Quality and Profit Assured Trusted Cloud Federation Formation: Game Theory Based Approach

Benay Kumar Ray, Avirup Saha, Sunirmal Khatua, *Member, IEEE*, and Sarbani Roy, *Member, IEEE*

Abstract—With more awareness and growth in the cloud market, demands for computational resources have increased in order to provide services to the cloud users. Sometimes it is difficult for an individual cloud service provider (CSP) to meet the level of promised quality of service (QoS) and to fulfill all types of resource requests dynamically. Cloud federation has become a consolidated paradigm in which group of cooperative CSPs share their unused resources with peers to gain some economic benefit. Hence, the cloud federation overcomes the limitation of each CSP for maintaining QoS during sudden spikes in resource demand. However, the presence of untrusted CSPs degrades the QoS of the services delivered through federation. Trusted CSPs are highly reputed in the federation as they can extend their resources and services to maintain the level of committed QoS by the member CSPs of the federation. Therefore, to guarantee delivery of committed QoS, it will be necessary to form a federation with trusted CSPs only. In this paper, we present a broker based cloud federation architecture. The cloud federation formation is modeled as a *hedonic coalitional game*. The main objective of this work is to find the most suitable and stable federation of trusted CSPs that will maximize the satisfaction level of each individual CSP on the basis of QoS and profit. The proposed coalitional game inspired cloud federation formation (CGCFF) algorithm has been extensively compared with selected existing techniques. Simulation results show that the set of federation formed by CGCFF is Nash-stable and performs better than these techniques in terms of satisfaction, quality and profit.

Index Terms—Cloud computing, broker, federation, coalition game theory, quality of service, satisfaction level, trust

1 Introduction

Cloud computing has earned an overwhelming popularity as a paradigm to deliver services and resources through virtualization and on demand provisioning techniques [1]. The main objective of a CSP is to deliver the services as per the service level agreement (SLA). Delivering a cloud service with the committed QoS has often been a delicate balancing act where a service provider has to consider resource request traffic, resource utilization, infrastructure constraints due to sudden spikes in resources' demand and scaling issues. These factors need to be tackled efficiently as they are directly related to the performance of the underlying environment. To handle such scenarios, small IaaS CSPs may find it more challenging compared to the big enterprises and thus it may hamper their QoS. Here, small IaaS CSPs are those whose global market share is less compared to other big IaaS CSPs (like Amazon Web Services, Microsoft Azure) [2], [3]. As the awareness and knowledge about the benefits associated with cloud services increase, demands for cloud IaaS also increases. Thus, increasing the number of users like Facebook, Youtube, e-commerce business, etc. The workloads generated by this type of users are tremendous, bursty and unpredictable [4], [5], [6]. Hence it becomes difficult for an individual CSP (mainly for small cloud service providers) to fulfill all the resource requests with the QoS expectations of cloud users. On the other hand, in today's data-centers, virtual machines of CSPs are often over-provisioned to avoid poor performance. But over-provisioning of virtual machines, leads to wastage of resources like core, memory and storage. Moreover, during times of low demands of resource requests, utilization of resources will decrease and thus overall revenue earned by the CSPs will drop. Therefore, all the above mentioned problems faced by CSPs will necessitate them to form a federation for seamless provisioning of resources and services across different cloud providers.

Cloud federation is the practice of collaborating the cloud environments of two or more service providers where each CSP can share their resources with peers to gain economic benefits as well as to maintain the promised QoS. Cloud federation provides a considerable amount of benefit to cloud providers while providing services to cloud users. First of all, within the federation, providers can make extra profit by sharing idle or underutilized computing resources. Second, cloud federation enables providers to accommodate unexpected demands without having to build new points-of-presence. Third, with cloud federation, cloud service providers do not need to invest in new hardware to offer high-end and flexible services and can have a great impact on energy savings.

Forming cloud federation out of multiple heterogeneous CSPs is often a very complicated task due to the differences

- *B.K. Ray, A. Saha, and S. Roy are with the Department of Computer Science and Engineering, Jadavpur University, Kolkata, West Bengal 700032, India. E-mail: {benayray, saha.avirup}@gmail.com, sarbani.roy@cse.jdvu.ac.in.*
- *S. Khatua is with the Department of Computer Science and Engineering, University of Calcutta, Kolkata, West Bengal 700073, India. E-mail: skhatuacomp@caluniv.ac.in.*

Manuscript received 7 Apr. 2017; revised 10 Apr. 2018; accepted 30 Apr. 2018. Date of publication 8 May 2018; date of current version 9 June 2021. (Corresponding author: Sarbani Roy.)
Digital Object Identifier no. 10.1109/TSC.2018.2833854

in (i) pricing model, (ii) QoS values and (iii) trust values. The trust of each CSP is defined as the extent to which the CSP can deliver QoS as compared to its committed QoS. There has been some research work on federation formation mechanism recently. In [7], Mashayekhy et al. have mainly focused on maximizing overall profit of federation in terms of cost and price of virtual machine of each CSP. Further, Wahab et al. [8] considers maliciousness of CSPs in federation formation. Here, malicious members are those who refuse to share their resources with the other member CSPs within the federation. Their main aim is to find the federation partition (consisting of set of federations) that minimizes the malicious members. But their proposed work do not maximize the profit of federation. However, none of these works have considered the problem of maximizing both profit and QoS of federation simultaneously while considering each CSP's trust value. Profit is one of the important parameters for CSPs as they are able to earn some extra profit by sharing their idle or underutilized resources with other CSPs during low resource demand. Further, it is also important for the federation to maximize profit as it encourages CSPs to participate in federation formation. Next, availability (uptime) is one of the major QoS parameters for CSPs, in order to take part in federation formation. This is because, CSPs take part in federation to increase their availability of resources during peak time and guarantee greater availability (uptime) of resources (virtual machine) to the cloud users. Further, forming a federation between trusted CSPs is very important as, any CSP will prefer to be a part of a federation where other members are trusted. This is because, the federation will fail to satisfy the member CSPs if the delivered QoS differs from the committed QoS for some of the member CSPs and hence decreasing the overall Quality of service delivered through federation. In order to deliver the committed QoS through federation, it will be necessary to assess trust of each CSP, before allowing them to take part in federation.

In this paper, problem of cloud federation formation is modeled as hedonic coalition game where different CSPs form a federation to allocate their idle resources dynamically to cloud users. The main objective of our work is to maximize the overall profit and availability of federation formed among trusted CSPs. The proposed framework uses Beta Mixture Model to estimate the quality and trust of each CSP and thereafter allows trusted CSPs to join in federation. A broker (trusted third party) based cloud federation architecture is proposed for forming and maintaining the cloud federation in an efficient way. The cloud broker determines the individual satisfaction level (the satisfaction that any CSPs received while being part of federation and is defined by value of profit and availability that it gets in that federation) of each CSP in a federation and based on that it decides whether the CSP should join or leave that federation. Finally, it obtains a stable federation partition, where none of the CSPs gets better incentive on joining a different federation. We have performed extensive experiments using a real-life data set [10] and [11] to investigate the importance of our proposed model and thereafter compared our proposed mechanism with three existing works [7], [8] and [9]. The major contributions of the paper can be summarized as follows:

- A generic cost model is defined to evaluate the cost of a federation.
- Quality and trust are estimated based on Beta Mixture Model for cloud service providers.
- The satisfaction level of CSPs and trust are used in forming the cloud federation and the proposed algorithm converges to a Nash stable and individually stable federation partition.

The rest of this paper is organized as follows: Section 2 describes the related work. In Section 3, we explain the proposed cloud federation architecture and define availability of resources in federation. In Section 4, quality and trust are estimated for cloud federation and its corresponding service providers. Section 5 presents the modeling of the cloud federation as a hedonic coalition game and describes the CGCFF algorithm. Section 6 presents the performance evaluation results of the proposed framework and Section 7 concludes the paper.

2 Related Work

In the last few years, a lot of research has been done around federated cloud computing. In [12], Rochwerger et al. discussed the primary requirements for federation formation among CSPs. Goiri et al. [13] presented an analytical model that characterizes cloud federation and can be used to drive provider's decisions about resource outsourcing, insourcing, and shutdown of unused instances to save power. Hassan et al. [14] studied the problem of distributed resource allocation in horizontal dynamic cloud federation environment (HDCF) formed by different service providers. They proposed a game theory based solution to this problem that encourages service providers to form a federation. Wang et al. [15] presented a cloud brokerage service which reserves large pool of virtual machine from CSPs and provides the required virtual machine to users at discounted price. Yi et al. [16] proposed Cocoa, a container-based model to implement group buying system, to entice and combine cloud users with complementary resource demands in cloud. Thus helping CSPs to improve their resource utilization and to minimize cost. In [17], Tang et al. proposed UniDrive, a consumer cloud storage (CCS) app that collaborates cloud storage of multiple clouds and enhances the reliability, security and networking performance of the CCS services. Furthermore, [4] proposed a Least Cost per Connection (LCC) load balancing algorithm for hybrid cloud application, which helps to find the best public CSP for outsourcing along with adapting changes among multiple public CSPs. Niu et al. in [5] proposed a cost effective online algorithm to overcome the problem of workload generated by flash crowds of common e-commerce websites in a hybrid cloud. In [6], a solution for flash deal applications to withstand flash crowds with soft guarantee in hybrid cloud environment is proposed. Niyato et al. [18] also proposed a revenue sharing mechanism among service providers in federation based on stochastic linear programming. But their mechanism does not consider the cost incurred by each service provider. Mashayekhy et al. [7] proposed a cloud federation formation mechanism based on coalition game theory. Their main aim is to maximize profit of a formed federation. It is to be noted that, though they have used hedonic coalition game, their final output of the game is single stable federation instead of federation partition consisting of set of federations. Wahab et al. [8] tried to propose a solution for the problem of the existence of malicious services that tend to act inappropriately to illegally maximize their own benefits while being part of a federation. Here, malicious members are those that refuse to share their

resources with the other member CSPs within federation. Their main aim is to find the stable federation partition that minimizes the malicious members in the formed federation within the partition.

However, none of the previous works in the literature considered the profit, availability and trust based cloud federation formation between the service providers. Moreover, the proposed game theory based mechanism CGCFF gives importance to the profit, availability and trust, three main deciding factors in the cloud federation game. In this paper, we study the cooperative behaviour of multiple CSPs having different resource capabilities. In CGCFF, the trusted CSPs are selected first and then a trusted cloud federation is formed based on their satisfaction level (defined by availability and profit). Thus, it helps cloud providers in the decision making process of federation formation, so that it can improve resource utilization, profit and availability of resources. Finally, we analyze the stability of final federation partition formed by a set of federation using properties of hedonic game.

3 Proposed Cloud Federation Architecture

In this section, we discuss the cloud federation architecture, availability (i.e., uptime) of resources in a cloud federation and thereafter derive the cost and chargeable price of resources in a cloud federation. Further, we also discuss the profit obtained by federation and its corresponding member CSPs followed by satisfaction level of each CSP and federation.

Cloud broker plays an important role in providing cloud services to users in an efficient way [19]. In this paper, we have proposed a broker based cloud federation architecture to form the cloud federation among multiple trusted CSPs and to provide cloud services to users in a collaborative manner as depicted in Fig. 1. The proposed architecture considers two types of cloud users: (i) cloud users who can directly interact with the CSP for their requests and (ii) cloud users who can interact with the cloud broker with the intention to get cloud service with high availability and QoS. Thus, it handles two scenarios (i) CSPs request through federation manager (FM) for leasing their idle resources and for forming a federation with other trusted CSPs and (ii) CSPs outsource their excessive resource request (only those CSPs which are part of a federation) through the service manager (SM), whenever these CSPs run out of computing resources. Different components of the proposed cloud federation architecture, depicted in Fig. 1, are explained as given below:

- *Service Manager:* The main responsibility of this entity is to provide services to CSPs and cloud users through the formed federation. CSPs when run out of computing resources, send excessive request to SM, in order to outsource their requests to the members of federation to which they belong. Moreover, cloud users who directly interact with broker with the intention to get cloud service with high availability and QoS are also being served by SM. It also maintains the intermediate states of the deployed applications to virtual resources from multiple federations.
- *Federation Manager:* This entity forms the federation among multiple CSPs in an efficient way. It becomes the virtual owner of the virtual resources provided by different CSPs in the federation. Thus, FM can easily manage the service requests from SM.

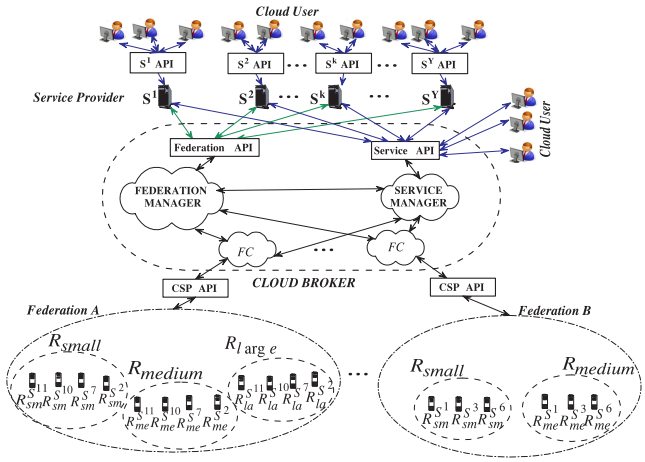

Fig. 1. Cloud federation architecture.

- *Federation Coordinator (FC):* For each federation, FM deploys a federation coordinator to manage the resources of newly formed federation. It also sends periodic updates to the FM about the state of the federation it manages. The FC groups similar resources ($R_{\mathscr{T}}$) according to their types ($\mathscr{T}$). For instance, the resource $R^{S^{id}}_{\mathscr{T}}$ denotes a resource from service provider S^{id} where $\mathscr{T} \in \{small, medium, large\}$ denotes different resource types.

3.1 Availability of Resources in Cloud Federation

Availability of resources is considered to be one of the major QoS parameters for cloud users [20]. In this paper, we have considered the uptime (measured as a fraction of total time) [21] of a resource as the availability of that resource.

QoS of cloud services are greatly affected by the unavailability (downtime) of resources. Such unavailability will reduce significantly by accessing the resources from a federation instead of individual CSPs. The cooperation among CSPs in federation is achieved when resources from other CSPs are used to fulfill the cloud user's service request in case the resources are unavailable for a certain CSP. The cooperation among service providers in the federation is managed by the FC as shown in Fig. 1. The arrangement order of CSPs S^{id} within the federation f_j is also maintained by FC and it is denoted by $f_j.\alpha$. Here, $f_j.\alpha$ denotes the order in which CSPs within federation deliver services and is given as

$$f_j.\alpha = < S^{id}_1, S^{id}_2, \ldots S^{id}_N > \ . \tag{1}$$

Where $S^{id}_{pos} \in f_j$, $id \in \{1, \ldots, Y\}$ and $pos \in \{1, \ldots, N\}$. S^{id}_{pos} denotes order in which CSPs will provide the cloud services. The size of the federation is $|f_j| = N$. Various notations used in the cloud federation architecture are given in Table 1. In this paper S^{id}_{pos} and S^{id} will be used interchangeably.

For better understanding, how cloud services are delivered through federation in which CSPs are arranged in order α is discussed as follows. On receiving the cloud user's requests of different types of resources $R_{\mathscr{T}}$ ($\mathscr{T} \in \{micro, small, medium, large, \ldots\}$), FC forwards the corresponding requests to the service provider S^{id}_1 which is the first member in the arrangement of $f_j.\alpha$, otherwise it will select next service provider i.e., S^{id}_2, the second member in the arrangement order $f_j.\alpha$. This process will continue until FC finds a service provider that can fulfill the required resources $R_{\mathscr{T}}$ in f_j.

TABLE 1
Notation

Symbols	Description
η	set of cloud service providers $\{S^{id}\}\ \forall\ id \in [1, Y]$ and id is the identity of service providers
S^{id}_{pos}	pos denotes the position of service provider S^{id} in arrangement order α
$f_j.\alpha$	α is the arrangement order of CSPs in federation f_j
$f_j.X$	X denotes different parameters of federation f_j
$f_j.S^{id}.T$	T denotes different parameters of service provider $S^{id} \in f_j$
$S^{id}.Z$	Z denotes different parameters of service provider S^{id}
$\varphi_{S^{id}}(f_j)$	profit obtained by S^{id} in federation f_j
Υ	set of federation formed by partitioning η
$R_{\mathscr{r}}$	denotes different types of resource where $\mathscr{r} \in \{small, medium, large\}$
$R^{S^{id}}_{\mathscr{r}}$	denotes $R_{\mathscr{r}}$ types of resources of service provider S^{id}

Here, each resource type $R_{\mathscr{r}}$ of CSP is considered as a component of federation f_j. Based on the availability (uptime) of service provider, the availability of any resource $R_{\mathscr{r}}$ (resource type) in f_j can be derived as follows:

The resource $R_{\mathscr{r}}$ is unavailable in the federation f_j if $R^{S^{id}_{pos}}_{\mathscr{r}}$ is unavailable for all $S^{id}_{pos} \in f_j$ (Fig. 1). Therefore, the probability of $R_{\mathscr{r}}$ to become unavailable is the probability Pr that all resources of $R^{S^{id}_{pos}}_{\mathscr{r}}$ within a federation are unavailable (down) (please note here,$\{R^{S^{id}_1}_{\mathscr{r}}, R^{S^{id}_2}_{\mathscr{r}}, \ldots, R^{S^{id}_N}_{\mathscr{r}}\}$ are N numbers of independent events as CSPs are independent entities). Let, $f_j.U$ denotes unavailability of $R_{\mathscr{r}}$ in federation f_j, then joint probability of unavailability of all the event $(R^{S^{id}_{pos}}_{\mathscr{r}} : S^{id}_{pos} \in f_j)$ is given as follows:

$$f_j.U = Pr\left(R^{S^{id}_1}_{\mathscr{r}}.u \cap R^{S^{id}_2}_{\mathscr{r}}.u \cap R^{S^{id}_3}_{\mathscr{r}}.u \cdots\cdots \cap R^{S^{id}_N}_{\mathscr{r}}.u\right). \quad (2)$$

Where $R^{S^{id}_{pos}}_{\mathscr{r}}.u$ are the event of unavailability of resource $R_{\mathscr{r}}$ of service provider S^{id}_{pos} at different position ($pos \in \{1 \ldots N\}$). Based on conditional probability, above equation can be expressed as

$$\begin{aligned} f_j.U = {} & Pr\left(R^{S^{id}_1}_{\mathscr{r}}.u\right) \times Pr\left(R^{S^{id}_2}_{\mathscr{r}}.u | R^{S^{id}_1}_{\mathscr{r}}.u\right) \\ & \times \cdots \times Pr\left(R^{S^{id}_N}_{\mathscr{r}}.u | R^{S^{id}_1}_{\mathscr{r}}.u, \ldots, R^{S^{id}_N}_{\mathscr{r}}.u\right). \end{aligned} \quad (3)$$

We assume that CSPs are independent and failure of any resource $R^{S^{id}_{pos}}_{\mathscr{r}}$ of service provider S^{id}_{pos} will not affect resources of other service providers. Therefore the equation can be expressed as

$$f_j.U = Pr\left(\prod_{pos=1}^{N} R^{S^{id}_{pos}}_{\mathscr{r}}.u\right). \quad (4)$$

The availability of any $R_{\mathscr{r}}$ in a federation f_j is given as

$$f_j.A = 1 - f_j.U \quad (5)$$

$$f_j.A = 1 - Pr\left(\prod_{pos=1}^{N} R^{S^{id}_{pos}}_{\mathscr{r}}.u\right). \quad (6)$$

Here, it shows that availability of $R_{\mathscr{r}}$ is based on availability (uptime) of independent service providers with equivalent distribution in a federation.

3.2 Cost of Federation

Let us denote a set of CSPs as $\eta = \{S^1, S^2, S^3, \ldots\ldots S^Y\}$. Suppose, each service provider S^{id} offers $R_{\mathscr{r}}$ types of resources, where each resource is differentiated based on the following parameters:

- $S^{id}.CO_{R_{\mathscr{r}}}$ is the number of cores of $R_{\mathscr{r}}$ type of resource of service provider S^{id}.
- $S^{id}.Me_{R_{\mathscr{r}}}$ is the amount of memory used by $R_{\mathscr{r}}$ type of resource of service provider S^{id}.
- $S^{id}.S_{R_{\mathscr{r}}}$ is the amount of storage used by $R_{\mathscr{r}}$ type of resource of service provider S^{id}.

The cost incurred by a service provider S^{id} to provide resource of type $R_{\mathscr{r}}$ can be given as follows:

$$\begin{aligned} S^{id}.C_{R_{\mathscr{r}}} = {} & S^{id}.CO_{R_{\mathscr{r}}} \times cost_{CO} + S^{id}.Me_{R_{\mathscr{r}}} \times cost_{Me} \\ & + S^{id}.S_{R_{\mathscr{r}}} \times cost_S. \end{aligned} \quad (7)$$

Where, $cost_{CO}$ is the cost of each core of $R_{\mathscr{r}}$ type of resource of S^{id}, $cost_{Me}$ is the cost of one GB of memory of $R_{\mathscr{r}}$ type of resource of S^{id} and $cost_S$ is the cost of one GB of storage of $R_{\mathscr{r}}$ type of resource of S^{id}.

However, the cost of a federation to provide resources of type $R_{\mathscr{r}}$ not only depends on the cost of each service provider S^{id} for $R_{\mathscr{r}}$ type of resource but also on the availability of $R_{\mathscr{r}}$ in S^{id} ($S^{id}.a_{R_{\mathscr{r}}}$). Thus cost incurred by federation to provide resources of type $R_{\mathscr{r}}$ is given by

$$\begin{aligned} f_j.Cost_{R_{\mathscr{r}}} = {} & S^{id}_1.a_{R_{\mathscr{r}}} \times S^{id}_1.C_{R_{\mathscr{r}}} + \sum\nolimits_{i=2}^{N} \prod\nolimits_{k=1}^{i-1} S^{id}_k.u_{R_{\mathscr{r}}} \\ & \times S^{id}_i.a_{R_{\mathscr{r}}} \times S^{id}_i.C_{R_{\mathscr{r}}}. \end{aligned} \quad (8)$$

Where $S^{id}_i.a_{R_{\mathscr{r}}}$ and $S^{id}_i.u_{R_{\mathscr{r}}}$ denotes the availability and unavailability of resource $R_{\mathscr{r}}$ in service provider S^{id}_i, which is placed in ith position of $f_j.\alpha$.

Therefore, the cost of federation f_j is defined as the sum of cost of each $R_{\mathscr{r}}$ of federation f_j provided by S^{id}_{pos} as shown below:

$$f_j.Cost_{\forall R_{\mathscr{r}}} = \sum_{\forall R_{\mathscr{r}}} f_j.Cost_{R_{\mathscr{r}}}. \quad (9)$$

3.3 Profit Gained by a Cloud Federation

The chargeable price of resource $R_{\mathscr{r}}$ in federation f_j is modeled based on availability (uptime) provided by that federation. To get high availability (uptime of service), cloud users will have to pay a higher price than that of QoS with lesser availability. We have assumed, each cloud federation with high availability will charge a higher price and is given by:

$$f_j.Price^{chargeable}_{R_{\mathscr{r}}} = \rho_{R_{\mathscr{r}}} \times f_j.A. \quad (10)$$

Where $\rho_{R_{\mathscr{r}}}$ is the fixed price of resource $R_{\mathscr{r}}$ whose availability is 1 and $f_j.A$ is the availability of resource $R_{\mathscr{r}}$ in federation f_j.

Thus the profit gained by federation f_j on providing resource $R_{\mathscr{r}}$ can be expressed as

$$f_j.Profit_{R_{\mathscr{r}}} = f_j.Price^{chargeable}_{R_{\mathscr{r}}} - f_j.Cost_{R_{\mathscr{r}}}. \quad (11)$$

Therefore, a federation can maximize its profit by minimizing $f_j.Cost_{R_{\mathscr{r}}}$ which depends on the ordering of service providers within the federation. As shown in Theorem 1, $f_j.Cost_{R_{\mathscr{r}}}$ is minimum when the service providers are arranged in increasing order of their cost for providing resource $R_{\mathscr{r}}$.

Theorem 1. *The expected cost of federation is minimum with the federation arrangement order $f_j.\alpha$ formed with increasing order of cost for providing resources.*

Proof. Let us denote the federation arrangement order as $f_j.\alpha = \{S_1^{id}, S_2^{id}, S_3^{id}, \ldots S_n^{id}\}$.

Then the expected cost of providing resource $R_{\mathscr{F}}$ in federation f_j can be given as below:

$$f_j.Cost_{R_{\mathscr{F}}} = S_1^{id}.a_{R_{\mathscr{F}}} \times S_1^{id}.C_{R_{\mathscr{F}}} + \cdots + (S_1^{id}.u_{R_{\mathscr{F}}} \ldots S_{n-1}^{id}.u_{R_{\mathscr{F}}}) \times S_n^{id}.a_{R_{\mathscr{F}}} \times S_n^{id}.C_{R_{\mathscr{F}}}.$$

Let us also denote

$$A = S_1^{id}.a_{R_{\mathscr{F}}} \times S_1^{id}.C_{R_{\mathscr{F}}} + \cdots + (S_1^{id}.u_{R_{\mathscr{F}}} \ldots S_{k-2}^{id}.u_{R_{\mathscr{F}}}) \times S_{k-1}^{id}.a_{R_{\mathscr{F}}} \times S_{k-1}^{id}.C_{R_{\mathscr{F}}}$$

$$B = (S_1^{id}.u_{R_{\mathscr{F}}} \ldots S_{k+1}^{id}.u_{R_{\mathscr{F}}}) \times S_{k+2}^{id}.a_{R_{\mathscr{F}}} \times S_{k+2}^{id}.C_{R_{\mathscr{F}}} + \cdots + (S_1^{id}.u_{R_{\mathscr{F}}} \ldots S_{n-1}^{id}.u_{R_{\mathscr{F}}}) \times S_n^{id}.a_{R_{\mathscr{F}}} \times S_n^{id}.C_{R_{\mathscr{F}}}.$$

Thus

$$f_j.Cost_{R_{\mathscr{F}}} = A + (S_1^{id}.u_{R_{\mathscr{F}}} \ldots S_{k-1}^{id}.u_{R_{\mathscr{F}}}) \times S_k^{id}.a_{R_{\mathscr{F}}} \times S_k^{id}.C_{R_{\mathscr{F}}} + (S_1^{id}.u_{R_{\mathscr{F}}} \ldots S_k^{id}.u_{R_{\mathscr{F}}}) \times S_{k+1}^{id}.a_{R_{\mathscr{F}}} \times S_{k+1}^{id}.C_{R_{\mathscr{F}}} + B.$$

Let us assume that $f_j.Cost_{R_{\mathscr{F}}}$ is minimum for which $S_1^{id}.C_{R_{\mathscr{F}}} \leq S_2^{id}.C_{R_{\mathscr{F}}} \leq S_3^{id}.C_{R_{\mathscr{F}}} \leq \ldots S_n^{id}.C_{R_{\mathscr{F}}}$ is not necessarily true i.e., there exists a k for which $S_k^{id}.C_{R_{\mathscr{F}}} > S_{k+1}^{id}.C_{R_{\mathscr{F}}}$.

Let us form a new federation arrangement order $f_j.\alpha' = \{S_1^{id}, \ldots, S_{k-1}^{id}, S_{k+1}^{id}, S_k^{id}, S_{k+2}^{id}, \ldots, S_n^{id}\}$.

Then the expected cost of federation will be

$$f_j.Cost'_{R_{\mathscr{F}}} = A + (S_1^{id}.u_{R_{\mathscr{F}}} \ldots S_{k-1}^{id}.u_{R_{\mathscr{F}}}) \times S_{k+1}^{id}.a_{R_{\mathscr{F}}} \times S_{k+1}^{id}.C_{R_{\mathscr{F}}} + (S_1^{id}.u_{R_{\mathscr{F}}} \ldots, S_{k-1}^{id}.u_{R_{\mathscr{F}}}, S_{k+1}^{id}.u_{R_{\mathscr{F}}}) \times S_k^{id}.a_{R_{\mathscr{F}}} \times S_k^{id}.C_{R_{\mathscr{F}}} + B$$

$$f_j.Cost'_{R_{\mathscr{F}}} - f_j.Cost_{R_{\mathscr{F}}} = (S_1^{id}.u_{R_{\mathscr{F}}} \ldots S_{k-1}^{id}.u_{R_{\mathscr{F}}})[S_k^{id}.a_{R_{\mathscr{F}}} \times S_{k+1}^{id}.a_{R_{\mathscr{F}}}(S_{k+1}^{id}.C_{R_{\mathscr{F}}} - S_k^{id}.C_{R_{\mathscr{F}}})]$$

$f_j.Cost'_{R_{\mathscr{F}}} - f_j.Cost_{R_{\mathscr{F}}} < 0$
since $S_k^{id}.C_{R_{\mathscr{F}}} > S_{k+1}^{id}.C_{R_{\mathscr{F}}}$
i.e., $f_j.Cost'_{R_{\mathscr{F}}} < f_j.Cost_{R_{\mathscr{F}}}$ which contradicts our assumption.

Therefore, $f_j.Cost_{R_{\mathscr{F}}}$ is minimum for which $S_1^{id}.C_{R_{\mathscr{F}}} \leq S_2^{id}.C_{R_{\mathscr{F}}} \leq S_3^{id}.C_{R_{\mathscr{F}}} \leq \ldots S_n^{id}.C_{R_{\mathscr{F}}}$ is necessarily true. However, the service providers are arranged in increasing order of availability in case there is a tie in cost. □

3.4 Distribution of Profit Between Members of Federation

The total profit $\mathbf{P}(f_j)$ obtained by federation f_j by utilizing all the contributed resource $R_{\mathscr{F}}$ by each CSP can be expressed as below:

$$\mathbf{P}(f_j) = \textstyle\sum_{\forall R_{\mathscr{F}}} f_j.Profit_{R_{\mathscr{F}}}. \tag{12}$$

Next, the profit obtained by S^{id} in federation f_j is given as follows:

$$\varphi_{S^{id}}(f_j) = \omega_{S^{id}} \times (\mathbf{P}(f_j) - \sum_{S^{id} \in f_j} \mathbf{profit}(\{S^{id}\})) + \mathbf{profit}(\{S^{id}\}). \tag{13}$$

Where, $\omega_{S^{id}}$ denotes the ratio of total number of resources $q^{S^{id}}$ of each type contributed by S^{id} in federation f_j over sum of total number of resources Q_{f_j} of each type contributed by all the member CSPs in federation and is given by $\omega_{S^{id}} = \frac{q^{S^{id}}}{Q_{f_j}}$. $\mathbf{profit}(\{S^{id}\})$ is the profit obtained by CSPs from $q^{S^{id}}$ number of resources while not being part of any federation.

From above Eq. (13) it is guaranteed that each CSP while being part of the federation will at least get the profit that it used to get alone. The extra profit obtained by federation is distributed among member CSPs based on their contribution of resources in federation.

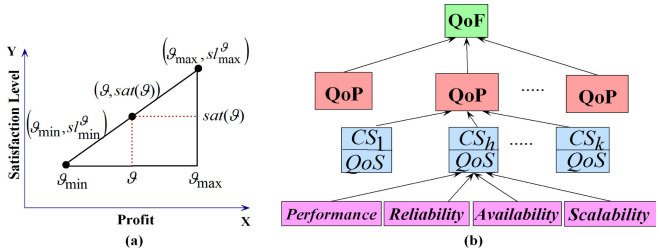

Fig. 2. (a) Satisfaction level based on profit, (b) Quality of federation.

3.5 Determination of Satisfaction Level

In economics satisfaction level is the satisfaction that any customer receives from any basket of goods [22]. In this paper we have used satisfaction level to measure satisfaction of (a) cloud service provider S^{id} in federation f_j ($v_{f_j}(S^{id}.sat)$) (b) a federation f_j ($f_j.sat$). *Profit* and *availability* are two important parameters for both CSPs and federation. Let us first highlight the importance of *Profit*. During low resource demand, CSPs can make extra profit by sharing their idle or underutilized resources with other CSPs. Thus CSPs will not take part in federation if they are not able to use their idle resources and earn some profit. Further, maximizing overall profit of federation will also help member CSPs to maximize their individual profit in federation. Next, *availability* is considered to be the most important QoS parameter for CSPs while forming federation. This is because, CSPs take part in federation to increase their availability of resource during peak time. Thus they are able to dynamically scale-up their resources' capacity and guarantee greater availability (uptime) of resource (virtual machine) to the cloud users [23]. Further, increasing overall availability of federation will also increase availability of individual CSPs in federation. Therefore to model satisfaction level, we have used two parameters *profit* $\vartheta \in \{\phi_{S^{id}}(f_j), \mathbf{P}(f_j)\}$ and *availability* $\xi \in \{f_j.A\}$. The satisfaction obtained for the profit ϑ (assuming satisfaction level increases with the increase in profit) can be calculated geometrically as shown in Fig. 2a and is given as follows:

$$\frac{(\vartheta - \vartheta_{min})}{(\vartheta_{max} - \vartheta_{min})} = \frac{sat(\vartheta) - sl_{min}}{(sl_{max} - sl_{min})} \tag{14}$$

$$sat(\vartheta) = \begin{cases} sl_{min} + \frac{|(\vartheta - \vartheta_{min})|}{|(\vartheta_{max} - \vartheta_{min})|} \cdot (sl_{max} - sl_{min}) \\ \qquad \vartheta_{\min} \leq \vartheta \leq \vartheta_{\max} \\ 1 \qquad \vartheta > \vartheta_{\max} \\ 0 \qquad \vartheta < \vartheta_{\min}. \end{cases} \tag{15}$$

Where,

- ϑ_{max} denotes the maximum expected profit. We assume maximum profit to be 200 percent of cost price
- ϑ_{min} denotes the minimum expected profit. We assume minimum profit to be 1 percent of cost price

TABLE 2
Observed Quality ($QoS^i_{attr_j}$) Data of CS_l for Different QoS Attributes

Days	(QoS_{attr})			
	performance (QoS_{attr_1})	reliability (QoS_{attr_2})	availability (QoS_{attr_3})	scalability (QoS_{attr_4})
Day1 ($QoS^1_{attr_j}$)	0.82	0.75	0.91	0.94
...	...	...	...	...
Day30 ($QoS^{30}_{attr_j}$)	0.89	0.92	0.86	0.90

- sl_{max} denotes maximum satisfaction level and value considered is 1.
- sl_{min} denotes minimum satisfaction level and value considered is .01.

Similarly we can calculate the satisfaction level based on availability ξ i.e., $sat(\xi)$ (assuming satisfaction level increases with the increase in availability). The value of ξ_{max} is 1 (maximum availability (uptime)) and ξ_{min} is 0.2 (minimum value for availability). The overall satisfaction level is calculated as a weighted average of $sat(\vartheta)$ and $sat(\xi)$. Therefore using Eq. (15), we can model total satisfaction level (based on *profit* and *availability*) of CSPs and federation as follows.

- Satisfaction level of a CSP S^{id} in a federation f_j

$$v_{f_j}(S^{id}.sat) = sat(\phi_{S^{id}}(f_j)) \times w_{\phi_{S^{id}}(f_j)} + sat(f_j.A) \times w_{f_j.A}, \tag{16}$$

where, $w_{\phi_{S^{id}}(f_j)} + w_{f_j.A} = 1$.

- Satisfaction level of a federation f_j

$$f_j.sat = sat(\mathbf{P}(f_j)) \times w_{\mathbf{P}(f_j)} + sat(f_j.A) \times w_{f_j.A}, \tag{17}$$

where, $w_{\mathbf{P}(f_j)} + w_{f_j.A} = 1$.

4 Quality and Trust Estimation

Cloud providers may gain economic benefits by providing cloud services through a federation instead of providing services individually as discussed in Section 3. However, the presence of untrusted CSPs within a federation will put obstacle towards adopting the federation, since the untrusted CSPs will degrade the overall Quality of Federation (QoF).

4.1 Quality

We have defined quality of federation in a hierarchical fashion as shown in Fig. 2b. Each cloud service CS_l consists of different QoS attributes (QoS_{attr_j}) like performance, reliability, availability, scalability, represented by the set QoS_{attr} i.e., ($QoS_{attr_j} \in QoS_{attr}$). Generally, QoS of each cloud service depends on the measured quality of each attribute. Here, quality of each attribute denotes the measure of delivered QoS for each attribute in the range [0, 1] and is denoted by $\{QoS^i_{attr_j}\}^t_{i=1}$ where, t is number of days. As an example, a snapshot of the measured quality $QoS^i_{attr_j}$ of QoS_{attr_j} for a cloud service say, CS_l during a month ($t = 30$) is given in Table 2. Finally in Eq. (18) the overall QoS of the cloud service CS_l of any ith day say, d_i is estimated by taking average of all $QoS^i_{attr_j}$ of ith day and is given as follows:

$$d_i = \frac{\sum_{j=1}^{|QoS_{attr}|} QoS^i_{attr_j}}{|QoS_{attr}|}. \tag{18}$$

TABLE 3
Estimated Overall QoS Data (i.e., d_i) of Cloud Service CS_l of any ith Day

day1 (d_1)	day2 (d_2)	day3 (d_3)	day4 (d_4)	day5 (d_5)	day6 (d_6)	day7 (d_7)
0.855	0.892	0.910	0.877	0.897	0.852	0.877
...	...	...	...	...	...	...
day24 (d_{24})	day25 (d_{25})	day26 (d_{26})	day27 (d_{27})	day28 (d_{28})	day29 (d_{29})	day30 (d_{30})
0.887	0.852	0.887	0.882	0.865	0.892	0.857

Here, $|QoS_{attr}|$ denotes the cardinality of set QoS_{attr}. The estimated QoS values for service CS_l with the example data (a snapshot is given in Table 2) are tabulated in Table 3.

Quality of provider (QoP) depends on the quality of each service provided by that provider (see Fig. 2b). Let $d = \{d_i\}^X_{i=1}$ denote the i.i.d (independent identically distributed) of observed data representing measurement of QoS in the range [0, 1] where X denotes the set of observed data of each cloud service as shown in Table 3. Here range 1 denotes the delivered quality of attributes to be 100 percent and range 0 denotes the delivered quality of attributes to be 0 percent. For each cloud service CS_l, let $\{d_{1l}, d_{2l}, \ldots, d_{Xl}\}$ be a vector of QoS from different time intervals (days) and these vectors are considered as distributions of CS_l. Each attribute i.e., QoS_{attr} can be viewed as an i.i.d variable. It is observed that the distribution satisfy the following two conditions (a) the QoS values of each cloud service lies within [0, 1] and (b) distribution of sample data of different CS_l (Table 3) may have different types of shape. We have used beta distribution for our statistical estimation of delivering QoP (Eq. (21)) as it satisfies the above mentioned conditions. Moreover, the standard Beta distribution with two shape parameters α and β is most frequently used probability distribution in statistics. One important advantage of Beta distribution is that its density function can take different shapes on the unit interval [0, 1] [24]. The beta distribution can also have positive as well as negative skewness based on the values of α and β. For detail explanation of Beta mixture model, refer to [25],[26]. Thus, distribution d of all CS_l follows a mixture density and QoP of any single S^{id} can be given by the density function

$$S^{id}.QoP(d) = \sum_{l=1}^{\kappa} \omega_l Beta_l(d; \alpha_l, \beta_l). \tag{19}$$

Where κ denotes the number of cloud services (components) of the corresponding service provider S^{id} in the mixture, α_l and β_l ($\alpha_l, \beta_l > 0$) denote the shape parameters of lth component. $Beta_l(\cdot)$ denotes the lth beta distribution and ω_l being the mixing weight, $\omega_l > 0$, $\sum_{l=1}^{\kappa} \omega_l = 1$. The density function of lth component (cloud service) of Beta distribution is given by:

$$Beta_l(d \,|\, \alpha_l, \beta_l) = \frac{\Gamma(\alpha_l + \beta_l)}{\Gamma(\alpha_l)\Gamma(\beta_l)} d^{\alpha_l - 1}(1-d)^{\beta_l - 1}. \tag{20}$$

From above equation $\Gamma(\cdot)$ denotes the gamma function and it is expressed by $\Gamma(\gamma) = \int_0^\infty b^{\gamma-1} exp(-b) db, b > 0$. We augment the data by introducing the latent indicator variable z_{il}, $(i = 1, \ldots, X), (l = 1, \ldots, \kappa)$ for each d_i. If $z_{il} = 1$, then it signifies that d_i comes from the components (cloud services) of the mixture, otherwise $z_{il} = 0$. Therefore the set of values z_{il} and

d_i together form a complete data. We assume every single vector $z_i = (z_{i1}, z_{i2}, \ldots, z_{i\kappa})$ is independent and identically distributed with probability of weight $\omega = (\omega_1, \ldots, \omega_\kappa)$. This is considered as prior distribution for z_i. We assume $\Theta = (\alpha_1, \beta_1, \ldots, \alpha_\kappa, \beta_\kappa)$ as a vector consisting of all unknown parameters α and β of κ number of components (cloud services).

We have used the expectation-maximization (EM) algorithm for the maximum likelihood estimation of the vector of unknown parameters (like α_l, β_l and ω_l) of the Beta mixture model [25],[26]. Based on estimated values of Beta mixture model (BMM) for specific service provider S^{id}, the probability density function of $QoP(d)$ is determined between [0.5, 1], which is the upper half of probability density function. Therefore $QoP_{delivered}$ of any S^{id} will be calculated as

$$S^{id}.QoP_{delivered} = \int_{0.5}^{1} \sum_{l=1}^{\kappa} \omega_l Beta_l(d, \alpha_l, \beta_l). \quad (21)$$

QoF depends on QoP of all service providers being part of that federation. Therefore QoF of federation f_j is given by:

$$f_j.QoF = \frac{\sum_{S^{id} \in f_j} (S^{id}.QoP_{delivered})}{|f_j|}. \quad (22)$$

4.2 Trust

Trust of a specific CSP determines the extent to which an existing CSP can be trusted to deliver services in federation and is defined by the equation below:

$$S^{id}.Trust = 1 - (S^{id}.QoP_{committed} - S^{id}.QoP_{delivered}). \quad (23)$$

Where,

- $S^{id}.QoP_{committed}$: is the measure of quality of service that service provider S^{id} promised to provide.
- $S^{id}.QoP_{delivered}$: is the measure of quality of service that service provider S^{id} originally delivered

The strength of trust will increase as the value of $S^{id}.Trust$ increases. Trust is highest at $S^{id}.Trust = 1$ and minimum when $S^{id}.Trust = 0$.

5 Cloud Federation Formation as a Hedonic Coalition Game

This section formulates the problem of cloud federation formation as a hedonic coalition game with non transferable utility and analyzes the properties of the game. The objective is to form a set of trusted coalition (federation) partition which restrains the untrusted CSPs to take part in federation and maximize the overall profit and quality (like availability) of federation by maximizing satisfaction level of federation.

5.1 Formulation of Hedonic Coalition Game

In game theory, a coalition game is a framework for modeling complex interactions among players where players' might collaborate with other players, by forming coalition. Coalition games are very often distinguished based on the utility or payoff that they allocate to each coalition. Hence, coalition game can either be transferable utility (TU) or non-transferable utility (NTU). In transferable utility game, the total utility obtained by any coalition $f \subseteq \eta$ can be shared in any manner among the members of that coalition (e.g., money). Whereas, in non-transferable utility game the utility of the coalition is non-dispensable and non -transferable (e.g., satisfaction, happiness) [8]. Therefore, coalition game with non-transferable utility can be defined as a pair (η, v) where η is the finite set of players and v is a characteristic function such that for any coalition $f_j \in \Upsilon$, $v(f_j)$ is a closed convex subset of $\mathbb{R}^{|f_j|}$ that contains the set of utility vectors that players in f_j can acquire [28].

Next, we mapped a cloud federation formation into a coalition game (η, v) with non transferable utility as follows:

- *Player:* Each player $S^{id} \in \eta$ denotes a CSP and each coalition corresponds to a federation.
- *Utility Function:* The characteristic function v quantifies the utility of a coalition. The utility of each CSP in coalition represents the CSP's satisfaction level while being part of a coalition and is defined in (Eq. (16)).

In the following we discuss some properties of the coalition game in cloud federation:

Property 1. *The proposed coalition game in cloud federation is a non transferable utility game.*

The utility of each CSP in coalition represents their satisfaction level obtained from that coalition. Whereas, satisfaction level is the satisfaction that any CSP receives from that coalition. Therefore satisfaction of any CSP in a coalition neither can be distributed nor can be transferred among the other member CSPs of coalition. Hence the proposed game is NTU game.

A hedonic game is a particular type of NTU game in which players have preference over which coalition they belong to and utility of any player in a coalition solely depends on the identity of the players in their coalition, but not on how other players are partitioned [27]. The players in hedonic games are typically understood to be self-interested, as they join the coalition based on their preference over the possible coalitions. Further, the coalition partition formed by hedonic coalition game is *Nash stable* and *individually stable* (see Section 5.2.1). Thus we believe that, this class of hedonic coalition formation game where (i) players preferences over partitions depend only on the members of his coalition and (ii) formed coalition partition is Nash stable and individually stable, is the best type of coalition game that can model federation (coalition) among the CSPs (players) based on their satisfaction level. The two key conditions which classify coalitional game as a hedonic game are as follows [27]:

Condition 1. *A coalition formation game is classified as hedonic if:*

(1) The utility of any player in a certain coalition depends solely on the members of that coalition.
(2) The coalition are formed based on the preference of the players over their possible coalitions' set.

Property 2. The proposed coalition game is a class of hedonic game.

The utility of each CSP in coalition represents its satisfaction level obtained from that coalition. Further, satisfaction level of CSP is the function of availability and profit. Therefore, availability of each CSP in coalition denotes the availability of coalition itself and from (Eq. (6)) it can be observed that the availability of federation depends on the member of coalition. Next, profit obtained by each CSP also depends on the members of coalition to which it belongs. This is because, from Eq. (8) it can be noted that, the incurred cost of any resource solely depends on the value of cost and availability of all the CSPs in coalition. Therefore,

incurred cost of resource delivered through coalition changes if member of the coalition changes. Thus, changing overall profit of coalition. Therefore, individual profit of CSPs will also change if the member of coalition changes. Thus, satisfaction level of CSPs in coalition will also change if member of coalition changes. Hence, utility of each CSP in a certain coalition solely depends on the members of that coalition, which satisfies the first condition.

For the second condition, in the remaining of this section, we are going to formally outline the player's preference relation over the coalition and compare them throughout the coalition formation process. However, before discussing the preference function, let us define initially the concept of coalition partition and thereafter preference relation [27].

In a hedonic coalition game, player chooses a coalition that the player prefers to be member of according to their preference over possible coalition partition, which is a set of coalition formed by partitioning η and defined as $\Upsilon = \{f_1, f_2, f_3, \ldots f_k, \ldots, f_y\}$, where $f_k \cap f_{k'} = \emptyset$ for $k \neq k'$ and 'y' is the total number of coalition partition for $1 \leq y \leq \eta$, and $\cup_{k=1}^{y} f_k = \eta$. Therefore, the concept of preference relation according to [27] can be define as follows:

Definition 1 (Preference Relation). *The preference of any player S^{id} can be represented by $(\succeq_{S^{id}})_{S^{id} \in f_k \subseteq \eta}$ (a complete, reflexive, and transitive binary relation). For any player $S^{id} \in \eta$, $f_k \succeq_{S^{id}} f_l$, denote that player S^{id} strongly prefers to be member of coalition f_k over coalition f_l or at least prefers to be member of both equally. Here $f_k \succ_{S^{id}} f_l$ denotes strict preference of player S^{id} to be a member of coalition f_k over coalition f_l.*

Based on the above definition of preference relation, the preference function of CSP S^{id} can be given as follows:

$$f_k \succeq_{S^{id}} f_l \Longleftrightarrow \mathscr{V}_{S^{id}}(f_k) \geq \mathscr{V}_{S^{id}}(f_l), \tag{24}$$

where, both $f_k \subseteq \eta$ and $f_l \subseteq \eta$ are any two coalition to which CSP S^{id} belongs and $\mathscr{V}_{S^{id}} : 2^{\eta} \longmapsto \mathbb{R}$ is a preference function for any CSPs S^{id} given as

$$\mathscr{V}_{S^{id}}(f) = \begin{cases} v_{f_j}(S^{id}.sat), & S^{id} \notin h(S^{id}) \\ -\infty & \text{otherwise,} \end{cases} \tag{25}$$

where, $v_{f_j}(S^{id}.sat)$ is given by Eq. (16), and $h(S^{id})$ denotes the history set of CSP S^{id}. The $h(S^{id})$ contains those coalition that CSP S^{id} has joined and then left, before the formation of current coalition partition Υ. The main reason behind defining preference function in Eq. (25) is to enable each CSP S^{id} to prefer that coalition in which its satisfaction is maximum. Further, it is to be noted from Eq. (25) that the preference value assigned by CSP S^{id} to a certain coalition, which is not in $h(S^{id})$ is equal to the value of utility that S^{id} obtains in this coalition. However, it is also noted from Eq. (25) that the preference value of S^{id} is $-\infty$ when it revisits the same coalition that it had left previously and thus S^{id} has no incentive to revisit the same coalition present in history set (This may be thought of as a basic learning process). In summary, a CSP prefers that coalition in which it achieves maximum value of satisfaction level and coalition with better satisfaction level is not present in history of player S^{id}.

5.2 Cloud Federation Formation Mechanism

In this section, we present hedonic coalitional game inspired cloud federation formation (CGCFF) algorithm that allows each trusted CSP to join with the federation in which it achieves maximum satisfaction such that satisfaction level of federation also increases in the final coalition partition. The CGCFF algorithm (Algorithm 1) is presented as follows. Initially CGCFF algorithm takes η as input and output final coalition partition Υ for each round. Instead of a single deterministic run, the CGCFF algorithm runs continuously managing the dynamic environment. However, in every round it generates a stable coalition partition. Initially the partition is empty and CGCFF calls $join(S^{id}, \Upsilon)$ method for each trusted $S^{id} \in \eta$ to build the initial coalition partition during first round. For the service provider under consideration, $join(S^{id}, \Upsilon)$ method tries to find the best federation (if any available in Υ) as per the preference rules. If there is any such federation for that service provider, then $join(S^{id}, \Upsilon)$ method will form a new federation by merging them, otherwise will keep that service provider as an identity federation. Thereafter, CGCFF calls $stabilize(\Upsilon)$ method to check whether any CSP wishes to leave its current federation and form a new federation, either a singleton or jointly with another set of CSP's. It is to be noted that, the $stabilize(\Upsilon)$ method continues until existing coalition partition converges to a Nash stable coalition partition.

After forming Nash stable coalition partition in the first round, next from the second round onward CGCFF continues to observe the status (QoS, QoP, QoF) of each CSP and federation, as their status may change over time. Such changes can affect the quality and trust values of the CSPs and for this CGCFF calls $updateQualityTrust(\Upsilon)$. Moreover, the cloud environment is very dynamic in nature, both from the resource request and resource usage point of view. Thus, to observe the changes in the environment, CGCFF periodically monitors ($monitor(\Upsilon)$) the underlying environment. Furthermore, to consider new CSPs in the environment CGCFF calls $getNewSPrequests()$. Once the number of newly joined CSP's has exceeded the threshold θ, the method $stabilize(\Upsilon)$ is again called to rearrange the partition into a Nash-stable coalition partition. It may be noted that the addition of a single CSP to the Nash-stable structure created by $stabilize(\Upsilon)$ does not destroy the stability of the system. Hence, practically, when $\theta > 1$ then more CSPs are allowed to join the federation partition and thereby lead the system towards a potentially unstable state where a CSP may prefer another federation to its current one.

CGCFF monitors the environment to know if there arises the following situations (i) any federation whose resource availability is below a threshold value (ii) any unstable federation in terms of QoF and satisfaction level and (iii) any service provider whose satisfaction level in federation is below its individual satisfaction level. If the resource availability of a federation is less than a predefined threshold, then CGCFF calls $mergeFD(f, \Upsilon)$ to find a best federation for merge. On the other hand, when the QoF or the satisfaction of any federation is less than the predefined thresholds, CGCFF removes the weak CSPs (in terms of minimum QoP) from that federation to restore its QoF and satisfaction at some higher level. The removed CSPs are collected in the list $Splitlist$. This is done by the function $split(f)$. The service providers are also removed and collected in the $Splitlist$ if the satisfaction level of service provider is higher in identity federation ($f_{\mathbf{I}}$) than the satisfaction level of service provider in federation f_j ($v_{f_j}(S^{id}.sat) \leq v_{f_{\mathbf{I}}}(S^{id}.sat)$). Finally, the CSPs in $Splitlist$ are merged with some federation in Υ based on the defined preference rules. This is done by the function $mergeSP(Splitlist, \Upsilon)$.

Algorithm 1. CGCFF(η)

Input: η, a set of providers
Output: Υ, a stable partition
$\Upsilon \longleftarrow \{\emptyset\}$;
for all $S^{id} \in \eta$ **do**
  while $S^{id}.Trust \geq Threshold_{Trust}$ **do**
    $join(S^{id}, \Upsilon)$;
  end while
end for
$stabilize(\Upsilon)$;
REPEAT
{
$updateQualityTrust(\Upsilon)$;
$monitor(\Upsilon)$;
$\eta' = getNewSPrequests()$;
for all $S^{id*} \in \eta'$ **do**
  while $S^{id*}.Trust \geq Threshold_{Trust}$ **do**
    join(S^{id*}, Υ); // joining of new CSPs S^{id*}
  end while
end for
if $newlyjoinedSP > \theta$ **then**
  $stabilize(\Upsilon)$;
end if
}

Algorithm 2. stabilize(Υ)

REPEAT
{
$PartitionUnchanged \longleftarrow TRUE$;
for all $f \in \Upsilon$ **do**
for all $S^{id} \in f$ **do**
  $singleton \longleftarrow FALSE$;
  $f \longleftarrow f \backslash \{S^{id}\}$; // S^{id} is taken out from federation f
  if $f = \{\}$ **then**
    $singleton \longleftarrow TRUE$;
    $\Upsilon \longleftarrow \Upsilon \backslash \{f\}$;
  end if
  $join(S^{id}, \Upsilon)$; //S^{id} join to any federation
  $ProviderStabilized \longleftarrow FALSE$;
  if $singleton$ **then**
    if $\{S^{id}\} \in \Upsilon$ **then**
      $ProviderStabilized \longleftarrow TRUE$; //$S^{id}$ again form singleton federation
    end if
  else
    if $S^{id} \in f$ **then**
      $ProviderStabilized \longleftarrow TRUE$; //$S^{id}$ join to same federation f
    end if
  end if
  $PartitionUnchanged \longleftarrow PartitionUnchanged \wedge ProviderStabilized$;
end for
end for
}
UNTIL $PartitionUnchanged$ //method $stabilize(\Upsilon)$ continues to repeat until partition Υ is Unchanged

The main computational complexity of CGCFF algorithm lies in the $stabilize(\Upsilon)$ method, i.e., the process of converging existing partition to a Nash stable coalition partition at each round. The outer for loop of $stabilize(\Upsilon)$ method iterates over all federations f in Υ, which is at most n (for singleton federations). The inner for loop iterates over all members of the federation f, which is again at most n (for a grand federation). The two nested for loops, can however be run over all combinations of federations feasible in Υ. The number of such combinations is O(2^n). Thus, the worst-case complexity of $stabilize(\Upsilon)$ is O($n^2 \times 2^n$), although it may in practice converge much faster if conditions are propitious.

5.2.1 Analysis of the Federation Game

Wu et al. [28] and Wahab et al. [8] analyzed three properties of hedonic game [27] for their proposed hedonic game based mechanism. Similarly, in this section, we have also analyzed these properties of the hedonic game with respect to our proposed CGCFF algorithm i.e., (i) CGCFF converges to a final coalition partition (ii) the final coalition partition is Nash stable and (iii) the final coalition partition is also individually stable.

Algorithm 3. join(S^{id}, Υ)

maxSat $\longleftarrow v_{f_I}(S^{id}.sat)$; // single CSP is considered as identity federation
$bestFD \longleftarrow \emptyset$;
$h(S^{id}) \longleftarrow \{\emptyset\}$;
for all $f \in \Upsilon$ **do**
  $dummyFD \longleftarrow f \cup S^{id}$;
  $dummyFD \longleftarrow arrangementOrder(dummyFD)$;
  if $(v_{dummyFD}(S^{id}.sat) \geq maxSat \,\&\, dummyFD.sat \geq f.sat)$ **then**
    maxSat $\longleftarrow v_{dummyFD}(S^{id}.sat)$;
    $bestFD \longleftarrow f$;
  end if
  S^{id} update $h(S^{id})$ by adding its recently left federation;
end for
if $bestFD \neq \emptyset$ **then**
  $\Upsilon \longleftarrow \Upsilon \setminus bestFD$;
  $bestFD \longleftarrow bestFD \cup S^{id}$;
  $bestFD \longleftarrow arrangementOrder(bestFD)$;
  $\Upsilon \longleftarrow \Upsilon \cup bestFD$;
else
  $newFD \longleftarrow \{S^{id}\}$;
  $\Upsilon \longleftarrow \Upsilon \cup newFD$;
end if

Algorithm 4. monitor(Υ)

for all $f \in \Upsilon$ **do**
  if $f.resource \leq f.Th_{resource}$ **then**
    $\Upsilon \longleftarrow \Upsilon \setminus f$
    mergeFD(f, Υ);
  end if
  if $(f.QoF \leq f.Th_{QoF} \parallel f.sat \leq f.Th_{sat})$ **then**
    SplitList $\longleftarrow$ split(f);
  end if
  for all $S^{id} \in f$ **do**
    if $v_f(S^{id}.sat) \leq v_{f_I}(S^{id}.sat) \parallel S^{id}.Trust \leq Threshold_{Trust}$ **then**
      $f \longleftarrow f \setminus S^{id}$;
      $f.size$ - - ;
      SplitList $\longleftarrow S^{id}$;
    end if
  end for
end for
if $SplitList.size > 0$ **then**
  mergeSP($SplitList, \Upsilon$);
end if

Theorem 2. *The CGCFF algorithm always converges to a final coalition partition (Υ^r_f) composed of a number of disjoint coalitions.*

Proof. CGCFF algorithm runs continuously and periodically at each round monitors the changes in the coalitions (due to join of new CSPs or change in QoS, QoP and trust of CSPs) of the coalition partition. So, let us consider any initial partition $\Upsilon_{initial} = \Upsilon_1^{r(t)}$ at time t of round r. The CGCFF calls $stabilize(\Upsilon_1^{r(t)})$ method for partition $\Upsilon_1^{r(t)}$ to find the Nash stable coalition partition. Hence the method $stabilize(\Upsilon_1^{r(t)})$ runs on the partition $\Upsilon_1^{r(t)}$ and checks whether any CSP wishes to leave its current coalition f_j and join any other coalition $f_k \in \Upsilon_1^{r(t)}$. It is to be noted that the $stabilize(\Upsilon_1^{r(t)})$ method of CGCFF algorithm is comprised of a sequence of switch operation (any CSP can leave its current coalition f_j and join any other coalition $f_k \in \Upsilon_1^{r(t)}$ based on the preference rule defined in Eqs. (24) and (25)). Hence for every switch operation the current partition $\Upsilon_1^{r(t)}$ at time t changes to another Thus $stabilize(\Upsilon_1^{r(t)})$ method continues to switch from one partition to another until there is no change in partition. Hence occurrence of any switch operation in $stabilize(\Upsilon_1^{r(t)})$ method leads to a new partition. Further, based on the fact that the number of coalitions in a partition is finite and given by the Bell number [29], it can be said that the number of switch operations in $stabilize(\Upsilon_1^{r(t)})$ method is finite and the switch operations will gradually lead to a final partition Υ_f^r. Hence CGCFF algorithm always converges to a final partition Υ_f^r consisting of disjoint federations at round r. □

The stability of the final partition Υ_f^r formed by the convergence of $stabilize()$ method of CGCFF algorithm, can be analyzed by using the stability concept of hedonic game [27]. First we define some useful definition and thereafter we analyze the properties of hedonic game.

Algorithm 5. mergeFD(f, Υ)

maxSat $\longleftarrow f.sat$;
bestFD $\longleftarrow \emptyset$;
for all $f' \in \Upsilon \mid f \neq f'$ **do**
  $dummyFD \longleftarrow f' \cup f$;
  if $(dummyFD.sat \geq maxSat)$ & $(dummyFD.sat \geq f'.sat)$ & $(dummyFD.A \geq f.A)$ & $(dummyFD.A \geq f'.A)$ **then**
    maxSat $\longleftarrow dummyFD.sat$;
    $bestFD \longleftarrow f'$;
  end if
end for
if $bestFD \neq \emptyset$ **then**
  $\Upsilon \longleftarrow \Upsilon \setminus bestFD$;
  $bestFD \longleftarrow bestFD \cup f$;
  $bestFD \longleftarrow arrangementOrder(bestFD)$;
  $\Upsilon \longleftarrow \Upsilon \cup bestFD$;
else
  $\Upsilon \longleftarrow \Upsilon \cup f$;
end if

Definition 2. *The partition Υ is Nash-stable if $\forall S^{id} \in \eta$, $f_\Upsilon(S^{id}) \succeq_{S^{id}} f_j \cup \{S^{id}\}\ \forall f_j \in \Upsilon \cup \{\emptyset\}$.*

In other words, a coalition partition Υ is said to be Nash stable if no player has an incentive to leave its current coalition f_m and join some other coalition f_n in Υ or act alone in such a way that the existing coalition partition has changed, assuming other coalitions have not changed.

Definition 3. *The partition Υ is individually stable if there do not exist any $S^{id} \in \eta$ and a coalition $f_k \in \Upsilon \cup \{\emptyset\}$, such that $f_k \cup \{S^{id}\} \succ_{S^{id}} f_\Upsilon$ and $f_k \cup \{S^{id}\} \succeq_{S^{id}} f_k$, where f_Υ denotes the $f_i \in \Upsilon$ such that $S^{id} \in f_k$.*

In other words, a coalition partition Υ is said to be individually stable if no player (S^{id}) has an incentive to leave its existing federation to join another existing federation f_k (possibly empty) without making the members of f_k worse off. Further, from [27], it can be shown that every Nash-stable coalition partition is also an individually stable.

Algorithm 6. split(f)

$SplitList \longleftarrow (\emptyset)$;
while $(f.QoF \leq f.Th_{QoF} \parallel f.sat \leq f.Th_{sat}) \&\& (f.size > 0)$ **do**
  $W_{S^{id}} \longleftarrow findWeakSP(f)$; // Returns S^{id} with minimum QoP
  $SplitList \longleftarrow SplitList \cup W_{S^{id}}$;
  $f \longleftarrow f \setminus W_{S^{id}}$;
  $f.size$ - - ;
end while

Algorithm 7. mergeSP($SplitList$, Υ)

for all $S^{id} \in SplitList$ **do**
  if $S^{id}.Trust \geq Threshold_{Trust}$ **then**
    join(S^{id}, Υ);
  end if
end for

Theorem 3. *The final partition Υ_f^r produced by $stabilize()$ method of CGCFF algorithm is Nash stable and individually stable.*

Proof. Assume that the final partition Υ_f^r output by the $stabilize()$ method of proposed CGCFF algorithm at any round r is not Nash-stable. However, there exist a CSP S^{id} and federation $f_k \in \Upsilon_f^r$ such that $f_k \cup S^{id} \succeq_{S^{id}} f_l$. Thus CSP S^{id} can carry out a switch operation, which results in changing of existing partition Υ_f^r to new partition $\Upsilon_{f(*)}^r$ such that $\Upsilon_{f(*)}^r \neq \Upsilon_f^r$. Thus, contradicting with the fact that Υ_f^r is the result of the convergence of the $stabilize()$ method of CGCFF algorithm (Theorem 2). Therefore, any partition Υ_f^r which is formed by $stabilize()$ method of proposed CGCFF algorithm is Nash-stable and thus from [27], this formed partition Υ_f^r is also individually stable. □

Proposition 1. *A federation $f_l \in \Upsilon$ formed by the CGCFF algorithm is trusted.*

Proof. A federation is said to be trusted if its QoF is greater than threshold QoF and trust level of each member of these federation is above threshold level i.e., $f_l.QoF \geq f.Th_{QoF}$ and $\forall S^{id} \in f_l$, $S^{id}.Trust \geq Threshold_{Trust}$. QoF of each federation depends on the value of QoP of each CSP. If value of QoP of any CSP decreases (in federation) then, overall QoF value of that federation will also decrease. Initially in CGCFF only those CSPs S^{id} such that $S^{id}.Trust \geq Threshold_{Trust}$ are allowed to join the partition Υ. In the procedure $monitor(\Upsilon)$ all federation f_l such that $f_l.QoF \leq f.Th_{QoF}$ are split by extracting the weakest CSP, i.e., the CSP with minimum QoP. In the procedure $mergeSP()$ these weak CSPs are allowed to rejoin the partition if their trust level is above the threshold, but if their QoP values are still low and result in the QoF values of the federations they join to fall below the corresponding threshold, they will

TABLE 4
Parameters

Parameter	Value
η	20
$Threshold_{Trust}$	0.5
$f.Th_{resource}$	20% of capacity
$f.Th_{QoF}$	0.5
$f.Th_{sat}$	0.5
large instance (vCPU, Memory(GB), Storage(GB))	2, 8, 100
xlarge instance (vCPU, Memory(GB), Storage(GB))	4, 16, 200
2xlarge instance (vCPU, Memory(GB), Storage(GB))	8, 32, 500
large instance (availability, chargeable price($)($\rho_{large}$))	1, 0.215
xlarge instance (availability, chargeable price($)($\rho_{xlarge}$))	1, 0.431
2xlarge instance (availability, chargeable price($)($\rho_{2xlarge}$))	1, 0.862

once again be expelled from the partition by the *monitor* procedure. Hence in the stable state of the partition there exist no federation f_l such that $f_l.QoF < f.Th_{QoF}$ and no CSP S^{id} such that $S^{id}.Trust < Threshold_{Trust}$. Consequently all federations $f_l \in \Upsilon$ are trusted. □

6 Experimental Evaluation

In this section, we first describe the experimental setup used to perform the simulation and then evaluate the result of the proposed cloud federation formation mechanism by extensive experiments.

6.1 Experimental Setup

We have conducted our experiment on Intel(R) Xeon(R) CPU E3-1226 v3 @ 3.30 GHz processor with 32 GB of RAM in 64-bit Windows 7 environment. We have compared the performance of CGCFF with other three existing mechanisms, namely (i) Profit-based Cloud Federation Formation Mechanism (CFFM) [7], (ii) Malicious-based Hedonic Coalition Formation (MHCF) [8] and (iii) QoS-based coalition formation mechanism (QCFM) [9]. The CFFM mechanism considers the price and cost of each CSP and tries to maximize overall profit of the formed federation. MHCF mechanism considers maliciousness of CSPs while forming federation. Here, malicious members are those who refuse to share their resources with the other members of the coalition. Their main aim is to find the coalition structure that minimizes the malicious members. QCFM mechanism considers many QoS parameters like availability, response time, reliability in the community formation process. However, the comparison is feasible, as all of these mechanisms are based on coalition game. All the algorithms are written in Python and for the purpose of estimating parameters (α, β) of the Beta mixture model, we have used statistical software R. The coalition formation model used in CFFM, MHCF and QCFM are very different from the proposed approach, but we have tried to make the experiment environment as fair and comparable to ours as possible. We have used similar data and eight numbers of CSPs as used by CFFM [7] and MHCF [8]. The chargeable price of virtual machine (VM) is taken similar to the price of VM charged by Amazon EC2 [11]. The cost of VM is assumed to be half of the chargeable price. The price of eight different regions of Amazon EC2 represents the chargeable price of each CSP and is given in Table 5. The QoS data of each CSP of last 30 days are partially collected from CloudHarmony [10] dataset and partially generated synthetically. CloudHarmony has maintained QoS data like availability of different CSPs for the past 30 days. The generated data is used to estimate the quality of delivered services for each CSP. In this simulation, we have considered 3 types of VM instances (large, xlarge and 2xlarge) of the General Purpose, Current Generation of Amazon EC2 identified by the series m3. We assume each CSP contributes at least 5 instances of each type (large, xlarge, 2xlarge) in the federation. Some of the parameters used in our experiment and its corresponding values are given in Table 4. Some of the measured parameters used for comparison are given below:

- Average satisfaction level of federation: $\frac{\sum_{f_j \in \Upsilon}(f_j.sat)}{|\Upsilon|}$.
- Average quality of federation: $\frac{\sum_{f_j \in \Upsilon}(f_j.QoF)}{|\Upsilon|}$.
- Average availability of federation: $\frac{\sum_{f_j \in \Upsilon}(f_j.A)}{|\Upsilon|}$.
- Average profit of federation: $\frac{\sum_{f_j \in \Upsilon}(\mathbf{P}(f_j))}{|\Upsilon|}$.
- Average individual satisfaction level of CSPs in federation: $\frac{\sum_{f_j \in \Upsilon}\sum_{S^{id} \in f_j}(v_{f_j}(S^{id}.sat))}{|\eta|}$.
- Average individual profit of CSPs: $\frac{\sum_{f_j \in \Upsilon}\sum_{S^{id} \in f_j}(\phi_{S^{id}}(f_j))}{|\eta|}$.
- Average size of all federation: $\frac{\sum_{f_j \in \Upsilon}(|f_j|)}{|\Upsilon|}$.

6.2 Experimental Results

First, we study in Fig. 3 the performance of the different mechanisms CGCFF, MHCF, CFFM and QCFM when all the CSPs are trusted (see Section 4.2). It is observed that, the average satisfaction of federation partition formed by CGCFF is maximum compared to CFFM, followed by QCFM and then MHCF as shown in Fig. 3a. This is because, the focus of CGCFF is to maximize profit (Fig. 3b) and availability (Fig. 3c) of federation simultaneously and hence maximizes the satisfaction of overall federation. In Fig. 3b, the overall average profit of federation partition formed by CGCFF is slightly better than CFFM followed by QCFM and MHCF. This is because in case of CFFM mechanism, at a

TABLE 5
Cloud Service Provider Prices, ($ per Hour)

Amazon EC2 Regions	US East (N. Virginia)	US West (Northern California)	EU (Ireland)	Asia Pacific (Singapore)	Asia Pacific (Tokyo)	Asia Pacific (Sydney)	South America (Sao Paolo)	AWS GovCloud (US)
CSP	S^1	S^2	S^3	S^4	S^5	S^6	S^7	S^8
large	0.133	0.154	0.146	0.196	0.193	0.186	0.19	0.168
xlarge	0.266	0.308	0.293	0.392	0.385	0.372	0.381	0.336
2xlarge	0.532	0.616	0.585	0.784	0.77	0.745	0.761	0.672

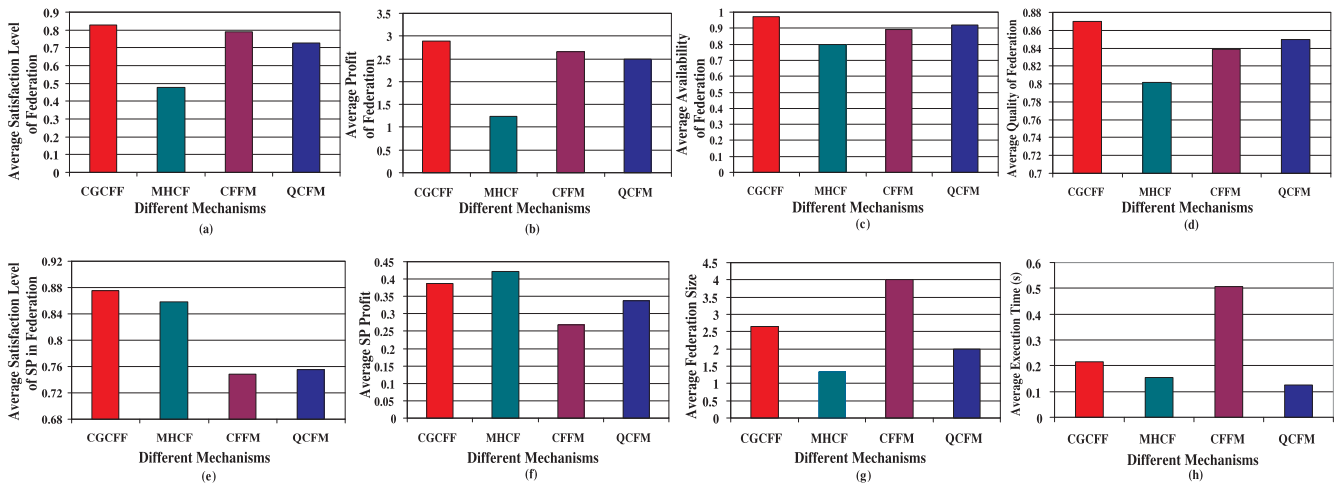

Fig. 3. Comparison of (a) Average satisfaction level of federations, (b) Average profit of federations, (c) Average availability of federations, (d) Average quality of federations, (e) Average satisfaction level of CSPs in federations, (f) Average profit of CSPs in federations, (g) Average sizes of federations and (h) Average execution time of federation formed by CGCFF, MHCF, CFFM and QCFM on the same set of 8 CSPs without considering the influence of untrusted CSPs.

time, a single partition is given as output, and it is observed that for each successive partition the profit of federation is always decreased when compared with the previous partition thus resulting in decrease of overall average profit of federation. Moreover, in terms of the average availability of the federation, as shown in Fig. 3c, CGCFF outperforms other mechanisms. This is due to the reason that, CGCFF mechanism main aim is to maximize the availability (uptime) of the federation, which is one of the important parameters in a cloud environment. Next in Fig. 3d, the average quality of the federation, formed by four mechanisms is compared. It is noticed that, CGCFF leads to QCFM following as a close second, followed by CFFM and then MHCF. Though, the main objective of QCFM mechanism is to increase the QoS of coalition partition, Fig. 3d shows that the average quality of federation formed by QCFM is not maximized. The reason is that QCFM mechanism follows a greedy approach to maximize the QoS value of coalition partition and hence consequently ends up finding the local maximum value of QoS for respective coalition partition. Furthermore, the average quality of federation formed by CFFM mechanism is lower compared to CGCFF and QCFM. This may be due to the fact that CFFM, in order to maximize profits of the federation, may select CSPs with low QoS value (CSPs with poor QoS can have less service delivery cost) thus decrease the overall quality of federation. Moreover, from Figs. 3b, 3c and 3d, it is also observed that, the average value of profit, availability and quality of federation obtained by MHCF mechanism is least compared to all four mechanisms. This is probably due to the presence of malicious CSPs as MHCF tries to minimize the maliciousness between CSPs and they have not considered the importance of maximizing overall profit, availability and quality of federation. Our main objective here is to show, profit and quality parameters are important for increasing overall satisfaction of federation, which is demonstrated through this experimental study.

Next, in Fig. 3e, the average satisfaction level of CSPs in federation is shown. It is observed that, CSPs in federation formed by CGCFF mechanism are more satisfied in terms of profit (see Fig. 3f) and availability (see Fig. 3c) compared to other three mechanisms. Fig. 3f shows that the average profit of CSPs in federation formed by MHCF mechanism is high compared to all other mechanisms. This is due to the fact that the average size (see Fig. 3g) of federation formed by MHCF mechanism is much less and thus the profit margin obtained by CSPs is more. In case of CFFM mechanisms the average profit is found to be lower. Here the reason is that CFFM in order to maximize profits of the federation, increases the size of the federation (see Fig. 3g). Thus, due to larger federation size, the share of profit obtained by CSPs in federation by CFFM decreases compared to other mechanisms. Fig. 3g shows the average federation size of all four mechanisms. From this set of experiments, it is observed that CFFM forms the largest federations, followed by CGCFF, QCFM and then MHCF. Fig. 3g shows that the size of federation formed by MHCF mechanism is not as much compared to all other mechanisms. This is because, MHCF mechanism tries to form federation such that maliciousness between member CSPs is minimized. Therefore leading to smaller size federation. In terms of execution time (Fig. 3h), CFFM has by far the highest complexity, followed distantly by CGCFF, MHCF and QCFM. Average execution time for CFFM is high as it finds the best possible combination of CSPs (out of an available set) using exhaustive search. Finally from above discussion of Fig. 3, it can be concluded that CGCFF mechanism performs better compared to other three mechanisms for most of the parameters.

Fig. 4 compares the performance of all four mechanisms when untrusted CSPs are present in the existing set of service providers. The effects of the untrusted CSPs have been compared as their percentage gradually increases from 0 percent (fully trusted case) to 20, 43, 60 percent and finally 80 percent. This is done by adding different numbers (2, 6, 12 and 32) of untrusted CSPs to the existing set of eight trusted CSPs. From Figs. 4a, 4b, 4c and 4d, it is depicted that, with the increase in number of untrusted CSPs, except average profit of federation (for some mechanism), the average satisfaction level, availability and quality of the federations formed by CGCFF are always steady and maximum compared to CFFM and QCFM which display fluctuating trends and MHCF which displays a steadily decreasing trend. The value of the satisfaction level of federation directly depends on the value of profit and availability of federation. Thus, from Figs. 4b and 4c, it is observed that,

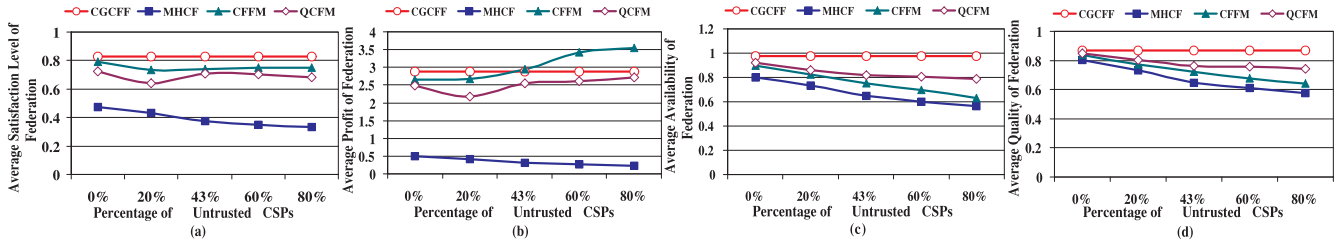

Fig. 4. Comparison of (a) Average satisfaction level of federations, (b) Average profit of federations, (c) Average availability of federations, (d) Average quality of federations, formed by CGCFF, MHCF, CFFM and QCFM when untrusted CSPs are increases from 0 to 80 percent.

with the increase in number of untrusted CSPs, there is an almost negligible effect on the average value of profit and availability of federation partition formed by CGCFF mechanism. The reason is that our mechanism considers only trusted CSPs during the federation formation process. Therefore, resulting in steady value for average satisfaction level (see Fig. 4a) of federation partition. Moreover, from Fig. 4b, it is observed that CFFM mechanism outperforms our proposed CGCFF mechanism when percentage of untrusted CSPs increases above 20 percent. The reason for this is as follows: First, the main objective of the CFFM mechanism is to maximize the overall profit of the formed federation partition. So CFFM always tries to maximize the profit of formed federation by forming federation of larger size. Second, when CFFM mechanism run on the data set consisting of untrusted CSPs, CFFM outputs a large size (present large number of CSPs) federation partition consisting of both trusted and untrusted CSPs. Hence as the size (number of CSPs) of the formed federation partition increases with the increase in the percentage of untrusted CSPs, the overall profit of the formed federation also increases. This is because, presence of more number of CSPs in federation will contribute more resources and as a result overall profit earned by federation will be more. However, CGCFF mechanism considers only trusted CSPs during federation formation process. Thus, with the increase in number of untrusted CSPs, there is no effect on the profit of federation partition formed by CGCFF mechanism. Therefore, resulting in steady value for average profit of federation partition. In case of MHCF mechanism, it is noticed that, the average satisfaction of federation decreases with the increase in number of untrusted CSPs. Figs. 4b and 4c show that the average value for profit and availability of federation decreases with the increase in number of untrusted CSPs. This is due to the reason that, first, MHCF mechanism does not maximize the overall profit, availability and quality of federation. Second, MHCF mechanism has not taken into account the trusted CSPs (in terms of delivering committed QoS to cloud users) while forming federation thus reducing overall profit, availability and quality (see Fig. 4d) of federation. Further, in case of CFFM and QCFM, from Fig. 4a it is observed that, with the increase in the percentage of untrusted CSPs, the average satisfaction level of federation for both the mechanisms decreases and is always lower when compared with the value of satisfaction level of federation in the absence of untrusted CSPs. However, though satisfaction level of federation formed by CFFM and QCFM mechanism is affected due to rise in number of untrusted CSPs, the main reason for this can be depicted from Figs. 4b and 4c which show the average profit and availability of federation (note that satisfaction level is function of profit and availability). Fig. 4b shows that there is not much effect on average profit of federation when the percentage of untrusted CSPs increases up to 80 percent. This is because, trust of CSPs are defined in terms QoS. Thus, increasing number of untrusted CSPs will not have much effect on the average value of profit of federation. Therefore, effect of profit on satisfaction level of federation will not be much. Moreover, from Fig. 4c it is observed that the average availability of federation formed by CFFM and QCFM mechanism is adversely affected by the increase in the percentage of untrusted CSPs. This is because, both CFFM and QCFM mechanism have not considered the availability and trust (trust is defined in terms of QoS) of CSPs while forming a coalition. Hence, as shown in Fig. 4a, the satisfaction level of federation formed by CFFM and QCFM mechanism is poor due to decrease in average availability of federation. Further, from Fig. 4d, it is observed that the average quality of federation for both mechanisms, is also adversely affected by the increase in the percentage of untrusted CSPs. This is again because, both CFFM and QCFM mechanisms have not considered the quality and trust of CSPs while forming a coalition. Figs. 4a, 4c and 4d reveal that the CGCFF mechanism outperforms all three MHCF, CFFM, QCFM mechanisms. It is also worth to note from Figs. 4a, 4b, 4c and 4d that the MHCF mechanism performs the worst compared to CFFM and QCFM mechanisms.

Next, we study the performance of some bigger games and delineate some further results as shown in Fig. 5. In this experiment, the number of CSPs in η is varied (25, 50, 100, 150 and 200) for three different cases of threshold values (0.5, 0.6, 0.7) of the trust (i.e., 50, 60 and 70 percent trusted CSPs respectively). Thereafter, we have analyzed two different *scenarios*, (i) effect on federation when the number of CSPs is varied from 25 to 200 and (ii) effect of three different cases of threshold on formed federation partition for the increasing number of CSPs. For the experiment, it is considered that, each CSP contributes at least 20 numbers of each instance of type large, xlarge and 2xlarge.

At first we discuss Fig. 5 according to *scenario* (i). Fig. 5a reveals that the average satisfaction level of federation is relatively high for 25 CSPs, then falls abruptly from 25 CSPs to 50 CSPs and thereafter remains almost uniform in all three cases for threshold values of trust. This is because, the value of satisfaction level is directly correlated with profit and availability of federation, thus from Figs. 5b and 5c it is noticed that, the average value of profit and availability has also fallen from 25 CSPs to 50 CSPs and thereafter the fluctuation in these values is very less when CSPs increases from 50 to 200. The reason for dropping in values of average profit, availability and quality (see Figs. 5b, 5c and 5d) of the federation is the decrease in average size of the federation (see Fig. 5f). Fig. 5f reports that when the size of the federation is

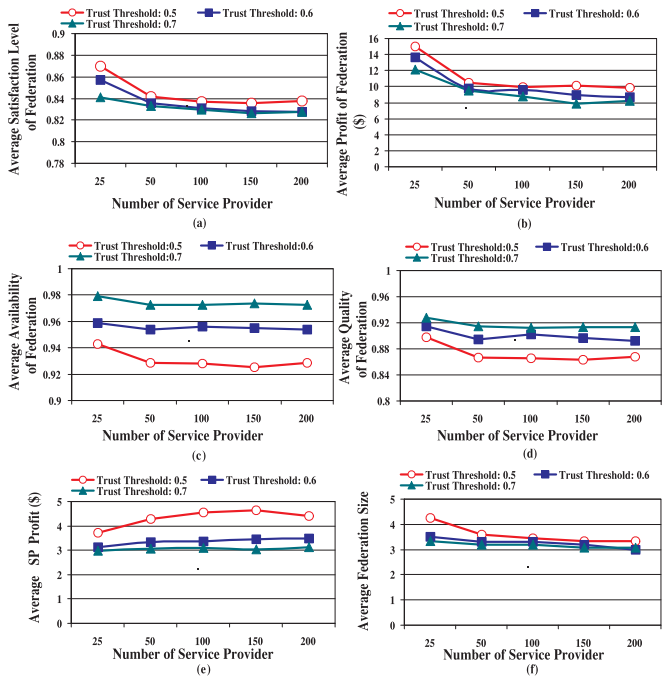

Fig. 5. Comparison of (a) Average satisfaction level of federations, (b) Average profit of federations (c) Average availability of federations (d) Average quality of federations, (e) Average profit of CSPs in federations and (f) Average sizes of federations formed by CGCFF for different sets of service providers with increasing order of threshold of trust.

around 4.4 (when number of CSPs are 25), the average value of profit, availability and quality of the federation is high and thereafter when average size of federation decreases (from 50 CSPs to 200 CSPs), the value of profit, availability and quality of federation also diminishes. This is due to the fact that when there are more members in the federation they can contribute more resources in the federation. Thus increasing overall profit and availability of federation. We can observe the similar effect on average profit of CSPs in Fig. 5e which is poor for large size federations and improves when the size of the federation decreases (see Fig. 5f).

Now, we analyze the performance of three different cases of threshold values of trust on federation partition for the increasing number of CSPs (i.e., *scenario* (ii)) from Fig. 5. Fig. 5a reveals that the average satisfaction level of federation is maximum when the threshold value is 0.5 and gradually decreases when the threshold is increased from 0.5 to 0.6 and from 0.6 to 0.7. This is because, the value of satisfaction level is directly related to the value of profit and availability of federation. Thus, from Figs. 5b and 5c it is observed that the presence of less untrusted CSPs (threshold values increased from 0.5 to 0.7) in the federation can increase the availability of the federation, but on the other hand profit decreases with the rise in threshold value from 0.5 to 0.7. Further, it should also be noted that, with each increased threshold level of trust (example from 0.5 to 0.6), for all selected CSPs (threshold level is 0.6) the average delivered QoS will always be better compared to average delivered QoS of CSPs selected earlier (threshold level is 0.5) as observed from Figs. 5c and 5d. However, delivering service with high QoS will increase the cost of cloud service. Therefore, decreasing the overall profit of federation (see Fig. 5b) when the threshold value increases from 0.5 to 0.7.

7 Conclusion

In this paper, we propose a mechanism that improves the CSPs' capabilities to address users' increasing demands. With the advent of cloud computing technology, it has become utmost important to extract benefits out of idle or underutilized resources maintained by different cloud service providers to yield more revenue. This requirement has led to the birth of the concept of cloud federation. A hedonic coalitional game has been formulated to model the federations among the CSPs. Beta-mixture approach has been used to determine the trust level of each CSP in the federation. We have devised a heuristic approach to build the set of federations (Υ), which is a Nash-stable coalitional structure and is also individually stable. The proposed CGCFF mechanism is robust due to: (i) CSPs can withstand the higher resource demands by increasing their resource availability in the federation and (ii) CSPs can utilize their idle and underutilized resources by sharing their resources with other members of the federation during low resource demands. Moreover, the benefit of CGCFF mechanism is that, it allows, trusted CSPs to join the federation in which their satisfaction is maximized. A comprehensive performance evaluation has been carried out for the proposed federation formation game. Even though this paper proposes a federation formation technique for cloud service providers, it can also be applied for other service providers which (a) adhere to availability constraints of resources, (b) aim to maximize profit of services within a federation and (c) are affected by trust of individual service providers.

Cloud federation enables overloaded CSPs to distribute its computing load by migrating the VM from overloaded CSPs to other under-loaded CSPs of federation. VM migration is also important to increase the reliability and availability of cloud services on occurrence of faults in the datacenters of CSPs. Thus, in our future work we will address issues related to fault and QoS violation due to VM migration.

Acknowledgments

This research work is supported by Visvesvaraya PhD Scheme for Electronics and IT, Digital India Corporation, Government of India.

References

[1] L. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: Towards a cloud definition," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 50–55, 2008.

[2] Gartner.com. (2017). [Online]. Available: https://www.gartner.com/newsroom/id/3808563

[3] Canalys.com. (2017). [Online]. Available: https://www.canalys.com/static/press_release/2017/media-alert-060217-cloud-infrastructure-market-49-intensifying-glabal-data-center-competition.pdf

[4] F. Liu, B. Luo, and Y. Niu, "Cost-effective service provisioning for hybrid cloud applications" *Mobile Netw. Appl.*, vol. 22, no. 2, pp. 153–160, 2017.

[5] Y. Niu, B. Luo, F. Liu, J. Liu, and B. Li, "When hybrid cloud meets flash crowd: Towards cost-effective service provisioning," in *Proc. IEEE Conf. Comput. Commun.*, 2015, pp. 1044–1052.

[6] Y. Niu, F. Liu, X. Fei, and B. Li, "Handling flash deals with soft guarantee in hybrid cloud," in *Proc. IEEE INFOCOM*, 2017, pp. 1–9.

[7] L. Mashayekhy, M. M. Nejad, and D. Grosu, "Cloud federations in the sky: Formation game and mechanism," *IEEE Trans. Cloud Comput.*, vol. 3, no. 1, pp. 14–27, Jan.–Mar. 2015.

[8] O. Abdul Wahab, J. Bentahar, H. Otrok, and A. Mourad, "Towards trustworthy multi-cloud services communities: A trust-based hedonic coalitional game," *IEEE Trans. Serv. Comput.*, vol. 11, no. 1, pp. 184–201, Jan./Feb. 2018.
[9] E. Khosrowshahi-Asl, J. Bentahar, H. Otrok, and R. Mizouni, "Efficient community formation for web services," *IEEE Trans. Serv. Comput.*, vol. 8, no. 4, pp. 586–600, Jul./Aug. 2015.
[10] CloudHarmony. (2016). [Online]. Available: https://cloudharmony.com/
[11] Amazon.com. (2016). [Online]. Available: https://aws.amazon.com/ec2/pricing/
[12] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, et al., "The reservoir model and architecture for open federated cloud computing," *IBM J. Res. Develop.*, vol. 53, no. 4, pp. 535–545, 2009.
[13] I. Goiri, J. Guitart, and J. Torres, "Characterizing cloud federation for enhancing providers' profit," in *Proc. IEEE 3rd Int. Conf. Cloud Comput.*, 2010, pp. 123–130.
[14] M. Hassan, B. Song, and E. Huh, "Distributed resource allocation games in horizontal dynamic cloud federation platform," in *Proc. IEEE Int. Conf. High Perform. Comput. Commun.*, 2011, pp. 822–827.
[15] W. Wang, D. Niu, B. Li, and B. Liang, "Dynamic cloud resource reservation via cloud brokerage," in *Proc. IEEE 33rd Int. Conf. Distrib. Comput. Syst.*, 2013, pp. 400–409.
[16] X. Yi, F. Liu, D. Niu, H. Jin, and J. C. S. Lui, "Cocoa: Dynamic container-based group buying strategies for cloud computing," *ACM Trans. Model. Perform. Eval. Comput. Syst.*, vol. 2, no. 2, 2017, Art. no. 8.
[17] H. Tang, F. Liu, G. Shen, Y. Jin, and C. Guo, "UniDrive: Synergize multiple consumer cloud storage services," in *Proc. 16th Annu. Middleware Conf.*, 2015, pp. 137–148.
[18] D. Niyato, A. V. Vasilakos, and Z. Kun, "Resource and revenue sharing with coalition formation of cloud providers: Game theoretic approach," in *Proc. 11th IEEE/ACM Int. Symp. Cluster Cloud Grid Comput.*, 2011, pp. 215–224.
[19] N. Grozev and R. Buyya, "InterCloud architectures and application brokering: Taxonomy and survey," *Softw.: Practice Experience*, vol 44, no. 3, pp. 369–390, 2014.
[20] (2016). [Online]. Available: http://www.networkworld.com/article/3020235/cloud-computing/and-the-cloud-provider-with-the-best-uptime-in-2015-is.html
[21] Amazon.com. (2016). [Online]. Available: http://aws.amazon.com/ec2/sla/
[22] D. Besanko and R. R. Braeutigam, *Microeconomics*, 3rd ed. Hoboken, NJ, USA: Wiley, 2008.
[23] N. Samaan, "A novel economic sharing model in a federation of selfish cloud providers," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 12–21, Jan. 2014.
[24] P. Johnson and M. Beverlin, "Beta distribution," in *Continuous Univariate Distributions–2*. New York, NY, USA: Wiley, 1970, pp. 37–56.
[25] M. Bouguessa, "Modeling outlier score distributions," in *Proc. Int. Conf. Adv. Data Mining Appl.*, 2012, pp. 713–725.
[26] M. A. T. Figueiredo and A. K. Jain, "Unsupervised learning of finite mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 3, pp. 381–396, Mar. 2002.
[27] A. Bogomolnaia and M. O. Jackson, "The stability of hedonic coalition structures," *Games Econ. Behavior*, vol. 38, no. 2, pp. 201–230, 2002.
[28] D. Wu, Y. Cai, R. Q. Hu, and Y. Qian, "Dynamic distributed resource sharing for mobile D2D communications," *IEEE Trans. Wireless Commun.*, vol. 14, no. 10, pp. 5417–5429, Oct. 2015.
[29] D. Ray, *A Game-Theoretic Perspective on Coalition Formation*. London, U.K.: Oxford Univ. Press, Jan. 2007.

Benay Kumar Ray is currently working toward the PhD degree in computer science and engineering at Jadavpur University, India.

Avirup Saha is currently working toward the PhD degree in computer science and engineering in IIT Kharagpur, India.

Sunirmal Khatua received the PhD degree from Jadavpur University, Kolkata, India, in December, 2016. He is currently an assistant professor with the Department of Computer Science and Engineering, University of Calcutta, India. He is a member of the IEEE.

Sarbani Roy received the PhD degree from Jadavpur University, in July 2008. She is an associate professor with the Department of Computer Science and Engineering, Jadavpur University, India. Her research interests include wireless sensor networks, cloud computing, IoT, smart environments, and social network analysis. She is a member of the IEEE and ACM.