The Journal of Supercomputing (2019) 75:885–929
https://doi.org/10.1007/s11227-018-2620-2

Toward maximization of profit and quality of cloud federation: solution to cloud federation formation problem

Benay Kumar Ray [1] · Avirup Saha [2] · Sunirmal Khatua [3] · Sarbani Roy [4]

Published online: 26 September 2018

Abstract
The emergence of cloud computing has led to an astronomical growth in the computing services provided by vendors over the cloud interface. This has led to the paradigm of cloud federations where a group of CSPs collaborate to form a federation for seamless provisioning of resource requests. In this paper, cloud federation formation framework is modeled as a multi-objective optimization problem with the trade-off between profit and QoS. Federation formation algorithms try to maximize the federation profit while maintaining a balance between the QoS and the profit of the members of the federation. We have applied *Linear Scalarization* as well as *ε-constraint* method to find the pareto-optimal solution to this multi-objective optimization problem. A heuristic-based algorithm for cloud federation formation following the integer linear program is proposed. We perform extensive experiments to investigate the performance of our proposed mechanism and show that our proposed mechanism yields optimized solution to the general problem of profit/QoS trade-off.

Keywords Brand value · Cloud computing · Cloud broker · Cloud federation · Quality of service

✉ Sarbani Roy
sarbani.roy@jadavpuruniversity.in; sarbani.jucse@gmail.com; sarbani.roy@gmail.com

Benay Kumar Ray
benayray@gmail.com; benayray@cusb.ac.in

Avirup Saha
saha.avirup@gmail.com

Sunirmal Khatua
skhatuacomp@caluniv.ac.in

[1] Department of Computer Science and Engineering, Central University of South Bihar, Fatehpur, Gaya, India

[2] Department of Computer Science and Engineering, IIT Kharagpur, Kharagpur, India

[3] Department of Computer Science and Engineering, University of Calcutta, Kolkata, India

[4] Department of Computer Science and Engineering, Jadavpur University, Kolkata, India

1 Introduction

Cloud federation is the system of integrating the cloud environments of several cloud service providers (CSPs) where it is possible for a CSP to outsource its resources to federation members to increase revenue [1]. However, to motivate CSPs to form federation, it is necessary for each CSP to maintain good reputation, strong security, privacy and QoS. Cloud service providers obtain immense advantage through formation of cloud federation. First of all, each CSP (big or small) while being part of cloud federation can generate extra revenue by utilizing idle or underutilized computing resources. Second, a cloud federation allows each CSP to expand its geographic footprints and manage unprecedented resource demand without having to built new points of presence and scale up its resource capabilities dynamically. Therefore, CSPs are able to maintain committed QoS in terms of scalability and availability. Moreover, with the advent of cloud federations, CSPs are not obliged to invest in additional infrastructure to improve their QoS offering and, therefore, this can greatly save great cost, time and energy.

1.1 Motivation

In recent years, there have been many research works involving cloud federation formation. Niyato et al. [2] have presented a hierarchical cooperative game model that helps CSPs to form federation and share the obtained profit among them. Mashayekhy et al. [1] in their study have presented cloud federation formation mechanism based on hedonic coalitional game, which aims to maximize the overall profit of the formed federation. Wahab et al. [3] concentrate on the dealing with malicious services that are apt to act unethically to illegally pursue their own ends at the expense of the federation. In this regard, they proposed a trust-based hedonic coalition game that enables CSPs to form a federation partition by minimizing maliciousness between the CSPs. Further, it is to be noted that none of the previous works has addressed the problem of trade-off among QoS and profit of formed federation simultaneously while considering each service provider's *brand value*. The service provider's brand value (BV) can represent its market reputation, goodwill and preference of cloud users. QoS is a major parameter with respect to cloud user and each CSP delivers cloud service with different QoS value. Hence, the federation fails to satisfy cloud users if the QoS delivered does not match the QoS committed for some of the federation members. Thus, in order to deliver and guarantee QoS to cloud user, it is necessary to maximize the QoS delivered through federation. Further, maximization of profit is also important for the federation as it incentifies service providers to participate in federation formation by renting their idle or underutilized computing resources and earn some extra revenues. In addition to QoS and profit of federation, *brand value* of participating service providers is also one of the important parameters, as it identifies the market reputation or goodwill of service providers.

1.2 Contribution

In this paper, cloud federation formation framework is modeled for the case, when request is made by a user (like big enterprise) to the cloud broker, consisting of the requirement of number of computing resources and a preferred individual service provider (“seed CSP”) through which the users seek to get resource services. Here, it is assumed that a particular “seed CSP” alone cannot fulfill cloud user’s resource requests. So the following “seed CSP” will be required to form a federation with other available CSPs to provide the requested resources to cloud users. Thus, it necessitates the *seed CSP* to find the best set of service providers out of available CSPs (which are interested to take part in federation) to form the federation.

Further, the problem of cloud federation formation is modeled as a multi-objective optimization problem with the trade-off between profit and QoS. The primary objective of our work is to optimally balance the overall profit and QoS of formed federation while considering each service provider’s *brand value*. It is to be noted that the goals of maximizing the QoS delivered by the federation caters to the clients and maximizing the profit earned by the federation caters to the CSPs. We have used beta mixture model to estimate the QoS of federation and its corresponding CSPs. We design a set of joining rules for cloud federation formation mechanism based on the QoS and brand value of service providers. The proposed mechanism distributes the profit of service providers within the federation based on their contribution in the federation, and it is also proved that individual profit gained by service providers in federation is always greater than the profit while not being part of a federation. Moreover, our work is the first in this domain that considers QoS, profit and brand value of federation and its corresponding member CSPs. We propose an algorithm based on integer linear program (ILP) to form the best federation. We also propose a heuristic-based algorithm for cloud federation formation following the ILP and our model. We perform extensive experiments to investigate the importance of our proposed mechanism and thereafter compare our proposed mechanism with two existing mechanisms proposed by Mashayekhy et al. [1] and Wahab et al. [3]. The major contributions of the paper can be summarized as follows:

- Proposing multi-objective optimization problem for cloud federation formation such that the overall QoS and profit of federation may be pareto optimal.
- Heuristic-based algorithm for cloud federation formation following the ILP and our model is also proposed.
- Proposing beta mixture model-based technique to estimate the QoS of individual service providers and federation.
- We have performed a series of extensive experiments to evaluate the performance of our model using different methodologies.

The paper is structured as follows. Section 2 describes the related works. In Sect. 3, QoS values of service providers and federation are estimated. Further, this section also explains the importance of brand value. Section 4 describes our proposed cloud federation formation framework. In Sect. 5, joining rules for service providers taking part in federation are defined. Section 6 describes the algorithm for cloud federation formation. Section 7 analyzes the performance of the proposed framework, and Sect. 8 concludes the paper.

2 Related work

In recent years, cloud computing has become a very active field of research [4]. Consequently, a lot of research has been done around federated cloud computing. The research work done in the different areas of cloud federation is described in this section.

2.1 Cloud federation architecture/model

Rochwerger et al. [5] have discussed the main reasons for cloud federation formation. Further, to support these requirements Rochwerger et al. [6] introduced the RESERVOIR model, a European research initiative that has the goal of overcoming the problem of scalability in cloud computing by introducing federated clouds in which providers with superfluous resources may lease them out to other providers who are temporarily in need of them. Several challenges in a multi-cloud environment, such as dynamic service elasticity, admission control, policy-driven placement optimization and cross-cloud virtual networks, were discussed along with their solutions. However, this did not include a federation formation mechanism. [7] describe a system called a "Cross-Cloud Federation Manager" which allows a cloud to create a federation with other clouds according to a three-phase model which comprises (1) discovery, (2) match-making and (3) authentication. Two types of clouds are considered (home and foreign) where home clouds cannot service the demands and are compelled to outsource the requests to the foreign clouds. In [8], Nordal et al. describe a mechanism called Balava for coordinating computations across multiple clouds involving data with confidentiality constraints. Yang et al. [9] describe a model of business-oriented federated cloud computing that supports computationally intensive real-time online interactive applications (ROIA). They discuss a method whereby multiple independent infrastructure providers can cooperate seamlessly to provide scalable IT infrastructure and QoS-assured hosting services for ROIA. Their model boasts of a unique business layer that can provide an enhanced security features and can trigger the on-demand resource provisioning across multiple infrastructure providers, hence helping to maximize the customer satisfaction, business benefits and resource usage. Altmann et al. [10] presented a cost model for federated hybrid clouds. The proposed cost model was applied as a part of a cost minimization algorithm, COMBSPO, that was used for making service placement decisions in hybrid clouds.

2.2 Resources provisioning in federated cloud

Van den Bossche et al. [11] addressed the issue of outsourcing workloads to external clouds in order to optimize performance as well as resource efficiency of a data center while satisfying quality-of-service restrictions through a linear programming approach (binary integer program). Goiri et al. [12] present cloud federation as a means for a provider to dynamically outsource resources to other providers in response to demand variations, as well as to rent out part of its unused resources to other providers. To this end, they introduced several decision equations to govern when to outsource resources to extraneous providers, insource (rent out) free resources, or shut down unused nodes to conserve power. Hassan et al. [13] have tackled the problem of distributed resource allocation in the emerging horizontal dynamic cloud federation (HDCF) platform. They have proposed a game-theory-based solution to this problem that ensures mutual benefits to provide incentive to the cloud providers to form a HDCF platform. They examine two resource allocation games—cooperative and noncooperative games, to analyze interaction among cloud providers in a HDCF environment. Also, both centralized and distributed algorithms are presented to find optimal solutions which have low overhead and robust performance. Toosi et al. [14] presented resource provisioning policies that help CSPs make decisions to increase their resource utilization and profit. Their main motive was to design a policy that helps CSPs decide on different types of incoming requests, and whether to outsource, reject or terminate VMs, whenever less profit is obtained. Chaisiri et al. [15] propose an optimal cloud resource provisioning (OCRP) algorithm by formulating a stochastic programming model in order to address the problem of advance reservation of resources as per different long-term plans as well as multiple provisioning stages. Their algorithm factors in the uncertainty involved with the consumer's future demand and the future resource prices while making reservations. Messina et al. [16] presented a decentralized solution, which enables finding trusted resources and allocation of the same into a federation. The proposed work also helps cloud users or providers to find most suitable collaborators by avoiding a search over the whole available set. Lee et al. [17] proposed a distributed resource allocation (DRA) approach to solve resource competition in the federated cloud environment. The proposed approach groups tasks according to communication behavior to minimize communication overhead and tries to allocate grouped tasks to achieve equilibrium when resource competition occurs. Abdi et al. in [18] address the problem of resource allocation for bag-of-tasks (BoT) workflows in a federation of clouds and formulate it as an integer linear programming problem. The proposed model minimizes financial cost including fees for running VMs and fees for data transfer, and fulfills deadline and resource constraints in the clouds.

2.3 Revenue maximization within the federation

Li et al. [19] presented an efficient algorithm for trading and scheduling resources in a federation environment so that the net profit of participating CSPs gets maximized. They have used double-auction-based mechanism for trading virtual machines between service providers. Their proposed mechanism is strategy proof, individual rational and ex-post budget balanced. Zant et al. [20] have proposed a revenue sharing model in the federated cloud environment. They have conducted numerical analysis to evaluate their proposed revenue sharing model. Rebai et al. [21] proposed a solution for optimally allocating distributed resources (virtual machine) among multiple CSPs within the existing federations. They have formulated integer linear programming to increase the revenues of CSPs based on resources contributed within the federation.

2.4 Federation formation mechanism

Niyato et al. [2] describe a hierarchical cooperative game model that helps CSPs to form federation and share the obtained profit among them. However, their study does not consider the operation cost of CSPs while providing resources. Further, they have also not considered different types of virtual machines. Mashayekhy et al. [1] in their study have overcome the aforementioned limitations. They have proposed a hedonic coalitional game for cloud federation formation mechanism, with the objective of maximizing the profit earned by the federation. Resources are allocated to maximize profit according to an integer linear programming (ILP) problem. Based on this, they proposed an algorithm which outputs a stable federation structure, in which the member CSPs do not have any incentive to abandon their federations or otherwise perturb the federation structure. Another paper that deserves special attention is the work of Wahab et al. [3] which concentrates on the dealing with malicious services that are apt to act unethically to illegally pursue their own ends at the expense of the federation. Such malicious behavior generally manifests itself as denial of service which affects vital parameters of a cloud federation such as availability, response time and throughput. To deal with such malicious services, a framework for trust establishment that is resilient to collusion attacks that occur to mislead trust results is proposed, along with a bootstrapping mechanism that capitalizes on the endorsement concept in online social networks to assign initial trust values. Finally, the authors propose a trust-based hedonic coalitional game that enables services to distributively form trustworthy multi-cloud communities. Bellaiche et al. [24] propose a cloud federation formation model that assesses the security risk levels of CSPs. After quantifying the security risk of CSPs, they design a hedonic coalitional game to model the cloud federation formation process with a preference relation that is based on the security risk levels and reputations of CSPs. In [25], cloud federation formation has been modeled as a hedonic coalitional game which tries to find the most best stable federation of trusted CSPs that will maximize the satisfaction level of each individual CSP on the basis of QoS and profit. In [26], the authors model the problem of forming cloud federations as a hedonic coalition game, which maximizes profit and minimizes migration cost within members of the same federation. Finally, [27] emphasizes on different approaches for cloud federation formation based on game theory and also highlights the importance of trust (soft security) in federated cloud environment. Different models for cloud federation formation using coalition game and the role of a cloud service broker in cloud federation are presented there.

Existing cloud federation formation mechanisms [1–3] have not considered the problem of trade-off among overall QoS and profit of federation simultaneously by considering brand value of each CSP while forming federation. Further, they have also not presented solution for the scenario mentioned as follows: In federated cloud environment, CSPs can dynamically increase their computing resource capability by collaborating with other service providers. So, any CSP (in our case *seed CSP*, when it

Table 1 Summary of previous research works in cloud federation

Year	Authors	Objective	Description
2009, 2011	Rochwerger et al. [5,6]	Overcoming the problem of scalability in cloud computing	Introduced cloud federation and its primary requirements for formation such that CSPs may rent out their idle resources to other CSPs
2010	Celesti et al. [7]	To add some enhancement on existing cloud architecture which adds new federation capabilities	Introduced a Cross-Cloud Federation Manager in the existing cloud architecture which allows CSPs to create a federation with other CSPs
2010	Van den Bossche et al. [11]	Maximize the utilization of the private cloud	Proposed a binary integer program (IP) formulation that minimizes the cost of outsourcing workload using a mix of public and private clouds
2010	Goiri et al. [12]	Introduced several decision equations to govern when to outsource resources to extraneous CSPs and rent out free resources to other CSPs	Presented a complete characterization of CSPs' federation in the cloud and introduced many decision equation for CSPs
2011	Nordal et al. [8]	Develop a system which can be intended as confidentiality-aware cloud federation system, which will support integral management of computations of both private and public cloud	Describe a Balava system for managing computations spanning across multiple clouds involving data with confidentiality constraints
2011	Hassan et al. [13]	To handle the resource allocation problem in horizontal dynamic cloud federation (HDCF)	Proposed a game-theoretic-based solution that ensures mutual benefits to CSPs and encourages them to form HDCF
2011	Toosi et al. [14]	Design a policy that helps CSPs to decide on different types of incoming requests, and whether to outsource, reject or terminate VMs	Presented a resource provisioning policy that helps CSPs in the decision-making process to increase their resource utilization and profit
2011	Niyato et al. [2]	Design a model to form a federation among CSPs	Presented a cooperative game model that helps CSPs to form federation and share the obtained profit among member CSPs
2012	Yang et al. [9]	To develop a business-oriented federated cloud computing model to support computationally intensive ROIA	The proposed model assures multiple CSPs to cooperate seamlessly to provide scalable IT services for ROIA

Table 1 continued

Year	Authors	Objective	Description
2012	Chaisiri et al. [15]	Minimizing both under-provisioning and problems under the demand uncertainty in cloud environments	Propose an optimal cloud resource over-provisioning algorithm to minimize the total price and cost for provisioning resources in a certain time period
2013	Li et al. [19]	Maximize the net profit of CSPs while trading VMs in federation	They have used double-auction-based mechanism for trading VMs between CSPs. They have also proved that their proposed mechanism is strategy proof, individual rational and ex-post budget balanced
2013	Wang et al. [22]	To enable the hybrid cloud environment to raise the resource utilization rate of the private cloud and to diminish task response time as much as possible	Propose an algorithm which exploits runtime estimation and several fast scheduling strategies for near-optimal resource allocation, which results in high resource utilization rate and low execution time in the private cloud
2014	Zant et al. [20]	Design a revenue sharing model in the federated cloud environment	Presented a revenue sharing model for member CSPs of federation and compared it with existing technique like shapley value
2014	Altmann et al. [10]	Suggest a cost model for federated hybrid clouds environment	Presented a cost model for federated hybrid clouds and used these models within a cost minimization algorithm, COMBSPO, which makes decisions for service placement
2015	Rebai et al. [21]	Increase the revenues of CSPs in federation	Solution is proposed for optimally allocating VMs among member CSPs of federation and increasing the revenues of CSPs in federation
2015	Messina et al. [16]	Propose a trust-based approach for large-scale federation utility computing infrastructures	Provide a decentralized solution to both cloud user and provider to enable trusted resources finding and allocation into a federation
2015	Mashayekhy et al. [1]	Form a federation partition for a given set of CSPs to maximize overall profit of federation	They have proposed a game-theory-based cloud federation formation mechanism, which maximizes the profit earned by the federation and forms a stable partition of cloud federations

Table 1 continued

Year	Authors	Objective	Description
2016	Wahab et al. [3]	Form a set of federation partition for given set of CSPs by minimizing maliciousness between CSPs	They have presented a trust-based hedonic coalitional game that enables services to distributively form trustworthy multi-cloud communities
2017	Ye et al. [23]	Maximizing expected profit and resource utilization and minimizing risk of CSPs	Design a reinsurance-emulated collaboration mechanism in a broker-based cloud federation where each CSP determines its resource retention for its future demand.
2017	Lee et al. [17]	To solve resource competition in the federated cloud environment	Their approach groups tasks according to communication behavior to minimize communication overhead and tries to allocate grouped tasks to achieve equilibrium when resource competition occurs.
2018	Abdi et al. [18]	To solve the problem of resource allocation for bag-of-tasks (BoT) workflows in a federation of clouds	Formulate an integer linear programming problem which minimizes financial cost and fulfills deadline and resource constraints in the clouds
2018	Bellaiche et al. [24]	Propose a model for cloud federation formation that assesses the security risk levels of CSPs	Design a hedonic coalitional game to model the cloud federation formation process with a preference relation that is based on the security risk levels and reputations of CSPs.
2018	Ray et al. [25]	To find the best stable federation of trusted CSPs that maximizes the satisfaction level of individual CSPs w.r.t. QoS and profit	Present a broker-based architecture for a cloud federation with a cloud federation formation algorithm inspired by coalitional games
2018	Ray et al. [26]	Design a cloud federation formation model that maximizes profit and minimizes VM migration cost within the cloud federation	Model the problem of forming federation among CSPs as a hedonic coalitional game with a utility function depending on profit and migration cost

runs out of computing resources, may need some extra support of computing resources, in order to deliver service to cloud users and maintain promised QoS. Hence, *seed CSP* will be required to collaborate (form federation) with other CSPs to increase the existing computing capabilities. In our case, the individual CSP through which federation formation is initiated is called *seed CSP*. Thus, finding the best combination of CSPs to form federation with *seed CSP* where overall profit and QoS of federation can be maximized is an important research problem. Furthermore, existing cloud federation formation mechanisms have used hedonic-game-based model, to form the federation of CSPs when CSPs have preference over which group they belong to. In other words, their main objective is to find a federation partition for a given set of CSPs. But in our case, federation formation is biased in favor of a particular CSP (*seed CSP*) in which seed CSPs choose members of federation based on his preference over a number of CSPs available to take part in federation. To the best of our knowledge, our work is the first that provides a solution for the above-discussed scenario for *seed CSP* to find the best combination of CSPs such that overall QoS and profit of federation may be pareto optimal. Further, a summary of previous research works in cloud federation is given in Table 1.

3 Preliminary

In this section, we have explained the preliminary concepts that we have used in the rest of the paper.

3.1 QoS of service provider

Due to increasing demands of IaaS services, managing and maintaining the committed QoS by each service provider has become an important concern. This is reflected in works such as that of Wang et al. [22] which incorporates QoS considerations in scheduling parallel tasks in hybrid clouds. QoS is considered to be an important parameter for the success of CSPs. If CSPs do not deliver QoS as they have committed, CSPs may lose their market reputation, trust of cloud user and consequently their enterprise brand value. QoS of any cloud service delivered by service providers represents the measure of QoS attributes like performance, availability and reliability. QoS is one of the major concerns to cloud users, because if service providers have to face serious network issues or server failure and have to discontinue service, then the cloud user can experience interrupted cloud service. The need to provide uninterrupted service in cloud federations has engendered research into preserving scalability commitments through techniques such as "reinsurance" which distribute risk among the member CSPs [23]. Without delivering high QoS, the advantages associated with using cloud services are diminished. Therefore, if advantages of using cloud service are reduced, then businesses and individual cloud users will host their own IT infrastructure. The importance of delivering high-end QoS is immense, in order to attract cloud users (Table 2).

Table 2 Notation

Symbols	Description
SP_s^*	The *seed CSP*, i.e., the CSP through which federation formation request is initiated
p_j^s	Chargeable price of jth types of instance by SP_s^*
$SP_i.Q$	The value of delivered QoS of CSP SP_i
$SP_i.BV$	*Brand value* of CSP SP_i
$SP_i.p_j$	Chargeable price of jth types of instance by SP_i
$SP_i.cost_j$	Total incurred cost of jth types of instance by SP_i
$SP_i.x_j$	Number of instances of type j contributed by SP_i
$SP_s^*.\tau$	The QoS tolerance of the *seed* CSP SP_s^*
$SP_i.Profit^s$	The profit obtained by CSP SP_i within federation
F_j	jth federation
$\|F_j\|$	Cardinality or size of federation F_j
$F.Q$	The overall QoS of federation F
$F.BV$	*Brand value* of federation F
Q_{attr}	Set of QoS attributes like availability, performance, scalability and reliability
$\|Q_{attr}\|$	Cardinality of set Q_{attr}

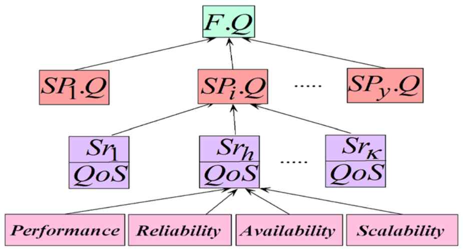
We have defined quality of service provider ($SP_i.Q$) hierarchically as shown in Fig. 1. Let us consider QoS of each cloud service as a measure of QoS attributes (Q_{attr_j}) like performance, availability, scalability, reliability, denoted by the set Q_{attr}, i.e., ($Q_{attr_j} \in Q_{attr}$). The QoS of each attribute is taken in the range [0, 1] and is denoted by $\{q^i_{attr_j}\}_{i=1}^{t}$, where i denotes interval in days and t denotes total number of days. Therefore, for each Q_{attr_j} there will be a set of values for $q^i_{attr_j}$ lying in the range [0, 1]. Here, a value of 1 means that the quality is 100 % and the value 0 signifies that the quality is 0 % of the maximum value. Table 3 provides a small snapshot of the measured Q_{attr} of three different cloud services say, Sr_1, Sr_2 and Sr_3 for a month [25]. Finally, the QoS of the cloud service Sr_l can be estimated by the equation given below:

$$d_i = \frac{\sum_{j=1}^{|Q_{attr}|} Q_{attr_j}}{|\ Q_{attr}\ |}. \tag{1}$$

The estimated QoS values for these services with the example data are given in Table 4.

The overall QoS of a CSP depends on the QoS of each component service. Let the i.i.d (independent and identically distributed) distribution of observed QoS values be denoted by $d = \{d_i\}_{i=1}^{X}$, where X represents the set of observed data of each CSP as given in Table 4. For each CSP Sr_l, let $\{d_{1i}, d_{2i}, \ldots, d_{Xi}\}$ be a vector of $Q(d)$ from different time intervals (days) which constitutes the distribution of Sr_l. It is noticed that the distribution satisfies the following two conditions: (a) the QoS values of each CSP $\in$ [0, 1] and (b) the shapes of the distributions of sample data of different CSPs Sr (Table 4) may vary. Keeping in view the above-mentioned conditions, we

Fig. 1 Quality of federation



have adopted the beta distribution to statistically model the overall QoS delivered by service providers, (Eq. 4) as it satisfies both of them.

The standard beta distribution with two shape parameters α and β is very popular in statistics. It has the important advantage that its density function can assume different shapes within the unit interval [0, 1] [28]. It can be positively as well as negatively skewed depending on the values of the parameters α and β. We direct the readers to [29–31] for a detailed treatment of the beta mixture model. Thus, all CSPs Sr_l have a distribution d of QoS attributes which follow a mixture density given by the beta mixture model. Consequently, the QoP of any given CSP SP_i can be expressed by the following density function:

$$SP_i.Q(d) = \sum_{l=1}^{\kappa} \omega_l Beta(d; \alpha_l, \beta_l). \tag{2}$$

Here, κ represents the number of component cloud services of the CSP SP_i, and α_l and β_l ($\alpha_l, \beta_l > 0$) represent the shape parameters of lth service (component). ω_l is the mixing weight such that $\omega_l > 0$ and $\sum_{l=1}^{\kappa} \omega_l = 1$. Further, the density function of lth cloud service is given by:

$$Beta_l(d \mid \alpha_l, \beta_l) = \frac{\Gamma(\alpha_l + \beta_l)}{\Gamma(\alpha_l)\Gamma(\beta_l)} d^{\alpha_l - 1}(1-d)^{\beta_l - 1}. \tag{3}$$

In the above equation, $\Gamma(\cdot)$ represents the gamma function and it is given 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 is a component of the mixture model, otherwise $z_{il} = 0$. We assume each $z_i = (z_{i1}, z_{i2}, \ldots, z_{i\kappa})$ is i.i.d. with prior probabilities $\omega = (\omega_1, \ldots, \omega_\kappa)$. We also consider a parameter vector $\Theta = (\alpha_1, \beta_1, \ldots, \alpha_\kappa, \beta_\kappa)$ of the unknowns α and β of κ components of the beta mixture model (CSPs).

We employ the expectation maximization (EM) algorithm to obtain an MLE estimate of the vector Θ [29,30]. Based on these estimated parameters for a given CSP SP_i, its corresponding p.d.f. $Q(d)$ is integrated over the range [0.5, 1] (the upper half of the range of the p.d.f). Hence, we can calculate the QoS delivered by each CSP SP_i as:

$$SP_i.Q = \int_{0.5}^{1} \sum_{l=1}^{\kappa} \omega_l Beta(d, \alpha_l, \beta_l). \tag{4}$$

Table 3 Sample data of Sr_1, Sr_2 and Sr_3 for different QoS attributes Q_{attr}

Days	$Sr_1(Q_{attr})$				$Sr_2(Q_{attr})$				$Sr_3(Q_{attr})$			
	Performance	Reliability	Availability	Scalability	Performance	Reliability	Availability	Scalability	Performance	Reliability	Availability	Scalability
Day1 ($q^1_{attr_j}$)	0.84	0.72	0.93	0.89	0.82	0.84	0.91	0.90	0.91	0.84	0.88	0.79
Day2 ($q^2_{attr_j}$)	0.92	0.82	0.90	0.92	0.44	0.42	0.52	0.56	0.48	0.56	0.48	0.38
…	…	…	…	…	…	…	…	…	…	…	…	…
Day29 ($q^{29}_{attr_j}$)	0.84	0.82	0.86	0.85	0.46	0.34	0.48	0.58	0.64	0.66	0.72	0.77
Day30 ($q^{30}_{attr_j}$)	0.87	0.89	0.90	0.91	0.85	0.84	0.89	0.78	0.62	0.66	0.82	0.69

Table 4 Estimated QoS data $Q(d)$ of cloud services Sr_1, Sr_2 and Sr_3

	Day_1	Day_2	Day_3	Day_4	Day_5	Day_6	Day_7	Day_8	Day_9	Day_{10}	Day_{11}	Day_{12}	Day_{13}	Day_{14}	Day_{15}
Sr_1	0.84	0.89	0.92	0.86	0.82	0.83	0.79	0.83	0.90	0.91	0.89	0.91	0.84	0.90	0.91
Sr_2	0.86	0.48	0.44	0.67	0.54	0.74	0.41	0.86	0.46	0.44	0.66	0.45	0.66	0.48	0.89
Sr_3	0.85	0.47	0.54	0.55	0.72	0.68	0.76	0.84	0.65	0.66	0.75	0.76	0.68	0.70	0.65

	Day_{16}	Day_{17}	Day_{18}	Day_{19}	Day_{20}	Day_{21}	Day_{22}	Day_{23}	Day_{24}	Day_{25}	Day_{26}	Day_{27}	Day_{28}	Day_{29}	Day_{30}
Sr_1	0.84	0.89	0.90	0.88	0.88	0.91	0.88	0.89	0.90	0.88	0.82	0.91	0.91	0.84	0.89
Sr_2	0.45	0.56	0.68	0.58	0.72	0.54	0.68	0.82	0.58	0.68	0.55	0.77	0.49	0.46	0.84
Sr_3	0.57	0.56	0.53	0.77	0.62	0.68	0.78	0.52	0.66	0.50	0.68	0.66	0.69	0.69	0.69

3.2 QoS of the federation

In federation, a group of service providers collaborate together to deliver uninterrupted cloud services to cloud users. These service providers deliver cloud services through federation with different QoS. So, overall QoS of a federation depends on the QoS delivered by all service providers being part of that federation. Further, it should be noted here that the service providers join a federation to obtain some benefits in terms of profit or availability [1,25]. Thus, in order to be a part of federation and to avail benefits associated with federation, the service providers will always try to deliver services with good QoS. Therefore, the QoS values of services delivered by each member of federation will not be exceptionally high or low. Hence, we take the average QoS of all the member service providers of a federation as the QoS of the overall federation and it is given by:

$$F.Q = \frac{\sum_{SP_i \in F_j} (SP_i.Q)}{|F_j|}. \tag{5}$$

3.3 Brand value of cloud service providers

A brand value of any cloud service provider or enterprise is an intangible asset. Like any other asset, brand of a CSP is required to be maintained, well understood and invested. However, by determining a CSP's brand value, it can be expressed exactly how much CSPs can enable business growth and contribute to revenue generation. Every cloud enterprise possesses some extra values other than its tangible assets. This value represents a goodwill or market reputation of a cloud enterprise, which is gradually developed after providing a long term of continuous cloud services. Therefore, the goodwill or reputation of any cloud enterprise can be referred to as its loyalty base for providing services based on committed QoS and other agreed upon service-level agreements (SLAs) with cloud users. Hence, there is a strong influence of goodwill or reputation on brand values of CSPs. Therefore, CSPs having the higher reputation or goodwill should have the higher brand value. Thus, alternatively a good brand (having higher brand value) is something which cloud users highly demand and for which they are ready to pay some extra price [32,33]. But it might happen that some CSPs who are new in market might provide high-quality (performance, availability, reliability and scalability) cloud services, but do not possess high brand value (as reputation or goodwill will take time to develop).

So in reality all cloud users have their own preferences over cloud service providers of particular brand value and QoS. But due to their budget constraints, their preference may vary and they will try to find CSPs of particular brand value and QoS in their budget. Hence, to emphasize the importance of the brand value of a CSP we have chosen to quantify it in terms of the prices charged by it individually for each VM instance given by $SP_i.p_j$ [33–35]. We define the brand value of a CSP SP_i as a quantity $SP_i.BV$ such that $0 \leq SP_i.BV \leq 1$ and

$$SP_i.p_j \propto SP_i.BV \quad \forall j \in \{1, 2, \ldots n\}, \tag{6}$$

where the proportionality constants depend on the individual VM instances. Further, please note that the brand value of individual CSPs is taken from G2 Crowd [36]. G2 Crowd provides G2 Score to each IaaS CSP, where G2 Score represents composite satisfaction rating and market presence score of corresponding CSPs, which is actually based on CSP's reputation and goodwill.

3.4 Brand value of federation

We call the member of the federation with the highest brand value the *dominant member* and its brand value is the *brand value of the federation*. The brand value of a CSP represents the market presence of the CSP and may not necessarily reflects its quality of service. Hence, for a federation F we have:

$$F.BV = max\{SP_i.BV|SP_i \in F\}. \tag{7}$$

It is to be noted that, in federation out of all member CSPs, the brand value of *seed CSP* will be maximum, i.e., the *seed CSP* will be the dominant member of the federation. Detailed explanation is given in Sect. 5.1.

4 Framework for cloud federation formation

A cloud federation framework is comprised of CSPs, cloud broker (CB) and cloud users. CB manages a set of registered cloud service providers and a set of formed federations among them. CB has knowledge of all the available cloud resources with each CSP. It also keeps track of delivered QoS and brand value of each CSP. All requests by cloud users, CSPs and cloud federations are processed through CB. CB is assumed to be a trusted third party and tries to satisfy the need of both cloud users and CSPs by providing an equitable scheme for both. CB is considered responsible for managing cloud user resource requests, initiating federation formation mechanism and distributing profit of federation between its member CSPs. Let us assume a set of cloud service providers $\eta = \{SP_1, SP_2, \ldots SP_m\}$ provide a resource to cloud users through CB. Each CSP can offer resources (virtual machine) of types $j \in \{1, \ldots, n\}$. Each jth kind of resource is characterized based on the number of cores co_j, the amount of memory me_j and the amount of storage st_j.

Each CSP intimates the CB about its total available cores, memory and storage to be shared in cloud federation after retaining specific capacity for its private users. We denote by $SP_i.Co$ the total number of cores, $SP_i.Me$ the total amount of memory, and $SP_i.St$ the total amount of storage. Again each CSP incurs cost when providing resources of jth type, and their costs may vary due to the varied types of infrastructure used and QoS delivered by them. Let the incurred cost of jth type of resource of service provider SP_i be given as $SP_i.cost_j$.

A cloud user sends a request to a CB, comprising the number of resources of each type needed and name of its preferred service provider (seed CSP). Let a typical user's resource requests be denoted by $R = \{r_1, r_2, \ldots, r_n\}$, where r_j is the number of resource requests of type j. Let us consider a scenario where a cloud user sends a request to the CB about his resource requirements of each type and his corresponding choice of a service provider SP_s^* (seed CSP). Suppose, due to high demand, preferred service provider SP_s^* is not able to fulfill the resource requests of some of the cloud users. The cloud federation broker on noticing this initiates the process of federation formation on behalf of SP_s^* with other available $SP_i \in \eta \backslash SP_s^*$. The main objective of CB is to find an optimal federation group for SP_s^* by striking a balance between the QoS and the profit of the members of federation. After forming a federation, service provider SP_s^* can easily fulfill the demand of cloud user. As cloud service is delivered through SP_s^* irrespective of the federation formation, we assume the same price p_j^s (denotes chargeable price of seed CSP SP_s^*) is charged by the federation per resource type j as it was charged by SP_s^*. This price is directly proportional to the brand value of the federation, defined as the maximum brand value of its constituents in Eq. 7. Formally,

$$p_j^s \propto F.BV. \tag{8}$$

As we will see later, this price will be constrained to be $SP_s^*.BV$ by our algorithms. Now, in Sect. 4.1 we will formulate the problem of cloud federation formation as an integer linear programming (ILP) problem.

4.1 ILP formulation of the federation formation problem

Cloud federation F is a set of CSPs $\{SP_1, SP_2, \ldots SP_y\} \subseteq \eta$, where each of the member CSPs shares a common interest in terms of resources, geographical distribution and economic benefits. The objective of federation formation is to maximize the profit P of the federation F while maintaining a balance between the QoS and the profit of the members of federation. Here, maximizing profit may need to compromise with the QoS, while maximizing QoS may need to compromise with the profit. Thus, Optimal Cloud Federation Formation (OPT-CFF) is a multi-objective optimization problem with the trade-off between profit and QoS. Hence, the two objective functions (profit and QoS) are given as follows:

$$f_{profit}(\{SP_i.x_j\}) = \sum_{i=1}^{y}\sum_{j=1}^{n} SP_i.x_j * \frac{\left(p_j^s - SP_i.cost_j\right)}{p_j^s} \tag{9}$$

$$f_{QoS}(\{SP_i.x_j\}) = \sum_{i=1}^{y}\sum_{j=1}^{n} SP_i.x_j * SP_i.Q. \tag{10}$$

The above two objective functions given in Eqs. (9) and (10) have to be maximized and given as:

$$Maximize(f_{profit}(\{SP_i.x_j\}), f_{QoS}(\{SP_i.x_j\})) \tag{11}$$

subject to $\{SP_i.x_j\} \in X$, where X is a feasible set of decision vectors, each element of which satisfies the following constraints:

$$\sum_{j=1}^{n} co_j \cdot SP_i.x_j \leq SP_i.Co \quad \forall i \in \{1, 2, \ldots, m\} \tag{12}$$

$$\sum_{j=1}^{n} me_j \cdot SP_i.x_j \leq SP_i.Me \quad \forall i \in \{1, 2, \ldots, m\} \tag{13}$$

$$\sum_{j=1}^{n} st_j \cdot SP_i.x_j \leq SP_i.St \quad \forall i \in \{1, 2, \ldots, m\} \tag{14}$$

$$\sum_{j=1}^{n} SP_i.x_j \geq 1 \quad \forall i \in \{1, 2, \ldots, m\} \tag{15}$$

$$\sum_{i=1}^{y} SP_i.x_j = r_j \quad \forall i \in \{1, 2, \ldots, m\}$$

$$SP_i.x_j \in \mathbb{N}^0. \tag{16}$$

The decision variable $SP_i.x_j$ represents the number of VM instances of type j contributed by SP_i. Since QoS lies in the range [0, 1], we divide the profit by the price of the seed CSP (p_j^s) to normalize it in the scale of QoS. The constraint given in Eq. (12) guarantees that the number of cores contributed by a CSP SP_i is less than its total number of available cores $SP_i.Co$. The constraint given in Eq. (13) ensures that the total memory contributed by a CSP SP_i is less than its total number of available memory $SP_i.Me$. Constraint in Eq. (14) ensures that the total storage contributed by a CSP SP_i is less than its total amount of available storage $SP_i.St$.

The constraint given in Eq. (15) ensures that a CSP contributes at least one VM to the federation. The constraint given in Eq. (16) ensures that the request is exactly satisfied.

In this paper, we have formulated the above multi-objective optimization problem as a single-objective optimization problem (linear scalarization) such that the solutions of this single-objective optimization problem are pareto-optimal solutions to the multi-objective optimization problem [37]. Hence, the optimization problem, OPT-CFF, based on *Linear Scalarization* is therefore stated as:

Linear scalarization

$$Maximize\left(w_{profit} * f_{profit}(\{SP_i.x_j\}) + w_{QoS} * f_{QoS}(\{SP_i.x_j\})\right) \tag{17}$$

$$Maximize \sum_{i=1}^{y} \sum_{j=1}^{n} SP_i.x_j \left(w_{profit} * \frac{\left(p_j^s - SP_i.cost_j\right)}{p_j^s} + w_{QoS} * SP_i.Q \right) \tag{18}$$

subject to

$$\begin{aligned}&((12), (13), (14), (15), (16)) \text{ and}\\&SP_i.x_j \in \mathbb{N}^0,\end{aligned}$$

where w is a weight vector parameter and is given as $w = (w_{QoS}, w_{profit})$ such that $w_{QoS} + w_{profit} = 1$

The objective function of OPT-CFF [Eq. (18)] maximizes an optimality score of federation F by ensuring that both the QoS and profit of the individual CSPs are considered while allocating the resources in the federation. A CSP with high QoS but low profit may be allocated the same number of resources as a CSP with low QoS but high profit. However, to prioritize profit over QoS we have also formulated the optimization problem as ε-constraint [37] which is given as follows:

ε-Constraint

$$Maximize \sum_{i=1}^{y}\sum_{j=1}^{n} SP_i.x_j(p_j^s - SP_i.cost_j) \tag{19}$$

subject to

$$\begin{aligned}&((12), (13), (14), (15), (16)) \text{ and}\\&SP_i.Q \geq SP_s^*.Q \qquad (20)\\&SP_i.x_j \in \mathbb{N}^0.\end{aligned}$$

Here, the QoS never goes below the QoS of the seed CSP as given by constraint (20). This is acceptable because the user choose the seed CSP based on its price and QoS.

4.2 Profit division among CSPs

Profit division is an integral part of a cloud federation framework. In order to have an incentive to join a federation, a CSP must be guaranteed to earn at least the same profit as it would earn if it were acting alone serving the same request. In our framework, the profit of a CSP is given by the sum over all VM types of the profit to be gained for that VM type times the number of instances of that VM type actually contributed by the CSP. Formally, for a CSP SP_i, its profit obtained within the federation is given by:

$$SP_i.Profit^s = \sum_{j=1}^{n} SP_i.x_j \times (p_j^s - SP_i.cost_j). \tag{21}$$

This can be represented as

$$SP_i.Profit^s = \sum_{j=1}^{n} SP_i.x_j \times p_j^s - \sum_{j=1}^{n} SP_i.x_j \times SP_i.cost_j. \tag{22}$$

That is,

$$SP_i.Profit^s = SP_i.Dividend^s - SP_i.Cost, \tag{23}$$

where $SP_i.Dividend^s = \sum_{j=1}^{n} SP_i.x_j \times p_j^s$ is the net *dividend*, received by the CSP, and $SP_i.Cost = \sum_{j=1}^{n} SP_i.x_j \times SP_i.cost_j$ is the net cost incurred by the CSP. When a CSP acts alone, its cost does not change. However, the payment (or dividend) received by it will change due to the fact that it will then charge its own price for the VM instance instead of the price charged by the federation. Thus, the estimated profit of a CSP when it acts alone will be given by

$$SP_i.Profit^i = SP_i.Dividend^i - SP_i.Cost. \tag{24}$$

A service provider SP_i delivers cloud service through *seed CSP*. Moreover, it is because of the *seed CSP* that a service provider SP^i is able to utilize its idle resources and earn some extra revenue. Thus, *seed CSP* takes some percentage of overhead charge say f (as agreed upon by them) over the extra profit earned by service provider SP_i which is given by

$$SP_s^*.Profit = \left(SP_i.Profit^s - SP_i.Profit^i\right) \times \frac{f}{100}. \tag{25}$$

Thus, net profit earned by service provider SP_i while being part of federation is given as:

$$SP_i.Profit = SP_i.Profit^i + \left(SP_i.Profit^s - SP_i.Profit^i\right) \times \frac{100 - f}{100}. \tag{26}$$

In course of defining the payoffs of the CSPs, we have ensured two important facts, firstly the participating CSPs are guaranteed to get higher profits in the federation than when they operate alone and secondly the dividends are distributed among the member CSPs in proportion to their active contributions to the federation.

Theorem 1 *The service providers while being part of federation earn greater profit than when they operate alone.*

Proof The price $SP_i.p_j$ charged by an individual CSP for a VM instance of type j and the price p_j^s charged by the federation F for the same VM type are governed by the relation

$$SP_i.p_j = p_j^s \times \frac{SP_i.BV}{F.BV}. \tag{27}$$

This follows from the relation between the prices charged by the federation and its brand value, given in Eq. 8, and the relation between the prices charged by SP_i and its own brand value, defined in Eq. 6. Hence, the estimated payment (or dividend) received by the CSP SP_i when it acts alone serving the same request is given by

$$SP_i.Dividend^i = \sum_{j=1}^{n} SP_i.x_j \times SP_i.p_j = \sum_{j=1}^{n} SP_i.x_j \times p_j^s \times \frac{SP_i.BV}{F.BV}. \tag{28}$$

Hence,

$$SP_i.Dividend^i = SP_i.Dividend^s \times \frac{SP_i.BV}{F.BV}. \tag{29}$$

Since $F.BV = max\{SP_i.BV | SP_i \in F\}$, it follows that $\frac{SP_i.BV}{F.BV} \leq 1\ \forall i \in \{1, 2, \dots m\}$. Thus, $SP_i.Dividend^i \leq SP_i.Dividend^s\ \forall i \in \{1, 2, \dots m\}$ and consequently

$$SP_i.Profit^i \leq SP_i.Profit^s\ \forall i \in \{1, 2, \dots m\}. \tag{30}$$

By virtue of Eq. 30, a CSP is guaranteed to reap at least as much profit within a federation as it would have received when it acted alone serving the same request. □

Theorem 2 *The proposed payoff scheme outlined in Sect.* (4.2) *distributes profit among the member CSPs based on their contributions to the federation.*

Proof The objective function of OPT-CFF (Eq. 18) allocates resources to each participating cloud provider to provide service to the client. Based on resource allocated to each provider, the profit of each service provider is calculated, by using Eqs. 26, 21 and 24. Therefore, it can be seen from Eqs. 26 and 21 that the profit is calculated based on the CSP's contribution of resources in the federation. Thus, the scheme described in Sect. (4.2) is guaranteed to distribute profit (using Eq. 26) among the member CSPs based on their contribution in the federation. □

5 Joining rules

In order to form federation, we define a set of joining rules for service providers taking part in federation formation. We define joining rules based on two constraints: (i) brand value and (ii) QoS. The following joining rules are discussed below:

5.1 Brand value constraint

The first condition for a CSP SP_i to join a federation F is

$$SP_i.BV \leq F.BV. \tag{31}$$

The above joining rule as shown in Eq. (31) prevents a seed CSP from partnering with another CSP of higher brand value. This is because, in the pricing model chosen here, the prices charged by the federation for its VMs are directly proportional to the brand value of the federation, which is the maximum brand value of its members. If a seed CSP accepts one of higher brand value into its own federation, it will have to charge more from its clients, which would be unfair to them. In effect, the clients would be getting more than they bargained for and paying more as well. Hence, it is to be noted here that, in the federation, out of all member CSPs, the brand value of *seed CSP* will be maximum and will represent the brand value of the federation.

5.2 QoS constraint

The second condition for a CSP SP_i to join a federation F is

$$SP_i.Q \geq (1 - SP_s^*.\tau)SP_s^*.Q, \tag{32}$$

where $SP_s^*.\tau$ is the QoS tolerance of the seed CSP. We define a QoS tolerance level, which ensures that the added members of lower brand value do not cause the delivered QoS to suffer too much. The smaller the tolerance, the higher will be the average QoS of the federation. If some CSPs exist which have low brand value (perhaps because they are new and have not yet gained the trust of the consumers) but have promised a high QoS, they will be selected by this rule. A federation F only takes on a new member if the ILP is not feasible, i.e., the request is too large to be serviced by the members of the current federation. If more than one CSP satisfies the above two conditions, the one with the maximum QoS is chosen. However, for federation formation the candidate CSPs are chosen to merge with federation in descending order of their QoS levels.

6 Algorithm for cloud federation formation

Initially, cloud users send a request R for resource to a cloud broker and also provide the name of their preferred service provider SP_i. On getting user's request, cloud broker selects preferred SP_s^* (seed CSP) to deliver cloud service. But if cloud broker finds that selected SP_s^* is not able to serve resource request alone, then SP_s^* is selected as the dominant member of the federation. Cloud broker on behalf of dominant SP_s^* initiates the process of federation formation, with the set η of CSPs who have declared themselves as candidates to take part in federation formation.

We formulate a greedy algorithm ServiceRequest (Algorithm 1) that allows a set of CSPs to service a resource request made to a particular CSP by a cloud user. It uses an auxiliary procedure which actually forms the federation of CSPs that will service the request. On executing algorithm ServiceRequest, first, the set η of CSPs with pre-defined contribution of total number of cores, memory and storage are defined. Next, it assigns virtual machine prices p_j proportional to the brand value of the dominant SP_i and runs the federation formation mechanism with SP_s^*, η and R as the inputs. We present here two versions of the federation formation mechanism. First, we propose an exhaustive cloud federation formation (ECFF) mechanism, which searches through all possible federations before finding the optimal federation and runs in time exponential in the time complexity of OPT-CFF. Therefore, as ECFF mechanism is computationally very costly, we have introduced the heuristic-based cloud federation formation (HCFF) mechanism, which uses a heuristic to determine the most suitable CSP to be included in the federation at each step and runs in time linear in the time complexity of OPT-CFF.

The ECFF mechanism first sets the flag *found* to False and then constructs the set S' which holds the candidate CSPs who satisfy the joining rules for this federation from the set S. Then, it examines each possible subset of this set and tries to form a valid federation from it. If a federation is feasible, the objective function of OPT-CFF is compared with the maximum value obtained so far. If it be greater than this value,

Algorithm 1 ServiceRequest

```
Require: Request R, seed CSP $SP_s^*$
Ensure: A suitable federation $F$ (if one exists), chargeable prices for multiple clients and payoffs for
    individual CSP's
1: Let $\eta$={$SP_1$,$SP_2$,...,$SP_m$} be the initial set of CSPs with pre-defined contributions of cores, memory
    and storage
2: $F$ = call HCFF($SP_s^*$, $\eta$, R)
3: if $F == \phi$ then
4:     declare error and terminate
5: else
6:     Allocate the resources in $F$ according to OPT-CFF
7:     Compute dividends for each CSP according to Eq. 21
8: end if
```

Algorithm 2 Exhaustive Cloud Federation Formation (ECFF)

```
Require: Seed CSP $SP_s^*$, set of CSP's S, Request R
Ensure: A suitable federation $F$ (if one exists) and a null set $\phi$ otherwise
1: found = False
2: $F$={$SP_s^*$}
3: Let $S'$={C|C ∈ S and $C.BV \leq F.BV$ and $C.Q \geq (1 - SP_s^*.\tau)SP_s^*.Q$}
4: maxValue = 0
5: for all each subset $S''$ of $S'$ containing SP do
6:     if OPT-CFF is feasible for $S''$ and R then
7:         currentValue = objective function of OPT-CFF
8:         if currentValue > maxValue then
9:             found = True
10:            F = $S''$
11:            maxValue = currentValue
12:        end if
13:    end if
14: end for
15: if (found == False) then
16:     return $\phi$
17: else
18:     return $F$
19: end if
```

it will be chosen as the new maximum. Thus, at the end we get the federation with the highest possible objective function of OPT-CFF. This federation is returned as F. If no federation is feasible, a null federation ϕ is returned to ServiceRequest.

In the HCFF mechanism, initially the flag *found* is set to False. This flag determines when a federation has been successfully formed. Initially, the seed CSP SP_s^* is considered to be the lone member of the proposed federation. This is because, if SP_s^* can serve the request on its own, it will not bring in other CSPs since that will be resulting in revenue division. The set S' which holds the candidate CSPs who satisfy the joining rules for this federation is constructed from the set S. In course of the while loop, this set will hold the remaining CSPs to be considered after every iteration. Within the while loop, we check whether the ILP OPT-CFF is feasible for the current federation or not. If so, then the flag *found* is set to true (signaling success). Otherwise, we consider the candidate SP' which maximizes the objective function of OPT-CFF when all requests can be satisfied by one provider, i.e., free of any resource constraint. This

Algorithm 3 Heuristic Cloud Federation Formation (HCFF)

```
Require: Seed CSP $SP_s^*$, set of CSP's S, Request R
Ensure: A suitable federation $F$ (if one exists) and a null set $\phi$ otherwise
1: found = False
2: $F$={$SP_s^*$}
3: Let $S'$={C|C ∈ S and $C.BV \leq F.BV$ and $C.Q \geq (1 - SP_s^*.\tau)SP_s^*.Q$}
4: while $S'$!=$\phi$ do
5:     if OPT-CFF is feasible for $F$ and R then
6:         found = True
7:         break
8:     end if
9:     Find $SP' \in S'$ such that $\sum_{j=1}^{n} r_j(w_{QoS} * SP'.Q + w_{profit} * (p_j - SP'.cost_j)) = MAX_{i=1}^{m}\{\sum_{j=1}^{n} r_j(w_{QoS} * SP_i.Q + w_{profit} * (p_j - SP_i.cost_j))\}$
10:    $F = F \cup SP'$
11:    $S'$ = $S'$ - {$SP'$}
12: end while
13: if (found==False) then
14:     return $\phi$
15: else
16:     return $F$
17: end if
```

is a simple heuristic we use to bring down the time complexity of the whole procedure drastically. This candidate SP' is added to the federation F and removed from the set S'. The while loop terminates either when a feasible federation has been formed (*found* = true), or when the set S' becomes empty. In the latter case, a federation is not feasible and a null federation ϕ is returned to ServiceRequest.

The ServiceRequest algorithm checks for the condition of the null federation and declares an error saying that the request cannot be satisfied, otherwise it uses Eq. (21) to compute dividends for individual CSPs.

6.1 Time complexity

The time complexities of ECFF, HCFF and *ServiceRequest* algorithms are given below:

Theorem 3 *The time complexity of the ECFF mechanism for n number of participating CSPs is $O(2^n \times f(n))$, where $f(n)$ is the time complexity of OPT-CFF.*

Proof The complexity of ECFF mechanism is dominated by the complexity of the while loop since the search through the set S to construct the set S' in line (3) takes only $O(n)$ time. For now, let us assume OPT-CFF takes $f(n)$ time where $f(n)$ is determined by the algorithm used internally by the ILP solver. In the worst case, the outer while loop iterates for $O(2^n)$ times (which is the cardinality of the power set of the input set of CSPs). The feasibility check in line (6) takes $O(f(n))$ time. Since the objective function of OPT-CFF may be calculated here and stored, there is no need to execute OPT-CFF more than once. All other operations take $O(1)$ time. Thus, the ECFF mechanism has a worst-case time complexity of $O(2^n \times f(n))$.

Theorem 4 *The time complexity of the HCFF mechanism for n number of participating CSPs is $O(n \times max(n, f(n)))$, where $f(n)$ is the time complexity of OPT-CFF.*

Proof The complexity of HCFF mechanism is dominated by the complexity of the while loop since the search through the set S to construct the set S' in line (3) takes only $O(n)$ time. We assume OPT-CFF takes $f(n)$ time where $f(n)$ is an unknown function of n and is determined by the algorithm used internally by the ILP solver. In the worst case, the outer while loop iterates for $O(n)$ times (when the combined resources of all the CSPs are required to satisfy the request, or the request cannot be satisfied). The feasibility check in line (5) takes $O(f(n))$ time. The search through the set S' in line (10) takes $O(n)$ time. All the other operations take $O(1)$ time. Thus, HCFF mechanism has a worst-case time complexity of $O(n \times max(n, f(n))$. If $f(n) \gg n$, the time complexity of HCFF mechanism will be $O(nf(n))$. □

ServiceRequest invokes OPT-CFF once in line (6), which takes $O(f(n))$ time which is less than the time complexity of HCFF mechanism. Since all the other operations in *ServiceRequest* take $O(1)$ time, the time complexity of ServiceRequest, too is either $O(n \times max(n, f(n))$ depending on which mechanism is used.

Let us now analyze the time complexity of OPT-CFF ($f(n)$). In our experiments, we use the free linear programming solver lpsolve which is based on the revised simplex method with branch-and-bound technique for solving integer linear programs [38]. This method has an exponential worst-case time complexity, i.e., $f(n) = O(2^n)$ [39]. However, this method has a polynomial average-case time complexity under various probability distributions of the input variables [40,41]. Hence, the worst-case time complexities of ECFF, HCFF and *ServiceRequest* are all $O(2^n)$. However, HCFF and *ServiceRequest* will generally have a polynomial average-case time complexity, whereas ECFF will always have an exponential average-case time complexity regardless of the probability distribution of the input variables.

6.2 Proof of convergence

In this section, we show that the proposed ECFF and HCFF mechanisms always converge to a valid federation if one exists.

Theorem 5 *The ECFF mechanism always converges to a valid federation if one exists.*

Proof We are predetermining the set of potential legal addition to the federation in line (3) satisfying the constraints delineated in the joining rules, i.e., Eqs. (31) and (32) by means of an exhaustive search. Then, we are constructing the power set of this set and exhaustively searching through it, trying to maximize the objective function of OPT-CFF. This ensures that when the request is not yet satisfied by the temporarily formed federation, all possible options are explored for finding a suitable match before declaring an error condition. Since the power set is finite, whether or not a suitable federation is found the loop will eventually terminate. □

Theorem 6 *The HCFF mechanism always converges to a valid federation if one exists.*

Proof We are predetermining the set of potential legal additions to the federation in line (3) satisfying the constraints delineated in the joining rules, i.e., Eqs. (31) and (32) by means of an exhaustive search. This ensures that when the request is not yet satisfied by the temporarily formed federation, all possible options are explored for finding a suitable match before declaring an error condition. Since this set is finite and its cardinality decreases by one on every iteration, even in the case where the break condition is never satisfied it is guaranteed to become empty eventually at which point the loop will terminate. We break out of the while loop the moment we find a federation that can satisfy the users' request. Otherwise if, even after adding all the available CSPs to the federation, we are still unable to satisfy the request, the while loop will terminate because then S will equal the null set ϕ. □

7 Experimental evaluation

In this section, we first describe our experimental setup and then evaluate the effectiveness of our solution.

7.1 Experimental setup

We have relied on data of Amazon EC2 and CloudHarmony, in order to test our scheme since such data are freely available [42,43]. We have taken the actual prices of VM as demanded by Amazon EC2 and have estimated costs of each VM at half of the stated price. We have taken cloud provider prices in eight different regions as prices demanded by individual CSPs (Table 5). This is because CSPs, like Amazon EC2, GoGrid, etc., have heterogeneous pricing model and virtual machine (VM) configuration. Thus, in order to keep homogeneous pricing model and virtual machine configuration for all the participating CSPs, we have considered VM prices of eight different regions of Amazon EC2 as the VM prices of eight different CSPs. The brand value of CSPs is taken from G2 Crowd [36,44] and is then normalized so that $0 \leq CSP.BV \leq 1$ (Table 6). G2 Crowd provides G2 Score to each IaaS CSP, where G2 Score represents composite satisfaction rating and market presence score of corresponding CSPs. We have, in accordance with our scheme, assigned brand values proportional to the prices of the VMs of individual CSPs, with Singapore assigning a highest brand value since it asks for the maximum prices. The delivered QoS attributes of CSPs for past thirty days are partially taken from CloudHarmony dataset and partially generated synthetically. We have considered data of twenty different delivered cloud service for each CSP, in order to estimate the quality of delivered service ($SP_i.Q$). The VMs considered are the large, xlarge and 2xlarge versions of the "general purpose, current generation" VM type offered by Amazon EC2 belonging to the m3 series (Table 7). The available resources of each CSP are assumed to be constant and equal to quadruple the resources demanded by the Type 3 VM (m3.2xlarge) which is the largest VM type we have considered (Table 8). The values of w_{profit} and w_{QoS} have been assumed equal since we intend to strike a balance between profit and QoS. High w_{profit} would result in a highly profitable federation with low QoS (similar to CFFM [45]), whereas a high

Table 5 Cloud provider prices ($ per hour)

Amazon EC2 regions CSP	US East (N. Virginia) SP_1	US West (Northern California) SP_2	EU (Ireland) SP_3	Asia Pacific (Singapore) SP_4	Asia Pacific (Tokyo) SP_5	Asia Pacific (Sydney) SP_6	South America (Sao Paolo) SP_7	AWS GovCloud (US) SP_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

Table 6 Cloud provider brand values

Amazon EC2 regions CSP	US East (N. Virginia) SP_1	US West (Northern California) SP_2	EU (Ireland) SP_3	Asia Pacific (Singapore) SP_4	Asia Pacific (Tokyo) SP_5	Asia Pacific (Sydney) SP_6	South America (Sao Paolo) SP_7	AWS GovCloud (US) SP_8
$SP_i.BV$	0.678	0.786	0.745	1.000	0.985	0.949	0.969	0.857

Table 7 Characteristics of VM instances (general purpose, current generation of Amazon EC2)

Parameter	Type 1 (m3.large)	Type 2 (m3.xlarge)	Type 3 (m3.2xlarge)
co_j (Intel Xeon E5-2670 v2 (Ivy Bridge) processors)	2	4	8
me_j (GB RAM)	7.5	15	30
st_j (GB SSD)	32	80	160

Table 8 Cloud provider constants

$SP_i.Co$ (# cores)	$SP_i.Me$ (GB RAM)	$SP_i.St$ (GB SSD)
32	244	640

w_{QoS} would result in a high QoS federation with low profit. Hence, we have avoided these extreme cases in the result and tried to analyze the result by establishing a profit/QoS trade-off. The tolerance level has been set to 0.5. The tolerance level can vary from 0 (no tolerance) to 1 (full tolerance). In the no-tolerance case, the algorithms ECFF/HCFF would never select a CSP with a lower QoS than the seed CSP, whereas in the full-tolerance case, the algorithm would select any CSP (with lower brand value than the seed CSP) regardless of its QoS. Here, we are constraining the selected CSP to have at least 50% of the QoS of the seed CSP, and therefore we set the tolerance level to 0.5. We consider this to be a reasonable lower bound for the QoS of the selected CSP. All arbitrary weights are assumed equal in all our equations. Our simulations have been conducted on Intel(R) Xeon(R) CPU E3-1226 v3 @ 3.30 GHz processor equipped with 32 GB of RAM in 64-bit Windows 7 environment. All programs are written in Python, and for the purposes of solving the integer linear program we have used the lpsolve Python API.

For experimental evaluation, at first we have compared the statistics for three different versions of our mechanisms (a) random-based cloud federation formation (RCFF), (b) exhaustive-based cloud federation formation (ECFF) and (c) heuristic-based cloud federation formation (HCFF). The RCFF mechanism constructs a random federation (Algorithm 4). For RCFF mechanism, a separate ILP has been created which shares the constraints of OPT-CFF but with the difference that it uses an objective function with random coefficients and it is not expected to maximize it either (OPT-CFF explicitly maximizes its objective function). ECFF and HCFF mechanisms have been described in detail in Sect. 6. By comparing the above three mechanisms, we find out the most effective mechanism out of the three in Sect. 7.2, and thereafter we compare our most effective mechanism with other two mechanisms proposed by Mashayekhy et al. [1] and Wahab et al. [3] in Sect. 7.3.

Further, for experimental purpose we generate a request vector as (number of requests for large type VM, number of requests for xlarge type VM, number of requests for 2xlarge type VM). Four types of request sizes have been considered—small-size request with vector (6, 5, 4), medium-size request with vector (9, 8, 9), large-size

Algorithm 4 Random Cloud Federation Formation (RCFF)

```
Require: Seed CSP $SP_s^*$, set of CSP's S, Request R
Ensure: A suitable federation $F$ (if one exists) and a null set $\phi$ otherwise
1: FormFederation(CSP SP, Set[CSP] S, Request R){
2: $found$ = False
3: $F$={SP}
4: $S'$={C|C ∈ S and $C.BV \le F.BV$ and $C.Q \ge (1 - SP_s^*.\tau)SP_s^*.Q$}
5: while($S'$!=$\phi$){
6: if $F$ can satisfy R subject to the constraints defined in Eqs. (12), (13), (14) and (16)) then
7:     $found$ = True
8:     break
9: end if
10: Randomly choose an $SP' \in S'$
11: $F = F \cup SP'$
12: $S'$ = $S'$ - {$SP'$}
13: if ($found$==False) then
14:     return $\phi$
15: else
16:     return $F$
17: end if
```

request with vector (17, 9, 12) and xlarge-size request with vector (24, 16, 14). Some of the measured performance metrics used are given below:

- Average profit of federation: $\frac{\sum_{s=1}^{y}\sum_{i=1}^{m} SP_i.Profit^s}{y}$. Here, s denotes the seed CSPs and y is the total number of CSPs in set η.
- Average individual profit of CSP in federation:
$$\frac{\sum_{s=1}^{y} \frac{\sum_{i=1}^{m} SP_i.Profit^s}{m}}{y}.$$
- Average quality of federation: $\frac{\sum_{s=1}^{y} F.Q^s}{y}$. Here, $F.Q^s$ is QoS of federation formed due to seed CSPs.
- Average individual profit of CSP when not part of a federation: $\frac{\sum_{i=1}^{y} SP_i.Profit^i}{y}$.
- Average optimality score is the value of objective function of OPT-CFF (Eq. 18). This value has been averaged over different runs with different seed CSPs.
- Average execution time: This value has been averaged over different runs with different seed CSPs.
- Average size of the federation is the value of m for each federation. This value has been averaged over different runs with different seed CSPs.

7.2 Experiment 1: result

In this section, we have compared the performance of three different mechanisms RCFF, ECFF and HCFF based on requests generated of different sizes and found out the most effective mechanism out of these three. Next, we have tried to investigate the effect on average profit and QoS of federations formed by different seed CSPs.

It is to be noted that all the comparisons are made based on *Linear Scalarization* technique. At last, we compare the performance of HCFF mechanism using (a) *Linear Scalarization* and (b) *ε-constraint*.

In Fig. 2a, we see that the average profit of the federation increases with the increase in request size. In this case, it is observed from Fig. 2a that the RCFF performs marginally better than the ECFF and HCFF mechanisms. But from Fig. 2b we can notice that the values of average QoS of federation (F.Q) for ECFF and HCFF are higher compared to RCFF, i.e., ECFF and HCFF mechanisms heavily outperform the RCFF mechanism. It can also be noticed from Fig. 2a, b that HCFF yields very close values of F.Q and profit to those obtained by ECFF. But from the above two figures, it is difficult to predict the most effective mechanism out of the three.

So we introduce the parameter optimality score, which is the most important criterion in our case to evaluate the federation formed by various mechanisms. The optimality score represents a balance between the QoS and the profit which are otherwise self-conflicting. This is also the central dogma of our work. Indeed, in our results RCFF tends to produce higher profits, whereas ECFF and HCFF tend to produce greater quality. So neither of these two factors alone can be used to judge the three mechanisms. Only the optimality score can provide a way to compare the three algorithms effectively. It can be observed from Fig. 2c that the ECFF produces the highest optimality score, followed by HCFF and then RCFF. It can also be observed from Fig. 2c that the RCFF yields much lesser optimality score compared to ECFF. From Fig. 2c, it is noticed that difference between the optimality score of HCFF and ECFF is negligible. Hence, from the above discussion it can be concluded that both HCFF and ECFF are better mechanisms compared to RCFF.

Figure 2d shows the execution time of three mechanisms RCFF, HCFF and ECFF. The execution time of ECFF in all the cases is very large compared to HCFF and RCFF mechanism. This is because it searches for the best possible federation combination out of available service providers. As HCFF and ECFF are better mechanisms compared to RCFF in terms of optimality score, we have not compared the execution time of RCFF. Furthermore, it can be observed from Fig. 2c that HCFF can obtain almost similar optimality score as ECFF and has much lesser execution time compared to ECFF. Thus, from the above discussion it can be concluded that the performance of HCFF mechanism is much better compared to ECFF. Figure 2e shows the average size of federation formed by RCFF, HCFF and ECFF for different request size. It is to be noted from Fig. 2e that for similar request size, the average sizes of federation formed by RCFF, HCFF and ECFF are same and size of federation increases with the increase in request size. But, though each mechanism obtained same federation size their execution times are different and it increases with the increase in request size (Fig. 2d).

Finally, Fig. 2f shows the average individual profit obtained by service providers (i) when they are part of federation and (ii) when they are not part of federation for different request sizes. We find that, for all three mechanisms RCFF, ECFF and HCFF, the average individual profit gained by a CSP while being part of a federation is always higher than the average individual profit gained by CSPs while not being part of a federation. From Fig. 2f, it is clearly revealed that according to our profit division scheme outlined in Sect. 4.2, a CSP always stands to gain as part of a federation

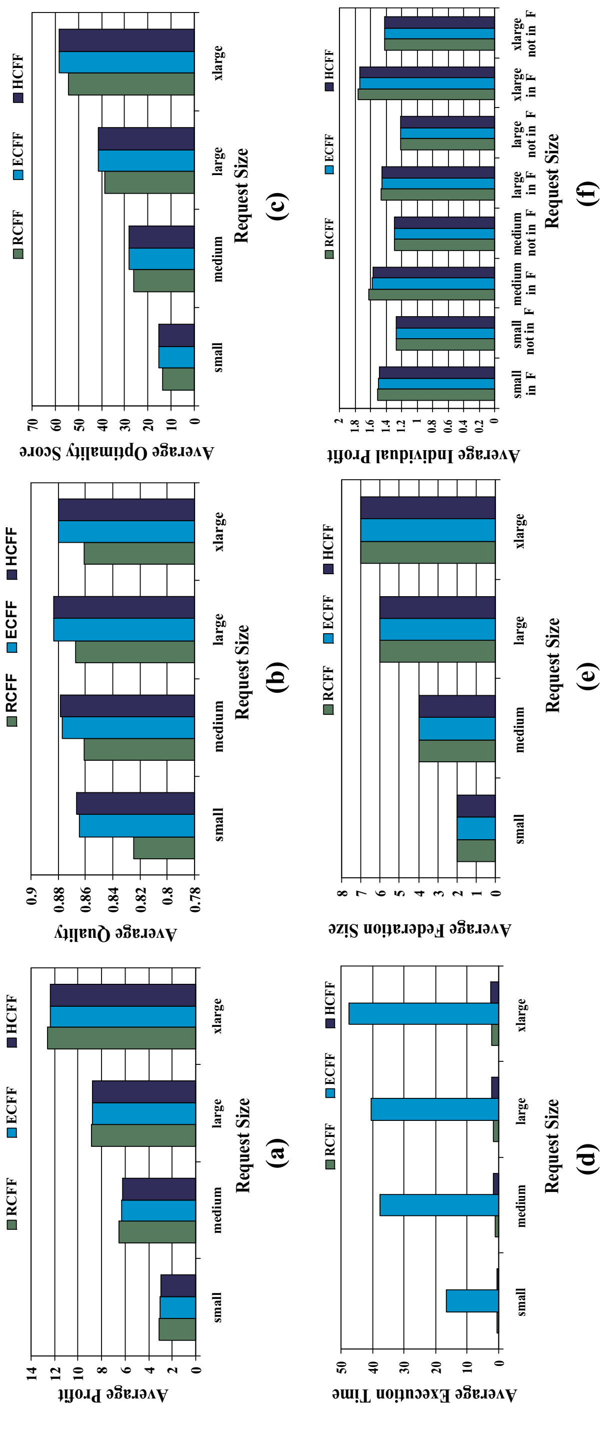


Fig. 2 **a** Average profit of federation, **b** average quality of federation F.Q, **c** average optimality score of federation, **d** average execution time, **e** average size of federation, **f** average individual profit of CSPs for RCFF, ECFF and HCFF mechanism for different requests

compared to the situation when it acts alone. Thus, it is profitable for CSPs to always take part in federation formation process.

Next, we investigate the effect on federation formed by different seed CSPs (service provider through which request is initiated) in case of medium-size request for HCFF mechanism. It is observed from Fig. 3 that federation formed by different seed CSPs for same request has different values of profit, quality and optimality score. From Fig. 3a, it is noticed that, when SP_4 is a seed CSP, profit of federation is least, whereas when SP_7 is seed CSP, the profit gained by federation is highest. But from Fig. 3b, it is noticed that average quality of federation is highest when seed CSP is SP_4, whereas it is lowest when seed CSP is SP_6. However, when we compare the optimality score of different seed CSPs in Fig. 3c, we found that the federation formed SP_4 has highest optimality score value of all seed CSPs. Thus, from the above discussion it is observed that for different seed CSPs profit and quality of federation may vary for same size request.

A further investigation is made to study the performance of three different proposed mechanisms, RCFF, ECFF and HCFF, by increasing the number of participating CSPs from 25 to 200. The request vectors for these cases are given in Table 9. The data for price, QoS and brand value are generated synthetically based on dataset provided by Amazon EC2, CloudHarmony and G2 Crowd. Based on the above discussion of Fig. 2a, it is known that the average profit obtained by RCFF mechanism is marginally better than ECFF and HCFF when number of CSPs is small. But from Fig. 4a, it is noticed that average profit of federation formed by RCFF mechanism is tremendously lower compared to ECFF and HCFF when number of CSPs increases from 25 to 200. This is because out of large number of available CSPs RCFF mechanism randomly chooses the participating CSPs for federation, and is thus not able to find the best combination of CSPs for federation. However, average profit of federation obtained by ECFF mechanism is found to be marginally better than HCFF mechanism. In Fig. 4b, c, we show the average quality and average optimality score of federation when number of CSPs increases from 25 to 200. In both the cases, ECFF and HCFF mechanisms outperform the RCFF mechanism. Moreover, from Fig. 4b, c it is also observed that difference between the average quality and average optimality score of federation formed by HCFF and ECFF is negligible. Hence, finally based on result of Fig. 4a–c we can again conclude that the proposed ECFF and HCFF mechanisms are better compared to RCFF mechanism.

Next, we compare the execution time of only ECFF and HCFF mechanism by increasing number of CSPs from 25 to 200. This is because, from the above discussion, it is clear that ECFF and HCFF mechanisms are better compared to RCFF mechanism. It is found from Fig. 4d that execution time of ECFF increases exponentially as the number of CSPs increases from 25 to 200. This is because ECFF mechanism searches through all possible federations before finding the optimal federation. But, execution time of HCFF is found to be increasing linearly, when the number of CSPs increases from 25 to 200 and is almost negligible when compared to ECFF mechanism. This is because HCFF mechanism uses a heuristic to find out the most probable optimal federation. Thus, based on the above discussion it is clear that, though the ECFF mechanism performs better in case of average profit, quality and optimality score of federation compared to HCFF, the difference is almost negligible when the number of

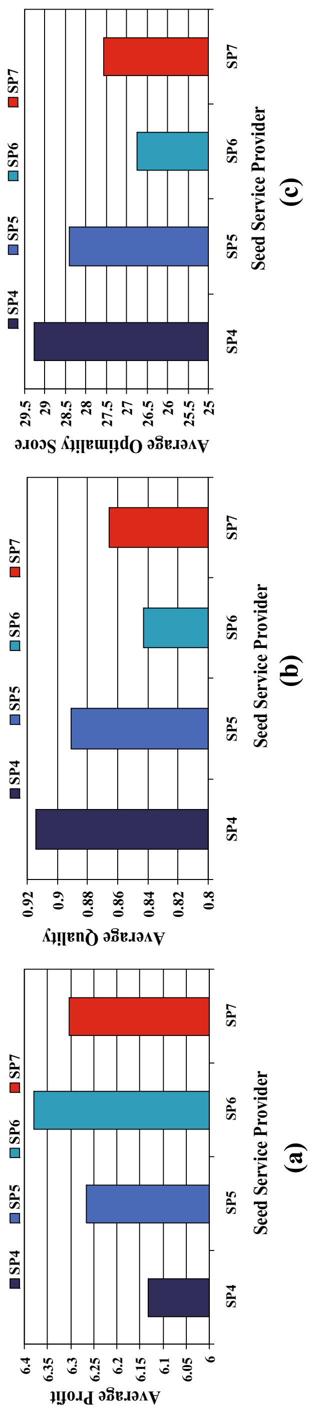


Fig. 3 **a** Comparison of average profit of a federation, **b** comparison of average quality of a federation and **c** comparison of average optimality scores, for different seed CSPs in case of medium request (9, 8, 9)

Table 9 Request vectors for varying numbers of CSPs in the pool ($| \eta |$) for Fig. 4

Pool size $\| \eta \|$	Type 1 VM (m3.large)	Type 2 VM (m3.xlarge)	Type 3 VM (m3.2xlarge)
25	60	50	40
50	110	100	90
100	220	210	200
150	330	320	310
200	440	430	420

CSPs increases from 25 to 200. But execution time of ECFF exponentially increases with the increase in number of CSPs, whereas that of HCFF mechanism increases linearly. Thus, it can be concluded that the overall performance of HCFF is better compared to ECFF.

Finally, we try to compare HCFF mechanism based on two proposed OPT-CFF (i) *Linear Scalarization* and (ii) *ε-constraint* by increasing number of CSPs from 25 to 200. From Fig. 4e, it is observed that with the increase in number of CSPs, average profit of federation formed by HCFF using *Linear Scalarization* and *ε-constraint* also increases. It is further noticed that on increasing CSPs from 25 to 100 the difference between the average profit of federation obtained by both the OPT-CFF techniques is almost negligible, but when number of CSPs increases from 100 to 200 the difference between the average profit of federation obtained by *Linear Scalarization* and *ε-constraint* technique is visible. This is due to the reason that *ε-constraint* technique strictly defines the $SP_i.Q \geq SP_s^*.Q$ condition, which restricts many CSPs from taking part in federation. Thus, eliminating those CSPs greatly maximize average profit of federation. However, the difference in average profit of federation obtained *Linear Scalarization* and *ε-constraint* technique is not too large and is within the acceptable limit. But on the other hand, from Fig. 4f it is observed that the average quality of federation formed by HCFF mechanism using *ε-constraint* technique is greater compared to federation formed by HCFF mechanism using *Linear Scalarization*. This is again due to the reason that *ε-constraint* technique strictly defines the $SP_i.Q \geq SP_s^*.Q$ condition and thus selects those CSPs whose quality is either equal to that of the seed CSP $SP_s^*.Q$ or greater than that of the seed CSP, therefore increasing the overall quality of federation. Thus, from the above discussion it can be concluded that both the OPT-CFF techniques are important as *Linear Scalarization* technique tries to maximize both quality and profit of federation, whereas *ε-constraint* technique tries to maximize profit such that quality of federation never goes below the QoS of the seed CSP.

We conclude from the above experimental results and analysis that HCFF is the most cost-effective mechanism in terms of time complexity compared to ECFF. However, it is also true that in spite of the results obtained by ECFF mechanism being the best (for average value of profit and quality of federation), HCFF mechanism yields very close results to those obtained by ECFF. Hence, it can be concluded that HCFF mechanism is the most effective mechanism.

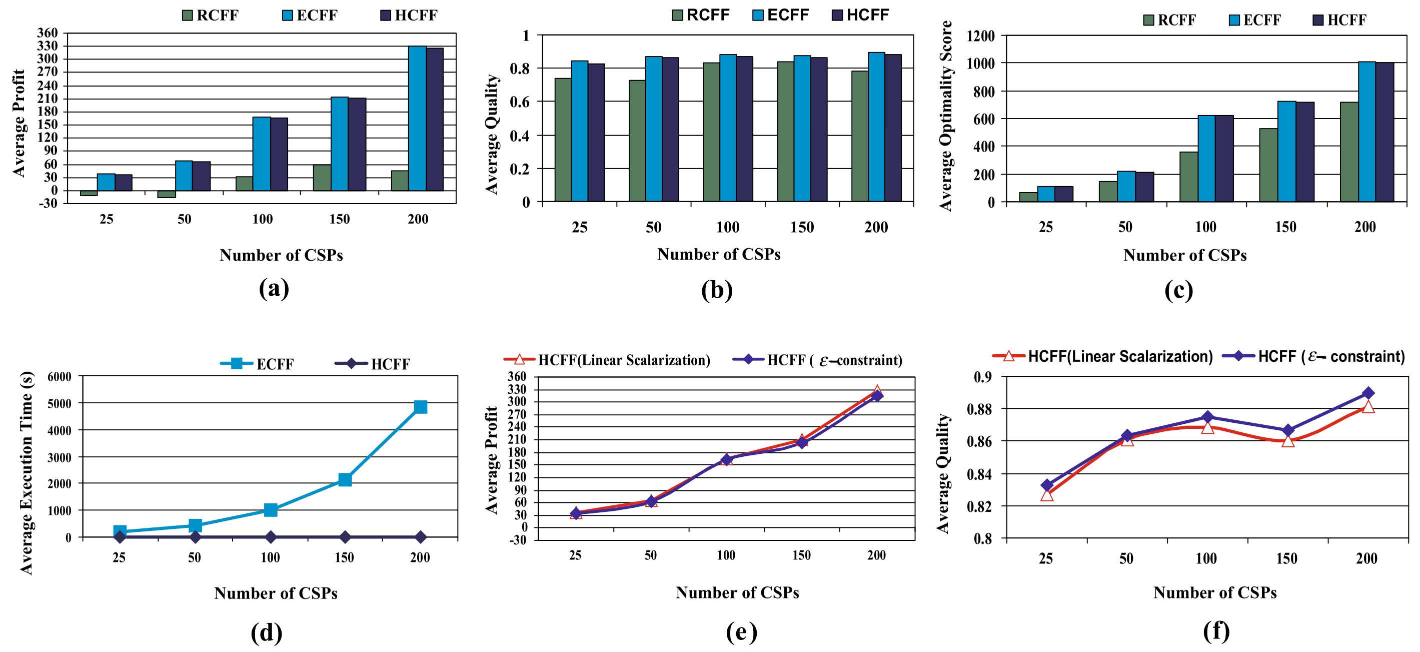
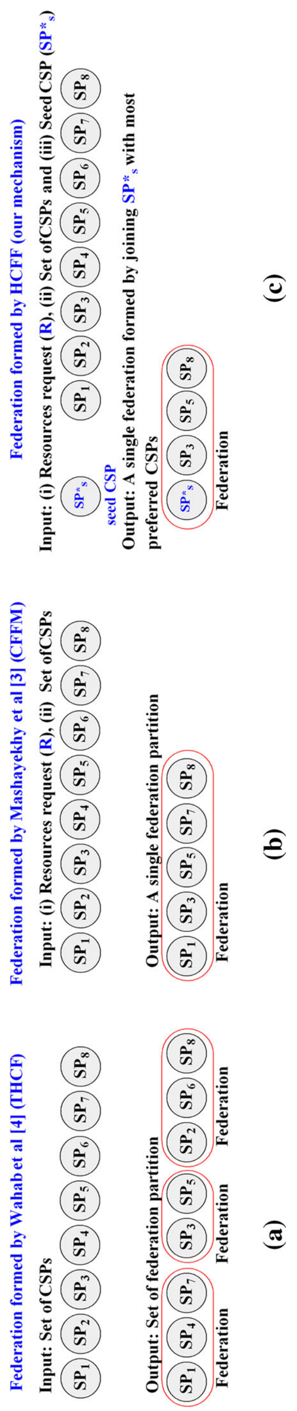


Fig. 4 **a** Comparison of average profit of a federation, **b** comparison of average quality of a federation, **c** comparison of average optimality scores, **d** comparison of average execution time, **e** comparison of average profit of federation obtained by HCFF mechanism for *Linear Scalarization* and *ε-constraint* and **f** comparison of average quality of federation obtained by HCFF mechanism for *Linear Scalarization* and *ε-constraint* by increasing number of CSPs

7.3 Experiment 2: result

In this section, we have compared the results obtained by our HCFF mechanism with other two mechanisms: (i) cloud federation formation mechanism (CFFM) proposed by [1] and (ii) trust-based hedonic cloud federation (THCF) mechanism proposed by Wahab et al. [3]. Before going into the details of experimental comparison, the difference in federation formed by all the three mechanisms are depicted in Fig. 5. Further, for this experiment we have used the same dataset of eight CSPs as used by CFFM [1] and THCF [3] to compare with our proposed mechanism (see Tables 5 and Sect. 7.1). For a detailed description of experimental process and setup, please refer to Sect. 7.1. Further, since CFFM and THCF mechanisms do not have a concept of a seed CSP and always take all the CSPs in the grand federation (federation of all CSPs) into account while forming their federations, we have chosen, as our seed CSP, SP_4 (Table 5, 6), which has the maximum brand value (1.000) of all the CSPs. This is because in that case the set of CSPs admissible in the federation becomes the grand federation itself. In Table 10, we have presented a theoretical comparison of CFFM and THCF mechanism with our proposed HCFF mechanism. The experimental comparison of CFFM and THCF mechanism with our proposed HCFF mechanism is presented below:

In Fig. 6, we have compared CFFM mechanism with our proposed HCFF mechanism based on small-size request (6, 5, 4). From Fig. 6a, it is observed that the average profit of federation formed by CFFM mechanism is slightly better compared to HCFF mechanism. This is because the main objective of the CFFM mechanism is to only maximize the overall profit of federation. On the other hand, HCFF mechanism tries to find pareto-optimal solution to the general problem of profit and QoS trade-off. Hence, in order to balance the trade-off between profit and QoS, the HCFF mechanism compromises with some profit and thus obtains slightly lower profit compared to CFFM mechanism. Furthermore, the HCFF mechanism always selects a federation with the minimum possible size, whereas CFFM selects a federation larger than necessary to serve the request with a view to maximizing the total profit earned by the federation. However, this results in a dilution of the profits for each individual member of the federation, since the profits must be shared among more CSPs. Hence, the individual profit gained by CSPs in the federation formed by CFFM mechanism is lower compared to the profit gained by individual CSPs in federation formed by HCFF (Fig. 6b). Furthermore, from Fig. 6c it is noticed that quality of federation formed by HCFF mechanism is higher compared to CFFM mechanism. This is because HCFF mechanism selects those CSPs as part of federation which maximize profit value without degrading the QoS value of the federation. Further, from the above discussion of Fig. 6a it is known that, in order to balance the trade-off between profit and QoS, the HCFF mechanism compromises with some profit, but at the same time it can also be observed from Fig. 6c that the QoS of federation increases and is greater than average QoS of federation formed by CFFM mechanism. Here, it should be noted that the CFFM mechanism only selects those CSPs as a part of federation which maximize the overall profit of federation. Therefore, it is possible that the selected CSPs, which maximize profit, have lower QoS value, thus decreasing the overall QoS of federa-



Federation formed by Wahab et al [4] (THCF)
Input: Set of CSPs
SP1 SP2 SP3 SP4 SP5 SP6 SP7 SP8
Output: Set of federation partition
SP1 SP4 SP7 Federation
SP3 SP5 Federation
SP2 SP6 SP8 Federation
(a)
Federation formed by Mashayekhy et al [3] (CFFM)
Input: (i) Resources request (R), (ii) Set of CSPs
SP1 SP2 SP3 SP4 SP5 SP6 SP7 SP8
Output: A single federation partition
SP1 SP3 SP5 SP7 SP8 Federation
(b)
Federation formed by HCFF (our mechanism)
Input: (i) Resources request (R), (ii) Set of CSPs and (iii) Seed CSP (SP*s)
SP*s seed CSP
SP1 SP2 SP3 SP4 SP5 SP6 SP7 SP8
Output: A single federation formed by joining SP*s with most preferred CSPs
SP*s SP3 SP5 SP8 Federation
(c)

Fig. 5 Different federation formation approaches by **a** Wahab et al. [3] (THCF), **b** Mashayekhy et al. [1] (CFFM) and **c** HCFF (our proposed mechanism)

Table 10 Theoretical comparison of THCF by Wahab et al. [3], CFFM by Mashayekhy et al. [1] and HCFF proposed by us

Point of comparison	THCF by Wahab et al. [3]	CFFM by Mashayekhy et al. [1]	HCFF proposed by us
Objective	Forms a federation partition by minimizing maliciousness between CSPs	Forms a federation by maximizing overall profit of federation	Forms a federation by maximizing overall profit and QoS of federation while considering brand value of CSPs
Methodology	Hedonic coalition game is used to form a federation	Hedonic coalition game is used to form a federation	Heuristic-based algorithm is proposed to form a federation
Maximizing or minimizing parameters	Overall maliciousness between CSPs is minimized	Overall profit of federation is maximized	Overall profit and QoS of federation are maximized
Types of federation formation	Forms a federation partition out of available set of CSPs as shown in Fig. 5a	Forms a single federation out of available set of CSPs as shown in Fig. 5b	Forms a most preferred federation for the *seed CSP* SP_s^* by joining with available set of CSPs as shown in Fig. 5c
Profit division	Profit division method is not defined	Uses Banzhaf value to divide profits	Profit division based on contribution of CSPs
Client/CSP interests	Only interests of CSPs are considered by decreasing maliciousness among CSPs	Only interests of CSPs are considered by increasing their profit in federation	Interests of both CSPs and cloud users are taken care of, viz., maximizing profit corresponds to the interest of CSPs and maximizing QoS corresponds to the interest of cloud users

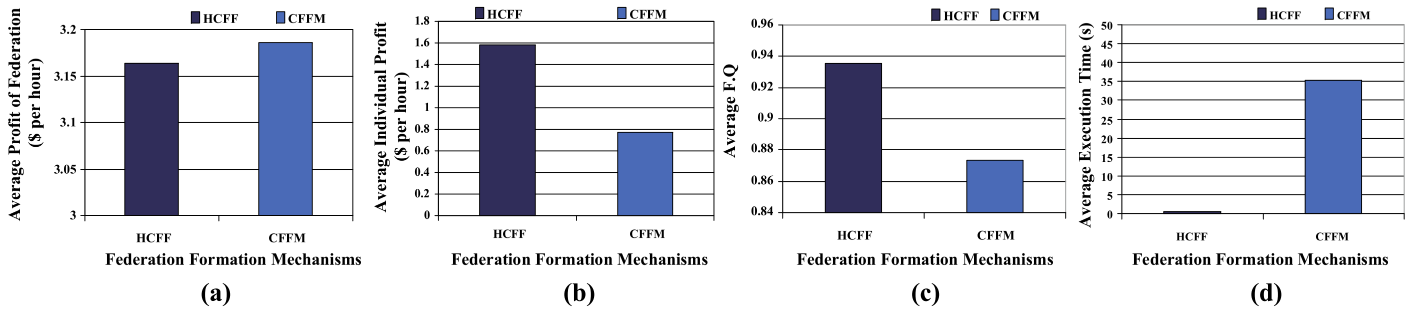


Fig. 6 **a** Comparison of average profit of a federation, **b** comparison of average individual profit of CSP in federation, **c** comparison of average quality of a federation and **d** comparison of average execution time of HCFF and CFFM mechanism

tion formed by CFFM mechanism. Lastly, Fig. 6d shows the execution time of the two mechanisms and it is seen that the HCFF mechanism heavily outperforms the CFFM mechanism. This is due to the reason that CFFM mechanism tries to find the best possible group of CSPs (which maximize profit of federation) out of a number of available CSPs by means of a high-complexity algorithm. The execution time of HCFF mechanism is almost negligible compared to CFFM.

In Fig. 7, we have compared the results obtained by our HCFF mechanism to the results obtained by the THCF mechanism based on small-size request (6, 5, 4). To be fair to their model, we have assumed a significant degree of maliciousness to be present between the existing service providers. It is seen from Fig. 7a–c that in the presence of malicious services, though HCFF mechanism manages to hold its own in case of profit (both profit of a federation and individual profit of a CSP), the QoS of federation is found slightly lower when compared to the THCF mechanism. This is because HCFF mechanism does not consider the malicious behavior of the CSPs while forming federation. Hence, it selects those CSPs as a part of federation who are malicious in nature and as a result QoS of federation gets diminished. But in the absence of any maliciousness among the existing service providers, we observe from Fig. 7d–f that HCFF heavily outperforms the THCF mechanism with respect to profit of federation, individual profit of CSPs in federation and quality of federation, except in the case of running time. This is because the main objective of THCF mechanism is to minimize maliciousness between CSPs but not to maximize profit and QoS of federation. Figure 7g shows the execution time of the two mechanisms, and it is seen that execution time of HCFF mechanism is comparatively greater than THCF mechanism. This is because, as HCFF mechanism uses an integer linear program to solve the proposed multi-objective optimization problem, it takes greater time to find the pareto-optimal solution of the proposed problem. Furthermore, from Fig. 7h it is observed that the size of the final federation formed by HCFF mechanism increases with the size of the request, whereas the federation size is constant for THCF mechanism. This is due to the reason that the THCF mechanism is independent of the request size and tries to find a federation partition that minimizes the number of malicious members in constituent federations, and therefore the size of federation is constant.

8 Conclusion

In this paper, we have presented a framework for cloud federation formation such that the solution of the trade-off among overall QoS and profit of federation is pareto optimal. The federation formation mechanism is formulated as an integer linear program (ILP) which maximizes the profit and QoS of the federation by striking a balance between the QoS and the profit of the CSPs taking part in federation. The framework takes care of need of both the cloud users and the cloud service providers. It is to be noted that the goal of maximizing the QoS delivered by the federation provides benefit to the cloud users and that of maximizing the profit earned by the federation provides benefit to the CSPs. A comprehensive performance evaluation has been carried out on a real cloud dataset for the proposed cloud federation formation mechanism. Our results show that our two mechanisms HCFF and ECFF yield very similar results. In certain

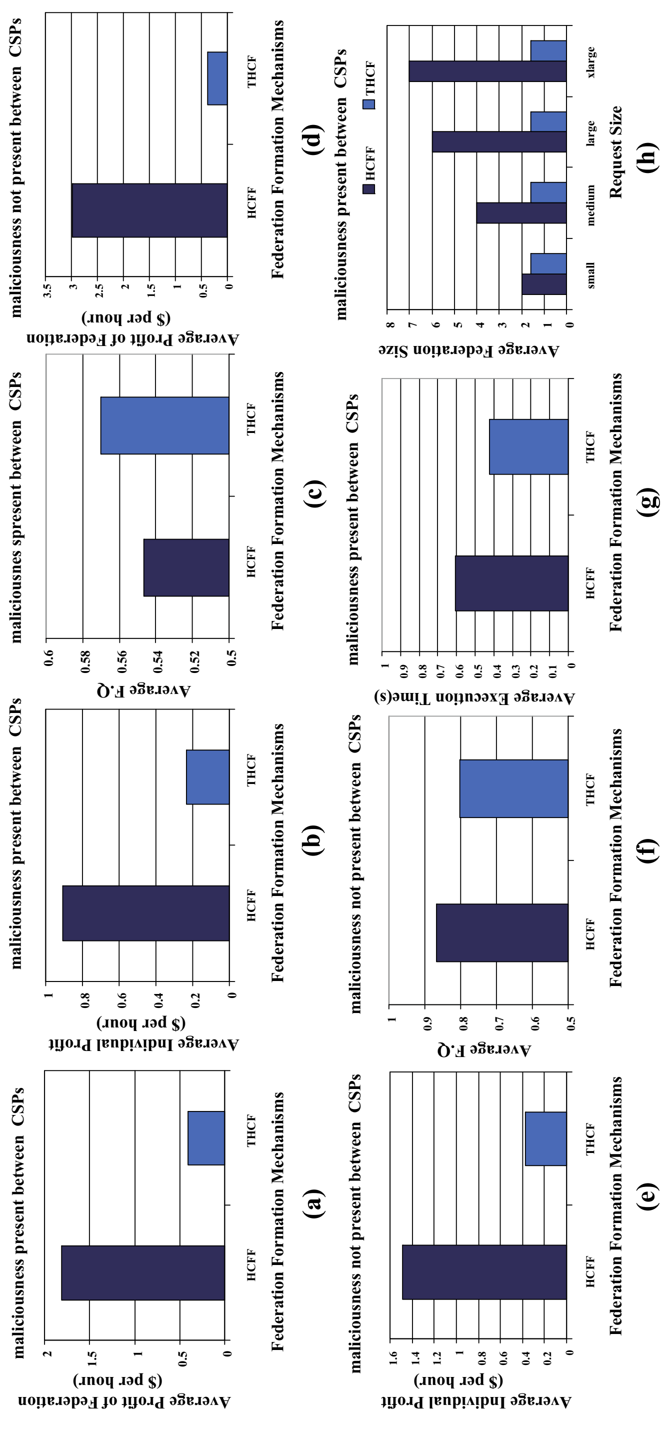


Fig. 7 **a** Comparison of average profit of a federation, **b** comparison of average individual profit of CSP in federation, **c** comparison of average quality of a federation, **d** comparison of average profit of a federation, **e** comparison of average individual profit of CSP in federation, **f** comparison of average quality of a federation, **g** comparison of average execution and **h** comparison of average size of federation formed by HCFF and THCF mechanisms

cases, ECFF performs better than HCFF. But the differences are marginal. However, the time complexity of ECFF is much higher than that of HCFF. Thus, HCFF might be a better choice for those who wish to execute the federation formation program in a heavily time-constrained situation. We have also compared our proposed mechanism with two other existing mechanisms CFFM [1] and THCF [3], and it is observed that the performance of our proposed mechanism HCFF is better in terms of profit and QoS of federation.

A federated system allows overloaded service providers to dispense their excess load by migrating their allocated virtual machines to under-loaded member service providers of their own federation. Further, virtual machine migration also allows individual service providers to increase availability and reliability of their services in the event of faults in the existing data centers of service providers. Therefore, in future it is necessary to study and address issues related to faults in the existing data centers and violation of QoS due to migration of virtual machines.

Acknowledgements This research is supported by National Fellowship for OBC, Government of India, UGC-UPE Phase-II at Jadavpur University and Visvesvaraya PhD Scheme of Ministry of Electronics & Information Technology, Government of India.

References

1. Mashayekhy L, Nejad MM, Grosu D (2015) Cloud federations in the sky: formation game and mechanism. IEEE Trans Cloud Comput 3(1):14–27
2. Niyato D, Vasilakos AV, Kun Z (2011) Resource and revenue sharing with coalition formation of cloud providers: game theoretic approach. In: 2011 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Newport Beach, CA, pp 215–224
3. Wahab OA, Bentahar J, Otrok H, Mourad A (2016) Towards trustworthy multi-cloud services communities: a trust-based hedonic coalitional game. IEEE Trans Serv Comput PP(99):1
4. Hung JC, Gangman Y (2017) Advances in next era cloud-empowered computing and techniques. J Supercomput 73(7):2843–2850
5. Rochwerger B, Breitgand D, Levy E, Galis A (2009) The reservoir model and architecture for open federated cloud computing. IBM J Res Dev 53(4):4:1–4:11
6. Rochwerger B, Breitgand D, Epstein A (2011) Reservoir when one cloud is not enough. IEEE Comput 44(3):44–51
7. Celesti A, Tusa F, Villari M, Puliafito A (2010) How to enhance cloud architectures to enable cross-federation. In: 2010 IEEE 3rd International Conference on Cloud Computing, Miami, FL, pp 337–345
8. Nordal A, Kvalnes A, Hurley J, Johansen D (2011) Balava: federating private and public clouds. In: Proceedings of IEEE World Congress on Services, pp 569–577
9. Yang X, Nasser B, Surridge M, Middleton S (2012) A business-oriented cloud federation model for real-time applications. Future Gener Comput Syst 28(8):1158–1167
10. Altmann J, Kashef MM (2014) Cost model based service placement in federated hybrid clouds. Future Gener Comput Syst 41:79–90
11. Van den Bossche R, Vanmechelen K, Broeckhove J (2010) Costoptimal scheduling in hybrid IaaS clouds for deadline constrained workloads. In: Proceedings of 3rd IEEE International Conference on Cloud Computing, pp 228–235
12. Goiri I, Guitart J, Torres J (2010) Characterizing cloud federation for enhancing providers' profit. In: IEEE 3rd International Conference on Cloud Computing, Miami, FL, pp 123–130
13. Hassan MM, Song B, Huh EN (2011) Distributed resource allocation games in horizontal dynamic cloud federation platform. In: IEEE 13th International Conference on High Performance Computing and Communications, Banff, AB, pp 822–827

14. Toosi A, Calheiros R, Thulasiram R, Buyya R (2011) Resource provisioning policies to increase IaaS providers profit in a federated cloud environment. In: Proceedings of 13th IEEE International Conference on High Performance Computing and Communications, pp 279–287
15. Chaisiri S, Lee BS, Niyato D (2012) Optimization of resource provisioning cost in cloud computing. IEEE Trans Serv Comput 5(2):164–177
16. Messina F, Pappalardo G, Rosaci D, Santoro C, Sarn GML (2015) A trust-aware, self-organizing system for large-scale federations of utility computing infrastructures. Future Gener Comput Syst 56:77–94
17. Lee YH, Huang KC, Shieh MR (2017) Distributed resource allocation in federated clouds. J Supercomput 73(7):3196–3211
18. Abdi S, PourKarimi L, Ahmadi M (2018) Cost minimization for bag-of-tasks workflows in a federation of clouds. J Supercomput 74(6):2801–2822
19. Li H, Wu C, Li Z, Lau FCM (2013) Profit-maximizing virtual machine trading in a federation of selfish clouds. In: Proceedings of IEEE on INFOCOM, Turin, pp 25–29
20. Zant BE, Amigo I, Gagnaire M (2014) Federation and revenue sharing in cloud computing environment. In: IEEE International Conference on Cloud Engineering, Boston, MA, pp 446–451
21. Rebai S, Hadji M, Zeghlache D (2015) Improving profit through cloud federation. In: 12th Annual IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, NV, pp 732–739
22. Wang WJ, Chang YS, Lo WT (2013) Adaptive scheduling for parallel tasks with QoS satisfaction for hybrid cloud environments. J Supercomput 66(2):783–811
23. Ye S, Liu H, Leung Y, Chu X (2017) Reinsurance-emulated collaboration mechanism in cloud federation. In: 2017 IEEE 10th International Conference on Cloud Computing (CLOUD). IEEE
24. Bellaiche M, Adel A, Talal H (2018) A cooperative game for online cloud federation formation based on security risk assessment. In: 2018 5th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2018 4th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom). IEEE
25. Ray B, Saha A, Khatua S, Roy S (2018) Quality and profit assured trusted cloud federation formation: game theory based approach. IEEE Trans Serv Comput. https://doi.org/10.1109/TSC.2018.2833854
26. Ray BK, Saha A, Roy S (2018) Migration cost and profit oriented cloud federation formation: hedonic coalition game based approach. Cluster Comput. https://doi.org/10.1007/s10586-018-2837-0
27. Ray BK, Sunirmal K, Sarbani R (2018) A game theoretic model for cloud federation, cloud computing for optimization: foundations, applications, and challenges. Springer, Cham, pp 73–97
28. Johnson P, Beverlin M (1970) Beta distribution, continuous univariate distributions-2. Wiley, New York, pp 37–56
29. Bouguessa M (2012) Modeling outlier score distributions, advanced data mining and applications. Springer, Berlin, pp 713–725
30. Figueiredo MAT, Jain AK (2002) Unsupervised learning of finite mixture models. IEEE Trans Pattern Anal Mach Intell 24(3):381–396
31. Ji Y, Wu C, Liu P, Wang J, Coombes KR (2005) Applications of beta-mixture models in bioinformatics. Bioinformatics 21(9):2118–2122
32. Value of brands (Online). https://www.b2binternational.com/publications/value-of-brands/. Accessed 17 May 2017
33. Holbrook MB (1992) Product quality, attributes, and brand name as determinants of price: the case of consumer electronics. Mark Lett 3(1):71–83
34. Firth M (1993) Price setting and the value of a strong brand name. Int J Res Mark 10(4):381–386
35. Shehzad U, Ahmad S, Iqbal K, Nawaz M, Usman S (2014) Influence of brand name on consumer choice and decision. IOSR J Bus Manag (IOSR-JBM) 16(6):72–76
36. G2 crowd based rating value for different IaaS service providers (Online). https://www.g2crowd.com/categories/infrastructure-as-a-service-iaas. Accessed 6 Sept 2017
37. Miettinen K (1999) Nonlinear multiobjective optimization. Springer international series in operations research and management science, vol 12. Springer US, New York
38. http://lpsolve.sourceforge.net/5.5/
39. Klee V, Minty GJ (1969) How good is the simplex algorithm? In: Shisha O (ed) Inequalities III (Proceedings of the Third Symposium on Inequalities held at the University of California, Los Angeles, California, 1–9 Sept 1969, Dedicated to the Memory of Theodore S. Motzkin). Academic Press, New York, pp 159–175 (1972)
40. Schrijver A (1998) Theory of linear and integer programming. Wiley. ISBN 0-471-98232-6 (mathematical)

41. Borgwardt KH (1987) The simplex method: a probabilistic analysis. Algorithms and combinatorics (study and research texts), vol 1. Springer, Berlin, p xii+268
42. Amazon EC2 based chargeable price of virtual machine instances (Online). https://aws.amazon.com/ec2/pricing/. Accessed 6 Sept 2017
43. CloudHarmony Service Status of different cloud service providers (Online). https://cloudharmony.com/status-group-by-regions. Accessed 6 Sept 2017
44. G2 crowd based rating value for Amazon EC2 (Online). https://www.g2crowd.com/products/amazon-ec2/reviews. Accessed 6 Sept 2017
45. Buyya R, Yeo C, Venugopal S, Broberg J, Brandic I (2009) Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. Future Gener Comput Syst 25(6):599–616
46. Ray BK, Khatua S, Roy S (2014) Negotiation based service brokering using game theory. In: IEEE International Conference on Applications and Innovations in Mobile Computing (AIMoC)
47. Grozev N, Buyya R (2014) Inter-cloud architectures and application brokering: taxonomy and survey. Softw Pract Exp 44(3):369–390