future internet

MDPI

Article

TRUST-ME: Trust-Based Resource Allocation and Server Selection in Multi-Access Edge Computing

Sean Tsikteris, Aisha B Rahman, Md. Sadman Siraj and Eirini Eleni Tsiropoulou *

Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87131-0001, USA; stsikteris1@unm.edu (S.T.); arahman3@unm.edu (A.B.R.); mdsadmansiraj96@unm.edu (M.S.S.)
* Correspondence: eirini@unm.edu

Abstract: Multi-access edge computing (MEC) has attracted the interest of the research and industrial community to support Internet of things (IoT) applications by enabling efficient data processing and minimizing latency. This paper presents significant contributions toward optimizing the resource allocation and enhancing the decision-making process in edge computing environments. Specifically, the TRUST-ME model is introduced, which consists of multiple edge servers and IoT devices, i.e., users, with varied computing tasks offloaded to the MEC servers. A utility function was designed to quantify the benefits in terms of latency and cost for the IoT device while utilizing the MEC servers' computing capacities. The core innovation of our work is a novel trust model that was designed to evaluate the IoT devices' confidence in MEC servers. This model integrates both direct and indirect trust and reflects the trustworthiness of the servers based on the direct interactions and social feedback from other devices using the same servers. This dual trust approach helps with accurately gauging the reliability of MEC services and ensuring more informed decision making. A reinforcement learning framework based on the optimistic Q-learning with an upper confidence bounds action selection algorithm enables the IoT devices to autonomously select a MEC server to process their computing tasks. Also, a multilateral bargaining model is proposed for fair resource allocation of the MEC servers' computing resources to the users while accounting for their computing demands. Numerical simulations demonstrated the operational effectiveness, convergence, and scalability of the TRUST-ME model, which was validated through real-world scenarios and comprehensive comparative evaluations against existing approaches.

Keywords: multi-access edge computing; Internet of things; reinforcement learning; game theory

check for updates

Citation: Tsikteris, S.; Rahman, A.B.; Siraj, M.S.; Tsiropoulou, E.E. TRUST-ME: Trust-Based Resource Allocation and Server Selection in Multi-Access Edge Computing. *Future Internet* **2024**, *16*, 278. https://doi.org/10.3390/fi16080278

Academic Editor: Franco Davoli

Received: 27 June 2024
Revised: 27 July 2024
Accepted: 30 July 2024
Published: 4 August 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The rise of new technologies, such as artificial intelligence, real-time video analytics, and virtual reality, has increased the expectations for the computing power, energy efficiency, and responsiveness of mobile devices within the Internet of things (IoT). Due to constraints in storage, computing capabilities, and battery life, IoT devices face significant challenges in locally handling demanding tasks. Multi-access edge computing (MEC) has emerged as a promising solution, enabling users to shift their computationally intensive tasks away from mobile devices to MEC servers by exploiting wireless connections [1].

However, the proliferation of IoT devices, including smartphones, tablets, and smart vehicles, which are driven by advancements in smart cities and intelligent transportation systems, has dramatically increased the volume of computation-intensive applications [2]. Thus, there has been a corresponding surge in workload and network traffic within MEC servers. In this paper, we propose the TRUST-ME model, which is a novel framework that enables users to autonomously choose an optimal MEC server for processing their computational tasks by considering their trust levels of the servers' quality of service (QoS),

communication capabilities, and computing characteristics. Additionally, the TRUST-ME model facilitates the MEC servers in efficiently allocating their computing resources to meet the users' latency constraints in a fair and computationally effective manner.

1.1. Related Work and Motivation

Several recent research works focused on the problem of *resource allocation and/or server selection* in MEC environments. A non-orthogonal multiple access or NOMA-based vehicle edge computing network model was designed in [3] and consists of several heuristic algorithms to jointly optimize the task offloading and MEC resource assignment to minimize the overall system's costs while ensuring delay tolerance for vehicular user equipment. An efficient D2D-assisted MEC computation offloading framework based on attention communication deep reinforcement learning (ACDRL) is introduced in [4] to enhance a system's performance by facilitating resource sharing and reducing the latency. An efficient algorithm for optimizing the energy consumption in an NOMA-enabled MEC system with strict latency requirements is designed in [5] to address both the resource allocation and the subchannel assignment through decomposition and optimization methods. A similar approach is followed in [6] by focusing on the energy-efficient and low-latency task execution for IoT devices through the joint sparse code multiple access (SCMA) codebook assignment, power allocation, CPU frequency scheduling, and task offloading policy optimization.

Optimization techniques and heuristics algorithms were designed to address the problem of resource allocation and/or server selection in MEC environments [7]. The joint task offloading and resource allocation problem in cooperative MEC server scenarios was studied in [8] by targeting the efficiency and fairness of the resource allocation process through a two-level algorithmic approach. A computationally efficient repeated auction model is proposed in [9] by utilizing a modified generalized second price algorithm to manage the resource allocation in the competitive MEC offloading service market while considering varying server workloads. Given the plurality of IoT devices in several applications supported by unmanned aerial vehicles (UAVs), an innovative alternative optimization approach for integrating UAV systems as both remote base stations and mobile edge servers using NOMA is proposed in [10] to minimize the latency and optimize the resource allocation. An MEC framework for IoT applications in air–ground-integrated wireless networks was analyzed in [11] with the aim to minimize the service latency while meeting the learning accuracy requirements and reducing the energy consumption through an optimized deep neural network (DNN) model decision, computation, communication resource allocation, and UAV trajectory control. A collaborative multi-UAV-assisted MEC system integrated with a terrestrial base station is proposed in [12] in order to minimize the total latency for IoT devices by optimizing the offloading decisions and resource allocations under the IoT devices' energy constraints. An MEC framework that jointly optimizes the caching, offloading, deployment, and resource allocation was analyzed in [13] to minimize the latency and energy costs.

Game theory has been extensively adopted in the recent literature to deal with the distributed optimization problem of resource allocation in MEC environments [14]. A task-offloading scheme for MEC environments is proposed in [15] to optimize the users' utility by leveraging potential games. A scheme based on the double deep Q-network (DDQN) to optimize service caching, resource allocation, and computation offloading in a MEC system is analyzed in [16] to reduce the delay and energy consumption costs. Focusing on the MEC system's effective resource utilization, the authors in [17] introduce a joint bandwidth and computing frequency allocation mechanism based on the Stackelberg games. A two-level bargaining-based incentive mechanism for task offloading and collaborative computing in MEC-enabled networks was analyzed in [18] to ensure the optimal resource allocation and load balance. An iterative algorithm that minimizes the latency in digital twin-aided MEC systems for industrial IoT was formulated in [19] to optimize the resource allocation through an alternating optimization approach combined with an inner convex approximation. A game-theoretic approach to optimize multiuser computing task offloading in a device-

enhanced MEC environment is presented in [20] with the aim to maximizing the aggregate offloading benefits by jointly optimizing the heterogeneous (i.e., communication and computing) resource allocation. A similar approach was followed in [21] based on a multistage Stackelberg game, where in the first stage, a contract-based incentive mechanism was introduced to motivate the IoT nodes to share idle resources, and in the second stage, the optimal pricing strategies are derived to optimize the IoT nodes' utilities.

Moreover, an energy-efficient game-theoretic approach to support the resource allocation for the IoT nodes in a MEC environment is proposed in [22] with the aim to optimize the IoT nodes' task offloading decisions based on their personal latency and energy consumption constraints. A similar decentralized game-theoretic approach was followed in [23] with the aim to optimally allocate mobile devices to nearby MEC servers to minimize the latency and communication interference of the IoT devices. The latter approach was further extended in [24] to address the interference-aware software-as-a-service user allocation problem in MEC environments by demonstrating the effectiveness and efficiency of a decentralized algorithm to find its Nash equilibrium. A decentralized-edge-node-grouping algorithm in MEC networks was investigated in [25] by addressing the system uncertainties and proving the convergence to a unique Nash equilibrium with high performance. A fair and efficient mechanism for multi-resource allocation in MEC environments was studied in [26] and guarantees a Nash equilibrium in polynomial time and satisfies various fairness criteria.

Given the distributed nature of the decision-making process regarding the resource allocation in MEC environments, the principles of *reinforcement learning (RL)* have been widely adopted in the recent literature [27]. A deep-reinforcement-learning-based approach to optimize computation offloading and resource allocation in NOMA-MEC systems is proposed in [28] in order to significantly reduce the computational overhead. The authors in [29] integrated artificial intelligence into UAV data offloading in a multi-server MEC environment using game theory and reinforcement learning principles to enhance the efficiency and effectiveness. Various optimization algorithms for managing resources in a MEC environment were studied in [30], and the authors demonstrated their similar performances and highlighted the adaptability of deep-RL-based approaches to changing conditions. A software-defined network or SDN-based MEC system is introduced in [31] using a stochastic game-based resource allocation algorithm to effectively reduce energy consumption and processing latency in IoT systems. A novel AI-based mechanism using RL and Bayesian reasoning for autonomous MEC server activation was analyzed in [32] to enhance the system's reputation and manage computing costs while ensuring the users' quality of experience. An RL-based strategy for efficient task offloading and resource allocation in multi-edge server collaboration is presented in [33] to jointly optimize the task latency, the user service numbers, and the energy consumption. An RL-based approach and an enhanced deferred acceptance algorithm are discussed in [34] to jointly optimize the virtual network functions' (VNFs') placement, routing, and scheduling, and significantly reduce the service latency and resource consumption in MEC environments. A strategy for joint task type and vehicle speed-aware task offloading and resource allocation in vehicular MEC environments is proposed in [35] to optimize the energy costs and processing revenue while meeting the task delay constraints. A Q-learning-based reinforcement learning solution for optimizing dynamic resource allocation in 5G radio access network slicing supporting MEC systems is presented in [36] by significantly improving the network utility compared with myopic, random, and first-come-first-served approaches. A distributed framework using autonomous MEC servers and stochastic learning automata in IoT networks to optimize server activation, device–server associations, and transmission power for QoS satisfaction is presented in [37].

Additionally, a Markov-decision-process-based task offloading strategy in a vehicular edge-cloud computing network is introduced in [38] based on a dueling actor–critic reinforcement learning approach to effectively manage the resource allocation and minimize the task completion times. Similarly, a shared offloading strategy using deep reinforcement

learning (DRL) for vehicular edge computing (VEC) is discussed in [39] to reduce the delay and energy consumption by allowing vehicles to share computing results and adaptively select optimal offloading modes in complex IoT environments. Also, a hierarchical vehicular edge computing architecture is designed in [40] to enhance the global task computing efficiency by allowing neighboring regions to share idle resources. The authors of [41] analyzed and compared value-based, policy-based, and hybrid deep reinforcement learning algorithms in order to address the task-offloading challenges in edge computing environments, and they provided a thorough discussion regarding their respective strengths, weaknesses, and future research directions. A dynamic offloading decision algorithm combined with DRL was analyzed in [42] to optimize the task offloading and resource allocation in an MEC environment, and it significantly improved the energy efficiency and task completion time under varying wireless conditions. A novel offloading algorithm that combines meta-reinforcement learning with DRL is proposed in [43] to efficiently generate near-optimal task-offloading decisions in resource-constrained IoT environments. A DRL-based computation offloading scheme for virtual reality video transmission in MEC networks was analyzed in [44] in order to optimize the task management through a deep deterministic policy gradient algorithm. Similarly, a DRL-based computation-offloading scheme for UAV-enabled MEC environments is proposed in [45] with the aim to reduce the task execution delay, energy consumption, and overall offloading cost compared with conventional methods. Moreover, an autonomous partial offloading system for delay-sensitive tasks in multiuser IoT MEC environments is introduced in [46] and is based on Q-learning and deep deterministic policy gradient strategies to significantly reduce the devices' delay and enhance their quality of service. A meta-RL task-offloading algorithm was designed in [47] based on graph neural networks and enhances the adaptability and robustness in dynamic MEC environments by effectively managing the task dependencies and optimizing the sampling efficiency.

1.2. Contributions

Though significant research efforts have been devoted in the recent literature to deal with the joint problem of MEC server selection and resource allocation in MEC environments consisting of multiple IoT devices, the study of these problems remains highly fragmented, meaning that the recent research works either focus on the problem of MEC server selection or on the problem of resource allocation in a separate manner. Additionally, assessing the trustworthiness of the MEC servers' computing capabilities in order to meet the IoT devices' latency constraints has been largely overlooked. To address these research gaps, we propose the TRUST-ME model, which enables the autonomous selection of edge servers by the IoT devices for task offloading while considering the MEC servers' trustworthiness in delivering quality of service to the IoT devices. Furthermore, we introduce a distributed resource allocation framework based on the multilateral bargaining games to facilitate the fair allocation of the MEC servers' computing resources to the IoT devices based on the computing demands of the latter ones.

The main contributions of this research work are summarized below:

1. Initially, the TRUST-ME model is presented as consisting of multiple edge servers and multiple IoT devices, where the latter ones are characterized by different computing tasks to be fully offloaded to the MEC servers for further processing. The communications and computing characteristics of the IoT devices are presented and a utility function is designed to capture the IoT devices' benefit by the experienced latency and cost from utilizing the computing capacity of the selected MEC server.
2. A novel trust model of the IoT devices to the MEC servers' computing capabilities was designed and consisted of the direct and indirect trust of the devices, where the latter one was derived from the social ties between the IoT devices that have used the same MEC server to process their computing tasks. A reinforcement learning approach based on optimistic Q-learning with an upper bound confidence action

selection algorithm is presented to enable the IoT devices to autonomously select an MEC server.

3. A multilateral bargaining model is presented for resource allocation, enabling the MEC servers to allocate their computing capacity to the IoT devices' tasks by taking into account their computing demand and the fairness in service provision between the devices.
4. A detailed set of simulation-based experiments was performed to demonstrate the operational efficacy and performance convergence of the TRUST-ME model in terms of the MEC server selection and resource allocation. Moreover, a real-world scenario was analyzed by considering different types of computing applications requested by the IoT devices to demonstrate the TRUST-ME model's applicability. A thorough scalability analysis also quantified its efficiency and robustness. A detailed comparative evaluation against alternative MEC server selection and resource allocation approaches quantified the superiority of the TRUST-ME model over current state-of-the-art methods.

1.3. Outline

The remainder of this paper is organized as follows. Section 2 presents the TRUST-ME model, and Section 3 analyzes the IoT devices' trust model and the autonomous RL-based MEC server selection. Section 4 provides the analysis for the multilateral bargaining resource allocation of the MEC servers' computing capacity, and Section 5 provides a detailed simulation-based evaluation of the TRUST-ME model. Finally, Section 6 concludes the paper.

2. TRUST-ME System Model

We consider a multi-access edge computing (MEC) environment with a group of users $\mathcal{N} = \{1, \ldots, n, \ldots, N\}$ and a set of MEC servers $\mathcal{S} = \{1, \ldots, s, \ldots, S\}$. Each user has a computing task represented by $A_n = (B_n, C_n, \phi_n, T_n, e_n)$, where B_n [bits] is the total data size, and C_n [CPU-cycles] is the required processing cycles, which is defined as $C_n = \phi_n B_n$, with ϕ_n $[\frac{CPU-cycles}{bits}]$ denoting the computation intensity. Additionally, T_n [sec] and e_n [J] represent the user's latency and energy constraints, respectively. The users fully offload their data B_n to MEC server s. Each MEC server has a computing capacity F_s $[\frac{CPU-cycles}{sec}]$ and allocates its capacity among the users offloading to it. The allocation vector is denoted by $\mathbf{f_s} = [f_{s,1}, \ldots, f_{s,n}, \ldots, f_{s,N}]$ $[\frac{CPU-cycles}{sec}]$, satisfying $\sum_{\forall n \in \mathcal{N}} f_{s,n} = F_s$.

User n offloads its data to MEC server s using the non-orthogonal multiple access (NOMA) technique, with the successive interference cancellation (SIC) implemented at the MEC server (receiver). Assuming the users' channel gains are sorted as $G_1 \leq \cdots \leq G_n \leq \cdots \leq G_{N_s}$, where N_s is the number of users offloading their computing tasks to server s, the throughput for each user n is given by the following equation [48]:

$$R_{n,s} = W \log_2 (1 + \frac{P_n G_n}{I_0 + \sum_{n'=1}^{n-1} G_{n'} P_{n'}}) \quad (1)$$

where W [Hz] denotes the bandwidth, P_n [watts] denotes the transmission power, and G_n denotes the channel gain. It is noted that for simplicity in the notation, we considered that the users N_s associated with MEC server s have a corresponding channel gain and we denote the channel gain of user n as G_n. The corresponding set of users selecting server s is denoted as $\mathcal{N}_s\{a, \ldots, n, \ldots, N_s\}$. Also, I_0 denotes the power spectral density of zero-mean additive white Gaussian noise (AWGN).

User n's time overhead consists of the overhead to transmit and process its data and it is given as follows:

$$t_n = \frac{B_n}{R_{n,s}} + \frac{C_n}{f_{s,n}} \tag{2}$$

Based on the experienced latency and the cost to utilize the computing resources of the MEC server, each user experiences a utility by processing its computing task using the MEC server, which is given as follows:

$$u_n = \frac{\frac{1}{t_n}}{\frac{1}{\max\limits_{n \in \mathcal{N}}\{t_n\}}} - \beta_n \cdot \frac{\mathcal{C}_s f_{s,n}}{\mathcal{C}_s \max\limits_{s \in \mathcal{S}}\{F_s\}} \tag{3}$$

where t_n [sec] is the user's experienced latency to transmit and process its computing task, as given by Equation (2); $\mathcal{C}_s$ $[\frac{\$}{CPU-cycles}]$ is the MEC server's announced computing price to process the users' data; $f_{s,n}$ $[\frac{CPU-cycles}{sec}]$ is the amount of computing resources allocated by MEC server s to the user n's computing task; and β_n is a weight factor. It is noted that the user's utility u_n is unitless and the normalization is performed for demonstration purposes in order to capture the pure differences between the users.

3. Trust-Based Reinforcement-Learning-Enabled MEC Server Selection

The users evaluate the reliability (i.e., trust levels) of the MEC servers based on the quality of service that they provide for processing the offloaded computing tasks in order to ultimately identify the most dependable MEC server. The trust of the users to the MEC servers is assessed by considering both the direct trust $dtr_{s,n}$ and the indirect trust $itr_{s,n}$. The direct trust stems from the users' previous interactions with the MEC servers, while the indirect trust, or reputation and reliability, is built on the collective experiences of other users. This dual-trust approach improves the users' understanding of the service quality offered by the MEC servers, especially in the case where the direct interactions of the users with the MEC servers are infrequent.

3.1. Influencers' Direct Trust

The level of direct trust a user places in an MEC server depends on whether the MEC server meets the user's minimum utility threshold, which is symbolized as U_n, and its latency constraint T_n [sec]. This is measured through the ratings $\tilde{r1}_{s,n} \in [0,1]$ and $\tilde{r2}_{s,n} \in [0,1]$, respectively, which the user assigns to the MEC server based on the computing capacity that the latter one offers. The user's ratings are evaluated under two distinct conditions:

Case 1: If the user's requirements are satisfied, meaning $U_n \leq u_n$ and $T_n \geq t_n$, the MEC server receives a perfect satisfaction rating of 1 for the computing capacity service. Thus, the final rating given to the MEC server is $\tilde{r}_{s,n} = \hat{r}_{s,n}$.
Case 2: If the computing capacity service provided by the MEC server does not meet the user's utility and latency constraints, the user's satisfaction rating will reflect the gap between the actual utility received and the minimum required utility, and the latency constraint gap, respectively.

The ratings provided under the two cases considering the experienced utility and latency are given in Equation (4) and Equation (5), respectively:

$$\tilde{r1}_{s,n} = \begin{cases} \hat{r}_{s,n} & \text{, if } U_n \leq u_n \\ \frac{u_n - U_n}{U_n} & \text{, if } U_n > u_n \end{cases} \tag{4}$$

$$\tilde{r}_{2s,n} = \begin{cases} \hat{r}_{s,n} & \text{, if } T_n \geq t_n \\ \frac{T_n}{t_n} - 1 & \text{, if } T_n < t_n \end{cases} \tag{5}$$

By combining the two ratings in an overall rating $\tilde{r}_{s,n}$ provided by the user to the MEC server, the corresponding overall rating is derived as follows by considering that the user considers the satisfaction of its utility and latency constraints to be equally important:

$$\tilde{r}_{s,n} = 0.5 \cdot \tilde{r}_{1s,n} + 0.5 \cdot \tilde{r}_{2s,n} \tag{6}$$

In Case 1, the ratings $\tilde{r}_{1s,n}$ and $\tilde{r}_{2s,n}$ are non-negative and are both set to 1. In Case 2, the ratings for both variables are negative and within the range of $[-1, 0)$. To address this, we employed the min–max normalization technique, which is a common method for data normalization, to transform the piecewise rating function from $[-1, 1]$ to $[0, 1]$, as demonstrated below:

$$r_{s,n} = \frac{\tilde{r}_{s,n} - \min_{\forall i \in \mathcal{N}_{n,s}}\{\tilde{r}_{s,n}\}}{\max_{\forall i \in \mathcal{N}_{n,s}}\{\tilde{r}_{s,n}\} - \min_{\forall i \in \mathcal{N}_{n,s}}\{\tilde{r}_{s,n}\}} = \frac{\tilde{r}_{s,n} + 1}{2} \tag{7}$$

where $\mathcal{N}_{n,s}$ refers to the set of computing service times provided to user n by MEC server s.

The expression $f(t_i)$ captures the time decay factor, which reflects how the user's earlier ratings lose their relevance over time in terms of impacting the user's trust in an MEC server, and is defined as follows:

$$f(t_i) = \exp(-\lambda(t - t_i)) \tag{8}$$

where λ, t, and t_i are the rate of decay, current service time, and historical service time, respectively. The decay rate λ is used in the time decay function to reduce the impact of older feedback on the current trust value. If λ is high, then the older feedback will decay faster, and thus, the recent interactions will significantly influence the trust metric. This can be beneficial in dynamic environments where recent performance is more indicative of future behavior. On the other hand, if λ is low, then the feedback decays slowly, and thus, more weight is given to historical data. This can stabilize the trust metric in environments where performance is consistent over time.

Based on the above analysis, user n's direct trust $dtr_{s,n}$ of MEC server s is defined as follows:

$$dtr_{s,n} = \frac{\sum\limits_{i \in \mathcal{N}_{n,s}} r_{s,n} \cdot f(t_i)}{\sum\limits_{i \in \mathcal{N}_{n,s}} f(t_i)} \tag{9}$$

where the denominator $\sum\limits_{i \in \mathcal{N}_{n,s}} f(t_i)$ incorporates the exponential decay function for normalization purposes.

3.2. Influencers' Indirect Trust

In an edge computing environment, the users establish a variety of social connections, which are represented as $\mathcal{C} = \{1, \ldots, c, \ldots, C\}$ and include friendships, familial bonds, and acquaintances with strangers. The strength of the social connection c between any two users is indicated by η_c, where $\eta_c > 0$ for all $c \in \mathcal{C}$. The relationship vector between user n and user n' is denoted by $\mathbf{\Lambda}_{n,n'} = \{\Lambda_{n,n',1}, \ldots, \Lambda_{n,n',c}, \ldots, \Lambda_{n,n',C}\}$. If $\Lambda_{n,n',c} = 1$, it means that the users n and n' have a relationship of type c; otherwise, $\Lambda_{n,n',c} = 0$.

For a user n, the trustworthiness of MEC server s is influenced by recommendations from other users with whom the user has several social connections. Generally, the users who are reliable provide truthful recommendations, while the users with malicious intent provide deceptive feedback. The reliability factor $rl_{n,n'}$ serves as a weight for these recommendations provided by other users n' to user n, which reflects how trustworthy the other

users' feedback is. This factor is based on both the degree of their social relationship and the similarity between user n and user n', which is calculated as follows:

$$rl_{n,n'} = \frac{\eta_c \cdot \Lambda_{n,n',c} \cdot SIM_{n,n'}}{\max\limits_{c \in \mathcal{C}} \eta_c} \tag{10}$$

Let $\text{SIM}_{n,n'}$ denote the similarity score between the users n and n'. To standardize these scores, we used the normalization factor $\max\limits_{c \in \mathcal{C}} \eta_c$. In general, the users with more similar ratings will have a higher similarity score, which is calculated using the Pearson correlation coefficient. This coefficient measures the degree of similarity between the users n and n' by analyzing their ratings on the MEC servers that they both have been served from. The similarity score is given by

$$SIM_{n,n'} = \frac{\sum\limits_{s \in \mathcal{S}_{n,n'}} (r_{s,n} - \bar{r}_n)(r_{s,n'} - \bar{r}_{n'})}{\sqrt{\sum\limits_{s \in \mathcal{S}_{n,n'}} (r_{s,n} - \bar{r}_n)^2} \sqrt{\sum\limits_{s \in \mathcal{S}_{n,n'}} (r_{s,n'} - \bar{r}_{n'})^2}} \tag{11}$$

where $\bar{r}_n$ and $\bar{r}_{n'}$ represent the average ratings given by user n and user n', respectively. The set $\mathcal{S}_{n,n'}$ includes all the MEC servers that have provided computing services to both users n and n'.

From this analysis, the indirect trust $itr_{s,n}$ can be expressed as follows:

$$itr_{s,n} = \frac{\sum\limits_{n,n' \in \mathcal{N}_s, n \neq n'} rl_{n,n'} \cdot dtr_{s,n'}}{\sum\limits_{n,n' \in \mathcal{N}_s, n \neq n'} rl_{n,n'}} \tag{12}$$

where $\mathcal{N}_s$ denotes the set of users being served by MEC server s.

Different types of social connections, such as friendships and familial bonds, can significantly impact the strength of the social connection η_c of the indirect trust in an MEC environment. Friendships often imply a higher degree of trust and regular interaction. Recommendations from friends are typically considered reliable because friends are likely to share similar standards and expectations regarding service quality. Friends often interact frequently, which means they have a richer history of shared experiences with MEC servers. This historical data enhances the accuracy of the trust metric, as the feedback is based on multiple interactions over time. Familial bonds are usually the strongest social connections. Recommendations from family members are highly reliable because the family members generally prioritize each other's best interests and provide honest feedback. Acquaintances have a weaker social connection compared with friendships and familial bonds. Recommendations from acquaintances are considered less reliable, but still valuable, as they provide additional perspectives. Finally, recommendations from strangers carry the least weight due to the lack of a personal connection and historical interaction.

3.3. Influencers' Overall Trust

The users combine their direct trust (Equation (9)) and their indirect trust (Equation (12)) in order to form their overall trust in MEC server s, which is given as follows:

$$tr_{s,n} = \begin{cases} \kappa \cdot dtr_{s,n} + (1 - \kappa) itr_{s,n}, & \text{if } N_{n,s} \neq 0 \\ 0.5 & \text{otherwise} \end{cases} \tag{13}$$

where $N_{n,s}$ denotes the number of times that MEC server s has provided services to user n, and the weight factor κ is defined as follows:

$$\kappa = \frac{1 - \exp(-N_{n,s})}{1 + \exp(-N_{n,s})} \tag{14}$$

The role of the weight factor κ is to balance the impact of the direct and indirect trust into the overall trust score. Higher values of κ give priority to the users' personal experiences, while lower values allow the overall trust score to incorporate broader feedback for comprehensive trust evaluation.

In our proposed research work, the robustness of the direct and indirect trust metrics against varying network conditions and user behaviors is a critical aspect of the multi-access edge computing (MEC) environment's operation. The direct and indirect trust leverage (1) the users' historical interactions and (2) the collective experiences of other users to evaluate the trustworthiness of the MEC servers. It is noted that the direct trust is influenced by the user's previous interactions with the server, while accounting for the satisfaction (or not) of the user's utility and latency requirements. On the other hand, the indirect trust incorporates the recommendations provided by other socially connected users. By jointly considering the direct and indirect trust, the evaluation of the trust level of each user to the MEC servers is improved, even under scenarios where individual interactions are infrequent or the network conditions fluctuate. The main network conditions that can influence the users' trust are the bandwidth, channel gain conditions, and service load of the MEC servers. Focusing on the bandwidth, if the bandwidth varies, then the users' experienced throughput is affected. However, the direct trust metric takes into account the experienced latency and utility, which can reflect changes due to the bandwidth fluctuations. Focusing on the users' channel gains, which can also vary due to the users' mobility or even due to environmental factors, the direct trust metric adjusts to the channel gain conditions' variations as it incorporates the real-time performance of the MEC server. Focusing on the MEC server's load, which can vary depending on the users' offloaded tasks, the computing capacity allocated to each user can vary and ultimately affect the processing time. The indirect trust metric includes the feedback from other users, who may experience different server loads. Thus, the proposed approach helps in adjusting the trust score based on the server's performance under varying loads. Focusing on the users' behavior, the users who interact infrequently with the MEC servers depend on the input stemming from the indirect trust metric. The indirect trust aggregates the collective experiences of other users and provides a reliable trust score, even with limited direct interactions. For example, new users can trust servers with high indirect trust scores based on the feedback from established users. The indirect trust also accounts for potential malicious behavior by weighing the reliability of the feedback from other users. Specifically, recommendations from users with strong social connections and similar behavior patterns are given more weight, and thus, filter out unreliable or deceptive feedback from potentially malicious users.

3.4. Reinforcement-Learning-Based MEC Server Selection

The objective for the users is to independently choose an MEC server to efficiently handle their computational tasks. In this scenario, the users function as reinforcement learning (RL) agents that evaluate the potential reward $\rho_{n,a_s}^{(ite)}$ associated with each action a_s at each iteration ite of the RL algorithm. This process continues until they reach a stable and optimal MEC server selection. The possible actions available to the users are represented by the set $\mathcal{A} = \{a_1, \ldots, a_s, \ldots, a_S\}$. The reward $\rho_{n,a_s}^{(ite)}$ for a user depends on several factors, i.e., the computational capacity $f_{s,n}$ provided by the chosen MEC server s, the user's confidence in the quality of service offered by the MEC server, the proximity of the user to the MEC server (which affects the user's communication latency), and the MEC server's overall computing capacity. By combining all these factors, the reward experienced by user n when offloading its computing task to MEC server s is given as follows:

$$\rho_{n,a_s}^{(ite)} = w_1 \frac{f_{s,n}}{F_s} + w_2 tr_{s,n} + w_3 \frac{x}{d_{s,n}} + w_4 \frac{F_s}{\sum\limits_{\forall s \in \mathcal{S}} F_s} \tag{15}$$

where the trust level $tr_{s,n} \in [0,1]$ quantifies how much user n trusts MEC server s; $d_{s,n}$ [m] denotes the distance between MEC server s and user n; x [m] represents the closest distance of a user to the selected MEC server; and the weights satisfy the following conditions: $w_1 + w_2 + w_3 + w_4 = 1$, where $w_1, w_2, w_3, w_4 \in \mathbb{R}^+$. A higher value of w_1 increases the importance of the computational capacity ratio and gives priority to servers with higher available capacity relative to their total capacity. In this case, the users tend to prefer the servers with better computational resources more in order to optimize the task processing speed. A higher value of w_2 emphasizes the trust level, and thus, MEC servers with higher trust scores are preferred. In this case, the reliability and security of the task offloading process are improved, as the users are more likely to choose trusted servers. A higher value of w_3 increases the importance of proximity, and thus, the users tend to prefer closer servers to offload their tasks. In this case, the communication latency is decreased and the users' experience and system responsiveness are improved. Finally, a higher value of w_4 emphasizes the overall computing capacity and promotes the selection of servers with a larger total capacity. In this case, a balance of the load across the network is achieved, and the servers' overloading is prevented.

The optimistic Q-Learning with upper bound confidence (OQ-UCB) action selection RL algorithm is introduced for the users to autonomously choose an MEC server. According to the experienced reward $\rho_{n,a_s}^{(ite)}$ at the iteration ite of the OQ-UCB algorithm, each user updates its expected reward $Q_{a_s}^{(ite)}$ to select an MEC server in the next iteration, which is defined as follows:

$$Q_{a_s}^{(ite)} = Q_{a_s}^{(ite-1)} + \gamma(\rho_{n,a_s}^{(ite)} - Q_{a_s}^{(ite-1)}) \tag{16}$$

where $\gamma \in (0,1)$ represents the learning rate of the OQ-UCB algorithm. Every user chooses an MEC server based on their anticipated reward and uncertainty regarding the outcome, which can be described as follows:

$$a_s^{(ite)} = \arg\max_{a_s^{(ite)}} [Q_{a_s}^{(ite)} + c \cdot \sqrt{\frac{\ln(ite)}{N_{a_s}^{(ite)}}}] \tag{17}$$

where $c \in \mathbb{R}^+$ represents the confidence level and $N_{a_s}^{(ite)}$ denotes the number of times action a_s has been selected up to iteration ite. Equation (17) incorporates two main components. The first addresses the uncertainty related to the user's selection of an MEC server, where a consistent choice reduces the uncertainty, while variability in terms of choosing different actions encourages the exploration for an optimal server selection. The convergence of the OQ-UCB algorithm is achieved when the expected reward $Q_{a_s}^{(ite)} \geq \epsilon$, where $\epsilon = 95\%$ for at least one MEC server a_s. By utilizing the OQ-UCB algorithm, the users are enabled to autonomously determine their preferred MEC server choice to offload their computing tasks.

4. Multilateral Bargaining Resource Allocation

The proposed mechanism for the reinforcement-learning-based edge server selection enables each user to independently choose an MEC server for offloading their data computing tasks. This process allows the MEC servers to dynamically adjust their resource allocation (i.e., computing capacity) based on the number of users requiring service while considering the sequence of service requests and data-processing volumes requested by the users. The users engage in a competition between each other to access a larger share of the selected MEC server's computing capacity F_s to meet their latency constraints $T_n, \forall n \in \mathcal{N}$, and satisfy their minimum utility requests U_n.

This competitive interaction between the users who select the same MEC servers $\mathcal{N}_s$ can be modeled as a *multilateral bargaining game*. In this game, the users negotiate between each other to share the common pool of resources F_s available at the selected MEC server. The computing capacity allocated to each user is denoted as $f_{s,n}$. The bargaining operator for an individual user n is defined by the matrix $\mathcal{B}_n$, where $b_{n'n'} = \hat{t}_{n'}$; $\forall n \neq n', b_{nn'} = 1 - \hat{t}_{n'}$; and $\forall n \neq n', b_{nn} = 1$, with $\hat{t}_n = \frac{B_n}{\sum_{n \in \mathcal{N}_s} B_n}$. In the matrix $\mathcal{B}_n$, B_n represents the bargaining power of user n, which is determined relative to the total bargaining power of all the users selecting the same MEC server F_s:

$$\mathcal{B}_n = \begin{bmatrix} \hat{t}_1 & 0 & \dots & 0 & 0 & 0 & \dots & 0 \\ 0 & \hat{t}_2 & \dots & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots & \vdots & \vdots & \dots & \vdots \\ 1-\hat{t}_1 & 1-\hat{t}_2 & \dots & 1-\hat{t}_{n-1} & 1 & 1-\hat{t}_{n+1} & \dots & 1-\hat{t}_{N_s} \\ \vdots & \vdots & \dots & \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & \hat{t}_{N_s} \end{bmatrix} \tag{18}$$

Let the bargaining operator in the multilateral bargaining game be denoted by $\mathcal{B} = \mathcal{B}_1\mathcal{B}_2 \dots \mathcal{B}_{N_s}$, its corresponding characteristic polynomial be given by $c(\lambda) = \det(\lambda I - \mathcal{B})$, and each individual bargaining operator be partitioned as follows:

$$\mathcal{B}_n = \begin{bmatrix} \mathcal{B}_{11}^{(n)} & \mathcal{B}_{12}^{(n)} \\ \mathcal{B}_{21}^{(n)} & \mathcal{B}_{22}^{(n)} \end{bmatrix} \tag{19}$$

where $\mathcal{B}_{11}^{(n)}$ is a scalar matrix, and $\mathcal{B}_{22}^{(n)}$ is a matrix with dimensions $(N_s - 1) \times (N_s - 1)$.

A similar partition is performed for the bargaining operator of the overall multilateral bargaining game, as follows:

$$\mathcal{B} = \begin{bmatrix} \mathcal{B}_{11} & \mathcal{B}_{12} \\ \mathcal{B}_{21} & \mathcal{B}_{22} \end{bmatrix} \tag{20}$$

Based on the above analysis and following the principles of bargaining game theory [49,50], the share function of each user is given as follows:

$$\begin{aligned} &\mathbb{S}_n(\hat{t}_{n+1}, \dots, \hat{t}_{N_s}, \hat{t}_1, \dots, \hat{t}_{n-1}) \\ &= \det(I - \mathcal{B}_{22}^{(n)} \dots \mathcal{B}_{22}^{(N_s)} \mathcal{B}_{22}^{(1)} \dots \mathcal{B}_{22}^{(n-1)}) = \det(I - \mathcal{B}_{22}) \end{aligned} \tag{21}$$

and each user's optimal share of the F_s computing resource of the selected MEC server is derived below:

$$f_{s,n} = \frac{\hat{t}_n^{n-1} \cdot \mathbb{S}_n(\hat{t}_{n+1}, \dots, \hat{t}_{N_s}, \hat{t}_1, \dots, \hat{t}_{n-1})}{\frac{\partial c(\lambda)}{\partial \lambda}|_{\lambda=1}} F_s \tag{22}$$

A detailed flowchart of the TRUST-ME model is presented in Figure 1.

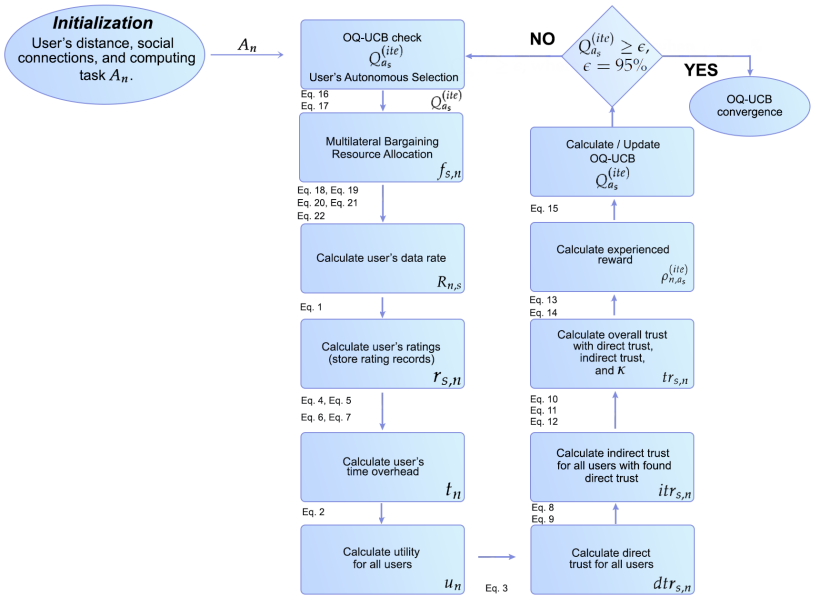

Figure 1. TRUST-ME flowchart.

5. Numerical Evaluation

In this section, the results are given of a detailed simulation-based numerical evaluation that was performed in order to demonstrate the operational characteristics and the priority of the TRUST-ME model. Specifically, its pure operation and performance are presented in Section 5.1 and a real-world application scenario of implementing the TRUST-ME model is discussed in Section 5.2. A detailed scalability analysis is performed in Section 5.3 in order to demonstrate the robustness and superiority of the TRUST-ME model. A thorough comparative evaluation is presented in Section 5.4 by considering different approaches in terms of enabling the users to select an MEC server and in terms of allocating the MEC servers computing resources.

The following simulation parameters were used in the rest of the analysis: $N = 12$, $S = 3$, $c = 1.5$, $\lambda = 2$, $\gamma = 0.8$, $w_1 = 0.05$, $w_2 = 0.85$, $w_3 = 0.05$, $w_4 = 0.05$, $\phi_n = 1$ $[\frac{CPU-cycles}{bits}]$, $e_n = 10^{-9}$ J, $f_c = 73.5$ GHz, $I_0 = 10 \cdot 10^{-22}$, $W = 5$ GHz, $P_n = 0.2$ W, $F_s = [3, 2, 1]$ $[\frac{GCPU-cycles}{sec}]$, $T_n = [20, 24, 27, 31, 34, 38, 42, 45, 49, 53, 56, 60]$ msec, $B_n = [2.00, 1.92, 1.84, 1.75, 1.67, 1.59, 1.51, 1.43, 1.34, 1.26, 1.18, 1.10]$ Mbits, $U_n = [90.0, 82.7, 75.5, 68.2, 60.9, 53.6, 46.4, 39.1, 31.8, 24.5, 17.3, 10.0]$, $\mathcal{C}_s = 0.007$ $[\frac{\$}{CPU-cycles}]$, $\beta_n = 10^{-1}$, $d_{1,n} = [14.14, 20.75, 28.20, 35.96, 43.88, 51.88, 59.93, 68.01, 76.11, 84.23, 92.36, 100.50]$ m, $d_{2,n} = [31.62, 35.08, 39.94, 45.75, 52.21, 59.09, 66.27, 73.66, 81.20, 88.85, 96.59, 104.40]$ m, and $d_{3,n} = [50.99, 53.20, 56.52, 60.77, 65.77, 71.36, 77.41, 83.82, 90.52, 97.44, 104.55, 111.80]$ m, unless otherwise explicitly stated.

5.1. Operation and Performance of the TRUST-ME *Model*

For this section, our goal was to present the pure operation and performance of the TRUST-ME model. Specifically, we considered an indicative setup that consisted of 12 users and three MEC servers, where the users with higher IDs were characterized by relaxed latency constraints, as they requested the processing of a smaller amount of data and imposed a smaller constraint in terms of the expected achieved utility. Figure 2a–d demonstrate the users' allocated computing capacity, their achieved data rate, and their experienced latency and utility while considering the selected MEC server following the optimistic Q-learning with the upper bound confidence action selection reinforcement learning algorithm. The results demonstrate that more users tended to select the MEC server that was characterized by a higher computing capacity, i.e., MEC servers with lower IDs. Furthermore, the results show that the users with lower IDs that entered into the the bargaining process early and are characterized by a computing task that requires higher computing capacity resulted in being allocated a corresponding higher computing capacity

by the selected MEC server. Furthermore, considering that the users resided relatively closer to the MEC server with higher computing capacity, they achieved a corresponding higher data rate, while also considering that users with lower IDs were in general closer to the MEC servers. By combining the two previous observations, we concluded that the users with lower IDs experienced lower corresponding latency by considering the selected MEC server, as they were allocated a higher computing capacity and they achieved a better data rate. The latter observation resulted in higher experienced utility for users with lower IDs relative to the corresponding selected MEC server given that the users who selected the same MEC server competed for its computing resources. Furthermore, Figure 3 illustrates the number of users associated with each MEC server and the corresponding total trust score accumulated by all the users who selected the corresponding MEC server. The results indicate a clear preference among the users for selecting the MEC server with a greater computing capacity. This preference correlated with an increased trust score for the service given the superior quality of service experienced when using higher-capacity MEC servers.

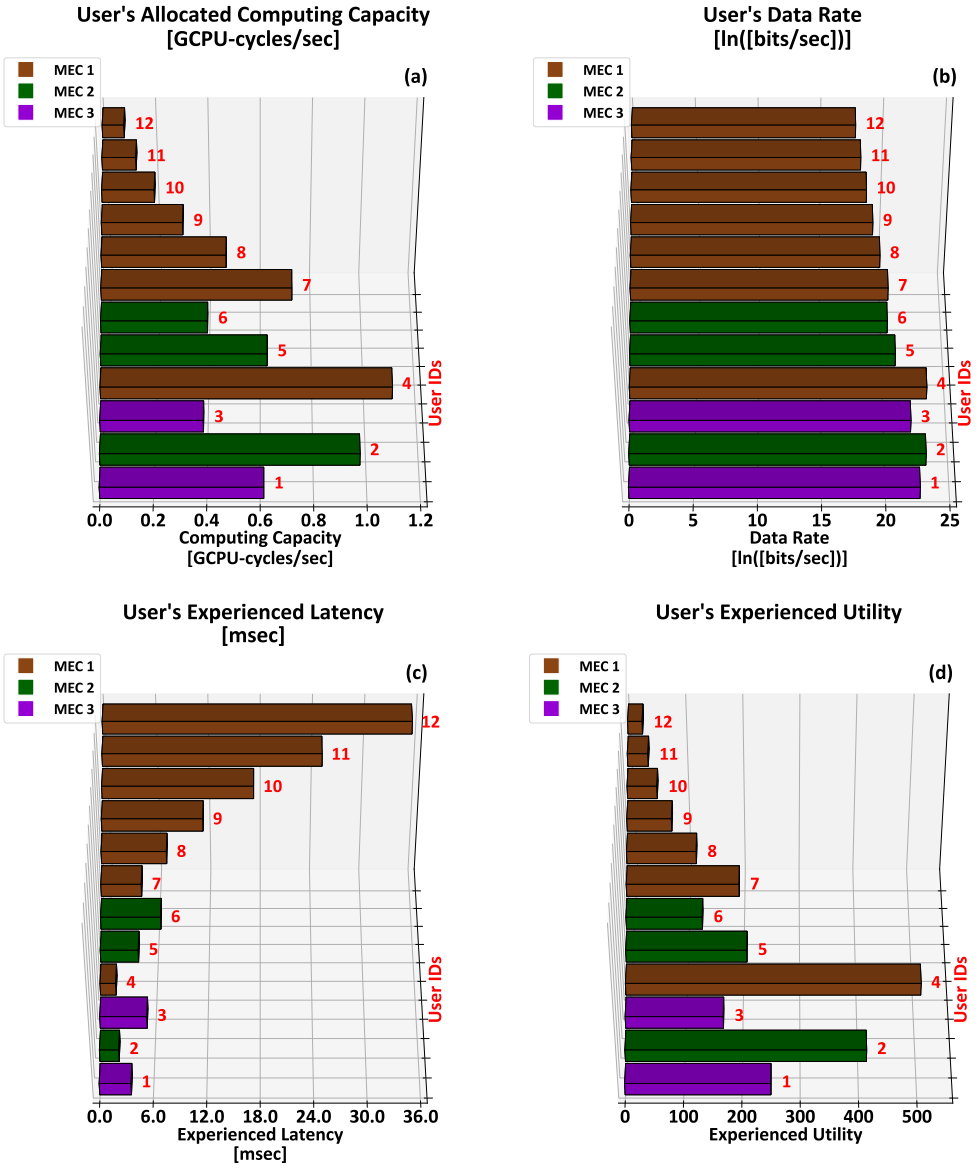

Figure 2. (**a**) Allocated computing capacity, (**b**) data rate, (**c**) experienced latency, and (**d**) experienced utility for the users and their respective MEC selected server.

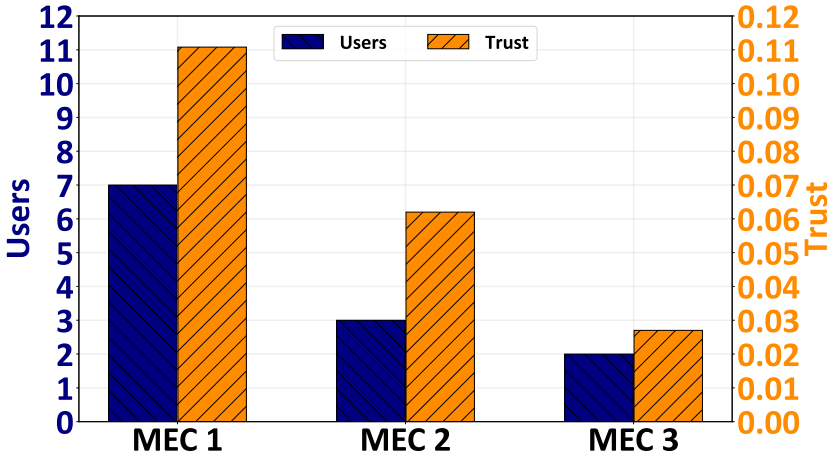

Figure 3. Trust analysis.

The convergence of the optimistic Q-learning with the upper bound confidence action selection reinforcement learning algorithm was also studied. Figure 4a,b demonstrate the OQ-UCB reward (Equation (15)) and the corresponding experience expected reward (Equation (16)) for all the users regarding their corresponding finally selected action over the iterations of the OQ-UCB algorithm, where different colors are used for the corresponding finally selected MEC servers by each user. The results demonstrate that the OQ-UCB algorithm converged fast (in less than 200 iterations, corresponding to a few milliseconds) to a stable action selection and the corresponding expected reward, i.e., Q-value, achieved values above 95% for all the users. Moreover, the results show that the users with lower IDs experienced a higher OQ-UCB reward given the higher trust that they created with respect to the corresponding MEC server, their proximity to the MEC servers, and the higher computing capacity that they were allocated by the selected MEC server.

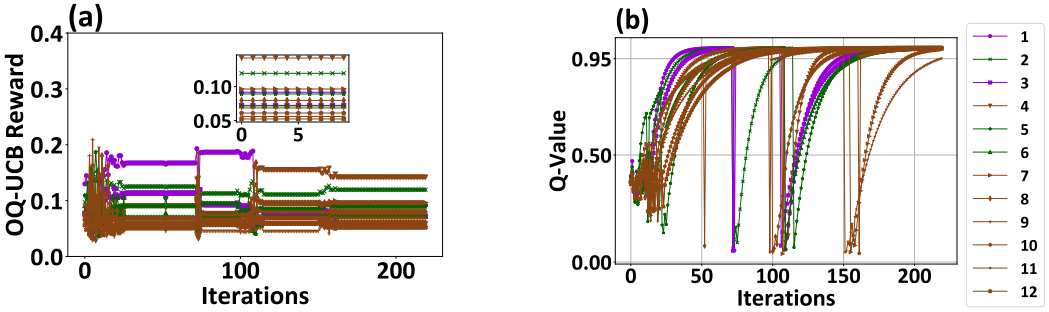

Figure 4. Convergence analysis: (**a**) Reward and (**b**) Q-value convergence.

5.2. A Real-World Application Scenario

As discussed in this section, we examined three representative applications requested by the users, with each one characterized by different levels of computing demand and data-processing requirements of the MEC servers. Specifically, we considered three distinct user groups, with each one consisting of four users that requested applications for augmented reality, real-time data analytics, and content caching. These applications represented high, medium, and low computing demands, respectively, in terms of the amount of data processed at the MEC servers. The augmented reality applications required significant computational resources for real-time image processing, object recognition, and overlay rendering. These applications were latency-sensitive and demanded quick responses to ensure a seamless user experience. The users that ran the augmented reality applications offloaded tasks such as image processing and rendering to the MEC servers, which allocated substantial computing capacity to meet the high processing requirements. Real-time data

analytics involved processing streaming data for insights and decision making. These applications required moderate computational resources and were moderately latency sensitive. Content caching applications involved storing and retrieving frequently accessed data to improve the access speed and reduce the latency. These applications required relatively low computational resources and were less latency sensitive.

Figure 5a illustrates the convergence of the OQ-UCB reward for the final action selected by the users with high, medium, and low computing demands. Figure 5b shows the average computing capacity allocated to the users by the selected MEC servers. The results indicate that the users that requested applications with higher computing demands experienced a higher OQ-UCB reward, as the servers allocated greater computing capacity to handle their computationally intensive tasks.

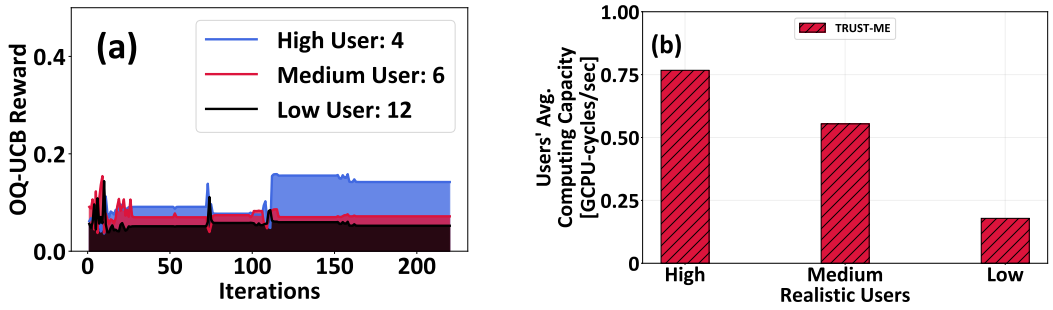

Figure 5. (**a**) Convergence of the OQ-UCB reward for the final action selected by users with high, medium, and low computing demands. (**b**) Average computing capacity allocated to users by the selected MEC servers.

5.3. Scalability Analysis

As discussed in this section, we analyzed the scalability of the proposed TRUST-ME model by varying the number of MEC servers and users. Figure 6a–c show the average computing capacity allocated to each user, the corresponding experienced average user's utility, and the resulting average latency experienced as the number of MEC servers and users changed. The results reveal that as the number of users increased and the number of MEC servers remained constant, the average computing capacity available to each user decreased, and thus, this caused an increased latency. This phenomenon occurred due to the finite computing capacity of the MEC servers, which when divided between more users, led to greater utilization of the MEC servers' resources, and thus, ultimately increased the users' experienced latency.

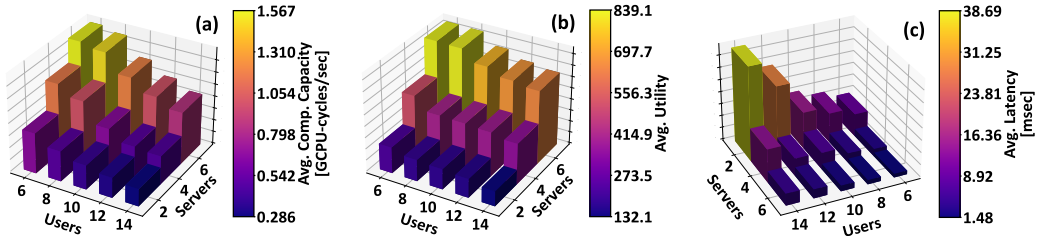

Figure 6. Scalability analysis.

5.4. Comparative Evaluation

As discussed in this section, we performed a comparative analysis of the proposed TRUST-ME model against two alternative scenarios:

- **Without trust:** the users selected an MEC server without considering the trust levels related to the services provided, and the resource allocation followed the multilateral bargaining game.

- **Proportional fair:** the users selected an MEC server using the proposed OQ-UCB algorithm, and the MEC servers' resources were allocated based on the proportional fairness by taking into account the users' data-processing needs, i.e., B_n.

Figure 7a–c show the percentage of the computing capacity allocated by the selected MEC servers to the three user groups (high, medium, and low computing demand), where in total, we considered a set of 12 users. Figure 8a–c show the average computing capacity allocation, the experienced average users' utility, and the corresponding latency experienced by users under each scenario.

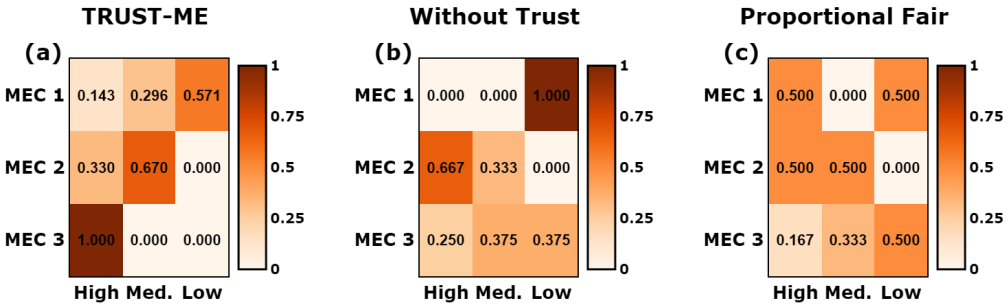

Figure 7. Comparative heat distribution plots showing the percentage allocations of the computing capacity by the selected MEC servers to high, medium, and low computing demand user groups.

The results reveal that the proposed TRUST-ME model achieved a more balanced distribution of the users across the MEC servers. The users with higher computing demands tended to select the MEC servers with lower capacities in order to dominate them and be allocated a large portion of their computing capacity. In the without trust scenario, the users with high computing demands predominantly chose the server with the medium computing capacity (MEC server 2), leaving the users with lower computing demands to utilize the higher-capacity server (Figure 7b). In the proportional fair scenario, the users with higher computing demands often selected the MEC servers with the high computing capacity, allowing them to dominate these servers and secure a larger share of its resources. Thus, the users with lower computing demands were distributed between the servers to capture their remaining computing capacities. The results also show that the without trust and proportional fairness resource allocation scenarios inadequately addressed the users' computing demands, where they experienced higher latencies due to insufficient and unbalanced resource allocation (Figure 8c).

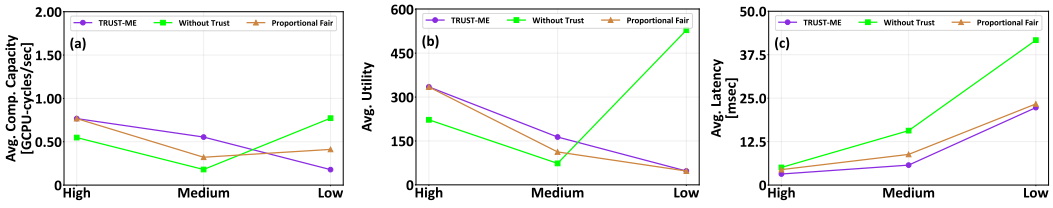

Figure 8. Comparative evaluation showing the (**a**) average computing capacity allocation, (**b**) average users' utility, and (**c**) latency experienced by users across different scenarios.

The proportional fairness (PF) strategy, while it allocated the resources based on the users' data processing needs, tended to allow users with high computing demands to dominate high-capacity servers. This approach can result in suboptimal resource utilization, where low-demand users do not receive sufficient resources. Also, the PF strategy did not account for the dynamic and often fluctuating network conditions as effectively as TRUST-ME. The users could experience varying levels of service quality due to the bandwidth fluctuations, server loads, and channel gain conditions. Moreover, it should be noted that

although the latency was only slightly higher by following the PF strategy, this approach did not fully address the need for adaptive and efficient resource allocation under varying demands and conditions. Additionally, the users with higher demands experienced better performance at the expense of those with lower demands under the PF strategy. On the other hand, the TRUST-ME model provided a more robust evaluation of the MEC server performance by jointly considering the direct and indirect trust. This adaptability ensured consistent and reliable service quality, even under fluctuating network conditions. Also, it should be noted that the execution time of the TRUST-ME model was very low (few msec), as discussed in Section 5.1, and TRUST-ME ensured lower latency by effectively balancing the resource allocation and adapting to real-time network conditions and user behaviors. The integration of trust metrics allowed for a more comprehensive assessment of the servers' performance and it led to more informed and efficient resource allocation decisions. This resulted in a consistent reduction in latency, as presented in Figure 8c. Moreover, TRUST-ME enhanced the overall user experience by providing a balanced and fair distribution of resources. Specifically, the users with varying demands could rely on the trust-based allocation to receive sufficient resources and ensure a satisfactory quality of service.

6. Conclusions

In this paper, the TRUST-ME model is presented by considering a multi-MEC servers multi-users edge computing environment, where the users are characterized by diverse computing tasks routed to the MEC servers for processing. A novel trust model is introduced for the users to assess the MEC servers' computing capabilities, integrating both direct and indirect trust mechanisms derived from the social interactions among the users using the same servers. A reinforcement learning approach based on the optimistic Q-learning with the upper confidence bound action selection algorithm is introduced to enable the autonomous MEC server selection by the users. A multilateral bargaining model was designed to support the resource allocation of the MEC servers' computing capacity to the associated users to ensure a fair distribution of the MEC servers' computing capacities based on the users' computing demands. Numerical simulations demonstrated the operational efficiency, convergence, and scalability of the TRUST-ME model, which were validated through real-world scenarios and simulations. Comparative evaluations against existing approaches quantified the superior performance of the TRUST-ME model in terms of the MEC server selection and resource allocation strategies.

Part of our current and future work includes the extension of the proposed trust model to incorporate dynamic trust updates based on real-time performance feedback from the MEC servers and the users, as well as adaptive learning mechanisms. Also, our goal is to implement the proposed model in an industrial IoT environment to support the users, i.e., IoT nodes, with real-time decision making in industrial IoT applications.

Author Contributions: Conceptualization and writing, S.T., A.B.R. and M.S.S.; methodology and supervision, E.E.T. All authors read and agreed to the published version of this manuscript.

Funding: The research of Tsiropoulou and Tsikteris was partially supported by the National Scienca Foundation, USA, Awards #2219617 and #2319994.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Liu, Y.; Zhong, P.; Yang, Z.; Li, W.; Li, S. Computation Offloading Based on a Distributed Overlay Network Cache-Sharing Mechanism in Multi-Access Edge Computing. *Future Internet* **2024**, *16*, 136. [CrossRef]
2. Kochetkova, I.; Leonteva, K.; Ghebrial, I.; Vlaskina, A.; Burtseva, S.; Kushchazli, A.; Samouylov, K. Controllable Queuing System with Elastic Traffic and Signals for Resource Capacity Planning in 5G Network Slicing. *Future Internet* **2023**, *16*, 18. [CrossRef]

3. Du, J.; Sun, Y.; Zhang, N.; Xiong, Z.; Sun, A.; Ding, Z. Cost-Effective Task Offloading in NOMA-Enabled Vehicular Mobile Edge Computing. *IEEE Syst. J.* **2023**, *17*, 928–939. [CrossRef]
4. Li, K.; Wang, X.; He, Q.; Yang, M.; Huang, M.; Dustdar, S. Task Computation Offloading for Multi-Access Edge Computing via Attention Communication Deep Reinforcement Learning. *IEEE Trans. Serv. Comput.* **2023**, *16*, 2985–2999. [CrossRef]
5. Liu, L.; Sun, B.; Tan, X.; Tsang, D.H.K. Energy-Efficient Resource Allocation and Subchannel Assignment for NOMA-Enabled Multiaccess Edge Computing. *IEEE Syst. J.* **2022**, *16*, 1558–1569. [CrossRef]
6. Liu, P.; An, K.; Lei, J.; Zheng, G.; Sun, Y.; Liu, W. SCMA-Based Multiaccess Edge Computing in IoT Systems: An Energy-Efficiency and Latency Tradeoff. *IEEE Internet Things J.* **2022**, *9*, 4849–4862. [CrossRef]
7. Yang, W.; Shi, L.; Liang, H.; Zhang, W. Trusted Mobile Edge Computing: DAG Blockchain-Aided Trust Management and Resource Allocation. *IEEE Trans. Wirel. Commun.* **2024**, *23*, 5006–5018. [CrossRef]
8. Zhou, J.; Zhang, X. Fairness-Aware Task Offloading and Resource Allocation in Cooperative Mobile-Edge Computing. *IEEE Internet Things J.* **2022**, *9*, 3812–3824. [CrossRef]
9. Habiba, U.; Maghsudi, S.; Hossain, E. A Repeated Auction Model for Load-Aware Dynamic Resource Allocation in Multi-Access Edge Computing. *IEEE Trans. Mob. Comput.* **2024**, *23*, 7801–7817. [CrossRef]
10. Do-Duy, T.; Huynh, D.V.; Garcia-Palacios, E.; Cao, T.V.; Sharma, V.; Duong, T.Q. Joint Computation and Communication Resource Allocation for Unmanned Aerial Vehicle NOMA Systems. In Proceedings of the 2023 IEEE 28th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Edinburgh, UK, 6–8 November 2023; pp. 290–295. [CrossRef]
11. Deng, C.; Fang, X.; Wang, X. UAV-Enabled Mobile-Edge Computing for AI Applications: Joint Model Decision, Resource Allocation, and Trajectory Optimization. *IEEE Internet Things J.* **2023**, *10*, 5662–5675. [CrossRef]
12. Tun, Y.K.; Dang, T.N.; Kim, K.; Alsenwi, M.; Saad, W.; Hong, C.S. Collaboration in the Sky: A Distributed Framework for Task Offloading and Resource Allocation in Multi-Access Edge Computing. *IEEE Internet Things J.* **2022**, *9*, 24221–24235. [CrossRef]
13. Liu, B.; Liu, C.; Peng, M. Computation Offloading and Resource Allocation in Unmanned Aerial Vehicle Networks. *IEEE Trans. Veh. Technol.* **2023**, *72*, 4981–4995. [CrossRef]
14. Chen, Y.; Wu, J.; Han, J.; Zhao, H.; Deng, S. A Game-Theoretic Approach-Based Task Offloading and Resource Pricing Method for Idle Vehicle Devices Assisted VEC. *IEEE Internet Things J.* **2024**, *11*, 21954–21969. [CrossRef]
15. Niu, Z.; Liu, H.; Ge, Y.; Du, J. Distributed Hybrid Task Offloading in Mobile-Edge Computing: A Potential Game Scheme. *IEEE Internet Things J.* **2024**, *11*, 18698–18710. [CrossRef]
16. Wang, L.; Zhang, G. Joint Service Caching, Resource Allocation and Computation Offloading in Three-Tier Cooperative Mobile Edge Computing System. *IEEE Trans. Netw. Sci. Eng.* **2023**, *10*, 3343–3353. [CrossRef]
17. Jiang, J.; Xin, P.; Wang, Y.; Liu, L.; Chai, Y.; Zhang, Y.; Liu, S. Computing Resource Allocation in Mobile Edge Networks Based on Game Theory. In Proceedings of the 2021 IEEE 4th International Conference on Electronics and Communication Engineering (ICECE), Xi'an, China, 17–19 December 2021; pp. 179–183. [CrossRef]
18. Chen, G.; Chen, Y.; Mai, Z.; Hao, C.; Yang, M.; Du, L. Incentive-Based Distributed Resource Allocation for Task Offloading and Collaborative Computing in MEC-Enabled Networks. *IEEE Internet Things J.* **2023**, *10*, 9077–9091. [CrossRef]
19. Van Huynh, D.; Nguyen, V.D.; Khosravirad, S.R.; Sharma, V.; Dobre, O.A.; Shin, H.; Duong, T.Q. URLLC Edge Networks With Joint Optimal User Association, Task Offloading and Resource Allocation: A Digital Twin Approach. *IEEE Trans. Commun.* **2022**, *70*, 7669–7682. [CrossRef]
20. Fang, T.; Yuan, F.; Ao, L.; Chen, J. Joint Task Offloading, D2D Pairing, and Resource Allocation in Device-Enhanced MEC: A Potential Game Approach. *IEEE Internet Things J.* **2022**, *9*, 3226–3237. [CrossRef]
21. Li, Y.; Yang, B.; Wu, H.; Han, Q.; Chen, C.; Guan, X. Joint Offloading Decision and Resource Allocation for Vehicular Fog-Edge Computing Networks: A Contract-Stackelberg Approach. *IEEE Internet Things J.* **2022**, *9*, 15969–15982. [CrossRef]
22. Irtija, N.; Anagnostopoulos, I.; Zervakis, G.; Tsiropoulou, E.E.; Amrouch, H.; Henkel, J. Energy Efficient Edge Computing Enabled by Satisfaction Games and Approximate Computing. *IEEE Trans. Green Commun. Netw.* **2022**, *6*, 281–294. [CrossRef]
23. Cui, G.; He, Q.; Chen, F.; Zhang, Y.; Jin, H.; Yang, Y. Interference-Aware Game-Theoretic Device Allocation for Mobile Edge Computing. *IEEE Trans. Mob. Comput.* **2022**, *21*, 4001–4012. [CrossRef]
24. Cui, G.; He, Q.; Xia, X.; Lai, P.; Chen, F.; Gu, T.; Yang, Y. Interference-Aware SaaS User Allocation Game for Edge Computing. *IEEE Trans. Cloud Comput.* **2022**, *10*, 1888–1899. [CrossRef]
25. Li, Q.; Ma, X.; Zhou, A.; Luo, X.; Yang, F.; Wang, S. User-Oriented Edge Node Grouping in Mobile Edge Computing. *IEEE Trans. Mob. Comput.* **2023**, *22*, 3691–3705. [CrossRef]
26. Meskar, E.; Liang, B. MAGIKS: Fair Multi-Resource Allocation Game Induced by Kalai-Smorodinsky Bargaining Solution. *IEEE Open J. Commun. Soc.* **2022**, *3*, 797–810. [CrossRef]
27. Li, Z.; Ju, H.; Ren, Z. A Learning Game-Based Approach to Task-Dependent Edge Resource Allocation. *Future Internet* **2023**, *15*, 395. [CrossRef]
28. Shang, C.; Sun, Y.; Luo, H.; Guizani, M. Computation Offloading and Resource Allocation in NOMA–MEC: A Deep Reinforcement Learning Approach. *IEEE Internet Things J.* **2023**, *10*, 15464–15476. [CrossRef]
29. Fragkos, G.; Kemp, N.; Tsiropoulou, E.E.; Papavassiliou, S. Artificial Intelligence Empowered UAVs Data Offloading in Mobile Edge Computing. In Proceedings of the ICC 2020—2020 IEEE International Conference on Communications (ICC), Dublin, Ireland, 7–11 June 2020; pp. 1–7. [CrossRef]

30. Poltronieri, F.; Stefanelli, C.; Tortonesi, M.; Zaccarini, M. Reinforcement Learning vs. Computational Intelligence: Comparing Service Management Approaches for the Cloud Continuum. *Future Internet* **2023**, *15*, 359. [CrossRef]
31. Wu, G.; Wang, H.; Zhang, H.; Zhao, Y.; Yu, S.; Shen, S. Computation Offloading Method Using Stochastic Games for Software-Defined-Network-Based Multiagent Mobile Edge Computing. *IEEE Internet Things J.* **2023**, *10*, 17620–17634. [CrossRef]
32. Fragkos, G.; Lebien, S.; Tsiropoulou, E.E. Artificial Intelligent Multi-Access Edge Computing Servers Management. *IEEE Access* **2020**, *8*, 171292–171304. [CrossRef]
33. Shi, B.; Wu, Y. Task Offloading and Resource Allocation Strategies Among Multiple Edge Servers. *IEEE Internet Things J.* **2024**, *11*, 14647–14656. [CrossRef]
34. Abouaomar, A.; Cherkaoui, S.; Mlika, Z.; Kobbane, A. Service Function Chaining in MEC: A Mean-Field Game and Reinforcement Learning Approach. *IEEE Syst. J.* **2022**, *16*, 5357–5368. [CrossRef]
35. Huang, X.; He, L.; Chen, X.; Wang, L.; Li, F. Revenue and Energy Efficiency-Driven Delay-Constrained Computing Task Offloading and Resource Allocation in a Vehicular Edge Computing Network: A Deep Reinforcement Learning Approach. *IEEE Internet Things J.* **2022**, *9*, 8852–8868. [CrossRef]
36. Shi, Y.; Sagduyu, Y.E.; Erpek, T. Reinforcement Learning for Dynamic Resource Optimization in 5G Radio Access Network Slicing. In Proceedings of the 2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), Pisa, Italy, 14–16 September 2020; pp. 1–6. [CrossRef]
37. Apostolopoulos, P.A.; Tsiropoulou, E.E.; Papavassiliou, S. Game-theoretic Learning-based QoS Satisfaction in Autonomous Mobile Edge Computing. In Proceedings of the 2018 Global Information Infrastructure and Networking Symposium (GIIS), Thessaloniki, Greece, 23–25 October 2018; pp. 1–5. [CrossRef]
38. Binh, T.H.; Son, D.B.; Vo, H.; Nguyen, B.M.; Binh, H.T.T. Reinforcement Learning for Optimizing Delay-Sensitive Task Offloading in Vehicular Edge–Cloud Computing. *IEEE Internet Things J.* **2024**, *11*, 2058–2069. [CrossRef]
39. Peng, X.; Han, Z.; Xie, W.; Yu, C.; Zhu, P.; Xiao, J.; Yang, J. Deep Reinforcement Learning for Shared Offloading Strategy in Vehicle Edge Computing. *IEEE Syst. J.* **2023**, *17*, 2089–2100. [CrossRef]
40. Hou, Y.; Wei, Z.; Zhang, R.; Cheng, X.; Yang, L. Hierarchical Task Offloading for Vehicular Fog Computing Based on Multi-Agent Deep Reinforcement Learning. *IEEE Trans. Wirel. Commun.* **2024**, *23*, 3074–3085. [CrossRef]
41. Abdulazeez, D.H.; Askar, S.K. Offloading Mechanisms Based on Reinforcement Learning and Deep Learning Algorithms in the Fog Computing Environment. *IEEE Access* **2023**, *11*, 12555–12586. [CrossRef]
42. Fang, J.; Qu, D.; Chen, H.; Liu, Y. Dependency-Aware Dynamic Task Offloading Based on Deep Reinforcement Learning in Mobile-Edge Computing. *IEEE Trans. Netw. Serv. Manag.* **2024**, *21*, 1403–1415. [CrossRef]
43. Zhang, Z.; Wang, N.; Wu, H.; Tang, C.; Li, R. MR-DRO: A Fast and Efficient Task Offloading Algorithm in Heterogeneous Edge/Cloud Computing Environments. *IEEE Internet Things J.* **2023**, *10*, 3165–3178. [CrossRef]
44. Xu, X.; Song, Y. A Deep Reinforcement Learning-Based Optimal Computation Offloading Scheme for VR Video Transmission in Mobile Edge Networks. *IEEE Access* **2023**, *11*, 122772–122781. [CrossRef]
45. Huda, S.M.A.; Moh, S. Deep Reinforcement Learning-Based Computation Offloading in UAV Swarm-Enabled Edge Computing for Surveillance Applications. *IEEE Access* **2023**, *11*, 68269–68285. [CrossRef]
46. Deng, X.; Yin, J.; Guan, P.; Xiong, N.N.; Zhang, L.; Mumtaz, S. Intelligent Delay-Aware Partial Computing Task Offloading for Multiuser Industrial Internet of Things Through Edge Computing. *IEEE Internet Things J.* **2023**, *10*, 2954–2966. [CrossRef]
47. Li, Y.; Li, J.; Lv, Z.; Li, H.; Wang, Y.; Xu, Z. GASTO: A Fast Adaptive Graph Learning Framework for Edge Computing Empowered Task Offloading. *IEEE Trans. Netw. Serv. Manag.* **2023**, *20*, 932–944. [CrossRef]
48. Diamanti, M.; Charatsaris, P.; Tsiropoulou, E.E.; Papavassiliou, S. Incentive Mechanism and Resource Allocation for Edge-Fog Networks Driven by Multi-Dimensional Contract and Game Theories. *IEEE Open J. Commun. Soc.* **2022**, *3*, 435–452. [CrossRef]
49. Rubinstein, A. Perfect equilibrium in a bargaining model. *Econom. J. Econom. Soc.* **1982**, *50*, 97–109. [CrossRef]
50. Jin, Y.; Zhang, J. A closed form characterization of the stationary outcome in multilateral bargaining. *Front. Econ. China* **2013**, *8*, 272–287.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.