Mobile Cloud Computing: A Stochastic Game-Theoretic Approach”.

[16] F. Li, E. Ge, W. Hu, and R. Xia, “A two-level game theoretic approach for task offloading in mobile edge computing,” *Engineering Applications of Artificial Intelligence*, vol. 136, p. 108819, Oct. 2024, doi: 10.1016/j.engappai.2024.108819.

[17] X. Chen, “Decentralized Computation Offloading Game for Mobile Cloud Computing,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 4, pp. 974–983, Apr. 2015, doi: 10.1109/TPDS.2014.2316834.

[18] X. Lyu, H. Tian, C. Sengul, and P. Zhang, “Multiuser Joint Task Offloading and Resource Optimization in Proximate Clouds,” *IEEE Transactions on Vehicular Technology*, vol. 66, no. 4, pp. 3435–3447, Apr. 2017, doi: 10.1109/TVT.2016.2593486.

[19] S. Bi and Y.-J. A. Zhang, “Computation Rate Maximization for Wireless Powered Mobile-Edge Computing with Binary Computation Offloading,” Mar. 22, 2018, *arXiv*: arXiv:1708.08810. Accessed: Feb. 17, 2024. [Online]. Available: http://arxiv.org/abs/1708.08810

[20] Z. Tong, X. Deng, F. Ye, S. Basodi, X. Xiao, and Y. Pan, “Adaptive computation offloading and resource allocation strategy in a mobile edge computing environment,” *Information Sciences*, vol. 537, pp. 116–131, Oct. 2020, doi: 10.1016/j.ins.2020.05.057.

[21] Z. Tong, X. Deng, J. Mei, L. Dai, K. Li, and K. Li, “Stackelberg game-based task offloading and pricing with computing capacity constraint in mobile edge computing,” *Journal of Systems Architecture*, vol. 137, p. 102847, Apr. 2023, doi: 10.1016/j.sysarc.2023.102847.

[22] V. Cardellini *et al.*, “A game-theoretic approach to computation offloading in mobile cloud computing,” *Math. Program.*, vol. 157, no. 2, pp. 421–449, Jun. 2016.

[23] P. Liu, J. Li, and Z. Sun, “Matching-Based Task Offloading for Vehicular Edge Computing,” *IEEE Access*, vol. 7, pp. 27628–27640, 2019, doi: 10.1109/ACCESS.2019.2896000.

[24] Y. Zhang *et al.*, “An Efficient Caching and Offloading Resource Allocation Strategy in Vehicular Social Networks,” *IEEE Transactions on Vehicular Technology*, vol. 73, no. 4, pp. 5690–5703, Apr. 2024, doi: 10.1109/TVT.2023.3332905.

[25] H. Wu, J. Geng, X. Bai, and S. Jin, “Deep reinforcement learning-based online task offloading in mobile edge computing networks,” *Information Sciences*, vol. 654, p. 119849, Jan. 2024, doi: 10.1016/j.ins.2023.119849.

[26] T. Zhang, D. Xu, A. Tolba, K. Yu, H. Song, and S. Yu, “Reinforcement-Learning-Based Offloading for RIS-Aided Cloud–Edge Computing in IoT Networks: Modeling, Analysis, and Optimization,” *IEEE Internet Things J.*, vol. 11, no. 11, pp. 19421–19439, Jun. 2024, doi: 10.1109/JIOT.2024.3367791.

[27] F. Dai, G. Liu, Q. Mo, W. Xu, and B. Huang, “Task offloading for vehicular edge computing with edge-cloud cooperation,” *World Wide Web*, vol. 25, no. 5, pp. 1999–2017, Sep. 2022, doi: 10.1007/s11280-022-01011-8.

[28] A. Ait Temghart, M. Marwan, M. Ouaskou, and M. Baslam, “Towards Efficient Computation Offloading in Multi-user MEC Environments: A Game-Theoretic Approach,” in *Digital Technologies and Applications*, vol. 1101, S. Motahhir and B. Bossoufi, Eds., in Lecture Notes in Networks and Systems, vol. 1101. , Cham: Springer Nature Switzerland, 2024, pp. 133–143. doi: 10.1007/978-3-031-68675-7_14.

[29] N. Chauhan and R. Agrawal, “A Probabilistic Deadline-aware Application Offloading in a Multi-Queueing Fog System: A Max Entropy Framework,” *J Grid Computing*, vol. 22, no. 1, p. 31, Mar. 2024, doi: 10.1007/s10723-024-09753-7.

[30] J. B. Rosen, “Existence and Uniqueness of Equilibrium Points for Concave N-Person Games,” *Econometrica*, vol. 33, no. 3, p. 520, Jul. 1965, doi: 10.2307/1911749.

[31] D. Gabay and H. Moulin, “On the Uniqueness and Stabil - ity of Nash Equilibria in Noncooperative Games,” *Applied Stochastic Control in Econometrics and Management Sciences*, Jan. 1980.

[32] G. P. Cachon and S. Netessine, “Game Theory in Supply Chain Analysis,” in *Models, Methods, and Applications for Innovative Decision Making*, M. P. Johnson, B. Norman, N. Secomandi, P. Gray, and H. J. Greenberg, Eds., INFORMS, 2006, pp. 200–233. doi: 10.1287/educ.1063.0023.

[33] R. Abraham, J. E. Marsden, and T. Ratiu, *Manifolds, Tensor Analysis, and Applications*, vol. 75. in Applied Mathematical Sciences, vol. 75. New York, NY: Springer New York, 1988. doi: 10.1007/978-1-4612-1029-0.