Computer Networks and Communications
https://ojs.wiserpub.com/index.php/CNC

Article

Optimizing Cloud Resource Allocation in Federated Environments through Outsourcing Strategies

Arash Mazidi

Department of Computer Engineering, Aliabad Katoul Branch, Islamic Azad University, Aliabad Katoul, Iran
E-mail: arash_mazidi_67@yahoo.com

Received: 12 April 2024; **Revised:** 14 June 2024; **Accepted:** 20 June 2024

Abstract: Cloud computing enables users to access required resources, with the invention of high-end devices leading to exponential increases in cloud resource requests. This poses significant challenges for resource management due to the scale of the cloud and unpredictable user demands. This paper presents an approach to managing resources during peak request periods for virtual machines (VMs) by leveraging cloud federation, outsourcing requests to other federation members. An algorithm is proposed to initiate cloud federation and allocate customer requests within it. The primary objectives are to increase the profit of cloud providers and improve resource utilization. An ensemble algorithm maximizes profit using both the proposed algorithm and three established ones. Experimental results demonstrate that our method outperforms existing approaches in profit, resource utilization, and rejected requests in most scenarios.

***Keywords*:** cloud resource management, cloud federation, request outsourcing

1. Introduction

With the advancement of information technology, computational tasks are now expected to be performed consistently and ubiquitously. Consequently, there's a growing demand for accessing computing services rather than investing in costly hardware and software. To address these needs, cloud computing has emerged. In this model, users lease physical infrastructure from third-party providers like Google or Amazon instead of owning it [1, 2].

Cloud computing creates virtual resources that users can access. Software and data reside on servers and are provided to users on demand. However, cloud resources such as network bandwidth, CPU cycles, and memory space are not static; if left unused, they become idle, resulting in wasted capacity within the data center. Providers are thus compelled to allocate resources to meet user demands and maintain quality of service (QoS). To address these challenges, providers can tap into unused resources from other providers, facilitating a cloud federation [3].

A cloud federation enables Infrastructure as a Service (IaaS) providers to address resource shortages during peak demand for virtual machines (VMs) by outsourcing requests to other federation members. Figure 1 illustrates a cloud federation setup, where each provider operates autonomously to serve its clients. IaaS providers have the capability to terminate spot VMs, freeing up resources. Therefore, they must carefully consider pricing, profitability, and QoS. Cloud service providers often accept numerous new requests to maximize profits while ensuring QoS based on the service level agreement (SLA) [4].

DOI: https://doi.org/10.37256/cnc.2220244747

The core of this approach lies within the cloud exchange center (CEC), which maintains a repository of service information. When Infrastructure as a Service (IaaS) providers encounter a shortage of local resources, they submit a request to the CEC for resource allocation to other federation members. The CEC's responsibility is to compile a roster of suppliers along with their respective service costs to fulfill these requests. A pivotal component, the cloud coordinator, determines whether to redistribute additional resources from other cloud providers. Within a cloud federation, providers share a portion of their unused capacity with fellow members [5, 6]. Hence, the cloud federation is positioned as a mechanism to optimize resource utilization, ultimately boosting profitability. Providers can lease out their surplus capacity to those in need, mitigating the wastage of underutilized resources. By leveraging other federation members, providers can also mitigate request rejections and circumvent local infrastructure limitations [7].

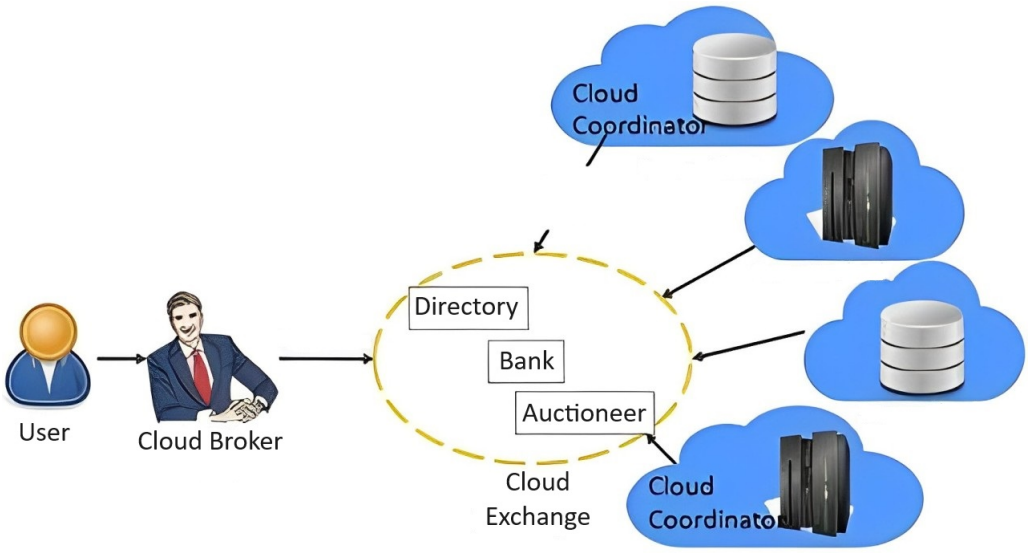

Figure 1. Federated Cloud Architecture [4].

Outsourcing requests isn't just about cloud federation; it's also about boosting application capacity or scalability across clouds, distributed grids, and clusters [8–10]. Regarding cloud federation, Hammoud et al. proposed a method using genetic algorithms and evolutionary game theory to address creating profitable federated clouds while maintaining stability among members amid dynamic strategies that could compromise Quality of Service (QoS). Their evolutionary game model, by achieving a stable state where no provider has an incentive to redistribute resources, yielded better profit and QoS outcomes [11]. Mashayekhy et al. offered a model for constructing cloud federations tailored for executing Map-heavy/Reduce-heavy programs, considering trust and reputation among participating cloud providers. This mechanism ensures high reliability and profitability for the providers [5]. Ray et al. suggested a proactive fault tolerance system within federations, predicting issues based on CPU temperature. Their approach treats fault tolerance as a multi-objective optimization problem, aiming to maximize profit and minimize migration costs while maintaining resource balance [12]. Furthermore, Kumar Ray et al. introduced a broker-based architecture for cloud federation, using a hedonic coalitional game to form federations that enhance provider satisfaction in terms of QoS and profit [13]. Ahmed et al. proposed a QoS-aware trust evaluation mechanism to assist in selecting helpful providers for cloud service federation across various domains [14]. Zhang et al. enhanced task prioritization by introducing high priority queues within the cloud and implementing a two-level asynchronous scheduling paradigm. They also devised two multi-resource fair scheduling methods for clouds and federations, adjusting user task priority dynamically using a modified weighting factor [15]. Ma et al. suggested a technique to translate member contributions in a cloud federation into a QoS metric [16]. Mohebbi Moghadam et al. explored the interaction between independent cloud providers and smart grid utilities within demand response programs,

proposing federation development among cloud service providers [17]. Kostler et al. introduced the NetFed network federation technique, facilitating inter-cloud operations via shared overlay networks, optimizing infrastructure selection for specific applications [18]. Saidi et al. conducted a systematic and comprehensive review of the recent literature published between 2016 and 2023 to address the challenges and highlight the current state of research in the task scheduling and VM placement to resource allocation in cloud computing [19].

However, the practice of distributing a portion of capacity as spot VMs poses challenges for IaaS providers. Spot VMs are terminated by providers when the current price exceeds what clients are willing to pay for resource usage. Consequently, spot VMs are often offered at a reduced cost compared to on-demand VMs. While spot VMs enhance provider profits and data center efficiency, this research focuses on addressing the issue of VM provisioning to maximize provider profit.

This study endeavors to tackle the challenge of outsourcing alongside the potential termination of spot VMs within the context of a data center. The primary objective is to formulate policies that empower service providers with a range of options for managing incoming requests, encompassing refusal, outsourcing, or the termination of rented spots to optimize resource allocation. Ultimately, the overarching aim remains the enhancement of provider profitability. Therefore, we propose a cloud resource management approach based on the ability of outsourcing the requests in the cloud federation. It outsources requests whenever provider has not available resources to other members to reach the maximum profit for the providers and resource utilization. The proposed strategies designed to assist service providers in making decisions that enhance resource utilization and profitability.

The key contributions of this research are as follows:

- Selection of the most suitable cloud facility during peak demand
- Augmentation of cloud federation profitability
- Mitigation of SLA violations through cloud federation utilization

2. Proposed Cloud Resource Management Approach

2.1 *Problem Statement*

In this section, we discuss the proposed approach. IaaS cloud computing providers provide a variety of VMs and pricing schemes that can be utilized to execute applications and meet other client needs. Three different price models request-based, reserved, and spot are available on Amazon EC2. We consider two services with pricing based on Amazon (request-based and spot services). Each cloud provider also has customers and a data center. Customers who are willing to tolerate the possibility of termination can lower the cost of using a VM by using spot VMs.

According to this approach, users submit a spot VM request that specifies the quantity of spot VMs needed as well as the proposed price (PP) for using a VM per hour. As a result, if the service request is greater than the going rate, the request is placed in a queue and is not processed till the spot price drops below the PP. Therefore, VMs won't stop operating until the customer decides to do so or until their cost surpasses the PP. The supplier terminates the low-PP VMs and switches them out for higher-PP VMs or demand-based requests in the case of a resource shortfall. The likelihood of VM termination is thereby decreased by increased PP.

We take into consideration both persistent and one-time spot VM requests. While persistent requests are maintained in the data center for re-execution until completion, one-time requests are spot requests that do not restart after the provider ends them. Additionally, if providers only offer a few resources, SLA violations may rise. As a result, suppliers should employ alternative strategies to fulfill new demand-based requests and so boost their profits. The suggestion is obtaining resources from other cloud service providers who can support new demand-based requests. Therefore, providers ought to be a part of a cloud federation so that they can sell their idle resources to other members in the federation for less money than they would charge a customer.

In addition to SLAs, an agreement between federation members is required that it is called federation level agreements (FLAs). Each provider must charge their shared resources according to the data center's idle capacity in accordance with the FLA. Equation (1) is used to determine the current cost of a resource per hour in the federation [4].

$$F = \frac{M_p - M_{idle}}{M_p} \times (F_{\max} - F_{\min}) + F_{\min} \tag{1}$$

where, F is the resource price in the federation, M_p, M_{idle}, $F_{\max}$ and $F_{\min}$ are total capacity, idle capacity in the data center, maximum price of demand-based resource and minimum accepted price by providers, respectively. The provider does not sell its resources at lower price of $F_{\min}$. Therefore, this pricing mechanism will lead to more resources with cheaper price for providers.

2.2 *Three Primary Request Management Policies*

2.2.1 *Non-Federated Totally In-House (NFTI) Policy*

According to this policy, if a demand-based request is received by the provider and the provider has already reached its capacity, the provider will first think about terminating the spot VMs with a lower price to accommodate a more useful demand-based request. The request will be denied if this action does not supply enough resources for the new request. To determine the maximum profit that a provider may make without a federation, this policy is regarded as a fundamental one [4].

2.2.2 *Federation-Aware Outsourcing Oriented (FAOO) Policy*

In accordance with this principle, once a demand-based request reaches the provider, any fully used provider checks other federation providers. After that, it assigns the request to the provider with the cheapest pricing. Spot VMs will be terminated as the final step in responding to new demand-based requests if outsourcing is not an option [4].

2.2.3 *Federation-Aware Profit Oriented (FAPO) Policy*

This approach is predicated on the premise that, while the termination of spot VMs reduces profitability, the replacement of spot VMs with demand-based requests boosts profits. Additionally, the termination of spot VMs might cause spot prices to rise. Decisions under this policy are predicated on the provider's immediate profit at time Δt [4].

2.3 *Resource Management in the Cloud Federation*

The proposed cloud resource management in the cloud federation is presented in this section. We have first proposed an algorithm which is shown in Part 1 of Algorithm 1, to initiate a cloud federation. Then we introduce a new algorithm to allocate costumer requests to resources (Part 2 of Algorithm 1). Furthermore, we proposed an ensemble algorithm using our algorithms and three previous works (NFTI, FAOO, and FAPO) which is in Algorithm 2. In fact, the ensemble algorithm combines the algorithms with different policies and functions.

In the first part of the proposed approach, N is the number of providers in the federation, k and η are the number of VMs in a provider and federation identification number, respectively. In the lines 4–5, the i^{th} cloud provider is initiated, and a data center is initiated to it. In line 6, the created cloud provider i is connected to the cloud federation with identification number η. Then, lines 7–8 assign k VMs to i^{th} cloud provider. Finally, i^{th} cloud provider is added to the cloud federation.

Algorithm 1 Proposed approach—Part 1: Federation initiation, Part 2: Resource management

Part 1: Cloud Confederation Initiation
Input: N: Number of providers in the federation, k: Number of VMs in the provider, η: Federation identification number
for i = 1 **to** N **do**
  $cp_i \leftarrow$ InitializeCloudProvider() // cp_i is the i^{th} cloud provider
  Initiate a data center for cp_i
  AssignCloudProviderToConfederation(cp_i, η)
  Initiate VM_1 to VM_k for cp_i
  cp.add(cp_i)
end for
Part 2: Resource management in the federation
Input: Δt: time, N: Number of providers in the federation
for i = 1 **to** N **do**
  if request is demand-based **then**
    if $VM_{available_i}(\Delta t) > VM_{demand_i}(\Delta t)$ **then**
      $VM_{available_i}(\Delta t) \leftarrow VM_{available_i}(\Delta t) - VM_{demand_i}(\Delta t)$
      $VM_{demand_i}(\Delta t) \leftarrow 0$
    else
      $VM_{demand_i}(\Delta t) \leftarrow VM_{demand_i}(\Delta t) - VM_{available_i}(\Delta t)$
      $VM_{available_i}(\Delta t) \leftarrow 0$
      if $VM_{available_j}(\Delta t) > VM_{demand_i}(\Delta t)$ **then**
        $VM_{available_j}(\Delta t) \leftarrow VM_{available_j}(\Delta t) - VM_{demand_i}(\Delta t)$
        $VM_{demand_i}(\Delta t) \leftarrow 0$
      else
        $VM_{demand_i}(\Delta t) \leftarrow VM_{demand_i}(\Delta t) - VM_{available_j}(\Delta t)$
        $VM_{available_j}(\Delta t) \leftarrow 0$
        if $VM_{demand_i}(\Delta t) > 0$ **then**
          while $VM_{demand_i}(\Delta t) \neq 0$ **do**
            Terminate $VMSpot_i$ by $provider_i$ and add it to $VM_{available_i}(\Delta t)$
            $VM_{available_i}(\Delta t) \leftarrow VM_{demand_i}(\Delta t)$
          end while
        else if $VM_{available_i}(\Delta t) > VMSpot_i(\Delta t)$ **then**
          $VM_{available_i}(\Delta t) \leftarrow VM_{available_i}(\Delta t) - VMSpot_i(\Delta t)$
          $VMSpot_i(\Delta t) \leftarrow 0$
        else
          $VMSpot_i(\Delta t) \leftarrow VMSpot_i(\Delta t) - VM_{available_i}(\Delta t)$
          $VM_{available_i}(\Delta t) \leftarrow 0$
        end if
      end if
    end if
  end if
end for

Second part of the proposed approach demonstrates the proposed algorithm for dynamic resource management in a cloud federation. The algorithm manages cloud provider members and resource allocation in the federation at the time Δt. It is assumed that the number of available VMs for the cloud provider i is equal to the total capacity of its VMs.

Then, we examine the available VMs to provide service to all demand-based requests. If available VMs are more than demand-based requests, all requests are allocated to VMs (Lines 14–16). Otherwise, we would not have available VMs (lines 18–30). Then, if all demand-based requests have not been allocated to VMs, they can be allocated to rented VMs of other providers of cloud federation. Therefore, demand-based requests are allocated to VMs of cloud federation providers. If there was some requested, VM spot should be terminated and use the VMs for requests. Finally, based on this policy and the remaining resources of provider, it executes spot requests (lines 31–36).

In this study, ensemble algorithm is proposed to manage the resources of the cloud federation. It is a combination of four algorithms (Proposed, NFTI, FAOO, and FAPO). The ensemble algorithm serves n requests of the cloud provider at time Δt. It evaluates the profit for provider using four algorithms. It also examines the number of requests that can be executed by each algorithm. Therefore, total profit is based on profit of serving and renting the resources. Then, resource allocation will be done based on profits of algorithms at time interval Δt.

Algorithm 2 Proposed Ensemble Algorithm

```
1: Input: N: Number of providers in the federation
2: for i←1 to N do
3:     scores[0]←Profit(Proposed) /* the profit provided by the Proposed algorithm*/
4:     scores[1]←Profit(NFTI) /* the profit provided by the NFTI algorithm*/
5:     scores[2]←Profit(FAOO) /* the profit provided by the FAOO algorithm*/
6:     scores[3]←Profit(FAOO) /* the profit provided by the FAOO algorithm*/
7: end for
8: switch (best(scores)):
9:     case 0: do request allocation using Proposed algorithm
10:    case 1: do request allocation using NFTI
11:    case 2: do request allocation using FAOO
12:    case 3: do request allocation using FAPO
13: end switch
14: end
```

The proposed approach introduces a unified initialization process for cloud providers, ensuring consistent configuration across the entire federation. This method reduces the complexity and potential errors associated with manual or ad-hoc configurations. It automatically assigns a predefined number of VMs to each cloud provider during initialization, streamlining the setup process. The ability to efficiently scale the federation by adding new providers and VMs without manual intervention significantly improves the scalability of cloud infrastructures. In addition, the automated and systematic initialization process reduces the time required to set up a new cloud federation, enabling faster deployment and integration of cloud resources. Furthermore, our algorithm introduces a hierarchical approach to resource allocation, prioritizing demand-based requests, and reallocating resources from other providers within the federation as needed. A novel mechanism to terminate spot VMs and reallocate them to high-priority requests ensures optimal use of resources and minimizes the risk of resource shortages. It ensures that resources are used optimally, reducing wastage, and improving overall efficiency. In addition, by prioritizing high-demand requests and reallocating resources dynamically, the algorithm enhances the quality of service, ensuring that critical requests are always met.

The ensemble algorithm combines four different resource management algorithms (Proposed, NFTI, FAOO, and FAPO), leveraging their unique strengths to maximize overall profit. It evaluates the profit from each algorithm and dynamically selects the most profitable one for resource allocation, ensuring that the most economically advantageous decisions are made.

3. Performance Evaluation of the Proposed Approach

The performance evaluation of the suggested resource management approach is presented in this section. The proposed approach is simulated using Cloudsim, and the parameters were altered in accordance with the environment's real-world characteristics to match the simulation experiments with reality. The experiments were carried out using a Windows 10 Professional computer with an Intel Core i7 processor, 12 GB of RAM, NetBeans IDE 7.3, and CloudSim Toolkit 3.0. It should be noted that each experiment was carried out ten times for greater accuracy, and the results are presented as an average.

3.1 *Experimental Metrics and Configuration*

Each cloud federation contains several providers, and providers have different VMs. To simplify the task, we have assumed that only one type of VM is provided by the providers. Table 1 presents the information of Three types of VMs.

Amazon Web Services (AWS) pricing is used by providers [1]. This means that VM costs is 0.085 per hour for customers. Regarding spot VMs, the provider charges customers based on the spot price, which fluctuates in accordance with the minimum PP in the system and the availability of resources on a regular basis. Each spot VM's price is determined for the entire hour at the start of each hour.

We assumed that customers do not propose PP higher than the requested price because the higher PP for resources are like demand-based requests. Thus, PP is a random value with uniform distribution between 0.020 and 0.085. We also use these values for F_{min} and F_{max} in Equation (1), respectively.

Table 1. Information of different types of VMs.

Type of VM	Large	Medium	Small
CPU (MIPS)	4000	3500	3000
RAM (MB)	8192	4096	2048
Price of execution ($)	1.1	0.88	0.64
Price of Initialization ($)	2.5	2	1.5

We used a 9-days workload of the Lublin model. We consider the number of nodes on each Lublin request as the number of VMs, and this number is limited to 32 per request.

We have used the profit, number of rejected VM requests and utilization to assess the impact of proposed approach.

Profit: This metric is defined as the profit of provider during a time Δt for demand-based and spot requests, and cooperation with federation members, which is shown in Equation (2). Further, we ignore the operating costs.

$$Profit(\Delta t) = Revenue(\Delta t) - Cost(\Delta t) \tag{2}$$

Utilization: This metric is the ratio of the hours of using the VMs to the maximum hours of VMs at time interval Δt, which is shown in Equation (3).

$$Utilization(\Delta t) = \frac{Used\ VMs\ (\Delta t)}{All\ VMs\ in\ (\Delta t)} \tag{3}$$

Number of rejected VMs requests: This metric shows the number of rejected requests for VMs. This rejection is just for demand-based requests because providers will never reject spot requests. It should be noted that this metric does not indicate the number of rejected requests because each request may contain more than one VM.

Total evaluation: This metric is calculated based on the normalized form of the average of previous metrics. We first calculate the average of profits, number of rejected VMs, and utilization separately. Then, we normalize the values between 0 and 1.

3.2 *Experimental Results*

Experimental results of the proposed approach and other related approaches that are reviewed early are presented in this section.

3.2.1 *Evaluation of Spot Request Percentage*

In this scenario, we consider five providers in the federation, and we change the ratio of spot request to evaluate the methods. Figure 2 shows the impact of spot requests percentage on the methods. When spot requests are few, the NFTI method has suitable performance, because it has no specific policy. The proposed method and ensemble method are the best with each percentage of spot requests. Specially, these methods have best performance with 30% spot requests. Furthermore, the resource utilization of proposed and ensemble methods has been excellent in the middle intervals because it has been kept at 90%. The resource utilization has also increased as rejections have decreased. When the number of rejected VMs decreased, providers with proposed and ensemble methods get more profit. In general, the proposed and ensemble methods have best performance among its counterparts in different percentage of spot requests.

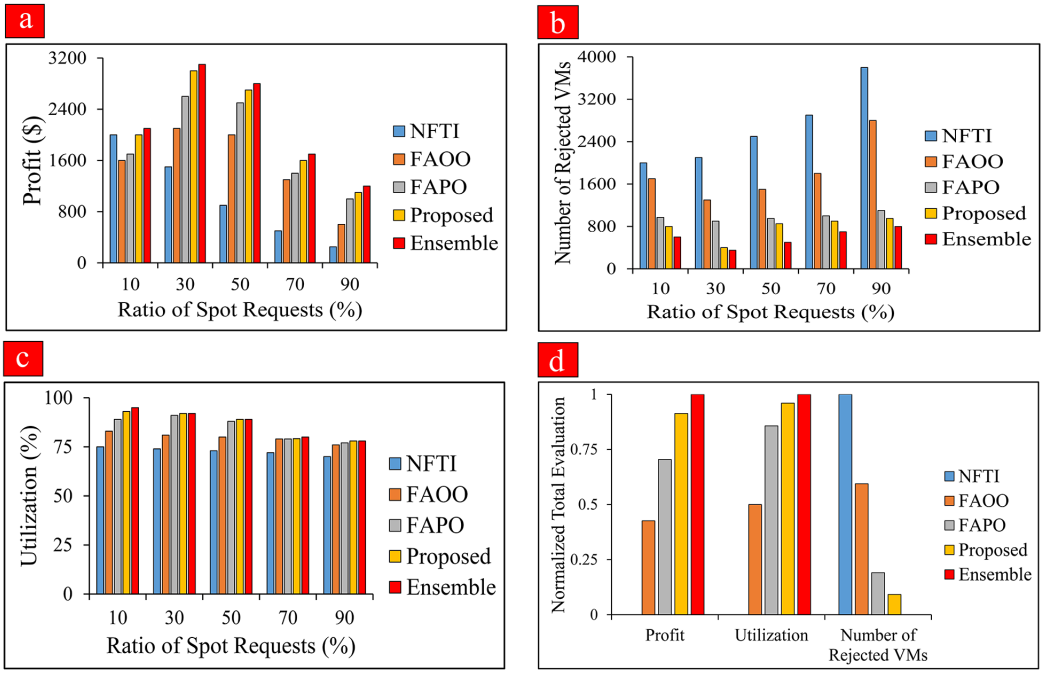

Figure 2. The evaluation of spot requests percentage.

3.2.2 *Evaluation of Number of Providers*

Figure 3 shows the evaluation of the number of providers. It shows when there are fewer providers, profit will be fewer. In the NFTI method, the utilization is fixed across different number of providers, indicating that there is no exact relationship between utilization and cloud providers. The results indicate the number of providers has no significant effect on the profit, utilization and VMs rejection of the NFTI method. In the FAPO and FAOO methods, when the providers are increased, the interactions between cloud providers will be increased and at the result, the utilization will be increased. In the proposed method, when the providers are increased, the number of rejected VMs are decreased, but the profit of providers due to outsource the requests don't increase. In general, the overall profit of the federation has increased. The federation's profit increases due to user satisfaction and reject reduction. An outstanding advantage has been observed in the ensemble method.

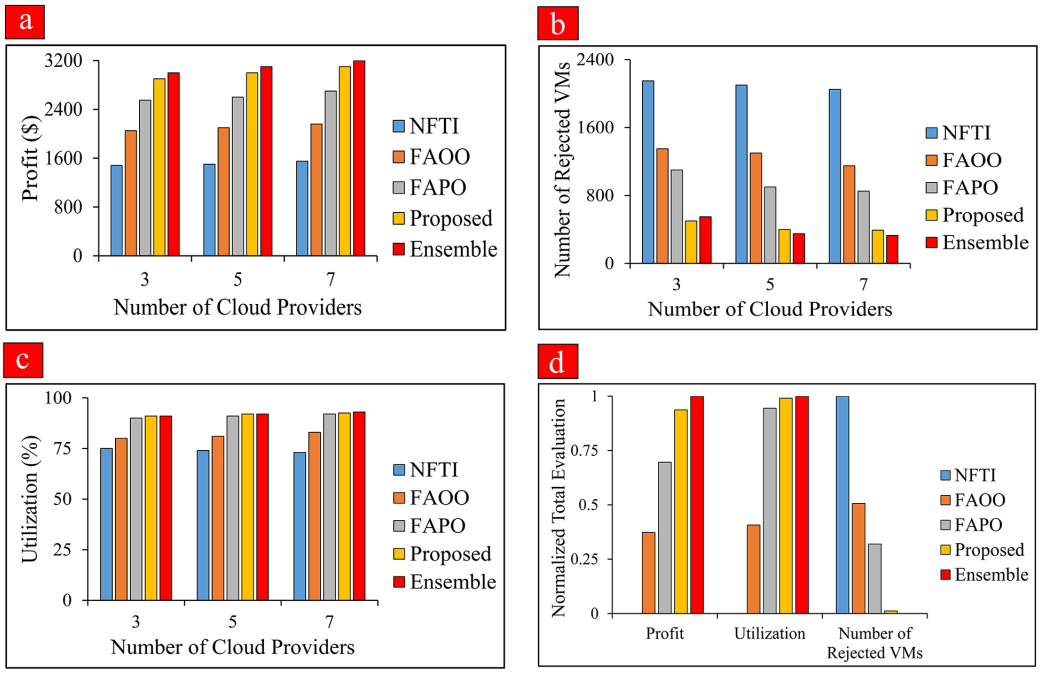

Figure 3. The evaluation of the number of providers.

3.2.3 *Evaluation of Number of Requests*

Figure 4 shows the evaluation of the amount of the requests. All methods increase the profit of providers with increasing the requests. The number of rejected VMs has increased but utilization has not changed, significantly. This is because with increasing resources, utilization has not changed in any of them. In the proposed method, the number of rejected VMs has increased and the utilization has been almost the same, but the profit has increased. This is because we have already estimated the desired number and rented the resources we want. In the ensemble method, the utilization is in an ideal state, and the profit has increased more than other methods.

3.3 *The Complexity of the Proposed Algorithm*

In this section, we discuss about the time complexity of two parts of the proposed algorithm. In the first part, we proposed a cloud confederation initiation algorithm, including:

Cloud Confederation Initiation
1 Iterate over N cloud providers.
2 Initialize each cloud provider.
3 Initiate a data center for each cloud provider.
4 Assign each cloud provider to the cloud federation.
5 Assign k VMs to each cloud provider.
6 Add each cloud provider to the cloud federation.

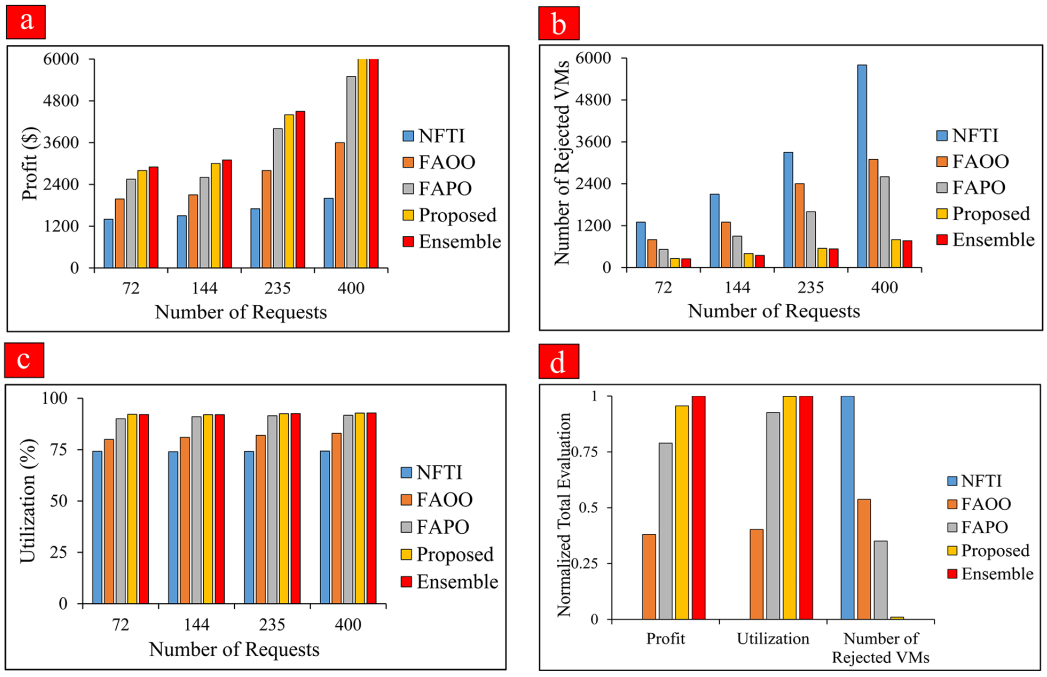

Figure 4. The evaluation of the number of requests.

Time complexity for this part of the algorithm:

Step 1 (Iteration): $O(N)$

Steps 2–4 (Initialization and Assignment): These steps involve constant-time operations for each provider, so they contribute $O(1)$ each per provider.

Step 5 (Assign k VMs): This involves a loop of k operations for each provider, contributing $O(k)$ per provider.

Overall, for N providers, the complexity of this part is: $O(N \times (1+k+1)) = O(Nk)$

In the second part of the proposed algorithm, we have these summarized lines:

Resource management in the federation
1. Iterate over N providers.
2. For each provider, check if a request is demand-based.
3. If so, compare available VMs with demand and allocate accordingly.
4. If demand exceeds availability, try allocating from other providers.
5. If still unmet, terminate spot VMs and allocate.

Time complexity for this part is:

Step 1 (Iteration): $O(N)$

Steps 2–5 involve nested conditionals and loops. The worst case occurs when there is a need to terminate and reallocate spot VMs, which involves additional iteration over VMs.

Steps 6–9 involve checking and allocation from another provider, which could be $O(N)$ in the worst case.

Termination and reallocation loop (Steps 16–18): In the worst case, it iterates over all VMs, leading to $O(k)$ operations.

Overall, for N providers, the complexity considering nested iterations and conditions is: $O(N \times (1+k+N)) = O(N^2+Nk)$

The overall complexity of the proposed algorithm is dominated by the resource management part due to its quadratic and linear terms, resulting in an overall complexity of $O(N^2+Nk)$. This means the performance of the system is heavily dependent on the number of providers N and the number of VMs k.

4. Discussion

The proposed algorithm for cloud confederation initiation systematically initializes and configures a federation of cloud providers, ensuring that each provider is assigned a specific number of virtual machines (VMs) and connected to the federation with a unique identification number. In addition, it is designed to handle a large number of providers (N) and VMs (k) efficiently, ensuring scalability.

Therefore, by methodically initializing each cloud provider and its resources, the algorithm minimizes the overall setup time for a cloud federation, enabling quicker deployment. Additionally, the structured approach ensures that all providers are uniformly initialized and integrated into the federation, promoting consistency and reliability in the cloud infrastructure.

Furthermore, the proposed algorithm dynamically manages resources based on demand, allocating available VMs to demand-based requests and reallocating resources from other providers if necessary. It includes a mechanism to terminate spot VMs and reallocate them to high-priority demand-based requests, ensuring optimal use of resources. Therefore, by dynamically reallocating resources based on demand, the algorithm ensures optimal utilization of available VMs, reducing wastage and improving efficiency. Further, the ability to reallocate resources from other providers and terminate spot VMs to meet demand-based requests enhances the quality of service, ensuring that high-priority requests are always fulfilled.

5. Conclusions and Future Works

In this study, we propose cloud resource management to enhance the profitability of Infrastructure as a Service (IaaS) providers through the utilization of cloud federation. Providers stand to gain from the federation by outsourcing requests to fellow members possessing idle resources. We conducted various experiments to evaluate the impact of provider decisions on performance metrics. The experimental findings indicate that the proposed methods result in increased provider profits and reduced VM rejection rates, while maintaining resource utilization at an acceptable level.

Moreover, the results reveal that demand-based requests yield greater profitability, especially for providers with a higher proportion of spot VMs and terminating spot VMs results in fewer disruptions to customer service. Conversely, outsourcing proves more lucrative when spot VMs are scarce, and terminating spot VMs leads to service interruptions. Additionally, the federation enables providers with low utilization to bolster profits by trading idle resources with other members.

Future research avenues could explore a security-aware approach to resource provisioning, meta-heuristic algorithms for federated resource management, and game theory methods to facilitate cooperation among federation members. Furthermore, the utilization of various types of VMs warrants consideration for future investigations.

Conflict of Interest

There is no conflict of interest for this study.

References

[1] A. Mazidi, M. Golsorkhtabaramiri, and M. Yadollahzadeh Tabari, "An autonomic risk- and penalty-aware resource allocation with probabilistic resource scaling mechanism for multilayer cloud resource provisioning," *Int. J. Commun. Syst.*, vol. 33, no. 7, May 2020, https://doi.org/10.1002/DAC.4334.

[2] A. Mazidi, M. Golsorkhtabaramiri, and M. Yadollahzadeh Tabari, "Autonomic resource provisioning for multilayer cloud applications with K-nearest neighbor resource scaling and priority-based resource allocation," *Softw. Pract. Exp.*, vol. 50, no. 8, pp. 1600–1625, Aug. 2020, https://doi.org/10.1002/SPE.2837.

[3] A. Mazidi, M. Mahdavi, and F. Roshanfar, "An autonomic decision tree-based and deadline-constraint resource provisioning in cloud applications," *Concurr. Comput.*, vol. 33, no. 10, May 2021, https://doi.org/10.1002/CPE.6196.

[4] A. N. Toosi, R. N. Calheiros, R. K. Thulasiram, and R. Buyya, "Resource provisioning policies to increase IaaS provider's profit in a federated cloud environment," in *Proc. 2011 IEEE Int. Conf. on HPCC 2011*, Banff, AB, Canada, Sep 2–4, 2011, pp. 279–287, https://doi.org/10.1109/HPCC.2011.44.

[5] L. Mashayekhy, M. M. Nejad, and D. Grosu, "A Trust-Aware Mechanism for Cloud Federation Formation," *IEEE Trans. Cloud Comput.*, vol. 9, no. 4, pp. 1278–1292, 2021, https://doi.org/10.1109/TCC.2019.2911831.

[6] A. Mazidi, E. Damghanijazi, and S. Tofighy, "An Energy-efficient Virtual Machine Placement Algorithm based Service Level Agreement in Cloud Computing Environments," *Circ. Comput. Sci.*, vol. 2, no. 6, pp. 1–6, 2017, https://doi.org/10.22632/ccs-2017-252-25.

[7] P. Varshney and Y. Simmhan, "AutoBoT: Resilient and Cost-Effective Scheduling of a Bag of Tasks on Spot VMs," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 7, pp. 1512–1527, Jul. 2019, https://doi.org/10.1109/TPDS.2018.2889851.

[8] M. Song and Y. Sang, "Secure Outsourcing of Matrix Determinant Computation under the Malicious Cloud," *Sensors*, vol. 21, no. 20, p. 6821, Oct. 2021, https://doi.org/10.3390/S21206821.

[9] L. Wang, Y. Tian, J. Xiong, "Achieving reliable and anti-collusive outsourcing computation and verification based on blockchain in 5G-enabled IoT," *Dig. Commun. Netw.*, vol. 8, no. 5, pp. 644–653, Oct. 2022, https://doi.org/10.1016/J.DCAN.2022.05.012.

[10] M. Suharmono, M. B. Alexandri, R. W. S. Sumadinata, H. A. Muhyi, "Outsourcing in supply chain: A bibliometric analysis," *Uncertain Supply Chain Manag.*, vol. 10, no. 4, pp. 1501–1508, Sep. 2022, https://doi.org/10.5267/J.USCM.2022.6.006.

[11] A. Hammoud, A. Mourad, H. Otrok, O. A. Wahab, H. Harmanani, "Cloud federation formation using genetic and evolutionary game theoretical models," *Future Gener. Comput. Syst.*, vol. 104, pp. 92–104, Mar. 2020, https://doi.org/10.1016/J.FUTURE.2019.10.008.

[12] B. K. Ray, A. Saha, S. Khatua, S. Roy, "Proactive Fault-Tolerance Technique to Enhhance Reliability of Cloud Service in Cloud Federation Environment," *IEEE Trans. Cloud Comput.*, vol. 10, no. 2, pp. 957–971, 2022, https://doi.org/10.1109/TCC.2020.2968522.

[13] B. K. Ray, A. Saha, S. Khatua, S. Roy, "Quality and Profit Assured Trusted Cloud Federation Formation: Game Theory Based Approach," *IEEE Trans. Serv. Comput.*, vol. 14, no. 3, pp. 805–819, May 2021, https://doi.org/10.1109/TSC.2018.2833854.

[14] U. Ahmed, A. Al-Saidi, I. Petri, O. F. Rana, "QoS-aware trust establishment for cloud federation," *Concurr. Comput. Pract. Exper.*, vol. 34, no. 3, p. e6598, Feb. 2022, https://doi.org/10.1002/CPE.6598.

[15] G. Zhang, R. Lu, W. Wu, "Multi-resource fair allocation for cloud federation," in *Proc. 21st IEEE Int. Conf. High Perform. Comput. Commun., 17th IEEE Int. Conf. Smart City, 5th IEEE Int. Conf. Data Sci. Syst., HPCC/SmartCity/DSS 2019*, Zhangjiajie, China, Aug. 10–12, 2019, pp. 2189–2194, https://doi.org/10.1109/HPCC/SMARTCITY/DSS.2019.00303.

[16] K. Ma, A. Bagula, H. Mauwa, A. Celesti, "Modelling Cloud Federation: A Fair Profit Distribution Strategy Using the Shapley Value," in *Proc. 2018 IEEE 6th Int. Conf. Future Internet Things Cloud, FiCloud 2018*, Barcelona, Spain, Aug. 6–8, 2018, pp. 393–398, https://doi.org/10.1109/FICLOUD.2018.00063.

[17] M. M. Moghaddam, M. H. Manshaei, M. N. Soorki, W. Saad, M. Goudarzi, D. Niyato, "On Coordination of Smart Grid and Cooperative Cloud Providers," *IEEE Syst. J.*, vol. 15, no. 1, pp. 672–683, Mar. 2021, https://doi.org/10.1109/JSYST.2020.2987017.

[18] J. Köstler, S. Gebauer, H. P. Reiser, "Network Federation for Inter-cloud Operations," *Lect. Notes Comput. Sci. (incl. subseries Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12718 LNCS, pp. 21–37, 2021, https://doi.org/10.1007/978-3-030-78198-9_2/COVER.

[19] K. Saidi, and D. Bardou, "Task scheduling and VM placement to resource allocation in cloud computing: challenges and opportunities," *Cluster Comput.*, vol. 26, no. 5, pp. 3069–3087, 2023, https://doi.org/10.1007/s10586-023-04098-4.