Quality-Assured Secured Load Sharing in Mobile Cloud Networking Environment

Snigdha Das, Manas Khatua, *Student Member, IEEE*, Sudip Misra, *Senior Member, IEEE*, and Mohammad S. Obaidat, *Fellow, IEEE*

Abstract—In mobile cloud networks (MCNs), a mobile user is connected with a cloud server through a network gateway, which is responsible for providing the required quality-of-service (QoS) to the users. If a user increases its service demand, the connecting gateway may fail to provide the requested QoS due to the overloaded demand, while the other gateways remain underloaded. Due to the increase in load in one gateway, the sharing of load among all the gateways is one of the prospective solutions for providing QoS-guaranteed services to the mobile users. Additionally, if a user misbehaves, the situation becomes more challenging. In this paper, we address the problem of QoS-guaranteed secured service provisioning in MCNs. We design a utility maximization problem for quality-assured secured load sharing (QuaLShare) in MCN, and determine its optimal solution using auction theory. In QuaLShare, the overloaded gateway detects the misbehaving gateways, and, then, prevents them from participating in the auction process. Theoretically, we characterize both the problem and the solution approaches in an MCN environment. Finally, we investigate the existence of Nash Equilibrium of the proposed scheme. We extend the solution for the case of multiple users, followed by theoretical analysis. Numerical analysis establishes the correctness of the proposed algorithms.

Index Terms—Mobile cloud networking, auction theory, Nash equilibrium, bandwidth distribution, load sharing

1 Introduction

Now-a-days mobile devices are inseparable components of our lives. Rapid reduction of hardware cost and the availability of numerous user-friendly applications such as health monitoring, gaming, entertainment, social networking, and trading applications make these devices more popular to the users. Despite the increase of usage of mobile devices, exploitation of their full functionalities is restricted by the limited energy supply, low bandwidth, low computational capacity, and seamless mobility. For overcoming these limitations, mobile devices are integrated with cloud computing in mobile cloud computing (MCC) systems. Mobile cloud networking (MCN), which one of the foundational aspects of MCC, is responsible for designing cost-efficient and secure service delivery across data communication networks. In MCN, a mobile user requests for expensive services to the cloud server though their respective interfacing gateways. The cloud server processes the service-requests, and, finally, provides the services to the users.

1.1 Motivation

On-demand service delivery is one of the basic characteristics of MCN. For providing real-time services over the wireless networks, dynamic allocation of resources such as bandwidth is essential. It may happen that the cloud service provider (CSP) has adequate bandwidth for providing the requested services to the mobile users, but the users are not able to harness it due to improper bandwidth management. Additionally, the cloud server has to assure a certain level of QoS in terms of service delay, reliability, jitter, and response time for encouraging the users to take services from it. Let us consider a cloud service provider, which provides different service demands to mobile users using a simple architecture, as shown in Fig. 1. When a mobile device changes its location from one place to another, the interfacing gateway also changes for making it feasible to offer uninterrupted service with adequate QoS from the cloud. After the change of location of a user, the current interfacing gateway may experience additional service load. We assume that the service request of all the users are independent of one another. Again, the connecting gateway may feel excessive service load when one or more mobile users increase their service demands. For maintaining the additional service request, proper bandwidth management is one of the prospective solutions. Recently, this problem was addressed by Misra et al. [1] by proposing an auction-based bandwidth distribution approach. The authors define the term "bandwidth distribution" as the proportional assignment of bandwidth to all the gateways, even if only a few gateways change their demands. Therefore, their proposed solution involves all the gateways in the distribution process, which we consider as additional overhead to the users with differential demands. We further observe that the overloaded gateway may be surrounded by underloaded gateways. In this case, a localized solution that only involves few local gateways can be computationally

- *S. Das, M. Khatua, and S. Misra are with the School of Information Technology, Indian Institute of Technology, Kharagpur, India. E-mail: {snigdhadas, manas.khatua, smisra}@sit.iitkgp.ernet.in.*
- *M.S. Obaidat is with the Department of Computer and Information Science, Fordham University, NY. E-mail: mobaidat@fordham.edu.*

Manuscript received 12 Nov. 2014; revised 27 June 2015; accepted 10 July 2015. Date of publication 16 July 2015; date of current version 6 Mar. 2019. Recommended for acceptance by D. Kliazovich.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TCC.2015.2457416

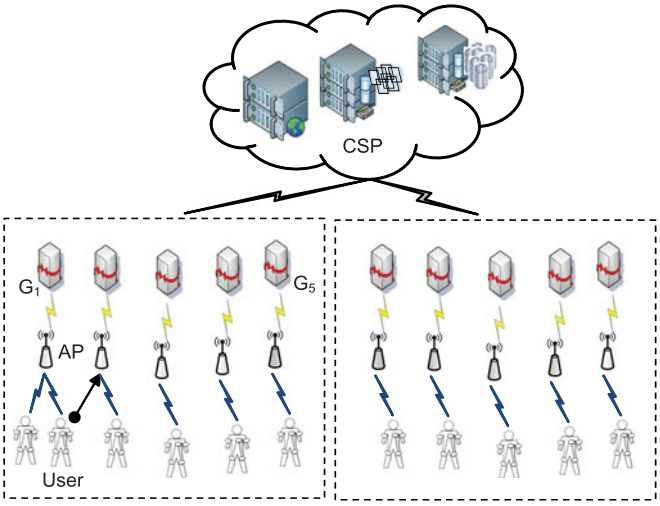

Fig. 1. Basic architecture of mobile cloud networks.

inexpensive with respect to the global solution that involves all the gateways of the network. These observations motivated us for designing a localized solution scheme, namely *quality-assured load sharing*. Thus, the mobile users are totally transparent about this load sharing process. During the processing time, the system will not accept any additional request. The system tracks those requests and processes in the next run. Additionally, optimal load sharing in a localized scheme cannot be guaranteed in the presence of a misbehaving gateway.

Definition 1 (Misbehaving Gateway). *A gateway is termed as a misbehaving gateway when it provides wrong information during the load sharing process for receiving additional privilege.*

As a consequence, the other gateways loose in the decision process. Nevertheless, the actual overloaded gateway may fail to maintain the required QoS to the mobile users connected to it. Therefore, a truthful load sharing scheme is required for avoiding such inefficiency of CSP.

1.2 Contributions

In this paper, we address the problem of QoS violation due to increase in load by a single user or multiple users as the CSP is responsible for providing QoS-guaranteed service provisioning in an MCN environment. Traditionally, load management is primarily achieved through load balancing. Our works step ahead by proposing a *load sharing* scheme for load management. At this juncture, it is pertinent to clarify that load sharing differs from the traditional concept of load balancing in that, while the former concerns sharing the additional demand with local gateways, the latter concerns equalizing the service demand among all the gateways. An architectural view of the problem is shown in Fig. 1. The dotted square in Fig. 1 represents the boundary of a local region, whereas all the gateways are in the global region. We assume that the CSP is authorized for the distribution of initial loads when global solution is required, and tracks the current QoS index level for all the connected gateways. As in the work by Misra et al. [1], we also consider QoS-guarantees in terms of *service delay*. In the load sharing process, our approach intelligently identifies the misbehaving gateways, and, then, avoids them from involving in the further execution process of the proposed scheme. We present load sharing as a *utility maximization* problem for all the gateways, in which the utility is formed using the *revenue* and *payoff* functions. The revenue function of a gateway is defined by two components—transmission delay-specific and service delay-specific revenue functions. The payoff function illustrates the pricing strategy between the overloaded gateway and its neighboring ones. We use an *auction theory*-based scheme to solve the optimization problem. Each neighbor of the overloaded gateway shares the expected QoS index, only if the mobile user in consideration is within the vicinity of it. The QoS index is representative of the interest of the gateway to take part in the auction-based load sharing process. The overloaded gateway measures the estimated QoS index based on its own knowledge and the information received from the CSP. Finally, the overloaded gateway verifies the estimated QoS index with the pre-specified expected value for including only the truthful gateways in the auction process. We, further, establish the existence of Nash Equilibrium (NE) in the proposed scheme. We theoretically analyze the upper and lower pricing limits, and prove that the algorithm reaches optimality. We extend the solution for multiple users' load increment in the presence of multiple misbehaving gateways. We summarize the main contributions of this paper as follows:

- We, theoretically, prove the requirement of load sharing in case of increase in demand by a user in MCN environment.
- We design a *utility maximization* problem for *quality-assured secured load sharing* (QuaLShare) in MCN with overloaded demand of a gateway.
- We analyze the *optimality criteria* and the existence of *Nash Equilibrium* for the proposed scheme, QuaLShare.
- We extend the solution for the multiple users' demand increment problem, followed by its theoretical analysis, for checking the optimality and Nash Equilibrium.

1.3 Paper Organization

The rest of the paper is organized as follows. In Section 2, we briefly present few relevant related works reported in the existing literature. In Section 3, we theoretically prove the necessity of the load sharing process along with the network model description. We present the utility maximization problem formulation followed by an auction-based solution approach for load sharing with single user in Section 4. In Section 5, we analyze the problem for the case of multiple users. We present the experimental results in Section 6. Finally, we conclude the paper in Section 7 with discussions about how this work can be extended in the future.

2 Related Work

Although in MCC mobile devices integrate with cloud for providing flexible, affordable, and capable service provisioning to the mobile users, the nascent integration faces major challenges due to finite power supply, low connectivity, presence of misbehaving users, and inefficient protocol design with respect to energy consumption, QoS provisioning, and resource allocation. Dinh et al. [7] and Fernando

TABLE 1
Different Resource Allocation Schemes and Respective Solutions with Parameter Metrics

Different schemes		Resource Type	Delay Metric	Misbehavior	Mobility	Distributed Algorithm	Local/Global Approach
AQUM: Mirsa et al. [1]		Bandwidth	✓	×	✓	×	Global
Stratus: Doyle et al. [2]		Data Center	✓	×	×	✓	Global
Zaman and Grosu [3]		Virtual Machine	×	✓[a]	×	✓	Global
Randles et al. [4]	Honeybee	Server	×	×	×	✓	Global
	Biased Random Sampling	Job	✓	×	×	✓	Global
	Active Clustering	System	×	×	×	✓	Global
Maguluri et al. [5]		CPU, Memory, Storage	✓	×	×	✓	Global
Grosu and Chronopoulos [6]		Computer	✓	✓[b]	×	✓	Global
QuaLShare (Our scheme)		Bandwidth	✓	✓	✓	✓	Local

[a]*User's misbehavior is considered in terms of truthfulness.*
[b]*User's selfishness is considered, whereas we consider the gateway's misbehaving characteristics.*

et al. [8] discussed about the subtleties behind enabling MCC. They highlighted several open issues such as security, resource management, network management, and quality of service provisioning. Sanaei et al. [9] described the present challenges and opportunities introduced by heterogeneity in MCC. Abolfazli et al. [10] indicated the existing issues on quality assurance in cloud based mobile devices. An energy-time efficient heterogeneous resource framework for hybrid mobile cloud computing has been studied by Sanaei et al. in [11]. A few recent works such as [12], [13], [14], and [15] emphasized on the existing security issues in cloud and mobile cloud environment. Such bare migration builds the motivational platform of our present work on secure load sharing in MCC.

Several works exist on load sharing and load balancing in other networks such as mobile computer network, homogeneous and heterogeneous system, and cognitive radio network. Othman and Hailes [16] discussed a power conservation strategy for mobile computers using load sharing. They reduced CPU power consumption by moving the jobs from a mobile host to a fixed host. Eager et al. [17] discussed a simple but adaptive load sharing policy for redistributing workload among users in homogeneous system. Another adaptive load sharing scheme for heterogeneous system was proposed by Mirchandaney et al. [18]. A game-theoretic framework was designed by Grosu and Chronopoulos [6] for obtaining a distributed, low complexity, and optimal load balancing scheme. Wang et al. [19] studied a resource allocation scheme with load balancing for maximizing the total data rate of all users in cognitive radio networks (CRNs). Wang et al. [20] proposed a novel auction-based approach for spectrum sharing in CRNs, in which the primary user (PU) allocates spectrum to the secondary users (SUs) based on the demand of the SUs, without violating its own performance. A stochastic differential game based dynamic bandwidth allocation problem was investigated by Zhu et al. [21] for both the SUs and the service providers (SPs). Guaranteed bandwidth allocation, which includes QoS requirement of the network components, was also proposed by Elias et al. [22]. Kun et al. [23] proposed a bandwidth allocation scheme for allocating equal amount of bandwidth to all users with the same type of service request. Zhang et al. [24] briefly summarized the different auction approaches for resource allocation in wireless systems. Although the existing works are similar to our present work, the former works do not consider mobility as well as security issue in their problem.

There exists very few works, specifically on load balancing in cloud networks. Nuaimi et al. [25] highlighted the existing challenges of load balancing and task scheduling in cloud computing. They compared the existing algorithms with respect to overall network load and time series. Randles et al. [4] studied three distributed approaches, namely Honeybee Foraging Behavior, Biased Random Sampling, and Active Clustering, for comparing different distributed load balancing schemes. Doyle et al. [2] proposed a scheme for scheduling incoming traffic to the nearest server to reach energy efficiency. They used Voronoi partitions for selecting low carbon emitted data center. A stochastic model of a cloud computing cluster was investigated by Maguluri et al. [5]. Zaman and Grosu [3] discussed an auction-based mechanism for dynamically allocating the virtual machines (VM). Papagianni et al. [26] proposed a unified resource allocation framework for cloud networks. Amamou et al. [27] discussed a service level agreement-aware dynamic bandwidth allocator (DBA), in which bandwidth is allocated among the VMs based on the application requirements. The mapping of cloud server and mobile users in MCC was studied by Das et al. [28]. A varying QoS requirement problem for different data incentive applications in cloud computing systems was studied by Lin et al. [29]. For distributing heterogeneous resources in mobile cloud systems, Yang et al. [30] proposed a combinatorial auction scheme. Wang et al. [31] introduced a double multi-attribute auction based resource allocation mechanism in cloud environment. However, the fundamental *differing characteristic* of the work presented in this paper from the existing ones is that we consider resource sharing in the presence of misbehaving users. Recently, Misra et al. [1] studied the bandwidth shifting and redistribution problems in a typical MCC environment. They studied proportional distribution of the available bandwidth among all the gateways in MCC. Therefore, the total cost for distribution is more as all the network gateways participate in the distribution process. A comparative study of the current state of the work is shown in Table 1.

Many security related works such as [15], [32], [33], [34], [35], and [13] also exist in a cloud computing environment. An anti-cheating bidding scheme was proposed by Hu et al. [12] for investigating the bidding strategy of bidders and finally securing the system from malicious bidders. Chen et al. [33] introduced three auction-based mechanisms for cheat-proof distributive spectrum allocation in cognitive radio based systems. However, none of these schemes addresses the problem of providing QoS-guaranteed load sharing in the presence of misbehaving users in MCN. In this paper, we propose a truthful auction-based QoS guaranteed load sharing scheme in MCN.

3 Mobile Cloud Network Model

Consider a simple mobile cloud environment with one CSP and I single channel gateways $\mathbf{G} = \{G_1, G_2, \ldots, G_I\}$ connected with the CSP through wireless channel. We further consider that each gateway G_i has K_i number of mobile users connected with it through a mobile network. Hence, $\mathbf{N}_i = \{N_{i1}, N_{i2}, \ldots, N_{iK_i}\}$, and, thus, $\mathbf{N} = \bigcup_{i=1}^{I} \mathbf{N}_i$. At this juncture, we should mention that, throughout the paper, the bold-faced characters represent the set of corresponding elements. Let us consider that the total allocated bandwidth vector for the gateways $\mathbf{G}$ is $\mathbf{B} = \{B_1, B_2, \ldots, B_I\}$. B_{tot} denotes the total available bandwidth of the CSP. If a mobile user requests any service from the CSP, the service is provided through the connecting gateway. We compute the service delay d_i of gateway G_i as follows [1]:

$$d_i = \frac{B_{tot} \sum_{k=1}^{|N_{K_i}|} T_{ik}}{E_i(1-\alpha_i)B_i}, \tag{1}$$

where T_{ik}, $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \ldots, \alpha_I\}$, and $\mathbf{E} = \{E_1, E_2, \ldots, E_I\}$ represent the ideal transmission delay required for accessing a service by the mobile node N_{ik}, if the entire B_{tot} is allocated to gateway G_i, the protocol overhead and the spectral efficiency of each wireless channel, respectively. In Table 2, we list few notations used in this paper.

TABLE 2
Summary of Notations

Notation	Description
G_i	Gateway i
I	Total number of gateways
N_{ik_i}	Set of users connected to a gateway G_i
B_i	Allocated bandwidth to a gateway G_i
B_{tot}	Total available bandwidth in CSP
d_i	Service delay associated with a gateway G_i
T_{ik}	Transmission delay required for accessing a service by a mobile user N_{ik} before load increment
$\widehat{T}_{ik}$	Transmission delay required for accessing a service by a mobile user N_{ik} after load increment
E_i	Spectral efficiency of a channel associated with G_i
α_i	Protocol overhead corresponding to a gateway G_i
U_i	Utility of gateway G_i
U_i^{init}	Utility of gateway G_i before load increment
r_i	Revenue per unit service delay received by G_i
q_i	Revenue per unit transmission delay received by G_i
ϑ_i	Payoff accumulated at G_i due to load sharing
d_{θ_i}	Threshold value of service delay corresponding to G_i
d_θ^{min}	Minimum delay threshold
d_θ^{max}	Maximum delay threshold
d_i^e	Expected service delay associated with a gateway G_i
$\bar{d}_i$	Expected QoS permissible index of gateway G_i
$\bar{d}_i^e$	Estimated QoS permissible index of gateway G_i
$\bar{d}_i^c$	Current QoS permissible index of gateway G_i
d_{ik}	Service delay occurred for user N_{ik}
Δp	Change in bid for single user load increment
H	Number of iteration
p	pricing value for single user load increment
$\overline{p}$	upper pricing limit for single user load increment
$\underline{p}$	lower pricing limit for single user load increment
$\Delta\mathcal{P}$	Change in bid for multiple users load increment
$\mathcal{P}$	pricing value for multiple users load increment
$\overline{\mathcal{P}}$	upper pricing limit for multiple users load increment
$\underline{\mathcal{P}}$	lower pricing limit for multiple users load increment
n	rate parameter
γ	Fractional constant
$\mathbf{G}_{j^*}$	Winner Gateway for S-QuaLShare
$\mathbf{G}^*$	Winner Gateway set for M-QuaLShare
L_{il}	Position of the user N_{il}

3.1 Necessity of Load Sharing

In MCN, the load of a gateway G_i increases when either a connected user increases its demand or a mobile user changes its connecting gateway from G_j to G_i due to location change. Therefore, the service delay of the gateway G_i is affected. For maintaining the QoS in terms of service delay, the gateway G_i calculates the total transmission delay for all the connected mobile users. In Theorem 1, we prove that the service delay differs with the heterogeneous service demand.

Theorem 1. *Let us consider that a mobile user N_{il} modifies its service demand. The service delay of the connecting gateway G_i before the demand change is unequal to that after the demand change, because of the variation in total transmission delay of G_i.*

Proof. We consider the case in which only one mobile user, say N_{il}, of the gateway G_i changes its service demand. Therefore, the transmission delay for the mobile user varies from T_{il} to $\widehat{T}_{il}$. We represent the service delay computation before and after the service demand change by $d_i(t_1)$ and $d_i(t_2)$, respectively. Using Equation (1), we compute $d_i(t_1)$ as follows:

$$d_i(t_1) = \frac{B_{tot}[\sum_{k=1,k\neq l}^{|N_i|} T_{ik} + T_{il}]}{E_i(1-\alpha_i)B_i}. \tag{2}$$

Similarly, we compute $d_i(t_2)$ as follows:

$$d_i(t_2) = \frac{B_{tot}[\sum_{k=1,k\neq l}^{|N_i|} T_{ik} + \widehat{T}_{il}]}{E_i(1-\alpha_i)B_i}. \tag{3}$$

As the values of T_{il} and $\widehat{T}_{il}$ are different, it can be inferred that,

$$d_i(t_1) \neq d_i(t_2). \tag{4}$$

This concludes the proof. □

Depending upon the transmission delay variation, the following scenarios occur:

Case I: We have $d_i(t_1) \geq d_i(t_2)$, if $T_{il} \geq \widehat{T}_{il}$. It means that QoS is not violated in terms of service delay.

Case II: We have $d_i(t_1) < d_i(t_2)$, if $T_{il} < \widehat{T}_{il}$. In this case, the decision of QoS is taken by the value of service delay threshold d_{θ_i} of the gateway G_i as follows:

Case IIa: The QoS provisioning by the gateway G_i is not violated if $d_i(t_2) \leq d_{\theta_i}$.

Case IIb: The QoS provisioning by the gateway G_i is violated if $d_i(t_2) > d_{\theta_i}$.

From Theorem 1, we observe that the QoS provisioning by a gateway fails when the modified service delay exceeds the service delay threshold. This necessitates the requirement of ensuring load sharing. Although the bandwidth distribution approach proposed by Misra et al. [1] is one of the useful remedies for the above problem, it involves all the gateways in decision making, and is thus, computationally expensive, as explained in Section 1. Unlike bandwidth distribution, the load sharing approach results in low overhead, as the solution approach does not involve all the gateways until and unless it is required.

4 Quality-Assured Secured Load Sharing for Single User Load Increment

In this section, we consider the case of single user load increment, whereas the consideration of multiple user load increment is discussed in Section 5. We formulate the utility function of the gateway for designing an optimal solution of the load sharing problem under single user load increment. Thereafter, we present an auction theory-based load sharing scheme, namely *Quality-assured Secured Load Sharing* (*QuaLShare*), followed by the theoretical analysis of the QuaLShare approach. Henceforth, throughout the paper, we use the prefixes 'S' and 'M' along with the name QuaLShare for differentiating the proposed solution under the single user and multiple users load increment, respectively.

4.1 Utility Function

We design a utility function for computing the overall payoff of a gateway. The utility function depends on both the transmission and service delays. The transmission delay specific revenue of a gateway is calculated for measuring the amount of service which is provided to the mobile users. The service delay specific revenue is computed for measuring the quality of service which is provided to the connecting mobile users. In addition to that, the utility function considers the payoff a gateway gains or loses due to load sharing. Considering the above three factors, the utility function of a gateway can be expressed as follows:

$$\mathbf{U} = \mathbf{R}^T(\mathbf{T}) + \mathbf{R}^Q(\mathbf{d}) + \vartheta, \tag{5}$$

where $\mathbf{d} = \{d_1, d_2, \ldots, d_I\}$ denotes the service delay vector and $\mathbf{T} = \{T_1, T_2, \ldots, T_I\}$ represents the transmission delay vector of the interfacing gateway. $\mathbf{R}^T(\mathbf{T})$ and $\mathbf{R}^Q(\mathbf{d})$ define the transmission delay-specific revenue function and the service delay-specific revenue functions, respectively, and ϑ denotes the amount of payoff a gateway receives from the load sharing solution. We model the three utility components $\mathbf{R}^T(\mathbf{T})$, $\mathbf{R}^Q(\mathbf{d})$, and ϑ as follows.

1) *Transmission delay specific revenue function modeling*: The increase in transmission delay increases the revenue of a gateway. So, we formulate the transmission delay specific revenue function as follows:

$$\mathbf{R}^T(\mathbf{T}) = \mathbf{q}\mathbf{T}, \tag{6}$$

where the vector $\mathbf{q} = \{q_1, q_2, \ldots, q_I\}$ denotes the revenue per unit transmission delay.

2) *Service delay specific revenue function modeling*: The service delay of a gateway is inversely proportional to the revenue of the gateway. Hence, we formulate the service delay-specific revenue function as follows:

$$\mathbf{R}^Q(\mathbf{d}) = (\mathbf{d}_\theta - \mathbf{d})\mathbf{r}, \tag{7}$$

where the vector $\mathbf{r} = \{r_1, r_2, \ldots, r_I\}$ defines the revenue per unit service delay.

3) *Load Sharing specific payoff modeling*: The vector $\vartheta = \{\vartheta_1, \vartheta_2, \ldots, \vartheta_I\}$ defines the payoff which is accumulated during the load sharing process. If ϑ_i equals zero, the connecting gateway only serves its own service request demand. The positive ϑ_i value indicates that the gateway is overloaded, and it presently shares excess load with its neighboring gateways. On the contrary, the negative ϑ_i denotes that the gateway serves the demands of its neighboring gateways.

Hence, the utility of a gateway G_i is defined as:

$$U_i = q_i \left[\sum_{k=1, k \neq l}^{|N_{ik_i}|} T_{ik} + \widehat{T}_{il} \right] + r_i(d_\theta - d_i) - \vartheta_i. \tag{8}$$

Finally, the utility maximization problem is expressed as follows:

$$\begin{aligned} maximize \quad & U_i \\ subject\ to \quad & d_i \leq d_{\theta_i}, i \in (1, I). \end{aligned} \tag{9}$$

4.2 S-QuaLShare Algorithm

Auction theory is a well-known procedure for purchasing and selling resources by offering prices according to bidding value. Therefore, we use an auction theory-based approach for designing the S-QuaLShare algorithm in MCN environment. In the S-QuaLShare algorithm, the overloaded gateway participates as a seller cum auctioneer, and the neighbors of the overloaded gateway act as buyers. We use the auction process for selecting such a truthful buyer, who maximizes the utility of the overloaded gateway among all the neighbors. We describe the basic steps of the S-QuaLShare algorithm as follows.

1) *Broadcast Pre-requisites*: We consider that, each gateway G_i knows the location information of all the users who are currently connected with it. The gateway G_i computes the transmission delay according to a user's demand. Initially, the overloaded gateway G_i broadcasts the following information of user N_{il} – transmission delay $\widehat{T}_{il}$ and location L_{il}.
2) *Expected value computation of QoS permissible index*: Each neighboring gateway $G_j, j \in Ne(G_i)$ computes

the expected delay d_j^e followed by the expected QoS permissible index $\bar{d}_j$ based on the transmission and service delays of all mobile users connected with it. The expressions of d_j^e and $\bar{d}_j$ are shown as follows:

$$d_j^e = \frac{B_{tot}[\sum_{k=1}^{|N_{K_j}|} T_{jk} + \widehat{T}_{il}]}{E_j(1-\alpha_j)B_j} \tag{10}$$

$$\bar{d}_j = \frac{d_j^e}{d_{\theta_j}}. \tag{11}$$

Finally, the gateway G_j shares $\bar{d}_j$ to the overloaded gateway G_i if the user N_{il} is within the proximity region of it.

3) *Runtime estimation of QoS permissible index*: The overloaded gateway G_i estimates the QoS permissible index $\bar{d}_j^e$ for a participating gateway G_j, depending on the present QoS permissible index $\bar{d}_j^c$, minimum delay threshold d_θ^{min}, and the service delay d_{il} that occurred for N_{il}. So,

$$\bar{d}_j^e = \bar{d}_j^c + \frac{d_{il}}{d_\theta^{min}}. \tag{12}$$

In this computation, the gateway G_i receives $\bar{d}_j^c$ and d_θ^{min} related to the overloaded gateway from the CSP.

4) *Misbehavior detection*: The overloaded gateway G_i compares the expected and estimated QoS permissible index for identifying misbehaving gateways. The gateway G_i accepts the interested gateways, as participants in the auction process, whose value of the above difference lies within ϵ. Remaining gateways are excluded from the process.

5) *Determine bid value*: The overloaded gateway submits a bid value p, which represents the minimum price for sharing the load at the initial stage. In every iteration, the bid balue is incremented by some positive quantity, Δp, as follows:

$$p(H+1) = p(H) + \Delta p. \tag{13}$$

G_i increments its bid value until its current value of utility exceeds the initial value.

6) *Bid selection*: The participating gateways accept the given bid value if and only if the expected value of future payoff is less than the current payoff value.

7) *Load allocation*: G_i allocates the user N_{il} to the gateway G_j^* who has accepted the bid. If multiple gateways accept the bid in the same iteration cycle, G_i assigns the user to the gateway G_j^* with minimum $\bar{d}_j^e$ as lesser QoS index denotes better service. Otherwise, we conclude that load sharing is not sufficient for providing services with adequate QoS guarantee. Henceforth, the CSP invokes the bandwidth distribution algorithm [1] for redistributing the bandwidth such that the modified distribution neutralizes the effects of QoS violation due to load increment.

8) *Payoff calculation*: The winning gateway G_{j^*} gets the benefit value $(p(H) - H^n\widehat{T}_{il})$ from the overloaded gateway G_i. Each loser gateway $G_{j\neq j^*}$ gets no benefit. Therefore, ϑ_i and ϑ_j^* are represented as follows:

$$\vartheta_i = p(H) - H^n\widehat{T}_{il} \tag{14}$$

$$\vartheta_j^* = -p(H) + H^n\widehat{T}_{il}. \tag{15}$$

The pseudo-code of the S-QuaLShare scheme is presented in Algorithm 1.

Algorithm 1. S-QuaLShare Algorithm

Inputs: $N_{il}(\widehat{T}_{il}, L_{il})$
Output: G_{j^*}

```
Invoke Misbehavior Detection algorithm for single user share as shown in Algorithm 2
if U_i(H+1) − U_i^init < 0 then
    if user not assign then
        Bandwidth Distribution required.
    end if
else
    if (max(d̄_j, d̄_j^e) ≤ 1) then
    G_i shares the bid p with the participant
        if any G_j* accepts the price then
        Assign the user to G_j*
        else
            if multiple gateways accept the price then
            Assign the user to G_j* with minimum d̄_j^e
            G_i pays ϑ_i to the assignee gateway
            else
            Revise the bid price p(H+1) = p(H) + Δp
            Goto Step 2
            end if
        end if
    end if
end if
```

Algorithm 2. Misbehavior Detection for Single User Share

Inputs: $N_{il}(\widehat{T}_{il}, L_{il})$
Output: G_{j^*}

```
G_i broadcasts N_il(T̂_il, L_il)
Each gateway G_j, j ∈ Ne(G_i) calculates d̄_j
G_j, j ∈ Ne(G_i) shares d̄_j if within proximity
Gateway G_i calculates d̄_j^e
if |d̄_j^e − d̄_j| > ϵ then
    Exclude misbehaving gateways from auction
end if
```

4.3 Pricing Limit

In this section, we determine the upper and lower pricing limits in the S-QuaLShare algorithm, as shown in Theorem 2. We further conclude from the theorem that any gateway does not accept the bid value in the auction process if the price is less than the lower pricing limit $\underline{p}$. Similarly, if the bid price is more than the upper pricing limit $\overline{p}$, the overloaded gateway does not execute the sharing process.

Theorem 2. *The upper and lower pricing limits are* $\bar{p}_i = (q_i + H^n)\widehat{T}_{il} - q_i T_{il}$ *and* $\underline{p}_i = [max(\bar{d}_j - \bar{d}_j^c)]d_\theta^{max} r_i$, *respectively.*

Proof. The overloaded gateway G_i shares the load of a mobile user only if it does not lose from sharing. It implies that $U_i - U_i^{init} \geq 0$, where $U_i = q_i \sum_{k=1,k\neq l}^{|N_{ik_i}|} T_{ik} + q_i\widehat{T}_{il} + r_i(d_{\theta_i} - d_i) - p_i + H^n\widehat{T}_{il}$ and $U_i^{init} = q_i \sum_{k=1,k\neq l}^{|N_{ik_i}|} T_{ik} + q_i T_{il} + r_i(d_{\theta_i} - d_i)$. So,

$$U_i - U_i^{init} \geq 0 \Rightarrow (q_i + H^n)\widehat{T}_{il} - q_i T_{il} \geq p. \tag{16}$$

From this inequality, we determine the upper bound of the pricing limit as follows:

$$\bar{p}_i = (q_i + H^n)\widehat{T}_{il} - q_i T_{il}. \tag{17}$$

Similarly, a gateway accepts allocation through the auction process if its utility is not negatively effected by the new sharing. The service delay of the gateway is increased due to the user sharing process. For including all the gateways in the auction process, we consider the maximum service delay increment equal to $[max(\bar{d}_j - \bar{d}_j^c)]d_\theta^{max}$, where $\bar{d}_j^c$ and d_θ^{max} denote the current QoS permissible index of the gateway G_j and the maximum delay threshold among the gateways, respectively. Now, we compute the lower price limit as follows:

$$\underline{p}_i = [max(\bar{d}_j - \bar{d}_j^c)]d_\theta^{max} r_i. \tag{18}$$

This concludes the proof. □

4.4 Nash Equilibrium

We investigate the existence of Nash Equilibrium of the S-QuaLShare algorithm, which is stated by the following theorem using the rate parameter n.

Theorem 3. *There exists Nash Equilibrium in the S-QuaLShare algorithm if and only if* $n\widehat{T}_{il}(\bar{p}_i - \underline{p}_i)^{n-1} = (\Delta p)^n$.

Proof. A neighboring gateway accepts the bid value only if the final payoff is higher than that in the previous iteration of the auction process. Let us assume that the bidding price is increased uniformly by Δp amount in each iteration H. Using the first and second derivatives of the final payoff function ϑ_i with respect to H, we determine the maximization condition, as follows:

$$H^{n-1} = \frac{\Delta p}{n\widehat{T}_{il}}. \tag{19}$$

If the maximum payoff value exists within the pricing range (shown in Theorem 2), the utility of all the gateways increases. Otherwise, the final utility of the overloaded gateway or the sharing gateway reduces. Therefore, we conclude that there exists Nash Equilibrium at the maximum payoff point, which depends on the rate parameter n. We, finally, measure the value of n using heuristic analysis. In other words, for the existence of Nash Equilibrium, the upper pricing limit $\bar{p}_i$ must be greater than or equal to the final bidding value ($\underline{p}_i + \Delta pH$). Hence, we obtain the necessary condition for the existence of Nash Equilibrium, as follows:

$$n\widehat{T}_{il}(\bar{p}_i - \underline{p}_i)^{n-1} = (\Delta p)^n. \tag{20}$$

This concludes the proof. □

5 Quality-Assured Secured Load Sharing for Multiple Users Load Increment

In this section, we extend the S-QuaLShare solution for multiple users load increment. We modify the utility function, and propose an auction theory-based load sharing scheme, namely *Quality-assured Secured Load Sharing for Multiple Users Load Increment* (*M-QuaLShare*).

5.1 Utility Function

We modify the utility function for computing total payoff of all the gateways due to multiple users' demand increment. Similar to the case of single user demand increment, the utility of the overloaded gateway, say G_i, is computed depending on the transmission delay, service delay, and payoff value, as described in Equation (8). The transmission and service delay-specific revenue functions are shown in Equations (6) and (7), respectively. Finally, we construct a utility maximization problem using the utility functions of the overloaded and the underloaded gateways. The utility maximization problem for all the gateways is expressed as follows:

$$\begin{aligned} maximize \quad & \mathbf{U} \\ subject\ to \quad & \mathbf{d} \leq \mathbf{d}_\theta. \end{aligned} \tag{21}$$

5.2 M-QuaLShare Algorithm Design

We adopt the merely identical auction scheme for sharing the load of multiple users among the neighboring gateways. The overloaded gateway G_i acts as an auctioneer cum bidder. The neighbors of G_i are treated as the consumer, and the loads are represented as the selling objects. The auction process picks up the reliable consumers among all the neighbors of G_i. The elemental steps of the M-QuaLShare algorithm are illustrated as follows:

1) *Broadcast pre-requisite*: We assume that each gateway G_i is aware of its own proximity region as well as the positions of all the connected mobile users. During the time of connectivity establishment, each gateway measures the transmission delay according to the user's demand. The overloaded gateway G_i starts the auction process by broadcasting the following details—$\mathbf{N} = \{N_{il_1}(\widehat{T}_{il_1}, L_{il_1}), N_{il_2}(\widehat{T}_{il_2}, L_{il_2}), \ldots, N_{il_x}(\widehat{T}_{il_x}, L_{il_x})\}$.
2) *Expected value computation of QoS permissible index*: Each neighbor $G_j, j \in Ne(G_i)$ measures the expected delay vector $\mathbf{d}_j^e = \{d_{jl_1}^e, d_{jl_2}^e, \ldots, d_{jl_x}^e\}$ and the expected QoS permissible index vector $\bar{\mathbf{d}}_j = \{\bar{d}_{jl_1}, \bar{d}_{jl_2}, \ldots, \bar{d}_{jl_x}\}$ for all the users using Equations (10) and (11), respectively. If any user is not in the proximity region of

$G_j, j \in Ne(G_i)$, the gateway G_j replaces the expected QoS permissible index for that user by $\chi(>1)$. Finally, each G_j shares $\bar{\mathbf{d}}_j$ with the gateway G_i, if at least one user is within the proximity region of it.

3) *Runtime estimation of QOS permissible index*: The gateway G_i estimates the QoS permissible index vector for all the participated gateway-user pair using Equation (12). Hence, we have,

$$\bar{\mathbf{d}}_j^e = \begin{bmatrix} \bar{d}^e_{1l_1} & \bar{d}^e_{1l_2} & \cdots & \bar{d}^e_{1l_x} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{d}^e_{Il_1} & \bar{d}^e_{Il_2} & \cdots & \bar{d}^e_{Il_x} \end{bmatrix}. \tag{22}$$

If the expected QoS permissible index of any gateway-user pair is greater than unity, G_i updates the estimate of QoS permissible index for that pair by $\chi(>1)$ value.

4) *Misbehavior detection*: The overloaded gateway G_i compares the estimated QoS permissible index with the expected QoS permissible index for each interested gateway-user pair. Finally, the gateway G_i selects the neighboring gateways as participants, who have the value of QoS index difference less than ϵ.

5) *Determine bid value*: At the starting phase, the overloaded gateway G_i submits the bid value $\mathcal{P}$, which denotes the minimum price per unit transmission time for sharing the load. During the auction process, it increments the bid value by a positive quantity $\Delta\mathcal{P}$, as follows:

$$\mathcal{P}(H+1) = \mathcal{P}(H) + \Delta\mathcal{P}. \tag{23}$$

The overloaded gateway G_i attracts the participants by increasing the bid value until the current utility reaches the initial utility value.

6) *Bid selection*: Any gateway G_j accepts the submitted bid value if the expected total payoff in the next iteration is lesser than the present total payoff.

7) *Load allocation*: The gateway G_i, initially, sorts the users, based on the required transmission time. From the list of gateways that accepts the submitted bid, a gateway G_j^* with minimum value of $\bar{d}^e_{jl_k}$ for the maximum transmission time $\widehat{T}_{il_k}$ in the list $\mathbf{T}_i$ is selected for temporarily reserving the user. The motivation behind such type of selection is to overcome the scenario where the total available resources of the neighbor gateways are sufficient to provide the services, but the individual ones are not. The temporary allocation process for the gateway G_j^* is continued until the users' transmission time list becomes empty. Likewise, the gateway G_i further continues the temporary reservation process until it temporarily reserve all the users, or the list of participating gateways becomes empty. Finally, the gateway G_i permanently assigns the users to the respective winner gateways $\mathbf{G}^*$ only if all the users are temporarily reserved. Otherwise, the process terminates unsuccessfully stating that the load sharing process is not sufficient to share the load among the neighbors due to the insufficiency of the resources.

8) *Payoff calculation*: The overloaded gateway pays the amount $(\mathcal{P}(H) - \gamma H^n)\sum \widehat{T}_{il_k}$ to the winning gateways $\mathbf{G}^*$, where $\sum \widehat{T}_{il_k}$ represents the total transmission time corresponding to each winning gateway, and γ is a constant value $(0 < \gamma < 1)$. The failed gateways do not get any benefit from the auction process. Hence, we determine ϑ_i and ϑ_j^* as follows:

$$\vartheta_i = (\mathcal{P}(H) - \gamma H^n)\widehat{\mathbf{T}}_i \tag{24}$$

$$\vartheta_j^* = -(\mathcal{P}(H) - \gamma H^n)\sum \widehat{T}_{il_k}. \tag{25}$$

The pseudo-code of the M-QuaLShare scheme is presented in Algorithm 3.

Algorithm 3. M-QuaLShare Algorithm

Inputs: $\mathbf{N} = \{N_{il_1}(\widehat{T}_{il_1}, L_{il_1}), \ldots, N_{il_x}(\widehat{T}_{il_x}, L_{il_x})\}$
Output: $\mathbf{G}^*$

Invoke Misbehavior detection algorithm for multiple users share, as shown in Algorithm 4
if $U_i(H+1) - U_i^{init} < 0$ **then**
  if all users not assigned **then**
    Bandwidth Distribution required.
  else
    if all users temporarily assigned **then**
      Assign all users and pay accordingly
    end if
  end if
else
  G_i shares the bid p with the participants
  if multiple gateways accept the price **then**
    Sort $\mathbf{T}_i$ of the user in descending order
    Call Temporary Allocation (Algorithm 5) for the gateway G_j with minimum $\bar{d}^e_{jl_k}$
    Repeat from Step 14 until all users temporarily assign or empty participants list
  else
    if any one G_j accepts the price **then**
      Sort $\mathbf{T}_i$ of the user in descending order
      Call Temporary Allocation (Algorithm 5)
    else
      Revise the bid price $\mathcal{P}(H+1) = \mathcal{P}(H) + \Delta\mathcal{P}$
      Goto Step 2
    end if
  end if
end if

Algorithm 4. Misbehavior Detection for Multiple Users Share

Inputs: $\mathbf{N} = \{N_{il_1}(\widehat{T}_{il_1}, L_{il_1}), \ldots, N_{il_x}(\widehat{T}_{il_x}, L_{il_x})\}$
Output: $\mathbf{G}^*$

G_i broadcasts $\mathbf{N}$
Each participant $G_j, j \in Ne(G_i)$ calculates $\bar{\mathbf{d}}_j$
$G_j, j \in Ne(G_i)$ shares $\bar{\mathbf{d}}_j$ if any N_{il_k} is within range
Gateway G_i calculates $\bar{\mathbf{d}}_j^e$
if $|\bar{d}^e_{jl_k} - \bar{d}_{jl_k}| > \epsilon$ **then**
  Exclude misbehaving gateway from auction process
end if

Algorithm 5. Temporary Allocation

Inputs: $\mathbf{N} = \{N_{il_1}(\widehat{T}_{il_1}, L_{il_1}), \ldots, N_{il_x}(\widehat{T}_{il_x}, L_{il_x})\}$
Output: $\mathbf{G}^*$
1: **if** $(max(\bar{d}_{jl_k}, \bar{d}^e_{jl_k}) \leq 1)$ **then**
2: Temporarily reserve the user for G^*_j
3: Reserve price ϑ^*_j
4: Exclude $\widehat{T}_{il_k}$ from $\mathbf{T}_i$
5: **end if**
6: Move to next transmission delay value in list $\mathbf{T}_i$
7: Repeat from Step 1 until the list $\mathbf{T}_i$ is moved to the end

5.3 Pricing Limit

In this section, we determine the upper and lower pricing limits in the M-QuaLShare algorithm, as shown in Theorem 4. We further conclude from the theorem that any gateway does not accept the multiple load share auction process, if the price is less than $\underline{\mathcal{P}}$. On the other hand, if the price is more than $\overline{\mathcal{P}}$, the overloaded gateway does not organize the multiple load sharing process.

Theorem 4. *The upper and lower pricing limit per unit transmission are represented as follows:*

$$\bar{\mathcal{P}}_i = (q_i + \gamma H^n) - q_i \frac{\sum_{k=1}^{x} T_{il_k}}{\sum_{k=1}^{x} \widehat{T}_{il_k}}$$

$$\underline{\mathcal{P}}_i = \frac{r_i d_\theta^{max}}{\sum_{k=1}^{x} \widehat{T}_{il_k}} \sum_{k=1}^{x} [max(\bar{d}_{jl_k} - \bar{d}^c_j)].$$

Proof. The overloaded gateway G_i shares the load of mobile users if and only if the gateway gets benefit from sharing. This implies that $U_i - U_i^{init} \geq 0$, where $U_i = q_i \sum_{h \neq l} T_{ih} + q_i \sum_{k=1}^{x} \widehat{T}_{il_k} + r_i(d_{\theta_i} - d_i) - (\mathcal{P} - \gamma H^n) \sum_{k=1}^{x} \widehat{T}_{il_k}$ and $U_i^{init} = q_i \sum_{h \neq l} T_{ih} + q_i \sum_{k=1}^{x} T_{il_k} + r_i(d_{\theta_i} - d_i)$. Hence, we have,

$$U_i - U_i^{init} \geq 0$$
$$\Rightarrow \frac{1}{\sum_{k=1}^{x} \widehat{T}_{il_k}} \left[q_i \sum_{k=1}^{x} \widehat{T}_{il_k} + \gamma H^n \sum_{k=1}^{x} \widehat{T}_{il_k} - q_i \sum_{k=1}^{x} T_{il_k} \right] \geq \mathcal{P}.$$

From this inequality, we determine the upper bound of the pricing limit as follows:

$$\overline{\mathcal{P}}_i = (q_i + \gamma H^n) - q_i \frac{\sum_{k=1}^{x} T_{il_k}}{\sum_{k=1}^{x} \widehat{T}_{il_k}}. \tag{26}$$

Likewise, a gateway takes part in the auction process if its utility is increased by new sharing. The gateway serves for the additional users due to load sharing. Therefore, the service delay of the gateway is increased. For incorporating all the gateways in the auction process, we consider that the average maximum service delay increment equals $\frac{d_\theta^{max}}{\sum_{k=1}^{x} \widehat{T}_{il_k}} \sum_{k=1}^{x} [max(\bar{d}_{jl_k} - \bar{d}^c_j)]$, where $\bar{d}^c_j$, d_θ^{max}, and $\sum_{k=1}^{x} \widehat{T}_{il_k}$ denote the current QoS permissible index of the gateway G_j, the maximum delay threshold among the gateways, and total transmission time of the load improved users, respectively. Now, we computes the lower price limit as follows:

$$\underline{\mathcal{P}}_i = \frac{r_i d_\theta^{max}}{\sum_{k=1}^{x} \widehat{T}_{il_k}} \sum_{k=1}^{x} [max(\bar{d}_{jl_k} - \bar{d}^c_j)]. \tag{27}$$

This concludes the proof. □

5.4 Nash Equilibrium

We investigate the existence of Nash Equilibrium of the M-QuaLShare algorithm, which is stated by the following theorem using the rate parameter n.

Theorem 5. *There exists Nash Equilibrium in the M-QuaLShare algorithm, if and only if* $n\gamma(\overline{\mathcal{P}}_i - \underline{\mathcal{P}}_i)^{n-1} = (\Delta\mathcal{P})^n$.

Proof. The neighboring gateways participate in the load sharing process only if the final payoff increases with respect to that in the previous iteration cycle. We consider that the bidding price is uniformly increased by Δp amount in each iteration H. Using the first and second derivatives of the final payoff function ϑ_i with respect to H, we determine the maximization condition as follows:

$$H^{n-1} = \frac{\Delta\mathcal{P}}{n\gamma}. \tag{28}$$

If the maximum payoff value exists within the pricing range (shown in Theorem 4), the utilities of all the gateways are increased. Otherwise, the final utility of the overloaded gateway (or the sharing gateways) is decreased. Therefore, we conclude that there exists Nash Equilibrium at the maximum payoff point, which depends on the rate parameter n. Finally, we derive the value of n using heuristic analysis. In other words, the upper pricing limit $\overline{\mathcal{P}}_i$ must be greater than or equal to the final bidding value $\underline{\mathcal{P}}_i + \Delta\mathcal{P}H$ for the existence of Nash Equilibrium. Hence, we obtain the necessary condition for the existence of Nash Equilibrium as follows:

$$n\gamma(\overline{\mathcal{P}}_i - \underline{\mathcal{P}}_i)^{n-1} = (\Delta\mathcal{P})^n. \tag{29}$$

This concludes the proof. □

6 Numerical Results

In this section, we illustrate numerical results of the proposed quality-assured secured load sharing algorithms in an MCN environment. At the outset, we describe an mobile cloud network scenario followed by the parameter settings. We prove the necessity of load sharing followed by bandwidth allocation to the gateways. We also show the optimality and Nash Equilibrium of both the load sharing algorithms for a single and multiple users' load increment in the presence of a misbehaving gateway. We executed each experiment 30 times, and the ensemble average values are plotted with 95 percent confidence interval. We use the *AQUM* algorithm proposed by Misra et al. [1] as a benchmark for comparing the quality of service and bandwidth allocation of the gateways in the load sharing scenario.

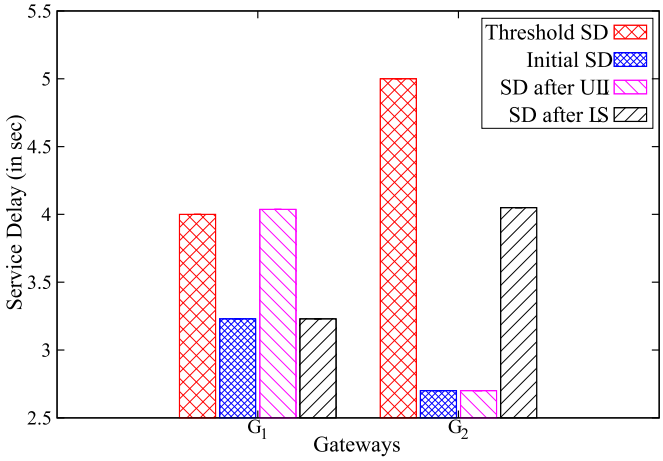

Fig. 2. Load sharing for single user load increment with overloaded gateway G_1 and winner gateway G_2.

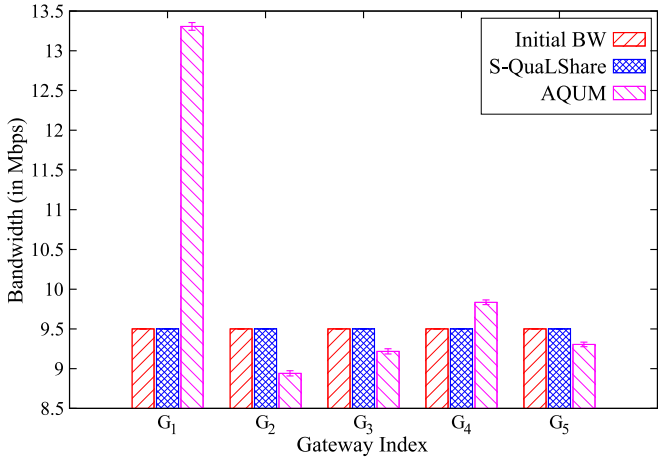

Fig. 3. Bandwidth allocation comparison between AQUM and S-QuaLShare.

6.1 Parameter Settings

Let us consider an MCN environment, as shown in Fig. 1, with one CSP and 10 gateways $(G_1, G_2, \ldots, G_{10})$. In MCN, the CSP dynamically allocates bandwidth to the gateways according to their demands for providing resource expensive services to the mobile users through the gateways. Each gateway G_i has four connected mobile users $(N_{i1}, N_{i2}, N_{i3}, N_{i4})$. We consider the revenue per unit transmission $q_i = 30, \forall i \in I$, revenue per unit service delay $r_i = 1, \forall i \in I$, and total available bandwidth $B_{tot} = 100$ Mbps. Throughout the experiments, we initially consider that the ideal transmission time for each mobile user equals 0.1 sec and the fixed SNR equals 10 dB. BER = $\{10^{-5}, 10^{-4}, 5 \times 10^{-5}, 10^{-5}, 5 \times 10^{-5}, 10^{-5}, 10^{-4}, 5 \times 10^{-5}, 10^{-5}, 5 \times 10^{-5}\}$ and $\boldsymbol{\alpha} = \{0.02, 0.008, 0.008, 0.008, 0.02, \quad 0.02, 0.008, 0.008, 0.008, 0.02\}$ represent the target BER and protocol overhead of all the gateways. The E_i of all the gateways are computed based on the values of the target BER and the SNR using the Equation (3) in [1].

In all the experiments, we consider that the gateway G_1 suffers from excess service load. Therefore, the gateway G_1 executes the load sharing process with the existing four neighboring gateways G_2, G_3, G_4 and G_5. The delay thresholds of all the gateways are $\{4, 5, 5, 5, 5\}$ sec. We further assume that, in the initial stage, the bandwidth is equally distributed among the gateways. Each gateway is allocated 9.5 Mbps bandwidth. Initially, the payoff of all the gateways is zero, as no one shares other's load.

The notations LS, SD_{th}, ULI, and SD, adopted in the legends of the figures, represent the cases of load sharing, service delay threshold, user load increment, and service delay, respectively.

6.2 Single User Load Increment

In all the experiments of single user load increment, we consider that the mobile user N_{11} connected with the gateway G_1 increases its demand. Consequently, the ideal transmission time for the modified demand increases from 0.1 to 0.2 sec. We further consider that the mobile user N_{11} is also within the vicinity of all the neighbors of G_1, so that all the neighbors participate in the auction process.

6.2.1 *Single User Load Sharing*

Fig. 2 establishes the necessity of load sharing due to single user load increment. Whenever the user N_{11} increases its demand, the ideal transmission time for providing a service increases. Consequently, the service delay of the connecting gateway G_1 increases, which ultimately crosses the threshold value. The service delay of the winner gateway G_2 remains undisturbed after the load increment of N_{11}. Finally, the service delay of G_2 increases after executing the single user load sharing process. Furthermore, it is evident that load sharing is sufficient for assuring service delay guarantee in G_1 as well as the winner gateway G_2, while the others remain unchanged.

6.2.2 *Bandwidth Distribution*

We compared the final bandwidth distribution among the gateways for the proposed single user load sharing algorithm and the existing bandwidth distribution algorithm, AQUM. We plotted the results in Fig. 3. In this experiment, we use the default settings for load sharing. For the AQUM algorithm, we adopt the similar data settings except that the additional shifting parameter is set 10.0 for all the gateways. In this figure, we observe that the bandwidth distribution remains undisturbed for the single user load sharing algorithm, whereas the bandwidth distribution of all the 10 gateways completely differ in the case of the AQUM algorithm. The CSP allocates more bandwidth to the gateway G_1, as it gets additional service request from its down-link user. Hence, we conclude that the gateways try to resolve the additional service request among themselves in the single user load sharing process, so that additional bandwidth is not required. Thus, our proposed algorithm does not suffer from any additional overhead due to bandwidth management.

We further compared the service delay of all the gateways for both the algorithms. Fig. 4 shows that only for the winner gateway G_2, the service delay increases in the single user load sharing process, while the others remain unaltered. Using the AQUM algorithm, the service delay of all the gateways change due to the varying bandwidth distribution. Hence, our proposed algorithm is similar to AQUM, and maintains the service delay of all the gateways within

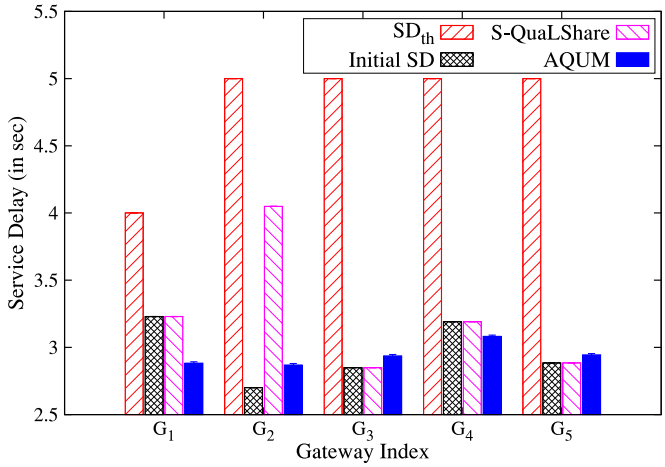

Fig. 4. Service delay comparison between AQUM and S-QuaLShare.

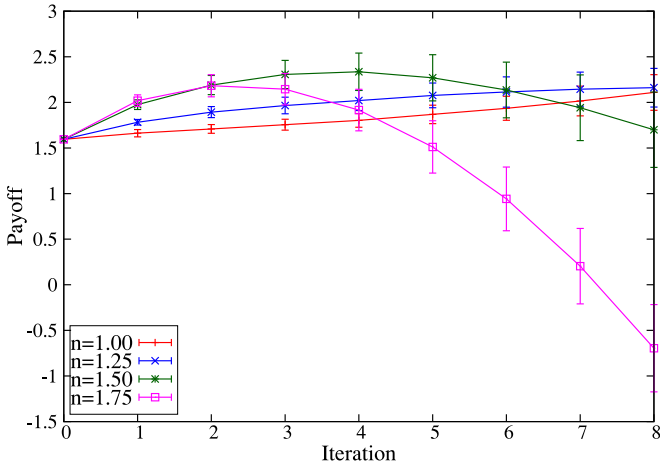

Fig. 6. Nash equilibrium of S-QuaLShare algorithm.

the service delay threshold value. In the S-QuaLShare algorithm, only the neighbors of the overloaded gateway participate. Hence, the gateways except the neighbors of the overloaded one remain isolated from the load sharing process and provide uninterrupted services to their associated mobile users, while the overloaded gateway involves in the load sharing process. Additionally, the proposed algorithm includes only the neighbors of the overloaded gateway with sufficient bandwidth. Thus, only a limited part of the MCC system is involved in the S-QuaLShare algorithm, whereas AQUM includes the entire MCC system. Hence, we conclude that using the proposed solution, the performance of S-QuaLShare is better than AQUM, due to the lower bandwidth management overhead and limited participation of the gateways.

6.2.3 Optimality

We test the optimality criteria of the single user load sharing algorithm. We performed this test with the default parameter settings. Fig. 5 shows that the payoff initially increases with the increase of the bid value. At the optimal point, the payoff value is maximum. The payoff decreases with the increase of the bid value beyond the optimal point. Therefore, the participating neighbor gateway accepts the bid at the optimal point to get the maximum benefit. We execute the single user load sharing process 30 times and plot the results with 95 percent degree of confidence. It is pertinent to mention that the length of the confidence interval is variable in nature due to the random selection of bid increment in each iteration. Further, the interval length is increasing in nature with the increase of iteration index because of the cumulative effect of Δp on the computation of p as shown in Equation (13). Finally, we observe that the algorithm reaches the optimal value on an average at iteration index 4 according to our experimental setup.

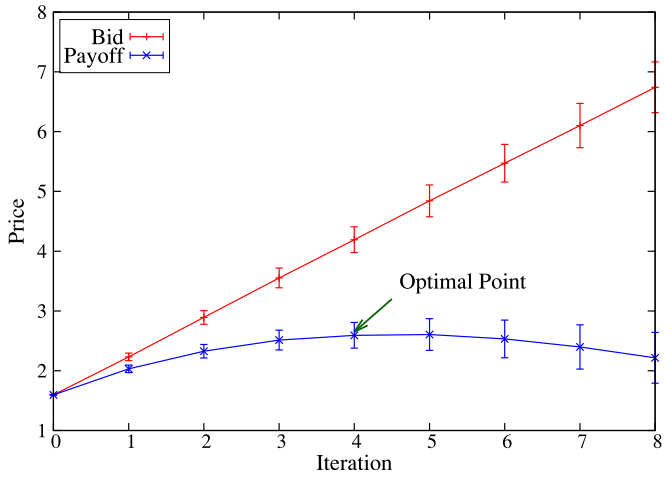

Fig. 5. Optimality of S-QuaLShare algorithm.

6.2.4 Nash Equilibrium

We measured the payoff value with varying n by keeping the values of the other parameters, as shown in Section 6.1. We observe in Fig. 6 that the payoff maximizes at iteration index 4 for $n = 1.5$. The maximum payoff indicates the existence of Nash Equilibrium in the proposed single user load sharing algorithm. We further investigate that the algorithm is unable to reach the maximum point for $n = 1.00$ and 1.25, as the bid value reaches the maximum limit and the algorithm terminates fast for $n = 1.75$ without getting optimal payoff.

6.3 Multiple Users Load Increment

In all the multiple users load increment experiments, we consider that the mobile users N_{11}, N_{12}, and N_{13} connected with the gateway G_1 increase their demand, so that the ideal transmission times for their demand increase from 0.1 sec to 0.2, 0.3, and 0.2 sec, respectively. The mobile users N_{11}, N_{12}, and N_{13} are within the vicinity of all the neighbors of G_1.

6.3.1 Multiple Users Load Sharing

Fig. 7 establishes the necessity of load sharing due to multiple users load increment. When the users N_{11}, N_{12}, and N_{13} increase their demands, the ideal transmission time of service demands increases. This affects the service delay of the connecting gateway G_1 and the value exceeds the threshold limit. In this experiment, multiple users load sharing ensures service delay guarantees in the overloaded gateway G_1, as well as the winner gateways G_3, G_2, and G_5. The service delay of the others remain the same during the sharing process.

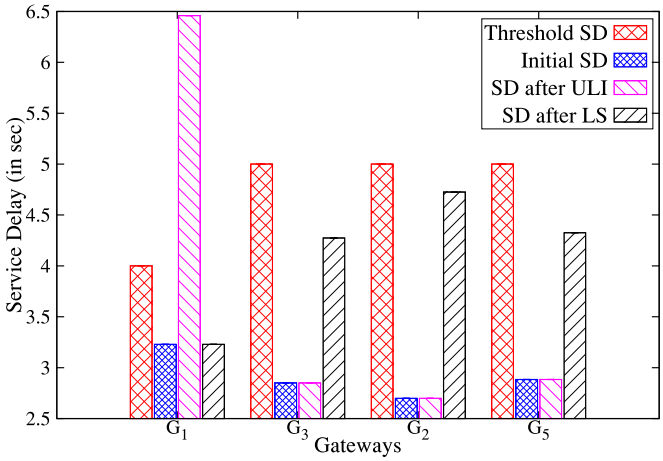

Fig. 7. Load sharing for multiple user load increment with overloaded gateway G_1 and winner gateways G_3, G_2 and G_5.

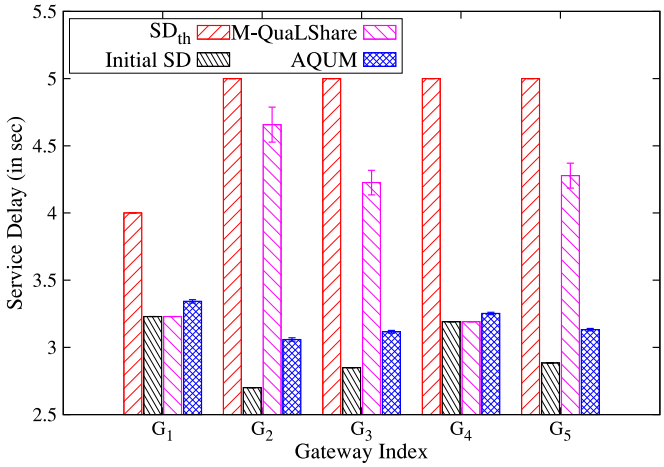

Fig. 9. Service delay comparison between AQUM and M-QuaLShare.

6.3.2 Bandwidth Distribution

We analyze the gateways' final bandwidth distribution for both the proposed multiple users load sharing and the AQUM algorithms. In AQUM, we use 10 as the value of the shifting parameter and the remaining all the values are the same as the default settings of the multiple users load sharing process. Fig. 8 shows that the amount of allocated bandwidth remains the same for all the gateways in the multiple user load sharing process, whereas that value differs for all the gateways used in the AQUM algorithm. As three users of gateway G_1 requests additional service, the CSP distributes the bandwidth to G_1 with a greater proportion. Therefore, we conclude that only a few gateways involve to resolve the incremental service request of a particular gateway in the multiple users load sharing process, while keeping the bandwidth amount the same. Hence, M-QuaLShare does not suffer from the overhead due to bandwidth management.

We further compared the service delay of all the gateways for both the algorithms. We plotted the received outcomes in Fig. 9. From this figure, we observe that the service delay increases only for the winner gateways in the multiple users load sharing process. The service delay varies for all the gateways using AQUM, as the bandwidth is completely redistributed among all the gateways, which are connected to the CSP. Additionally we observed that, similar to AQUM, M-QuaLShare also maintains the service delay within the service delay threshold. Besides the no bandwidth management overhead, the M-QuaLShare algorithm only considers the nearby neighbor gateways of the overloaded one. Therefore, the proposed algorithm involves minimal number of gateways for sharing the loads, whereas AQUM deals with the entire set of gateways present in the system. Hence, we conclude that the performance of M-QuaLShare is better than AQUM in the localized scenarios, where the overloaded gateway has sufficient underloaded neighbors.

6.3.3 Optimality

Fig. 10 shows the optimality of the proposed multiple user load sharing algorithm. In this experiment, we computed the payoff and bid values for each iteration cycle. The optimal point is a point at which the payoff is maximum and beyond this point the users do not receive any additional benefit. With the default parameter settings, we observe that at the beginning of the experiment the payoff increases with the increase of the bid value, the payoff value is maximum at the optimal point, and finally, the value decreases. We

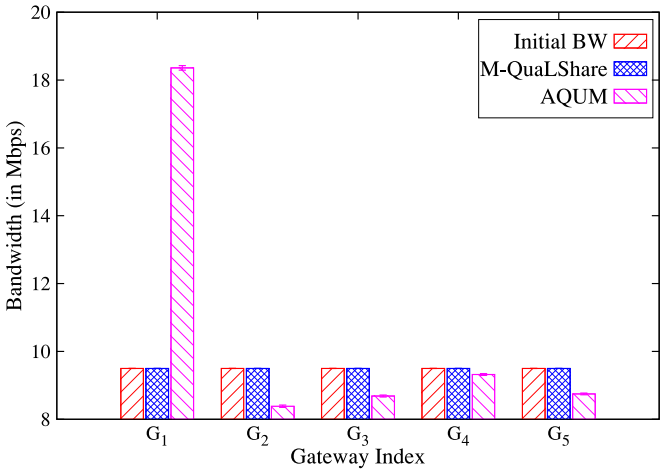

Fig. 8. Bandwidth allocation comparison between AQUM and M-QuaLShare.

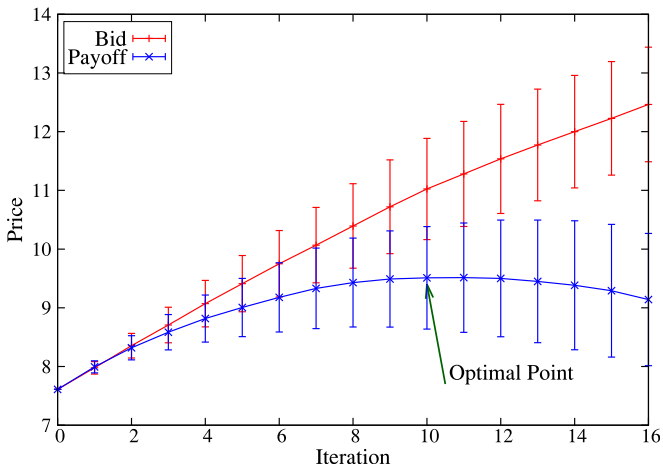

Fig. 10. Optimality of M-QuaLShare algorithm.

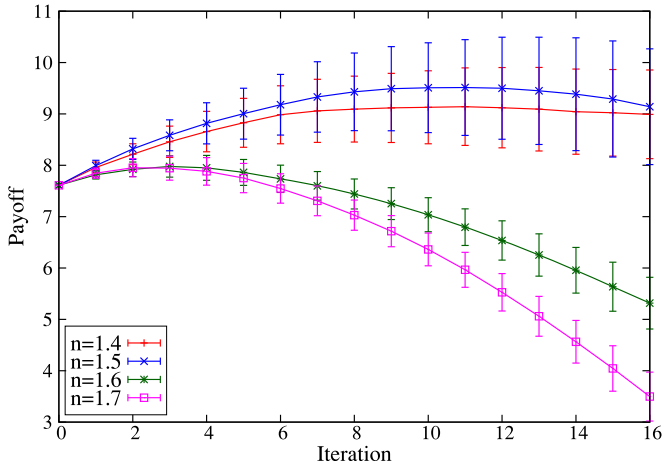

Fig. 11. Nash Equilibrium of M-QuaLShare algorithm.

perform the multiple users load sharing process 30 times, and on an average, we reach the optimality at iteration 10.

6.3.4 Nash Equilibrium

We computed the payoff value with varying n in the default data setup. Fig. 11 depicts that the payoff initially increases for all values of n. For $n = 1.5$, the payoff maximizes at iteration index 10. Additionally, the plotted result shows that the algorithm concludes within a fewer iterations for $n = 1.6$ and 1.7, before reaching the optimal value. On the other hand, for $n = 1.4$, the algorithm does not terminate, as the bidding value crosses the maximum limit. Hence, we conclude that the maximum payoff indicates the existence of Nash Equilibrium in the proposed multiple users load sharing algorithm.

7 Conclusion

In this paper, we identified and addressed the problem of load sharing for distributing the loads of the overloaded gateway in an MCN environment. The load sharing process differs from the traditional load balancing process [2] in that while the former concerns sharing the additional demand with local gateways, the latter concerns equalizing the service demand among all the gateways. We further consider the load sharing problem in the presence of misbehaving gateways, where they provide inadequate information for getting the access of the mobile users who have increased their demand. Thereafter, the winner misbehaving gateways share those users with their neighbors to gain additional benefit. We have proposed an auction-based QoS-guaranteed secure load sharing algorithm for maximizing the utility of the overloaded gateway in both single user and multiple users load increment scenarios. We have verified the requirement of load sharing through numerical analysis. We have also investigated the existence of Nash Equilibrium and the optimality criteria of the proposed algorithms.

Even though the proposed algorithms are able to distinguish the misbehaving gateways from all the participants using the information partially provided by the CSP, in real environments, this information sharing is not always feasible. Therefore, a self-computing scheme is required. In this work, we have considered single hop connection between the gateway and the mobile user. In the future, we also envision to work on an architecture having multi-hop connectivity between the gateway and the mobile user. On the other hand, we assume that the service request of all the mobile users are independent of one another. Hence, the discussion regarding the intra-dependency and inter-dependency of the users' services need to be discussed in the future.

References

[1] S. Misra, S. Das, M. Khatua, and M. S. Obaidat, "QoS-guaranteed bandwidth shifting and redistribution in mobile cloud environment," *IEEE Trans. Cloud Comput*, vol. 2, no. 2, pp. 181–193, Apr.-Jun. 2014.

[2] J. Doyle, R. Shorten, and D. OMahony, "Stratus: Load balancing the cloud for carbon emissions control," *IEEE Trans. Cloud Comput.*, vol. 1, no. 1, pp. 116–128, Jan.-Jun. 2013.

[3] S. Zaman and D. Grosu, "A combinatorial Auction-based mechanism for dynamic vm provisioning and allocation in clouds," *IEEE Trans. Cloud Comput.*, vol. 1, no. 2, pp. 129–141, Jul.-Dec. 2013.

[4] M. Randles, D. Lamb, and A. Taleb-Bendiab, "A comparative study into distributed load balancing algorithms for cloud computing," in *Proc. IEEE 24th Int. Conf. Adv. Inf. Netw. Appl. Workshops*, 2010, pp. 551–556.

[5] S. T. Maguluri, R. Srikant, and L. Ying, "Stochastic models of load balancing and scheduling in cloud computing clusters," in *Proc. IEEE INFOCOM*, 2012, pp. 702–710.

[6] D. Grosu and A. T. Chronopoulos, "Noncooperative load balancing in distributed systems," *J. Parallel Distrib. Comput.*, vol. 65, pp. 1022–1034, 2005.

[7] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: Architecture, applications, and approaches," *Wireless Commun. Mobile Comput.*, vol. 13, no. 18, pp. 1587–1611, Dec. 2013.

[8] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: A survey," *Future Generation Comput. Syst.*, vol. 29, no. 1, pp. 84–106, Jan. 2013.

[9] Z. Sanaei, S. Abolfazli, A. Gani, and R. Buyya, "Heterogeneity in mobile cloud computing: Taxonomy and open challenges," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 369–392, Jan.-Mar. 2014.

[10] S. Abolfazli, Z. Sanaei, E. Ahmed, A. Gani, and R. Buyya, "Cloud-based augmentation for mobile devices: Motivation, taxonomies, and open challenges," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 337–368, Jan.-Mar. 2014.

[11] Z. Sanaei, S. Abolfazli, A. Gani, and M. Chen, "HMCC: A hybrid mobile cloud computing framework," in *Proc. IEEE Int. Conf. Mobile Cloud Comput., Serv., Eng.*, 2015, pp. 157–162.

[12] H. Hu, Z. Li, and H. Hu, "An Anti-cheating bidding approach for resource allocation in cloud computing environments," *J. Comput. Inf. Syst.*, vol. 8, no. 4, pp. 1641–1654, Feb. 2012.

[13] C. Rong, S. T. Nguyen, and M. G. Jaatun, "Beyond lightning: A survey on security challenges in cloud computing," *Comput. Elect. Eng.*, vol. 39, no. 1, pp. 47–54, Jan. 2013.

[14] S. Abolfazli, Z. Sanaei, A. Gani, F. Xia, and L. T. Yang, "Rich mobile applications: Genesis, taxonomy, and open issues," *J. Netw. Comput. Appl.*, vol. 40, pp. 345–362, 2014.

[15] A. N. Khan, M. M. Kiah, S. U. Khan, and S. A. Madani, "Towards secure mobile cloud computing: A survey," *Future Generation Comput. Syst.*, vol. 29, no. 5, pp. 1278–1299, Jul. 2013.

[16] M. Othman and S. Hailes, "Power conservation strategy for mobile computers using load sharing," *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 2, no. 1, pp. 44–51, Jan. 1998.

[17] D. L. Eager, E. D. Lazowska, and J. Zahorjan, "Adaptive load sharing in homogeneous distributed systems," *IEEE Trans. Softw. Eng.*, vol. 12, no. 5, pp. 662–675, May. 1986.

[18] R. Mirchandaney, D. Towsley, and J. A. Stankovic, "Adaptive load sharing in heterogeneous distributed systems," *J. Parallel Distrib. Comput.*, vol. 9, pp. 331–346, 1990.

[19] H. Wang, J. Ren, and T. Li, "Resource allocation with load balancing for cognitive radio networks," in *Proc. IEEE Global Telecommun. Conf.*, Dec. 2010, pp. 1–5.

[20] X. Wang, Z. Li, P. Xu, Y. Xu, X. Gao, and H. Chen, "Spectrum sharing in cognitive radio networks-an auction-based approach," *IEEE Trans. Syst., Man, Cybern., Part B: Cybern.*, vol. 40, no. 3, pp. 587–596., Jun. 2010.
[21] K. Zhu, D. Niyato, and P. Wang, "Dynamic bandwidth allocation under uncertainty in cognitive radio networks," in *Proc. IEEE Global Telecommun. Conf.*, 2011, pp. 1–5.
[22] J. Elias, F. Martignon, and A. Capone, "An efficient dynamic bandwidth allocation algorithm for quality of service networks," in *Proc. Autonomic Netw.*, Sep. 2006, pp. 132–145.
[23] Z. Kun, D. Niyato, and P. Wang, "Optimal bandwidth allocation with dynamic service selection in heterogeneous wireless networks," in *Proc. IEEE Global Telecommun. Conf.*, 2010, pp. 1–5.
[24] Y. Zhang, C. Lee, D. Niyato, and P. Wang, "Auction approaches for resource allocation in wireless systems: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1020–1041, Jul.-Sep. 2013.
[25] K. A. Nuaimi, N. Mohamed, M. A. Nuaimi, and J. A. Jaroodi, "A survey of load balancing in cloud computing: Challenges and algorithms," in *Proc. IEEE 2nd Symp. Netw. Cloud Comput. Appl.*, 2012, pp. 137–142.
[26] C. Papagianni, A. Leivadeas, S. Papavassiliou, V. Maglaris, C. C. Pastor, and A. Monje, "On the optimal allocation of virtual resources in cloud computing networks," *IEEE Trans. Comput.*, vol. 62, no. 6, pp. 1060–1071, Jun. 2013.
[27] A. Amamou, M. Bourguiba, K. Haddadou, and G. Pujolle, "A dynamic bandwidth allocator for virtual machines in a cloud environment," in *Proc. IEEE Consum. Commun. Netw. Conf.*, Jan. 2012, pp. 99–104.
[28] S. Das, S. Misra, M. Khatua, and J. J. P. C. Rodrigues, "Mapping of sensor nodes with servers in a mobile Health-cloud environment," in *Proc. IEEE 15th Int. Conf. e-Health Netw., Appl. Serv.*, Lisbon, Portugal, Oct. 2013, pp. 481–485.
[29] J. W. Lin, C. H. Chen, and J. M. Chang, "QoS-aware data replication for data intensive applications in cloud computing systems," *IEEE Trans. Cloud Comput.*, vol. 1, no. 1, pp. 101–115, Nov. 2013.
[30] Z. Yang, N. Dusit, and W. Ping, "An auction mechanism for resource allocation in mobile cloud computing systems," in *Proc. 8th Int. Conf. Wireless Algorithms, Syst., Appl.*, Berlin, Heidelberg, 2013, pp. 76–87.
[31] X. Wang, X. Wang, C. li Wang, K. Li, and M. Huang, "Resource allocation in cloud environment: A model based on double Multi-attribute auction mechanism," in *Proc. IEEE Int. Conf. Cloud Comput. Technol. Sci.*, 2014, pp. 599–604.
[32] M. Zhou, R. Zhang, W. Xie, W. Qian, and A. Zhou, "Security and privacy in cloud computing: A survey," in *Proc. 6th Int. Conf. Semantics, Knowl. Grids*, 2010, pp. 105–112.
[33] Y. Chen, Y. Wu, B. Wang, and K. J. R. Liu, "Spectrum auction games for multimedia streaming over cognitive radio networks," *IEEE Trans. Commun.*, vol. 58, no. 8, pp. 2381–2390, Aug. 2010.
[34] S. Subashini and V. Kavitha, "A survey on security issues in service delivery models of cloud computing," *J. Netw. Comput. Appl.*, vol. 34, no. 1, pp. 1–11, Jan. 2011.
[35] A. Verma and S. Kaushal, "Cloud computing security issues and challenges: A survey," in *Proc. 1st Int. Conf. Adv. Comput. Commun.*, Kochi, India, Jul. 2011, pp. 445–454.

Snigdha Das received the BTech degree in information technology from the West Bengal University of Technology, India in 2010 and she is currently working toward the MS degree from the School of Information Technology, Indian Institute of Technology Kharagpur, India, and is working as a senior project officer in Development of Feasibility Assessment Model for Adaption of Underground Coal gasification Technology in North-Eastern Region of India funded by DeitY, Government of India. She was an associate software engineer in CSC for two years. Her current research interests include mobile computing and cloud computing.

Manas Khatua received the BTech degree in computer science and engineering from the University of Kalyani, India, in 2003, the MTech degree in information technology from Bengal Engineering and Science University, India, in 2007, and is currently working toward the PhD degree from the School of Information Technology, Indian Institute of Technology Kharagpur, India. Presently, he is a research assistant of the Singapore University of Technology and Design, Singapore. Prior to this, he was associated with Tata Consultancy Services, Kolkata, for two and half years. He was a lecturer of the Bankura Unnayani Institute of Engineering, India, for more than two years. His research interests include the performance evaluation and resource optimization in Wireless LANs. He also works on sensor networks, network security, and mobile cloud computing. He is one of the recipients of NIXI Fellowship 2014. He is a student member of the IEEE.

Dr. Sudip Misra received the PhD degree in computer science from Carleton University, in Ottawa, Canada. He is an associate professor in the School of Information Technology at the Indian Institute of Technology Kharagpur. Prior to this, he was associated with Cornell University (USA), Yale University (USA), Nortel Networks (Canada), and the Government of Ontario (Canada). He has received eight research paper awards in different conferences. He was awarded the IEEE ComSoc Asia Pacific Outstanding Young Researcher Award at IEEE GLOBECOM 2012. He was also awarded the Canadian Governments prestigious NSERC Post Doctoral Fellowship and the Humboldt Research Fellowship in Germany. He is a senior member of the IEEE.

Mohammad S. Obaidat (F'05) received the PhD degree in computer engineering from Ohio State University. He is an internationally well-known academic/ researcher/ scientist. He currently the chair and full professor of computer & information science at Fordham University. Among his previous positions are a chair of the Computer Science Department and the director of Graduate Program at Monmouth University, a dean of engineering at Prince Sultan University, and an advisor to the president of Philadelphia University. He served as an advisor to the president of Philadelphia University and as a president, senior VP, VP membership, and VP conferences of SCS. He has received extensive research funding and has published more than 38 books and 600 refereed technical articles. He is an editor-in-chief of three scholarly journals and an editor of many other international journals. He received numerous awards. He is a fellow of the IEEE and SCS.