ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
Indian Journal of Science and Technology, *Vol 8(S9), 103–110, May 2015*
DOI: 10.17485/ijst/2015/v8iS9/60951

A QoS Guaranteed Selection of Efficient Cloud Services

S. K. Mahalingam[1*] and N. Sengottaiyan[2]

[1]Anna University, Chennai - 600044, Tamil Nadu, India; mahalingamphd789@gmail.com
[2]Indira Institute of Engineering and Technology, Thiruvallur - 602001, Tamil Nadu, India; nsriram3999@gmail.com

Abstract

Objective: The main objective of this work is to select the optimal cloud services by considering price and Quality-of-Service (QoS) concession in the cloud computing environment. **Methods:** In this manuscript an innovative technique is introduced that is called Optimal Selection of Cloud Service with Price for QoS Concession (OSCSPQC). In this technique the more sophisticated constraints like accountability, agility, assurance of service, security and privacy, and usability are included for improving VM placement strategies. Moreover, if the cloud services are reserved by customers, price and QoS concession mechanism are used to facilitate both providers and customers indicate their preferences for price and QoS metrics and search for equally acceptable prices and QoS metrics. **Results:** The OSCSPQC method shows high performance in terms of user satisfaction, response time and total utility when compared to the existing A Novel Customer Hints Scheduling (ANCHS) method. The proposed OSCSPQC method considers the customer hints and QoS requirements for selecting the cloud service. If the cloud load is 1, the user satisfaction in OSCSPQC is 97.2%, the total utility is 89% and response time is 9500 ms. The comparison results shows that the proposed method achieves better performance when compared to the existing system. **Conclusion:** The findings demonstrate that the OSCSPQC method is presented and suggested that this method high performance.

Keywords: Cloud Computing, Cloud Negotiation, IaaS Cloud, Service Measurement, Virtual Machine Scheduling

1. Introduction

Cloud computing is an internet-based computing used to illuminate the different computing conceptions that include a huge number of computers connected through a real-time network[1]. The main benefit of using infrastructure-as-a-Service is that it conserves customers and applications from entire administrative process and resource allowance rules of the underlying machinery. Virtualization is used to build exploitation of VMs as a main task in the IaaS cloud. The fixed service-level policies are used by customers where they compute the mapping quality of computational resources to VMs. Generally, assessment function is a nontrivial process for it needs knowledge of both the application at hand and the policies modifiable resource sharing within the physical infrastructure. Hien Nguyen Van et al.[2] suggested an automatic virtual resource organization system for service hosting platforms. Deepal Jayasinghe et al.[3] suggested a method called structural control aware virtual machine assignment system to improve the performance.

William E. Walsh et al.[4] developed a distributed architecture in a realistic prototype data center that illustrates how the utility functions ensure an assembly of autonomic elements to frequently optimize the use of computational resources in a vibrant, heterogeneous environment. Akshat Verma et al.[5] developed the design and implementation of power-aware application assignment controller in the condition of a situation with heterogeneous virtualized server clusters.

Borja Sotomayor et al.[6] suggested a resource provisioning method for handling workloads in which join the requests for resources in a specific time period. Jens Dittrich et al.[7] developed the cloud computing has augmented more popularity because of its simplicity and its ability to provide cloud resources.

Alexandru Iosup et al.[8] suggested a technique that utilizes the idea regarding virtualization and the resource

*Author for correspondence

time-allocation for allocating the resources to the customers. Simon Ostermann et al.[9] presented a new framework for investigating the performance of the cloud services. Donald Kossmann et al.[10] suggested a different architecture to affect cloud computing for database application and gives the results of a comprehensive estimation of the previous commercial cloud services. Ian Foster et al.[11] proposed a Quality-of-Service design which merges resource reservation and application adaptation. A. Souvik Pal et al.[12] presented different virtualization methods which are used to solve the difficult overloads and multiple software architecture. Various methods are studied and conducted on virtualization a range of problems included have frequently been presented in separation of each other.

Nefeli[1] is a virtual infrastructure gateway that performs logical assignment of VMs onto physical nodes. This can be accomplished by customer-provided deployment hints. The workload is modeled as patterns of data flows; calculations control/points, and essential network connections, customers can distinguish approving VM layouts. The deployment hints includes: 1. Resource utilization patterns, 2. VMs that may turn into a performance block; 3. Section of the requested virtual infrastructure that can be sustained by the subsistence of special hardware support. These hints are developed by Nefeli so as to redeploy VMs in the cloud and accomplish effectual task-flow execution. This also considers the high-level VM location rules set by the cloud administration, which intents to achieve energy efficiency and load balancing. But this hint based scheduling method only considers customer constraints.

For providing effectual cloud services, these limited constraints are not adequate. For providing efficient cloud services, the high-level constraints are also considered. So, in this article Optimal Selection of Cloud Service with price for QoS concession (OSCSPQC) which computes the quality and prioritize Cloud services. In order to provide high satisfaction for the customers, it is necessary to provide service-level agreements through concession. To create such reservations, customers and providers are essential to do the following: 1. Customers and providers need to specify the preferences for price and QoS metrics and 2. Identify the reciprocally acceptable prices and Quality-of service metrics. But the problem is there is little concession support for QoS concession between consumers and providers. So, an optimization algorithm is proposed by organizing the number of concurrent proposals to reduce the computational complexity.

2. Scheduling Cloud Services based on Customer Hints and QOS Requirements

2.1 Customer Hints

Nefeli is a successful infrastructure gateway which utilizes customer-provided deployment hints to make intellectual assignment of virtual machines into the physical nodes. The workloads are formed as the patterns of data flows, calculations, control points and requisite network connections, so that customers can make creative VM layouts. The VM layouts are created as deployment hints. The customer hints contain: 1. A resource consumption patterns among the virtual machines; 2. VMs that may become a performance blockage; and 3. Portions of the requested virtual infrastructure that can be assisted by the survival of particular hardware support. The high-level placement rules are also considered by Nefeli that can be set by the cloud administration. The main intent of cloud administration is to involve energy efficiency and load balancing.

Nefeli is an infrastructure access which presents a layer between the customer and the Infrastructure-as-a-Service cloud services. It interfaces with the lower level cloud services to handle the VM cycle and execute basic administrative processes. This interface is responsible for permitting customers for particular requests and also organizes the VM deployment and migration.

For example, the customer hints are:

- *MTraf* = This denotes that to deploy the same host a set of virtual machines so as to minimize the traffic in the physical network.
- *Support VM* = This represents the reservation of single hosting node for a particular VM.

2.2 Customer QoS Requirements

It is very challenging to decide which provider can satisfy the QoS requirements for the customers because of the enlargement of public cloud offerings. In a cloud environment, the services provided by the cloud provider have different prices and performance levels. So, the customer has difficult to select the best service provider. The computation of service levels is complicated for different cloud services. So, to choose the preeminent cloud services some of the factors like Accountability, Agility, Assurance of Service, Security and Privacy, and Usability are measured to rank the cloud services. By using this evaluation, the customers differentiate the cloud services.

2.3 Measurable Factors

2.3.1 Accountability

Accountability is an important factor to create the confidence of a customer on any Cloud provider. This metric is evaluated when evaluating and scoring services like adaptability, compliance, data ownership, provider ethicality, sustainability, etc.

2.3.2 Agility

Agility is a significant metric which illustrated how quickly the new capabilities are included into IT infrastructures. The agility factor is depends on whether the service is flexible, transportable, and adjustable.

2.3.3 Assurance

This factor indicates the probability of a Cloud service performing as promised in the SLA. Therefore, consistency, resiliency and service faithfulness are significant metrics in selecting Cloud services.

2.3.4 Security and Privacy

It is very challenging to protect the data and privacy. When the data is hosted in other organizations, the stringent security strategies are needed.

2.3.5 Usability

The usability is a significant factor in the quick adoption of Cloud services. This factor depends on the metrics like simplicity, Installability, Learnability, and Compatibility.

2.4 QoS Metrics for Selecting Cloud Services

2.4.1 Average Response Time

Average response time is defined by $\sum_i \frac{T_i}{n}$ in which T_i denotes the between when a customer requested for an IaaS service and when it is actually available and n is the total number of IaaS service requests.

2.4.2 Flexibility

Flexibility is defined as the capability of the service provider to regulate the changes in services according to the customers' requests. The time taken to upgrading the service to a higher level is called flexibility.

2.4.3 Energy Efficiency

Energy efficiency is defined as the percentage of total facility power taken for storage and network capacity. The formula for calculating energy efficiency is as follows: $EE = \frac{\sum Storage\,capacity + \sum Network\,capacity}{Total\,Energy}$.

2.4.4 Reliability

Reliability indicates how well the cloud service works without failure at a particular time interval. Consequently, it is defined according to the mean time to failure guaranteed by the cloud provider and the previous failures qualified by the customers. It is evaluated by: $Reliabilty = 1 - \frac{n_{failure}}{n} * p_{mttf}$, in which n_failure represents the number of customers who experienced a failure at a particular time interval, n represents the number of customers, p_{mttf} denotes the promised mean time to failure.

2.4.5 Stability

Stability is defined as the changeability in the performance of a service. The formula for stability is as follows: $\sum \frac{\alpha_{avg.i} - \alpha_{sla.i}}{\frac{T}{n}}$ where α denotes computational unit of the resource, network unit of the resource; $\alpha_{avg.i}$ is the observed average performance of the customer, α_{slai} is the promised values in the SLA; T denotes the service time; and n denotes the total number of customers.

2.4.6 Availability

Availability is defined as the percentage of time for accessing the cloud service.

2.4.7 Usability

Usability is defined as simplicity of using a cloud service. This factor can be quantified as the average time qualified by the preceding customers of the Cloud service to activate, learn, install and understand it respectively.

2.5 Optimal Cloud Service Selection Based on Score Value

Let us consider V being the virtual Machines to be deployed and H denotes the group of physical nodes, M denotes that the function from V to H (M: V $\rightarrow$ H). Nefeli selects the profile which outfits the constraints articulated for the task-flow. The profile contains information collected form customer hints and customer QoS requirements. The customer restraints are merged with VM specifications which creates deployment patterns. These patterns are used to match with VM necessities to physical node resources.

The customer hints and customer QoS requirements are called constraints. Each and every restraint is defined as a utility function F that creates a single deployment profile. The input for a deployment profile is taken as a utility function and returns the degree of restraint satisfaction in the range of [0, 1]. F: $M_{all} \leftrightarrow$ [0, 1].

In this equation Mall denotes the set of all probable deployment profiles.

$$CC = \sum_{Hint_i \in H_s} W_i\, hint_i(m) + \sum_{QoS_i \in QoS_s} W_i\, QoS_i(m) \quad (1)$$

In equation (1) the customer constraints are set of all customer hints and customer QoS requirements. In this equation, H_s denoted the set of all hints, QoS_s denotes the set of QoS requirements, and w represents the respective weights.

The deployment profile m is allocated a score, which is calculated by the formula:

$$\text{Score(m)} = \sum_{CC_i \in C_s} wiCCi(m) \quad (2)$$

In this equation (2) C_s denotes the set of all restraints and w denotes the respective weights. The deployment profile which has the maximum score is taken as the most favorable profile,

$$(m_{opt}) \geq \text{Score}(m_{\downarrow}q), \qquad \forall m_{\downarrow}q \in M_{\downarrow}all, \quad (3)$$

In this equation (3), M_{all} denotes the set of all possible deployment profiles. To discover optimal deployment profiles are NP-hard so to utilize simulated annealing to attain reasonable approximations. The neighborhood N_m of a deployment profile m is denoted as,

$$N_{\downarrow}m = \{N \in M_{\downarrow}all \mid ProbN(v) \neq m(v)) = d, \forall v \in V\} \quad (4)$$

In the equation (4) V denotes the set of all VMs, M_{all} is the set of all profiles, d represents the probability for a VM to be deployed on a hosting node other than the one set by profile m. Increasing d results in wider neighborhoods and avoids us from getting fascinated at local optima. To find the near-optimal VM-to-host mapping, this method decouples the profile assessment and generation from the process. This facilitates to place restraints into two types:

2.5.1 Soft Constraints

The degree of satisfaction of constraints that belong in this class contributes to the overall quality of the produced profile.

2.5.2 Hard Constraints

Postulations placed in this group have to be fulfilled to their full extent. Figure 1 Shows the flow chart for scheduling cloud services.

Algorithm 1: A Novel Customer Hints Scheduling Algorithm

Input: Set of VM and Set of physical nodes, Customer Constraints

Output: Optimal selection of profiles

1. Deploy the set of VMs and set of physical nodes N.
2. Customer QoS Requirements and hints are given as input
3. // Customer hints
4. Mtraf // Mtraf = Minimum traffic in the physical network
5. SupportVM // SupportVM = Reserve single hosting node for a specific VM
6. // Customer QoS requirements
7. AVR = $\sum_i \frac{T_i}{n}$ // Computation of Average response time
8. // Where T_i = time between the customer I requested for IaaS service, n = Total number of requests
9. $Flexibility = \frac{Upgrading\ the\ service}{total\ time}$ // Computation of flexibility
10. $EE = \frac{\sum Storage\ capacity + \sum Network\ capacity}{Total\ Energy}$ // Computation of Energy efficiency
11. $Reliabilty = 1 - \frac{n_{failure}}{n} * p_{mttf}$ // Computation of reliability
12. // Where n_failure = Number of customers who experienced failure in a time interval, n = Number of customers, p_{mttf} = promised mean time to failure
13. Stability = $\sum \frac{\alpha_{avg.i} - \alpha_{sla.i}}{\frac{T}{n}}$ // Computation of Stability
14. // Where α = Computation unit, $\alpha_{avg,i}$ = observed average performance, $\alpha_{sla.i}$ = promised values in the SLA, T = Service Time, n = Total number of customers
15. $Availability = \frac{(total\ service\ time) - (total\ time\ for\ which\ service\ was\ not\ available)}{total\ service\ time}$ //Computation of Availability
16. // Computation of Usability
17. Customer constraints = Customer hints + Customer QoS requirements
18. Combine customer constraints and VM specifications
19. Create a deployment pattern
20. $CC = \sum_{Hint_i \in H_s} W_i\, hint_i(m) + \sum_{QoS_i \in QoS_s} W_i\, QoS_i(m)$ // CC = Customer Constraints, = Set of all hints, = Set of QoS requirements, and w respective weights.

3. Optimization Method for Reserving Cloud Services with Price and QOS Concession

The optimization technique is developed for reserving cloud services with price and QoS concession. In order to achieve customer satisfcation, customers and service providers require to create service-level agreements whenever making reservations for cloud services. It is essential for both consumers and providers to reach an agreement on the price and QoS of a cloud service. There are some difficulties in the creatio of an agreement between a consumer and a provider such as 1. To decide the price of the service and 2. To decide the quality of a service.

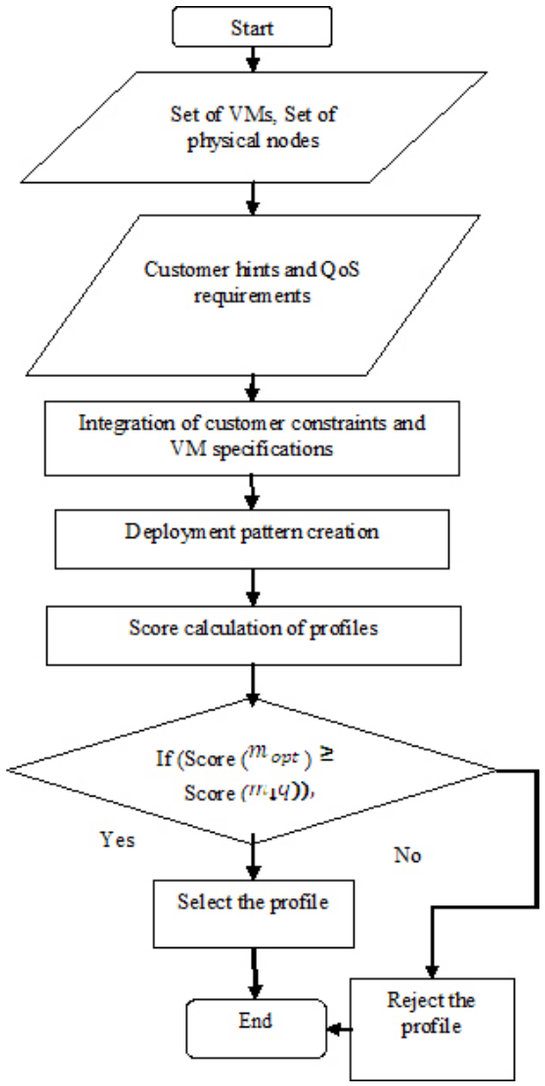

Figure 1. Flow chart for scheduling cloud services.

3.1 Price Utility for a Cloud Service

The consumers prefer the cheapest price for leasing a service; providers require to sell their Services at the highest prices. Let us consider IP_C and RP_C be the most preferred price and the least preferred price of a consumer agent. Let P represents a price that both agents reach an agreement. The price utility is calculated for both consumer and provider. S^P_{min} represents the minimum value that a customer and provider obtain for reaching an agreement at their particular reserve prices. To discriminate between not reaching an agreement and reaching an agreement at the reserve price, S^P_{min} is defined as 0.01. Suppose, if a customer and provider cannot attain an agreement before its concession deadline, they receive price value as zero. The range of the consumer's and provider's price value is $\{0\} \cup [S^P_{min},1]$.

3.2 QoS Utility for a Cloud Service

The utility value is also taken for Quality-of-service metrics like adaptability, reliability, stability and usability. Quality-of-Service is also another significant consideration for cloud service reservations. The QoS utility function model is considered for consumer and provider preferences for dissimilar time slots. Finally, the aggregation value is taken for both price and QoS metrics. The concession making method decides the amount of concession in each concession round, which denotes the reduction in an agent's expected total utility. So, by using this optimization method, to tradeoff between price and QoS concession is achieved and controlling the number of simultaneous proposals to reduce the computational complexity.

Figure 2 Shows the flow chart for optimization method for reserving cloud services with price and QoS concession.

Algorithm 2: A Novel Customer Hints Scheduling with price and QoS concession

Input: Set of VMs, set of physical nodes, Customer constraints, set of cloud service providers

Output: Optimal cloud service with price and QoS Concession

1. Deploy the set of VMs and set of physical nodes N
2. Set of cloud service providers and consumers give their price and QoS requirements
3. // Price utility for cloud service demanded by provider and customer
4. $$S^C_P(P) = \begin{cases} s^P_{min} + \left| \dfrac{LPP_C - P}{LPP_C - MPP_C} \right| & MPP_C \le P \le LPP_C \\ otherwise \end{cases} //$$

Price utility for Customer

5. //Where S = Price Utility for Customer, S^P_{min} = minimum utility value, LPP = least preferred price, MPP=Most preferred price, P = price for both agents reach an agreement

6. $$S^P_P(P)=\begin{cases} S^P_{min}+\left|\dfrac{P-LPP_P}{MPP_P-LPP_P}\right| & LPP_P \le P \le MPP_P \\ \end{cases} //$$

Price utility for Provider

7. // QoS Utility for cloud Services demanded by provider and customer
8. //Adaptability for Customer

9. $$S^C_A(A)^x=\begin{cases} s^A_{min} \quad T=T^x_{ah} or T=T^x_{at} \\ S^x_m, \quad T^x_{amh} \le T \le T^x_{amt} \\ S^x_m.\{\dfrac{(A-A^x_{ah})}{(A^x_{amh}-A^x_{ah})}\}^{\alpha^x_h}, \quad T^x_{ah} < T < T^x_{amh} \\ S^x_m.\left\{\dfrac{(A^x_{at}-A)}{A^x_{at}-A^x_{amt})}\right\}, \alpha^x_t \quad T^x_{amt} < T < T^x_{at} \end{cases}$$

10. // Adaptability for provider

11. $$S^P_A(A)=\begin{cases} s^A_{min}+(1-s^A_{min}).[1-(\dfrac{f^P_A(A)-1}{N^P_A-1})] \; FT_P \le T \le LT_P \\ 0, \quad otherwise \end{cases}$$

12. //Reliability for customer

13. $$S^C_R(R)^x=\begin{cases} s^R_{min} \quad T=T^x_{rh} or T=T^x_{rt} \\ S^x_m, \quad T^x_{rmh} £ T £ T^x_{rmt} \\ S^x_m.\{\dfrac{(R-R^x_{rh})}{(R^x_{rmh}-R^x_{rh})}\}^{\alpha^x_h}, \quad T^x_{rh} < T < T^x_{rmh} \\ S^x_m.\left\{\dfrac{(R^x_{rt}-R)}{R^x_{rt}-R^x_{rmt})}\right\}, \alpha^x_t \quad T^x_{rmt} < T < T^x_{rt} \end{cases}$$

14. //Reliability for provider

15. $$S^P_R(R)=\begin{cases} s^R_{min}+(1-s^R_{min}).[1-(\dfrac{f^P_R(R)-1}{N^P_R-1})] \; FT_P \le T \le LT_P \\ 0, \quad otherwise \end{cases}$$

16. //Stability for customer
17. /Stability for provider

18. $$S^P_{ST}(ST)=\begin{cases} s^{ST}_{min}+(1-s^{ST}_{min}).[1-(\dfrac{f^P_T(ST)-1}{N^P_{ST}-1})] FT_P \le T \le LT_P \\ 0, \quad otherwise \end{cases}$$

19. //Usability for customer

20. $$S^C_U(U)^x=\begin{cases} s^U_{min} \quad T=T^x_{uh} or T=T^x_{ut} \\ S^x_m, \quad T^x_{umh} \le T \le T^x_{umt} \\ S^x_m.\{\dfrac{(U-U^x_{uh})}{(U^x_{umh}-U^x_{uh})}\}^{\alpha^x_{uh}}, \quad T^x_{uh} < T < T^x_{umh} \\ S^x_m.\left\{\dfrac{(U^x_{ut}-U)}{U^x_{ut}-U^x_{umt})}\right\}, \alpha^x_t \quad T^x_{umt} < T < T^x_{ut} \end{cases}$$

21. //Usability for provider

22. $$S^P_U(U)=\begin{cases} s^U_{min}+(1-s^U_{min}).[1-\dfrac{f^P_U(U)-1}{(N^P_U-1)}] FT_P \le T \le LT_P \\ 0, \quad otherwise \end{cases}$$

23. //Energy efficiency for customer

24. $$E)^x=\begin{cases} s^E_{min} \quad T=T^x_{eh} or T=T^x_{et} \\ S^x_m, \quad T^x_{emh} \le T \le T^x_{emt} \\ S^x_m.\{\dfrac{(E-E^x_{uh})}{(E^x_{emh}-E^x_{eh})}\}^{\alpha^x_{eh}}, \quad T^x_{eh} < T < T^x_{emh} \\ S^x_m.\left\{\dfrac{(E^x_{ut}-E)}{E^x_{et}-E^x_{emt})}\right\}, \alpha^x_t \quad T^x_{emt} < T < T^x_{et} \end{cases}$$

25. // Energy efficiency for Provider

26. $$S^P_E(E)=\begin{cases} s^E_{min}+(1-s^E_{min}).[1-(\dfrac{f^P_E(E)-1}{N^P_E-1})] \; FT_P \le T \le LT_P \\ 0, \quad otherwise \end{cases}$$

27. Aggregate the price and QoS utility function

28. $$U_{total}=\begin{cases} 0 \quad if \; S_P(P) or S_Q(Q)=0 \\ w_p.S_P(P)+w_Q.S_Q(Q) \quad otherwise \end{cases}$$

29. $\Delta S_{tot}=S^t_{tot}.(\frac{t}{\tau})^{\lambda}$ // t = concession round, τ = concession deadline, λ = concession strategy

31. $S^{t+1}_{tot}=S^t_{tot}-S_{tot}$

30. Obtain the cloud service with price and QoS concession between the customer and provider.

4. Performance Evaluation

In the performance analysis, A Novel Customer Hints Scheduling (ANCHS) method and A Novel Customer Hints Scheduling with Price and QoS Concession (ANCHSPQC) method. In the novel customer hints scheduling method, user given their constraints and these constraints are combined with VM specifications to produce deployment profile. The score value is calculated for deployment profile and based on this value the optimal profile is selected. In the novel Customer Hints Scheduling with price and QoS

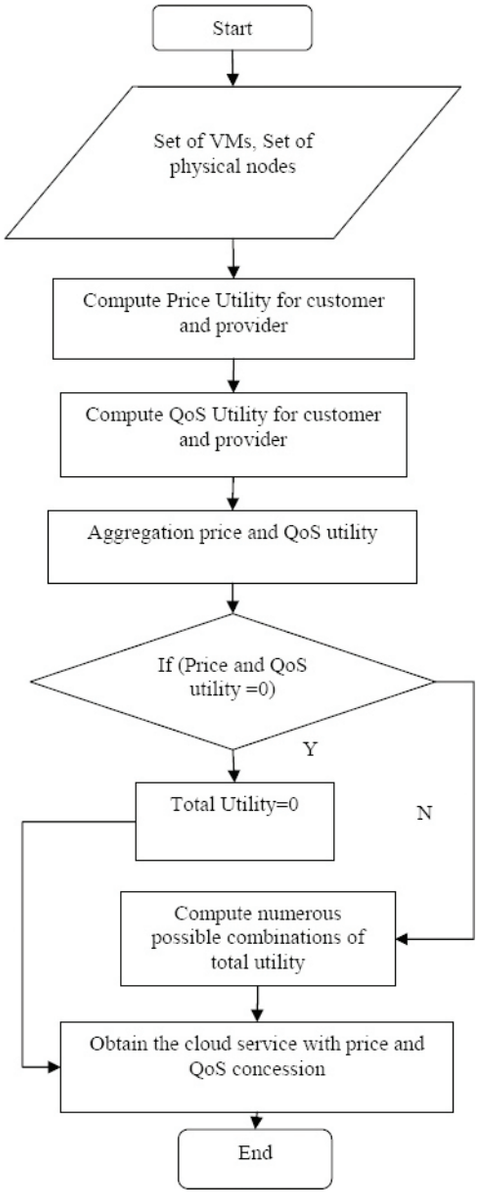

Figure 2. Flow chart for optimization method for reserving cloud services with price and QoS concession.

concession method, the price and QoS concession technique is used which ensures both providers and customers specify their preferences for price and QoS metrics. So, by using this method the trade-off between price and QoS concession is achieved by controlling the number of simultaneous proposals to reduce the computational complexity.

4.1 User Satisfaction Transmission capacity

Figure 3 shows that the user satisfaction level. In the X-axis the cloud load is taken. In the Y-axis User satisfaction

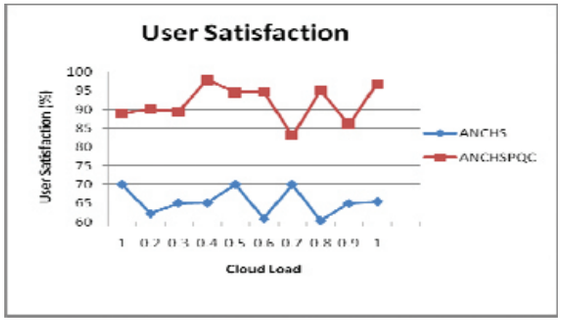

Figure 3. User Satisfaction.

is taken. The user satisfaction is defined as the degree of satisfaction provided by the cloud services. This method clearly shows that when the cloud load increases, the user satisfaction level is increases in the proposed ANCHSPQC method when compared to the existing ANCHS method.

4.2 Total Utility

Figure 4 shows that the total utility. In the X-axis the cloud load is taken. In the Y-axis total utility is taken. The total utility is defined as overall amount of satisfaction of a particular cloud service. This method clearly shows that when the cloud load increases, the total utility is increases in the proposed ANCHSPQC method when compared to the existing ANCHS method.

4.3 Response Time

Figure 5 shows that the response time. In the X-axis the cloud load is taken. In the Y-axis response time is taken. The response time is defined as the how fast the user request is processed. The average response time is taken in milliseconds. This method clearly shows that when the cloud load increases, the response time is decreases in the

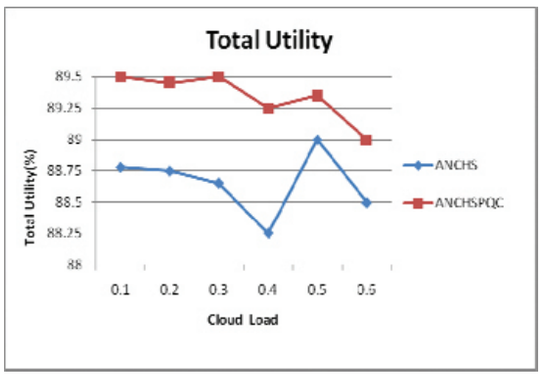

Figure 4 Total Utility.

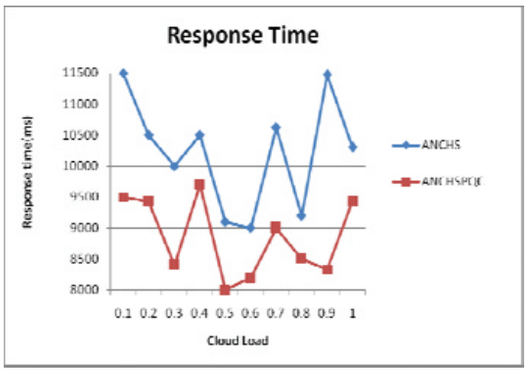

Figure 5. Response time.

proposed ANCHSPQC method when compared to the existing ANCHS method.

5. Conclusion

A new technique is proposed which is called Optimal Selection of Cloud Service with price for QoS Concession (OSCSPQC). This novel technique considers the high-level constraints such as Accountability, Agility, Assurance of Service, Security and Privacy, and Usability. Additionally, the price and time slot concession are also preferred for both providers and customers do the following: 1. Customers denote their preferences for price and QoS requirements and 2. Search for mutually acceptable prices and QoS metrics. But there is a little QoS concession between consumers and providers. So, the optimization algorithm is used to tradeoff between price and QoS concession by controlling the number of simultaneous proposals to reduce the computational complexity.

For future work, a cloud adaptive dynamic scaling mechanism is considered which could robotically scaling up and scaling down cloud infrastructures to accomplish changing workload based on performance of application level metric.

6. Reference

1. Tsakalozos K, Roussopoulos M, Delis A. Hint-based execution of workloads in clouds with Nefeli. IEEE Transactions on Parallel and Distributed Systems. 2013 Jun; 24(7):74–85.
2. Van HN, Tran FD, Menaud J-M. Autonomic virtual resource management for service hosting platforms. Proceedings of ICSE Workshop Software Engineering Challenges of Cloud Computing; 2009. p. 1–8.
3. Jayasinghe D, Pu C, Eilam T, Steiner M, Whalley I, Snible E. Improving performance and availability of services hosted on iaas clouds with structural constraint-aware virtual machine placement. Proceedings of IEEE International Conference Services Computing (SCC); 2011 Jul. p. 72–9.
4. Tesauro G, Kephart JO. Utility functions in autonomic systems. Proceedings of First International Conference on Autonomic Computing; 2004. p. 70–7.
5. Verma A, Ahuja P, Neogi A. pMapper: Power and migration cost aware application placement in virtualized systems, Proceedings on 9th ACM/IFIP/USENIX International Conference Middleware; 2008. p. 243–64.
6. Sotomayor B, Keahey K, Foster I. Combining batch execution and leasing using virtual machines. Proceedings of 17th International Symposium. High Performance Distributed Computing; 2008. p. 87–96.
7. Schad J, Dittrich J, Quiane-Ruiz J. Runtime measurements in the cloud: observing, analyzing, and reducing variance. Proceedings of the VLDB Endowment. 2010; 3:460–71.
8. Iosup A, Yigitbasi N, Epema D. On the performance variability of production cloud services. CA, USA: Proceedings of IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing. CA, USA; 2011 May. p. 104–13.
9. Iosup A, Ostermann S, Yigitbasi N, Prodan R, Fahringer T, Epema D. Performance analysis of cloud computing services for many-tasks scientific computing. IEEE Transactions on Parallel and Distributed Systems; 2011 Jun. p. 931–45.
10. Kossmann D, Kraska T, Loesing S. An evaluation of alternative architectures for transaction processing in the cloud. Proceedings of the 2010 International Conference on Management of Data, ACM; 2010. p. 579–90.
11. Foster I, Roy A, Sander V. A quality of service architecture that combines resource reservation and application adaptation. Proceeding of 8th International Workshop on Quality Service; 2000. p. 181–8.
12. Souvik A Pal B, Pattnaik PK. Classification of virtualization environment for cloud computing. Indian Journal of Science and Technology. 2013; 6(1):127–33.