Personalized QOS based Ranking Approach for Cloud Service Selection

G. Ramya
M.Tech (CNIS)
CBIT
Hyderabad

K. Radhika
Associate Professor
IT Department
CBIT

ABSTRACT

Cloud computing is an Internet-based computing model. This model enables access to resources and services on demand. Cloud computing users have applications with different Quality of Service requirements. On the other hand, there are different cloud service providers offering services with different qualitative characteristics. Determining the best cloud computing service for a specific application is a significant research problem. Ranking of cloud service providers compares different services offered by different providers based on quality of service, in order to select the most suitable cloud service provider.

QoS parameters provide valuable information for making optimal cloud service selection from a set of functionally equivalent service candidates. To obtain QoS values, real-world invocations on the service candidates are usually required. This project proposes a QoS ranking prediction framework for cloud services that eliminates delay and expenses involved in real-world service invocations. It makes use of the past service usage experiences of other users. This framework does not require any additional invocations of cloud services while making QoS ranking prediction.

The algorithm is implemented by considering both cost and benefit parameters such as Response time and throughput respectively using a database containing response time and throughput values of 300 users for 10 different cloud providers. Also, Sensitivity analysis is done by varying weights of individual QoS parameters to verify the correctness of the algorithm. It is observed from the results that the proposed cloud service selection algorithm is able to appropriately choose the best cloud service provider depending on the weights of the respective QoS parameters.

General Terms

Algorithms

Keywords

Cloud computing, Quality of Service, ranking prediction, personalization

1. INTRODUCTION

Cloud computing is Internet-based computing, where shared configurable resources such as infrastructure, plat-form, and software are provided as services to users. As there is exponential growth in revenues and with the strong promotion of leading industrial companies like Amazon, Google, Microsoft, IBM, etc., cloud computing is quickly becoming popular in recent years.

Cloud computing providers offer their services according to three fundamental models: Software as a service (SaaS), Platform as a service (PaaS), Infrastructure as a service (IaaS).In the business model using software as a service (SaaS), users are provided access to application software and databases.

In PaaS model, database platform as a service is provided by various cloud providers such as Google BigTable, Amazon SimpleDB, Fathom DB, Microsoft SDS etc. In SaaS model, document management as a service is provided as NetDocuments, Questys, DocLanding, KnowledgeTreeLive, SpringCM etc. by different providers. In Iaas model computation services provided by various providers like Amazon EC2, Sever Path GoGrid, Rackspace cloud servers, Verizon CaaS, Savvis Cloud Compute etc. Since there are number of cloud providers for a specific service, there is a need to have a decision support system that allows users to select a cloud provider from a set of equivalent cloud providers.

Cloud service providers can be classified based on functional and non-functional requirements. Functional requirements include the purpose of the cloud service that includes storage, computation, database, content management, document management, Development and Testing platform etc. Nonfunctional requirements of cloud services are described by quality-of-service (QoS) parameters. QoS parameters of cloud services provide valuable information to select optimum cloud service provider. QoS perceived by the client is affected by various factors such as system configuration, geographical location, mobility, internet connectivity of the client, etc. For the same cloud service provider, different cloud users may receive different levels of QoS and ranking of cloud service providers is not uniform for all the users. Hence, there is a need for personalized cloud service selection. The brute force approach of personalized cloud service QoS ranking evaluates all the candidate services at the user-side and rank the services based on the observed QoS values. Though this approach is simple, it is not feasible in reality, as it is costly and time consuming and involves large number of service invocations. This paper proposes a personalized QoS based ranking algorithm that selects the optimum cloud service provider based on QoS parameters experienced by similar users. Similar users are identified by using Kendall Rank Correlation Coefficient (KRCC). This algorithm is time saving, cost effective and eliminates explicit service invocations.

2. LITERATURE REVIEW

All Collaborative filtering methods are widely adopted in recommender systems [1], [2]. A memory-based approach is one type of the most widely studied collaborative filtering approaches. The most analyzed examples of memory-based collaborative filtering include user-based approaches [3], [4], item-based approaches [5], [6], [7], and their fusion [8], [7], [9], [10], [11]. User-based and item-based approaches often use the vector similarity method [12] and the PCC method [2] as the similarity computation methods. Compared with vector similarity, PCC considers the differences in the user rating style when calculating the similarity.

The rating-based collaborative filtering approaches try to predict the missing QoS values in the user-item matrix as accurately as possible. However, in the ranking-oriented scenarios, accurate missing value prediction may not lead to accuracy ranking prediction. Therefore, ranking-oriented collaborative filtering approaches are becoming more attractive. Liu and Yang [13] proposed a ranking oriented collaborative filtering approach to rank movies. Yang et al. [14] propose another ranking-oriented approach for ranking books in digital libraries. Different from these previous approaches [13], [14], this paper provides a comprehensive study of how to provide accurate QoS ranking for cloud services, which is a new and urgently-required research problem.

3. DESIGN

3.1 System Architecture

Quality-of-service can be measured at the server side or at the client side. While server-side QoS properties provide good indications of the cloud service capacities, client-side QoS properties provide more realistic measurements of the user usage experience. The commonly used client-side QoS properties include response time, throughput, failure probability, etc.

This paper mainly focuses on ranking prediction of client-side QoS properties, which likely have different values for different users (or user applications) of the same cloud service. Fig. 1 shows the system architecture of CloudRank framework, which provides personalized QoS ranking prediction for cloud services. The target users of the Cloud Rank framework are the cloud applications, which need personalized cloud service ranking for making optimal service selection. A user is called active user if he/ she is requesting ranking prediction from the CloudRank framework. A user can obtain service ranking prediction of all available cloud services from the CloudRank frame-work by providing observed QoS values of some cloud services. More accurate ranking prediction results can be achieved by providing QoS values on more cloud services, since the characteristic of the active user can be mined from the provided data.

Within the CloudRank framework, there are several modules. First, based on the user-provided QoS values, similarities between the active user and training users can be calculated. Second, based on the similarity values, a set of similar users can be identified. After that CloudRank algorithm is used to make personalized service ranking by taking advantages of the past service usage experiences of similar users. Finally, the ranking prediction results are provided to the active user.

from service-oriented applications, the usage experiences of cloud services are much easier to be obtained in the cloud environment. The cloud applications can invoke and record the client-side QoS performance of the invoked cloud services easily by using monitoring infrastructure services provided by the cloud platform. The cloud provider can collect these client-side QoS values from different cloud applications easily with approval of application owners. The framework can be used at both design time and runtime. At runtime, the cloud application may obtain new QoS values on some cloud services. By providing these values to our CloudRank server, new QoS ranking prediction can be obtained. Based on the service QoS ranking, optimal system reconfiguration can be achieved.

3.2 Statistical Methods Used in the Algorithm

P **Similarity Computation**

Ranking similarity computations compare users QoS values on the commonly invoked services. Given two QoS values on the same set of services, the Kendall Rank Correlation Coefficient (KRCC)] evaluates the degree of similarity by considering the number of inversions of service pairs which would be needed to transform one order into the other. The KRCC value of users u and v can be calculated by

$$Sim\ (u, v) = 2(C\text{-}D)/N(N\text{-}1) \tag{1}$$

where N is the number of services, C is the number of concordant pairs between two lists, D is the number of discordant pairs, and there are totally N(N-1)/2 pairs for N cloud services. Since C = N(N-1)/2-D then (1) equals to *Sim (u, v)* = 1 - 4D/(N(N-1)). Employing KRCC, the similarity between two service rankings can be calculated by

$$\text{Sim (u, v)} = 1- \frac{4\times \sum_{i,j\in I_u\cap I_v} \bar{I}((q_{u,i}-q_{u,j})(q_{v,i}-q_{v,j}))}{\left|I_u\cap I_v\right|\times(\left|I_u\cap I_v\right|-1)} \tag{2}$$

Where $I_u \cap I_v$ is the subset of cloud services commonly invoked by users u and v,

$q_{u,i}$ is the QoS value (e.g., response time, throughput, etc.) of service i observed by user u, and Ī(x)is an indicator function defined as

$$\bar{I}(x) = \begin{cases} 1 & \text{if } x < 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

From above definition, the ranking similarity between two rankings is in the interval of [-1,1], Where -1 is obtained when the order of user u is the exact reverse of user v, and 1 is obtained when order of user u is equal to the order of user v. Since KRCC compares service pairs, the intersection between two users has to be at least 2 ($\left|I_u \cap I_v\right|$ >= 2)for making similarity computation.

By calculating similarity values between the current active user with other training users, the similar users can be identified. Employing QoS values of dissimilar users will greatly influence the prediction accuracy. To address this problem, the users with negative correlations are excluded and employ only the Top-K similar users for making QoS ranking prediction. In the approach, a set of similar users S(u)is identified for the active user by

$$N(u) = \{v \mid v \in T_u, Sim(u, v)\}>0, v \neq u \} \tag{4}$$

Where T_u is a set of the Top-K similar users to the user u and Sim (u, v)> 0 excludes the dissimilar users with negative similarity values. The value of Sim (u, v) in 4 is calculated by (2).

Calculation of Preference

The ranking-oriented approaches predict the QoS ranking directly without predicting the corresponding QoS values. A user's preference on a pair of services can be modeled in the form of $\psi_k : I \times I \rightarrow IR$ where $\psi_k(i, j)$ > 0 means that quality of service i is better than service j when QoS parameter k is considered and is thus more preferable for the active user and vice versa. The value of the preference function $\psi_k(i, j)$ indicates the strength of preference for parameter k and a value of zero means that there is no preference between two services. The preference function $\psi_k(i, j)$ is anti-symmetric, i.e., $\psi_k(i, j)$ = -$\psi_k(j,i)$. $\psi_k(i,i)$ = 0 for all i ∈ I.

Given the user-observed QoS values on two cloud services, the preference between these two services can be easily derived by comparing the QoS values, where $\psi_k(i, j)$ =q_i-q_j. To obtain the preference values regarding pairs of services that have not been invoked or observed by the current user, the preference values of similar users S(u) are employed. The basic idea is that the more often the similar users in S(u observe service i as higher quality than service j, the stronger the evidence is of $\psi_k(i, j)$ > 0 for the current user. This leads to the following formula for estimating the value of the preference function $\psi_k(i, j)$, where service i and service j are not explicitly observed by the current user u

$$\psi_k(i, j) = \sum\nolimits_{v\in N(u)^{ij}} w_v\,(q_{v,i} - qj) \times w_k \tag{5}$$

where v is a similar user of the current user u, $N(u)^{ij}$ is a subset of similar users, who obtain QoS values of both services i and j, w_k is the weight assigned to QoS parameter k and w_v is a weighting factor of the similar user v, which can be calculated by

$$w_v = \frac{sim(u,v)}{\sum\nolimits_{v\in N(u)^{ij}} Sim(u,v)} \tag{6}$$

w_v makes sure that a similar user with higher similarity value has greater impact on the preference value prediction in (5).

Calculation of Confidence

The preference values can be obtained explicitly or implicitly. When the active user has QoS values on both the services i and service j, the preference value is obtained explicitly. On the other hand, the preference value is obtained implicitly when employing QoS information of similar users.

When differences in preference values are treated equally, it may compromise the QoS ranking prediction accuracy. By considering the confidence values of different preference values, the algorithm uses the following rules to calculate the confidence values:

. If the user has QoS values of these two services i and j. The confidence of the preference value is 1.

. When employing similar users for the preference value prediction, the confidence is determined by similarities of similar users as follows:

$$C(i,j) = \sum_{N(u)^{ij}} w_v\,Sim(u,v) \tag{7}$$

where v is a similar user of the current user u, $N(u)^{ij}$ is a subset of similar users, who obtain QoS values of both services i and j, w_k is the weight assigned to QoS parameter k and w_v is a weighting factor of the similar user v, which can be calculated by

$$w_v = \frac{sim(u,v)}{\sum\nolimits_{v\in N(u)^{ij}} Sim(u,v)} \tag{8}$$

4. IMPLEMENTATION

4.1 Algorithm

The algorithm used for the calculation of cloud service provider ranking is described in this section. Employed service set is the set of cloud service provider names for which the current/active user has the observed QoS values. Full service set is the set of all the cloud providers.

Algorithm: Cloud Rank

```
Input : An employed service set E, a full service set I, a preference function $\psi_k$, confidence values C, weights of parameter $w_k$
Output : a service ranking $\rho n$
F=E;
While F≠Ø do
    $t = arg \max \sum_{k=1}^{l} \psi_k \times w_k$ ;
    $\rho_e(t) = |E|-|F|+1$;
    F=F-{t};
end
foreach i∈ I do
    $\pi(i) = \sum_{j\in I}\sum_{k=1}^{l} C(i,j) \times \psi(i,j)_k \times w_k$
end
n=|I|;
while I≠Ø do
```

```
        t= $\max_{i\in I} \pi(i)$;
        $\rho n$ (t)=n-|I|+1;
        I=I-{t};
        Foreach $i \in I$ do
                $\pi(i) = \pi(i) - \sum_{k=1}^{l} C(i,j)\times\Psi(i,t)_k\times w_k$;
        end
end
while E≠Ø do
        e= arg $\min_{i\in E} \rho_e i$;
$index = \min_{i\in E} \rho n$ (i);
$\rho n$ (e)=index;
E=E-{e};
End
```

The detailed steps involved in the algorithm are explained as follows.

Step 1: Rank the employed cloud services in E based on the observed QoS values and weights of parameters. $\rho_e(t)$ stores the ranking, where t is a cloud service and the function $\rho_e(t)$ returns the corresponding order of this service. The values of $\rho_e(t)$are in the range of [1,|E|] where a smaller value indicates higher quality.

Step 2: For each service in the full service set I, calculate the sum of preference values with all other services by

$$\pi(i)=\sum_{k=1}^{l}\sum_{j\in I}\psi(i,j)\times C(i,j)\times w_k$$

. Since $\psi_k(i,i)$ =0 including $\psi_k(i,i)$ in the calculation does not influence the results. Larger $\pi(i$) value indicates more services are less preferred than i (i.e., $\psi_k(i,j)$ >0).In other words, service i should be ranked in a higher position.

Step 3 : Services are ranked from the highest position to the lowest position by picking the service t that has the maximum $\pi(t)$ value. The selected service is assigned a rank equal to n –|I|+1 so that it will be ranked above all the other remaining services in I. The ranks are in the range of [1,n], where n is the number of services and a smaller value indicates higher quality. The selected service t is then removed from I and the preference sum values $\pi(i$) of the remaining services are updated to remove the effects of the selected service t.

Step 4 : Step 3 treats the employed services in E and the non-employed service in I- E identically which may incorrectly rank the employed services. In this step, the initial service ranking ρn is updated by correcting the rankings of the employed services in E.

5. RESULTS AND ANALYSIS

The algorithm is implemented in java. The database is implemented in MySQL. Two cost and benefit parameters, response time and throughput are considered for evaluating the rank of cloud providers. The cloud service providers are given names as CS1, CS2,.., CS10. The database of previous users QoS values is populated with values of 300 different users. The QoS values given as input are shown in the table 1

Table 1 Input Values

Parameters	CS2	CS4	CS6
Response Time	2.4	2.2	9.7
Throughput	22	9.1	2.8

Response-time refers to the time duration between the user sending out a request to a service and receiving a response. Throughput represents the data transfer rate over the network. Response time values are expressed in seconds and throughput values are expressed in Kbps. The weights of the parameters, response time and throughput are 0.2 and 0.8 respectively. Then the ranking results are as shown in the table 2.

5.1. Sensitivity Analysis

Sensitivity analysis is a technique used to determine how different values of an independent variable will impact a particular dependent variable under a given set of assumptions.

Here the dependent variables for an active user are weights of response time and throughput. The sensitivity analysis includes to check whether a change in the weight of parameter reflects an appropriate change in the output ranks. If the weight of response time is increased, then ranks of the providers with better response time values should be improved.

As the values of weights of response time parameter is varied, the ranks for the cloud service providers also varies. It is depicted as shown in the table 3.

Here as the weight of response time is increased the providers who have lesser response time, would attain better ranks. The provider CS3 has good response time values as shown in the table 4. So its rank is incremented from 5 to 1 as the weight of response time is increased.

The provider CS10 has good throughput values. So it is better ranked when throughput weight is high i.e 1. But it has very high response time values, so the rank is degraded when the weight of response time is increased.

The graph corresponding to the table 3 is shown in the Fig 2 .As the weights of response time and throughput are changed, the ranks obtained by the cloud providers also changes.

The values of the preference function for each cloud provider varies with the weight. If the providers has high response time values, their preference value increases with the increase in weight of response time. It is depicted in Fig 3. Here the provider CS9 has better response time value. So its preference value increases as the weight of response time increases.

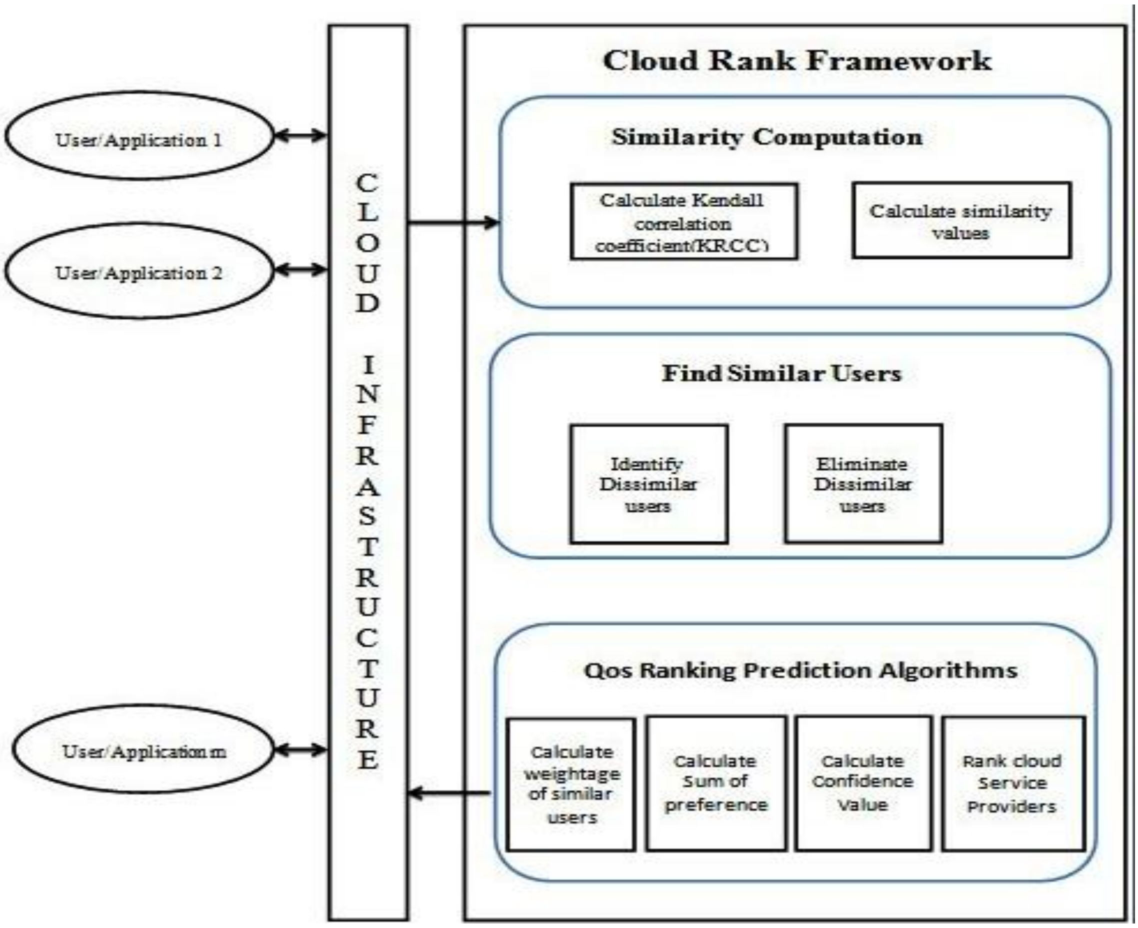
Cloud Rank Framework
Similarity Computation
Calculate Kendall correlation coefficient(KRCC)
Calculate similarity values
Find Similar Users
Identify Dissimilar users
Eliminate Dissimilar users
Qos Ranking Prediction Algorithms
Calculate weightage of similar users
Calculate Sum of preference
Calculate Confidence Value
Rank cloud Service Providers
CLOUD INFRASTRUCTURE
User/Application 1
User/Application 2
User/Application m

Fig. 1 System Architecture of Cloud Rank

Table 2 Rank Result

Rank	CSP	Value
1	CS2	24.2182
2	CS1	22.5143
3	CS3	21.23905
4	CS10	20.36104
5	CS6	20.0873
6	CS4	19.6125
7	CS9	18.8034
8	CS7	18.364
9	CS8	17.9058
10	CS5	16.8936

Table 3 Variation of Ranks with Varying Weights of Response Time

Response Time Weight	0	0.2	0.4	0.6	0.8	1
Throughput Weight	1	0.8	0.6	0.4	0.2	0
CS1	Rank 3	Rank 2	Rank 3	Rank 3	Rank 3	Rank 2
CS2	Rank 1	Rank 1	Rank 1	Rank 2	Rank 2	Rank 3
CS3	Rank 5	Rank 3	Rank 2	Rank 1	Rank 1	Rank 1
CS4	Rank 6	Rank 6	Rank 4	Rank 4	Rank 4	Rank 5
CS5	Rank 10	Rank 10	Rank 6	Rank 9	Rank 9	Rank 9
CS6	Rank 4	Rank 5	Rank 5	Rank 7	Rank 7	Rank 8
CS7	Rank 8	Rank 8	Rank 7	Rank 6	Rank 6	Rank 6
CS8	Rank 9	Rank 9	Rank 8	Rank 8	Rank 8	Rank 7
CS9	Rank 7	Rank 7	Rank 6	Rank 5	Rank 5	Rank 4
CS10	Rank 2	Rank 4	Rank 10	Rank 10	Rank 10	Rank 10

Table 4 Response Time Values

CS1	CS2	CS3	CS4	CS5	CS6	CS7	CS8	CS9	CS10
2.34	2.42	2.25	2.27	8.06	8.17	3.89	5.86	3.48	20.4
3.11	4.12	4.23	3	16.98	13.89	5.58	10.81	10.13	43.06
6.14	5.24	3.257	3.24	14.08	15.75	5.84	8.68	14.36	28.04

Table 5 Throughput Values

CS1	CS2	CS3	CS4	CS5	CS6	CS7	CS8	CS9	CS10
17.094	24.793	8.888	8.81	2.481	13.0966	5.141	5.119	5.747	25.49
12.861	14.563	4.728	6.666	1.177	7.7033	3.584	2.775	1.974	12.076
6.514	11.45	0.614	6.172	1.42	6.7936	3.424	3.456	1.392	18.544

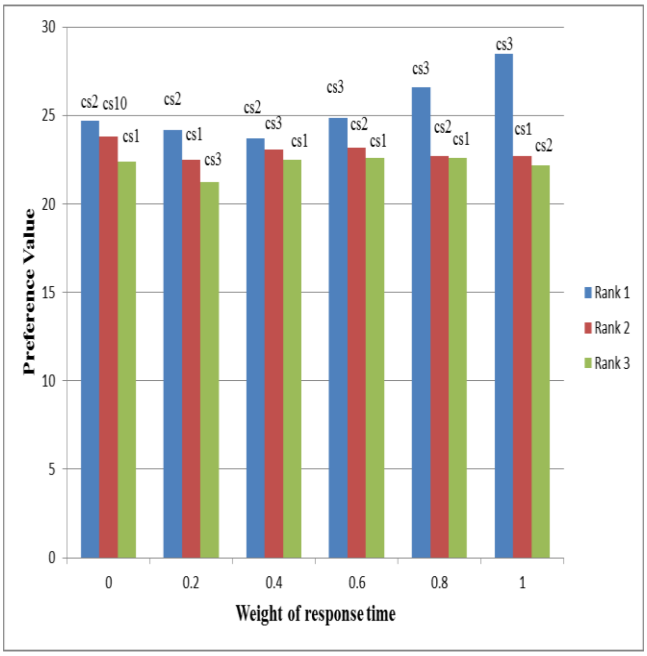

Fig 2 Rank Variation with respect to Response Time Weight

The corresponding preference values for the top three cloud providers in the graph is as shown in the table 6.

The values of the preference function for each cloud provider varies with the weight. If the providers has high response time values, their preference value increases with the increase in weight of response time. It is depicted in table 4. Here the provider CS9 has better response time value. So its preference value increases as the weight of response time increases.

Table 6 Preference Values of Top Cloud Providers

Weight of RT	Rank 1	Rank 2	Rank 3
0	24.7	23.8	22.4
0.2	24.2	22.5	21.23
0.4	23.71	23.05	22.5
0.6	24.86	23.2	22.6
0.8	26.6	22.7	22.6
1	28.48	22.68	22.19

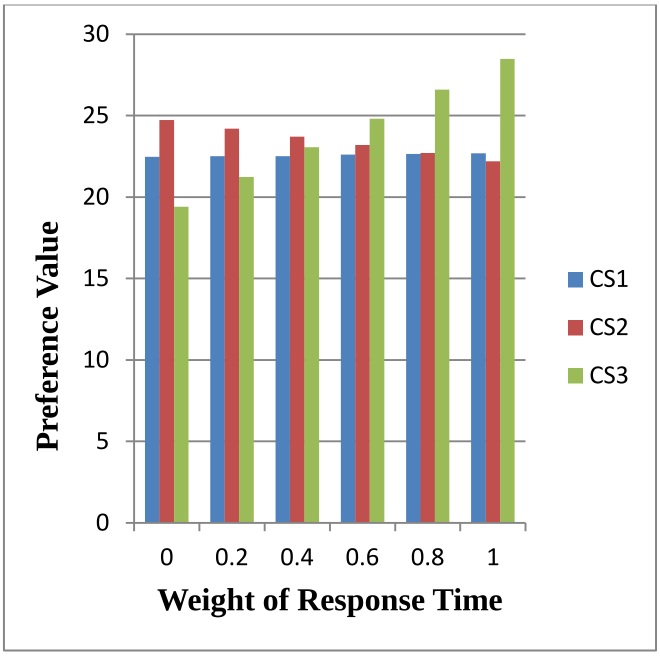

Fig 3 Variation of preference values with respect to weights of response time

The corresponding preference values of the cloud providers as depicted in the graph is as shown in the table 7.

Table 7 Preference Values For Varying Weights

Weight of Response Time	CS1	CS2	CS3
0	22.47	24.72	19.4
0.2	22.5	24.2	21.23
0.4	22.5	23.7	23.05
0.6	22.6	23.2	24.8
0.8	22.64	22.7	26.6
1	22.68	22.199	28.48

6. CONCLUSION AND FUTURE SCOPE

With the advent of increasing number of cloud service providers, personalized cloud service selection has become a crucial challenge for cloud users. This paper aims at developing a personalized QoS ranking prediction framework that considers QoS requirements of the user along with past usage experiences of other users. Also, this algorithm does not require any additional service invocations, when making QoS assisted ranking of cloud service providers. The proposed ranking approach identifies and aggregates the preferences between pairs of services. Cloud service providers are ranked by considering the weights of these parameters.

The algorithm is implemented by considering both cost and benefit parameters such as response time and throughput respectively. Also, Sensitivity analysis is done by varying weights of individual QoS parameters to verify the correctness of the algorithm. It is observed from the results that the proposed cloud service selection algorithm is able to appropriately choose the best cloud service provider depending on the weights of the respective QoS parameters. The project is implemented by considering both cost and benefit parameters.

7. REFERENCES

[1] B R. Burke, “Hybrid Recommender Systems: Survey and Experiments,” User Modeling and User-Adapted Interaction, vol. 12, no. 4, pp. 331-370, 2002.

[2] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “Grouplens: An Open Architecture for Collaborative Filtering of Netnews,” Proc. ACM Conf. Computer Supported Cooperative Work, pp. 175-186, 1994.

[3] J.S. Breese, D. Heckerman, and C. Kadie, “Empirical Analysis of Predictive Algorithms for Collaborative Filtering,” Proc. 14th Ann. Conf. Uncertainty in Artificial Intelligence (UAI ’98), pp. 43-52, 1998

[4] R. Jin, J.Y. Chai, and L. Si, “An Automatic Weighting Scheme for Collaborative Filtering,” Proc. 27th Int’l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR ’04), pp. 337-344, 2004

[5] M. Deshpande and G. Karypis, “Item-Based Top-n Recommendation,” ACM Trans. Information System, vol. 22, no. 1, pp. 143-177, 2004

[6] G. Linden, B. Smith, and J. York, “Amazon.Com Recommendations: Item-to-Item Collaborative Filtering,” IEEE Internet Computing, vol. 7, no. 1, pp. 76-80, Jan./Feb. 2003.

[7] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-Based Collaborative Filtering Recommendation Algorithms,” Proc. 10th Int’l Conf. World Wide Web (WWW ’01), pp. 285-295, 2001.

[8] H. Ma, I. King, and M.R. Lyu, “Effective Missing Data Prediction for Collaborative Filtering,” Proc. 30th Int’l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR ’07), pp. 39-46, 2007.

[9] J. Wu, L. Chen, Y. Feng, Z. Zheng, M. Zhou, and Z. Wu, “Predicting QoS for Service Selection by Neighborhood-Based Collaborative Filtering,” IEEE Trans. System, Man, and Cybernetics, Part A, to appear.

[10] Z. Zheng, H. Ma, M.R. Lyu, and I. King, “WSRec: A Collaborative Filtering Based Web Service Recommender System,” Proc. Seventh Int’l Conf. Web Services (ICWS ’09), pp. 437-444, 2009.

[11] Z. Zheng, H. Ma, M.R. Lyu, and I. King, “QoS-Aware Web Service Recommendation by Collaborative Filtering,” IEEE Trans. Service Computing, vol. 4, no. 2, pp. 140-152, Apr.-June 2011.

[12] J.S. Breese, D. Heckerman, and C. Kadie, “Empirical Analysis of Predictive Algorithms for Collaborative Filtering,” Proc. 14th Ann. Conf. Uncertainty in Artificial Intelligence (UAI ’98), pp. 43-52, 1998.

[13] N.N. Liu and Q. Yang, “Eigenrank: A Ranking-Oriented Approach to Collaborative Filtering,” Proc. 31st Int’l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR ’08), pp. 83-90, 2008.

[14] C. Yang, B. Wei, J. Wu, Y. Zhang, and L. Zhang, “Cares: A Ranking-Oriented Cadal Recommender System,” Proc. Ninth ACM/IEEE-CS Joint Conf. Digital Libraries (JCDL ’09), pp. 203-212, 2009.