Cloud Service ranking and Selection using Linear Programming

J. Preethi
Department of Computer Science and Engineering
SSN College of Engineering
Chennai, India

N. Sujaudeen
Department of Computer Science and Engineering
SSN College of Engineering
Chennai, India

T.T. Mirnalinee
Department of Computer Science and Engineering
SSN College of Engineering
Chennai, India

P. Venugopal
Department of Mathematics
SSN College of Engineering
Chennai, India

ABSTRACT

Cloud computing is basically an internet based computing, whereby shared configurable resources are provided to cloud service consumers as services on demand. As an increasing growth of cloud computing, many enterprises provide different cloud services to cloud service consumers. From cloud service consumer's perspective, it is difficult to choose an appropriate cloud service that satisfies their QoS requirements. As requirements of one cloud service consumer will vary from another, dynamic ranking has to be used to satisfy the requirements of different cloud service consumers. A simple model is needed to address the dynamic ranking of cloud services. The dynamic ranking and selection of cloud services is solved using Linear Programming(LP) model. This project considers quantifiable attributes such as processor speed, cost, etc. and some non-quantifiable attributes to rank various cloud services according to the requirements of cloud service consumer using Linear Programming(LP) technique.

Keywords

Cloud computing, QoS, Cloud Service Ranking, Service Selection, Linear Programming.

1. INTRODUCTION

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with service provider interaction[1]. Cloud Service Ranking is needed to cloud service consumers to choose appropriate cloud service from a pool of available cloud services. The QoS parameters such as response time, availability, security etc. are used to rank the cloud services based upon consumer’s requirements. The requirements will vary from one consumer to another consumer. The dynamic ranking for different consumers is needed. To achieve dynamic ranking and selection of cloud services, Linear Programming model is used. Linear Programming(LP) is a method to achieve the best outcome in a mathematical model whose requirements are represented by linear relationships. LP has many types of solutions. Given a system of m linear equations with n variables (m<n), the solution obtained by setting n-m variables equal to zero and solving for the remaining m variables is called as a basic solution. A solution (set of values for the decision variables) for which all of the constraints in the solver model are satisfied is called a feasible solution. A basic solution in which all the basic variables are non-negative is called a basic feasible solution. Any feasible solution which optimizes the objective function to reach its maximum (or minimum) value is called its optimal solution. If the values of the objective function can be increased or decreased then they are called as an unbounded solution.

The rest of the paper is organized as follows. Section 2 describes the related work. Section 3 deals with the system description of the proposed system. Experimental results are discussed in Section 4 and Section 5 concludes the paper.

2. RELATED WORK

RajkumarBuyya et al. [2] proposed a framework that measures the quality and prioritize cloud services. It computes few quantifiable QoS attributes proposed by Cloud Measurement Index Consortium (CSMIC) to rank the cloud services. Analytic Hierarchy Process (AHP) technique is used for ranking. There are three phases in this process. They are problem decomposition, judgment of priorities and aggregation of these priorities. In the first phase, the ranking of a complex problem is modeled in a hierarchy structure that specifies the interrelation among three kinds of elements, including the overall goal, QoS attributes and their sub attributes and alternative services. The second phase consists of a pairwise comparison of QoS attributes is done to specify their relative priorities and a pairwise comparison of cloud services based on their QoS attributes to compute their local ranks. In final phase, for each alternative service, the relative local ranks of all criteria are aggregated to generate the global ranking values for all the services. Though it provides a uniform way to evaluate the relative ranking of Cloud services for each type of QoS attribute, non-quantifiable QoS attributes were not considered for ranking. So, it is not compatible with various QoS attributes.

Z. Zheng et al. [3] proposed component ranking by taking the advantages of service candidates. CloudRank framework ranks the component using past component usage experiences of other different component users. This paper solves the major challenge of QoS-driven Cloud service quality ranking. Hence the locations of users are different, the Cloud service quality ranking of a user could not be redirected to another user. So, personalized QoS Ranking for cloud services is proposed to evaluate all the Cloud services at the user-side and rank the Cloud services based on the observed QoS performance. Hence the Greedy algorithm treats the explicitly rated items and the unrated items equally[8], it is not compatible with large invocations of service users.

X. Liu et al. [4] proposed generic QoS framework for cloud workflow systems that covers the major stages of a workflow lifecycle. The framework has components. They are QoS requirement specification, QoS-aware service selection, QoS consistency monitoring and QoS violation handling [2]. The software services which have higher quality then the QoS constraints can be selected by service selection component. The third component is QoS consistency monitoring at the execution stage. Because of dynamic nature of cloud, workflow execution states need to be kept under constant monitoring and QoS verification. The monitoring component detects QoS violation and some recovery actions should be taken to handle consistency. Though it covers all stages of Qos life cycle, QoS metrics are not identified and no mechanism to differentiate consumers based on requirements.

Jianmin Wang et al. [5] proposed a personalized QoS ranking prediction framework for cloud services, which requires no additional service invocations when making QoS ranking was proposed. By taking advantage of the past usage experiences of other users, the ranking approach identifies and aggregates the preferences between pairs of services to produce a ranking of services. By calculating similarity values between the current active user with other training users, the similar users can be identified. Dissimilar users are excluded as they will greatly influence the prediction accuracy. Two ranking prediction algorithms for computing the service ranking based on the cloud application designer's preferences are proposed. In CloudRank1, comparing the QoS values the preference between these two services can be calculated. The differences between preference values may be treated equally that hurts QoS ranking prediction accuracy. By considering the confidence values of different preference values CloudRank2 was proposed. Similar users whose have higher similarities will get higher confidence values. However CloudRank algorithms have provided the same quality of cloud services and this approach evaluates all the service candidates equally and Optimal VM allocation for each service need to be used[9].

TejasChauhan et al. [6] proposed the work addresses the issue of matching SLA parameters to find suitable cloud provider for particular application. Because of dynamic nature of cloud, the matching of SLA templates need to be dynamic and continuous monitoring of Quality of Service (QoS) is necessary to enforce SLAs. SLA template contains many parameters like cloud's resources (physical memory, main memory, processor speed etc.) and properties (availability, response time etc.). The first step is to define cloud model that contains cloud resources, properties, resource quantity etc. and requirement model that contains application's required resources and its quantity. The next step is to convert these models to graph structure. Next step is to find Pairwise Connectivity Graph. Final step is to find initial mapping between two models. This initial mapping is used in similarity flooding algorithm. Find the fix point that matches Qos of consumer's application and SLA of provider. It only matches service providers published capability with consumer's requirements. Thus it could not be able to track the change in cloud performance because of dynamic nature of cloud.

Maryam Solhi Lord et al. [7] proposed a linear programming technique is used for solving optimization problem. An optimization problem is a decision problem in which we are choosing among several decisions. To arrive at the solution, first we need to identify among the available decisions those that are feasible, and then choose the best one among the feasible alternatives. The objective was a linear function and all constraints were written as linear equations or inequalities. It is a flexible modeling tool to represent a lot of performance measures and restrictions on decisions. It is easy to understand and solve.

Dimitris Souravlias and Konstantinos E. Parsopoulos [8] proposed a neighborhood ranking for allocating the computational budget to the particles. The proposed approach allocates more function evaluations to particles with neighborhoods that increases complexity.

3. PROPOSED FRAMEWORK

The proposed architecture has three components. They are Cloud Service Consumers, Cloud Broker and Cloud Service Providers. Various Cloud Service Providers provide similar services to cloud service consumers. so, it is difficult to choose the appropriate cloud service that matches with cloud service consumer's requirements from their perspective. Dynamic cloud service ranking and selection helps cloud service consumers to select the particular services according to their requirements.

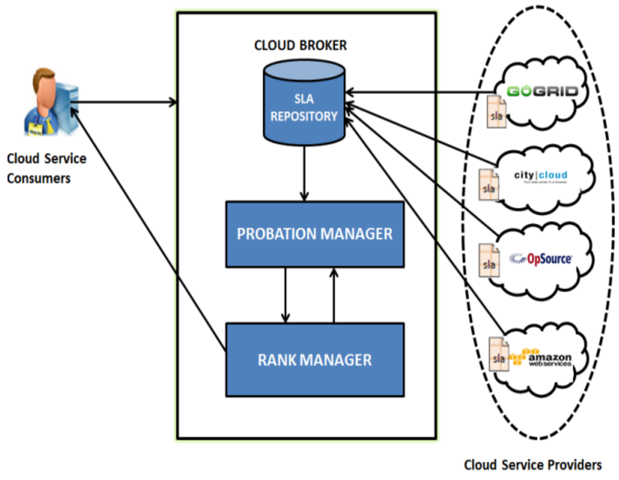

Fig. 1. Cloud Service Ranking and Selection Architecture

All cloud service providers registers with cloud broker using their SLA's. The Cloud Broker has SLA repository, probation manager and rank manager. The SLA's of cloud service providers are stored in the SLA Repository of cloud broker. when new service provider is added to the providers pool, then the SLA of new service is stored in the SLA repository. The Probation Manager takes SLA of new service from SLA repository and checks the parameters of SLA during probation period. After the probation period, the probation manager informs the rank manager with updated parameters. The Rank Manager has rank list that contains ranking of cloud services according to the SLA parameters. The rank manager updates the rank list with SLA parameters given by probation manager. If a service is identified that it is no longer used by consumers, then the rank manager gives the service to probation manager for validation. After validation of the service, the rank manager updates the rank list by adding validated service given by probation manager. when cloud service consumers request a service from cloud broker with priority of different QoS parameters, the cloud broker modifies the rank list according to each cloud service consumer's requirements and provides an appropriate ranking and selection of cloud services to cloud service consumers.

3.1 Build SLA Repository

The SLA from various cloud service providers for different cloud services are collected. The SLA document consists of the quantifiable and non-quantifiable QoS parameters which includes service name, cloud provider, security, availability, processor speed, processor cores, cost per hour, cost monthly, ram, storage, service credit, bandwidth, performance, etc. The SLA parameters from various cloud service providers are collected and stored in the SLA repository. The SLAs are maintained using database. The SLA of cloud service providers are given to cloud broker.

3.2 Design Cloud Broker

The SLA of cloud service providers are collected by cloud broker. It has two entities. They are Probation Manager and Rank Manager.

3.2.1 Probation Manager

The probation manager checks the SLA parameters and populates the list with SLA. It updates the SLA list whenever a new service enters. After getting SLA parameters, it pushes the parameters to rank list which is maintained by rank manager.

3.2.2 Rank Manager

The Rank Manager ranks the cloud services using their SLA parameters. If any service is found to be an outdated service, it gives the corresponding service SLA parameters to probation manager. Probation manager will check the SLA parameters and updated SLA parameters are replaced into the rank list. The rank list is also updated after insertion or updation of new service. when a cloud service that does not satisfies any of cloud service consumer's requirements, is identified and removed from rank list. After removing the particular cloud service, the rank list is updated.

3.3 Integrate Cloud Service Consumer's requirements with Cloud Broker

The cloud service consumer gives their requirements to broker after analyzing the available QoS parameters of cloud services maintained by cloud broker. Based on the order of requirements given by consumer, the weight is assigned to the requirements. The highest ordered requirement parameter will get the highest weight which is equal to the number of requirements entered by consumer. The next to highest will get the weight less than the highest. The weight for non-specified parameter by cloud service consumer is 0. The requirements and their corresponding weights is stored in the repository for further rank evaluation.

3.4 Cloud service ranking and selection using LP Model

The rank manager uses an objective function as a linear equation. Cloud service consumer specified constraints (linear equations) is used to maximize the objective function by Linear Programming. The services with SLA parameters that maximizes the objective function are given with higher rank and the highest service or top-N services in the rank list will be given to the cloud service consumer.

3.4.1 Mapping LP with proposed System

Let $S_1, S_2, \ldots, S_k$ be the k service providers and $p_1, p_2, \ldots, p_n$ be the n parameters offered by all the service providers. Let $v_{i1}, v_{i2}, \ldots, v_{in}$, $1 \le i \le k$, be the values of n parameters.Suppose the consumer chooses m service parameters based on his priority from the given n parameters $\le n$. Then these $m$parameters are assigned weights $(w_j : 1 \le j \le n)$as follows:

The weight of the parameters in the order of priority is $m+1, m, m-1, \ldots 4,3,2,1$ and the remaining $n-m$ parameters will be assigned 0 as all the parameters should be considered. Let us calculate the weighted sum$\sum_{j=1}^{n} v_{ij} w_j$ for each service provider S_i,$1 \le i \le k$ .This weighted sum signifies the preference of the consumer. The consumer is allocated the provider with maximum weighted sum. As the consumer should be given a best provider, the decision variables are S_1, S_2,…,S_k. This problem can be formulated as linear programming problem as follows.

The objective function is

Maximize (S_i), $1 \le i \le k$

Subject to the constraints

$S_i = \sum_{j=1}^{n} v_{ij} w_j$, $1 \le i \le k$

No. of parameters chosen by consumer, $m \le n$

Service name given by consumer$S_n \in S_i$, $1 \le i \le k$

The services with SLA parameters that maximizes the objective function are given with higher rank. The rank manager uses the weights according to cloud service consumer's interest and performs multiplication of values of parameters and their corresponding weight. Then the weighted sum of for each cloud service is calculated. The cloud services that satisfies the objective function are feasible services for cloud service consumer. The service that maximizes the weighted sum which satisfies the above objective function and satisfies all constraints of consumer is selected and given to cloud service consumers using linear programming model.

4. EXPERIMENTAL RESULTS

CloudSim is an extensible simulation toolkit that enables modeling and simulation of Cloud computing systems and application provisioning environments. The CloudSim toolkit supports both system and behavior modeling of Cloud system components such as data centers, virtual machines (VMs) and resource provisioning policies. CloudSim is used to rank various cloud services. The SLA of almost 30 cloud service providers were collected and stored in SLA repository of cloud broker. The SLA document of each cloud service contains quantifiable QoS parameters such as security, processor speed, processor cores, cost per hour, cost monthly, ram, storage, service credit and non-quantifiable QoS parameters which includes availability, performance and response time. The cloud service consumers give their requirements to broker, after analyzing the available QoS parameters of cloud services maintained by cloud broker. Based on the order of requirements given by cloud service consumer, the weight is assigned to the requirements. Assuming that the parameters chosen first will have higher precedence and will get the highest weight which is equal to the number of requirements entered by consumer. The following next will get the weight less than the highest. The weight for non-specified parameter by cloud service consumer is 0. The requirements and their corresponding weights is stored in the repository for further rank evaluation. The services with SLA parameters that maximizes the objective function are given with higher rank. The rank manager uses the weights according to cloud service consumer's interest and performs product of values of the parameters with their corresponding weight. Then the weighted sum of for each cloud service is calculated. The cloud service that satisfies the objective function are feasible services for consumer. The service that maximizes the weighted sum which satisfies the above objective function with set of all constraints of consumer is selected and given to cloud service consumers using linear programming model. The service is selected according to the requirements selected by cloud service consumers as follows

Table 1. Selection of cloud service when service type is not given

Requirements	Selected Cloud Provider
Security Processor core Cost monthly	Microsoft
Performance Service credit Availability Processor core	Joyent
Security Availability Processor core Cost monthly Performance Response time Service credit	Microsoft

The cloud service type is not given by cloud service consumers and requirements with order selected are security, processor core and cost monthly then the chosen cloud service is storage whose cloud provider is Microsoft. Our system allocates weight with security to have higher precedence followed by other parameters. The second iteration considered requirements with quantifiable QoS parameters such as service credit and processor cores and non-quantifiable QoS parameters as performance and availability, then the chosen cloud service is storage whose cloud provider is Joyent. The last iteration considered requirements with four quantifiable QoS parameters such as security, processor core, cost monthly and service credit and three non-quantifiable QoS parameters availability, performance and response time, then the chosen cloud service is storage whose cloud provider is Microsoft. Initially, only quantifiable QoS parameters were considered in the requirements and the selected service was storage whose cloud provider was Microsoft. The selection of cloud service is same in both cases but selection of cloud provider is different. Thus including non-quantifiable QoS parameters has moderate impact with the selection of cloud service and cloud provider when service type is not given.

The cloud service type is given by cloud service consumers as network then the chosen cloud service provider is Opsource. When cloud service type is given by cloud service consumers as storage then the chosen cloud provider is Hp. When the cloud service type is given by cloud service consumers as RDS then the chosen cloud provider is Amazon. In three cases, the selection of cloud service and cloud provider is different. Thus including non-quantifiable QoS parameters has huge impact in the selection of cloud service and cloud provider.

Table 2. Selection of cloud service when service type is given

Requirements	Selected Cloud Provider
Security Processor core Cost monthly	Opsource
Performance Service credit Availability Processor core	Hp
Security Availability Processor core Cost monthly Performance Response time Service credit	Amazon

When different set of QoS parameters were chosen, different cloud service was selected according to the number of QoS parameters and order of QoS parameters chosen by cloud service consumers using Linear Programming model.

5. CONCLUSION AND FUTURE WORK

The goal of the project is to propose the dynamic ranking and selection of cloud services considering both quantifiable and non-quantifiable QoS parameters to provide an appropriate service that satisfies almost all requirements of cloud service consumers using Linear Programming(LP). The proposed system has a middleware, a cloud broker that simplifies the ranking and selection of cloud services. It contains several components that ease the process by performing both offline (static) ranking which is done at the back ground and online (dynamic) ranking when cloud service consumer specifies the requirements. This improves the performance of the system as the offline job is separated from that of the online activity. Also we tried with several combinations of specifying quantifiable and non-quantifiable parameters and found that the latter has moderate impact in the cloud service selection as the former has high influence when the service type is not given. Also we run a series of iterations altering both the parameters and found that the non-quantifiable parameters also have huge impact in cloud service selection. No direct mapping was done, as cloud broker was used as mediator between cloud service consumers and cloud service providers. Prediction of cloud services was not performed. Time complexity of linear programming is high when number of services increases. Autonomic updation of SLAs with the broker can be considered. Also, we intend to try on other optimization techniques to improve the accuracy of cloud service selection.

6. REFERENCES

[1] P. Mell and T. Grance, “The NIST definition of cloud computing,” National Institute of Standards and Technology, Information Technology Laboratory, Technical Report Version 15, 2009.

[2] J. Saurabh Kumar Garg, Steve Versteeg and RajkumarBuyya, “A framework for ranking of Cloud computing services”, Future Generation Computer Systems 29 , 2013.

[3] Z. Zheng, Y. Zhang, and M.R. Lyu,“CloudRank: A QoS-Driven Component Ranking Framework for Cloud Computing”, in Proc. Int’l Symp. Reliable Distributed Systems (SRDS ’10), pp. 184-193, 2010.

[4] X. Liu, Y. Yang, D. Yuan, G. Zhang , W. Li and D.Cao, “A Generic QoS Framework for Cloud Workflow Systems”, Proc. Ninth IEEE Int. Conf. Dependable, Autonomic and Secure Computing, pp.713-720 , 2011.

[5] Z. Zheng, X. Wu, Y. Zhang, M. R. Lyu, and J. Wang, “QoS ranking prediction for cloud services”, IEEE Transactions on Parallel and Distributed Systems, vol. 99, no. Preprints, p. 1, 2012.

[6] T. Chauhan, S. Chaudhary, V. Kumar, and M. Bhise, “Service Level Agreement Parameter Matching in Cloud Computing”, World Cong. ICT , pp. 564-570, 2011.

[7] Maryam Solhi Lord and Samira MohebbiBazardeh, “Linear Programming and Optimizing the Resources”, Institute of Interdisciplinary Business Research, 2013.

[8] DimitrisSouravlias and Konstantinos E. Parsopoulos, “Particle Swarm Optimization with Budget Allocation through Neighborhood Ranking”, Genetic and Evolutionary Computation Conference, 2013.

[9] M. Subha and M. UthayaBanu, “A Survey on QoS Ranking in Cloud Computing”, International Journal of Emerging Technology and Advanced Engineering, Volume 4, Issue 2, February 2014