Ann. Telecommun. (2010) 65:705–712
DOI 10.1007/s12243-010-0193-z

The Representation and Computation of QoS Preference with Its Applications in Grid Computing Environments

Quan Liang · Yuanzhuo Wang

Received: 28 April 2009 / Accepted: 13 July 2010 / Published online: 23 July 2010

Abstract In Grid computing environment, quality of service (QoS) provisioning must be provided to the end users on the basis of their specific requirements. This paper proposes in a first step QoS attributes for Grid applications. In this matter, a mix of quantitative and qualitative parameters have to be considered. In the context, the analytical hierarchy process (AHP) technique [1] is a possible approach to formulate the QoS requirements of the users for Grid services. In order to apply QoS preference to actual application, we introduce a QoS function and a metric for user's satisfaction degrees. These both tools can be used as an evaluation criterion by the user. Subsequently, an algorithm of service selection considering the user's QoS preference is presented. Our empirical studies indicate that the application can reliably select the optimal service for users.

Keywords Grid computing · QoS preference · AHP · Satisfaction degree · Service selection

Foundation: National Natural Science Foundation of China (no. 60803123 and no. 60873193); Science Research Projects of Fujian University of Technology (no. GY-Z09009).

Q. Liang (✉)
Department of Computer and Information Science,
Fujian University of Technology,
Fuzhou 350108, China
e-mail: liangquanlq@gmail.com

Y. Wang
Institute of Computing Technology,
Chinese Academy of Sciences,
Beijing 100190, China

1 Introduction

It is necessary to provide reliable guarantee strategies to users according to their special QoS requirements in a Grid computing environment. At the same time, it needs to support the individual service, and reflects the "server on demand". The idea "sever on demand" can also rationalize the use of system resources, optimize the system resources' structure [2]. In the practical application, study of QoS preferences and further their algorithms of service selections are of great significance.

The definition and computational method for QoS parameters of a Grid service can reflect the difference among Grid technologies, such as Web applications, P2P, and traditional distributed applications. Joutsensaloa et al. studied weighted computation of the service price, and deduced the computation decryption of the price weight [3, 4]. Dai et al. provided a description of Grid service, and made an in-depth analysis and computation to the Grid service's reliability [5]. Liu et al. has studied the generic quality standards for Web service, and proposed the representation of QoS attributes matrix, as well as the method of QoS computation based on the matrix [6]. Amin et al. discussed the abstract model of the task decomposition and the scheduling system, and analyzed the corresponding strategies to determine QoS weight under Grid environment [7]. For Web service combination, the weighted computation of QoS attributes global optimization method has been studied in literature [8], this method may be suitable for Grid service across platforms. The service selection algorithms based on combination and optimization of QoS attributes were proposed in perspectives of the Web service and Grid service, respectively [9, 10]. The latter also gave fault-tolerant services to ensure the reliable performance of task processing. In addition, there are other

works also provided a number of effective solutions for service selection in Grid computing environment [11–15].

Although the previous works has made great contributions in QoS decryptions, QoS computation, QoS optimization and corresponding strategies for service selections, systematic researches on users' QoS preferences and service selection, methods for decryptions, and computation for QoS preferences in particular, are still very deficient, and this is exactly the goal and major contributions of the paper.

Specially, our work was carried out in a Grid computing environment and related experiments were also implemented in a Grid platform. Therefore, the corresponding research results are suitable for the Grid computing environment, but in the similar other network environment the results should also be used.

2 Representation and computation of QoS preference

In practical applications, a user has often two considerations to the service selection. On one hand, it is necessary to meet the QoS requirement while obtaining the quite good performance-to-price ratio. On the other hand, different users also pay different attentions to various QoS attributes besides the basic QoS requirement. For example, some users have a special requirement to the service price, but others possibly pay a higher attention to the service response time or credibility [16]. However, different levels of attention to QoS attributes are actually user's QoS preferences.

2.1 Representation

Suppose the candidate service set is $SC=\{S_1, S_2,\cdots, S_m\}$, the QoS attribute set of each candidacy service is $Q=\{q_1, q_2,\cdots, q_n\}$, then the information of user QoS preferences can be described with several forms as follows:

(1) Sorting relation vector (SRV): sorting QoS attributes can indicate various attention degree of the user. In $o^i=(o^i_1, o^i_2, \cdots, o^i_n)$, o^i_j stands for the user's preference for QoS attributes q_j, o^i_j indicates the order of preference of q_j in all QoS attributes, and the smaller the o^i_j, the greater the level of the user's preference.

(2) AHP judgment matrix (AJM): a comparison matrix is given in which QoS attributes can be compared with each other. Element α_{ij} may be given through the $1\sim9$ scale law [17] which proposed by Satty in AHP matrix $A=(a_{ij})_{n\times n}$, it expresses the user's preference to QoS attributes q_i and q_j. The matrix A satisfies $\alpha_{ij}>0$, $\alpha_{ii}=1$, $\alpha_{ij}\alpha_{ji}=1$, $\forall i, j\in N$.

(3) Language evaluation collection (LEC): the degree of user's QoS preference may be expressed through natural language or fuzzy language. The results, which are the language evaluation collection H, and be further used to perform relationship processing to H.

What needs to point out is that the above two forms SRV and LEC finally can be converted into the AJM form and solved based on AHP. The reason we did so because the solution technology of AHP is already very mature. In fact, other representations of QoS preferences ultimately can be converted into AHP representation (namely, AJM). For example, without loss of generality, the AHP matrix can be constructed according to the known QoS preference function (as Definition 1), and then QoS preferences can be computed based on the AHP matrix.

Definition 1 QoS preference function: $F(X)$ reflects the user's QoS preference information, it has n components, that is, $F(X)=(f(x_1), f(x_2),\cdots, f(x_n))$, they correspond to the QoS attribute preference function respectively. The binary relation "$\succ$" can be used to measure preference level of components, it indicates the meaning of "superior, pay more attention to", and so on.

The next section will explain how to convert and compute the QoS preference according to its representation. The specific description of the conversion from LEC into AJM and the processing strategy of AHP matrix will be presented. This description will also demonstrate the detailed process of the conversion and computation of the QoS preference.

2.2 Conversion and computation

We convert LEC into AJM and give a processing strategy of the AHP matrix. However, an important task is to construct the AHP judgment matrix, and reflect the user's QoS preferences through the comparison of the QoS attribute pairs.

The processing procedure of LEC is as follows. First of all, differentiate the various levels of language evaluation, then convert the level to the fuzzy figure presentation. For example, language evaluation collection H of user's QoS preferences can be divided into seven grades, namely, H is {Not Attention Completely(NAC), Not Attention Very much(NAV), Not Attention(NA), Less Attention(LA), Generally Attention(GA), Very Attention(VA), Most Attention(MA)}. Represent a language evaluation with a fuzzy figure with a trapezoidal membership function showed in Fig. 1, and then H can be converted into scaled form.

The fuzzy preference relation is necessary in order to get a fuzzy value scope of the language evaluation. So the fuzzy preference relation is defined as follows.

Definition 2 (Fuzzy preference relation) The difference of various elements in $Q=\{q_1, q_2,\cdots, q_n\}$ can be described by fuzzy preference relation $R^k \subset Q\times Q$, the corresponding membership function is $\mu_Q^k : Q \times Q \to [0,1]$. Supposes $b^k(q_i, q_j)\in R$ is the fuzzy preference relation between q_i and q_j, $q_i, q_j\in Q$, and $b^k(q_i, q_j)$ expresses the degree that q_i surpasses q_j, and has the following properties: (1) non-negativity $b^k(q_i, q_j)\geq 0$; (2) closure $b^k(q_i, q_j) + b^k(q_j, q_i) = 1$; (3) complementarity $b^k(q_i, q_i)=0.5$.

Then LEC can be converted into AJM according to Fig. 1. In this process, the conversion still has a variety of methods. We give a definition as follows (that is Definition 3). Therefore, a complementary judgment matrix can be constructed. Afterwards, it will be processed.

Definition 3 Suppose q_i and q_j are two fuzzy figures, fuzzy preference relation $b^k(q_i, q_j)$ is defined as [18]:

$$b^k(q_i, q_j) = \frac{D(q_i, q_j) + D(q_i \cap q_j, 0)}{D(q_i, q_j) + D(0, q_j)} \tag{1}$$

In this formula, $D(q_i, q_j)$ is the area that q_i controls q_j, $D(q_i, 0)$ is the area of q_i, $D(0, q_j)$ is the area of q_j, and $D(q_i \cap q_j, 0)$ is the intersection area of q_i and q_j.

According to Definition 3, we could obtain the fuzzy complementary judgment matrix $b^k = (b^k(q_i, q_j))_{n\times n}$ by this approach. The fuzzy complementary judgment matrix describes and reflects the information of user's QoS preferences. We could use the characteristic vector method (CVM) to deal with user's preference relationship.

The following gives the basic processing method to the AHP matrix. Take the reciprocal judgment matrix as an example. Assume the reciprocal judgment matrix is $H = (h_{ij})_{n\times m}$. The basic problem to process H with CVM is to calculate the largest characteristic root $\lambda_{\max}$ and the corresponding characteristic vector. The detailed steps are as follows:

(1) Standardize each column of H, $\overline{h}_{ij} = \frac{h_{ij}}{\sum_{k=1}^{n} h_{kj}}$, $i,j \in [1,n]$;

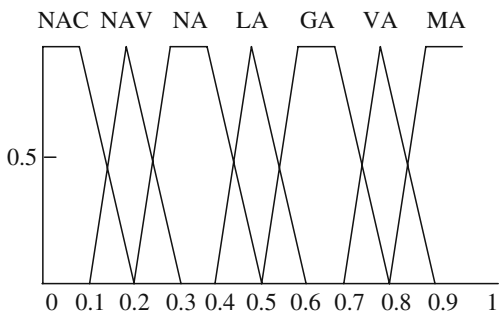


Fig. 1 The language evaluation scope of fuzzy value

(2) Add the element value of each row, $\overline{\omega} = \sum_{j=1}^{n} \overline{h}_{ij}$, $j \in [1,n]$;

(3) Standardize vector $\overline{\omega} = (\overline{\omega}_1, \overline{\omega}_2, \cdots, \overline{\omega}_n)^T$, $\omega = \frac{\overline{\omega}_i}{\sum_{j=1}^{n} \overline{\omega}_j}$, $i \in [1,n]$, the result $\omega=(\omega_1, \omega_2,\cdots, \omega_n)^T$ is exactly the characteristic vector;

(4) Calculate the largest characteristic root $\lambda_{\max}$, $\lambda_{\max} = \sum_{i=1}^{n} \frac{(H\omega)_i}{n\omega_i}$, in which $(H\omega)_i$ is the ith component of $H\omega$.

In order to ensure the credibility and accuracy, it is necessary to check the consistence. The ideal judgment matrix should satisfy the consistence condition. The consistence index (CI) is a stochastic average value, and it is generally believed that the consistence condition is satisfied as the exponent number of the matrix not more than 2. The consistence is acceptable when the consistence rate is less than 0.1, namely, $CR<0.1$, and the attribute weight vector derived from the matrix is also effective. Otherwise, the judgment matrix needs to be reconstructed. The following approximate computation is available for CI and CR of the fuzzy complementary judgment matrix.

$$\begin{cases} CI = \dfrac{\lambda_{\max} - n}{n-1} \\ CI = \dfrac{CI}{RI} \end{cases} \tag{2}$$

3 Application of QoS preference

3.1 User QoS satisfaction degree

It is necessary to give methods to evaluate service selection process which is based on the QoS preference. The evaluation should mainly consider from user's angle. QoS satisfaction degree is an important factor. We assume that QoS attributes and user expectations subject to normal distribution, which has a strong modeling capacity and continuity of the value.

Definition 4 QoS satisfaction degree: the degree of consistence between expected QoS values and the actual value, it can express with the normal distribution probability of QoS attribute q. If the expectation is q^*, the QoS satisfaction degree D follows:

$$D(q^*) = P(q^*) = p(Q \geq q^+), q^+ = \begin{cases} q, & q > q^* \\ q^* + |q^* - q|, & q \leq q^* \end{cases},$$

and

$$P(Q \geq q^+) = 1 - P(Q < q^+) = 1 - \int_{-\infty}^{q^+} f(q) d_q = 1 - \int_{-\infty}^{q^+} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(q-\mu)^2}{2\sigma}} d_q ,$$

where, μ is the expected value and σ the standard deviation.

3.2 Service selection with QoS preference

Assume that the candidacy service collection is $SC=\{S_1, S_2,\cdots, S_n\}$, QoS attribute collection $Q_i=(q_{i1}, q_{i2},\cdots, q_{im})$, and user expected QoS $Q_j^* = (q_1^*, q_2^*, \cdots, q_m^*)$. QoS preference function is $F(\omega)$. Firstly, construct the QoS matrix and calculate QoS preference vector $\overline{\omega}$. Secondly, calculate user satisfaction level $\overline{D}(S_i)$ for each service candidate through $\overline{\omega}$. Finally, select the relatively optimal service according to $\overline{D}(S_i)$. The detailed algorithm is as follows:

Algorithm 1 Service selection with QoS preference

(1) Calculate QoS preference vector $\overline{\omega}$ according to $F(\omega)$;
(2) Set i=1, k=0, D=0, start to screen SC;
(3) If i>1, then turn to step (7);
(4) Calculate $\overline{D}(S_i)$ of service S_i;
(5) If $\overline{D}(S_i) > D$, then $D = \overline{D}(S_i)$, k=i;
(6) $i = i + 1$, turn to step (3);
(7) If k=0, means that not any service is selected, turn to step (9);
(8) Service S=S_k is the optimal, return S;
(9) Exit the algorithm.

Without loss of generality, it uses $F(\omega)$ to represent QoS preferences in Algorithm *1*, rather than the three representations proposed in the paper (that is, SRV, LEC, and AJM). On the one hand, Algorithm 1 calculates the preference vectors $\overline{\omega}$ according to users' QoS preferences, and makes it as one of the basis for the service selection. Therefore, the selected services will be able to reflect user's QoS preferences. On the other hand, the algorithm uses a vector of $\overline{D}(S_i)$ to determine whether selected services can satisfy user's requirements, which is as another basis for service selection. Because QoS satisfaction degree (that is Definition 4) reflects the difference of value between the user's expectation and the actual, therefore it could determine whether to satisfy user's requirements. Algorithm 1 gives a balance between these two aspects, so the service selection on demand could be realized.

The next section implements computations of the two important vectors in Algorithm 1, which are $\overline{\omega}$ and $\overline{D}(S_i)$, they are respectively computed by Algorithm 2 and Algorithm 3.

3.3 Algorithm implementation

The computation of QoS preference vector $\overline{\omega}$ and user satisfaction degree $\overline{D}(S_i)$ is crucial, they are obtained by algorithm $\Pr ef(F(\omega), \overline{\omega})$ and algorithm $\text{Satis}(Q, \overline{\omega}, \overline{D}(S_i))$, respectively.

Algorithm 2 $\Pr ef(F(\omega), \overline{\omega})$

(1) Construct judgment matrix $H = (h_{ij})_{m\times m}$ or $B = (b_{ij})_{m\times m}$;
(2) Set k=0, calculate accumulation sum:

for j=1, 2,···, m do
{for i=1, 2,···, m do $k - k + h_{ij}$;
$\overline{h}_{ij} = h_{ij}/k$;
}

(3) Calculate matrix row-sum:

for i=1, 2,···, m do ω_i=0;
for j=1, 2,···, m do
for i=1, 2,···, m do $\omega_j = \omega_j + \overline{h}_{ij}$;

(4) Set k=0, obtain QoS preference vector:

for j=1, 2,···, m do $k = k + \omega_j$;
for j=1, 2,···, m do $\overline{\omega}_j = \omega_j/k$;
Let $\overline{\omega} = (\overline{\omega}_1, \overline{\omega}_2, \cdots, \overline{\omega}_m)$;

(5) Firstly set $\lambda_{\max}$=0, then calculate the largest characteristic vector $\lambda_{\max}$:

for i=1, 2,···, m do $(H\omega)_i = 0$;
for i=1, 2,···, m do
for j=1, 2,···, m do $(H\omega)_i = (H\omega)_i + h_{ij} \times \omega_j$;
for i=1, 2,···, m do $\lambda_{\max} = \lambda_{\max} + (H\omega)_i/(m \times \overline{\omega}_i)$;

(6) Carry on the consistence check:

If CR>0.1, then turn to step (1),else return $\overline{\omega}$ and exit the algorithm.

Algorithm 2 describes how to calculate the preference vector $\overline{\omega}$ according to information of user QoS preferences. Without loss of generality, $F(\omega)$ is still used to represent user QoS preferences (of course, it can also be any of SRV, LEC, and AJM). The algorithm summarizes the whole process of the QoS preference representation, conversion, and computation, according to $F(\omega)$. The algorithm first construct the AHP matrix and standardize it. Then the maximum eigenvector $\lambda_{\max}$ and the weight vector $\overline{\omega}$ are computed by the eigenvector method. Eventually, the consistency check is needed. The final result ω can reflect user QoS preferences, and is one of the bases of service selections.

Algorithm 3 $\text{Satis}(Q, \overline{\omega}, \overline{D}(S_i))$

1. Suppose single QoS satisfaction degree is d_{ij}, initiate d_{ij}:

for j=1, 2,···, m do d_{ij}=0;

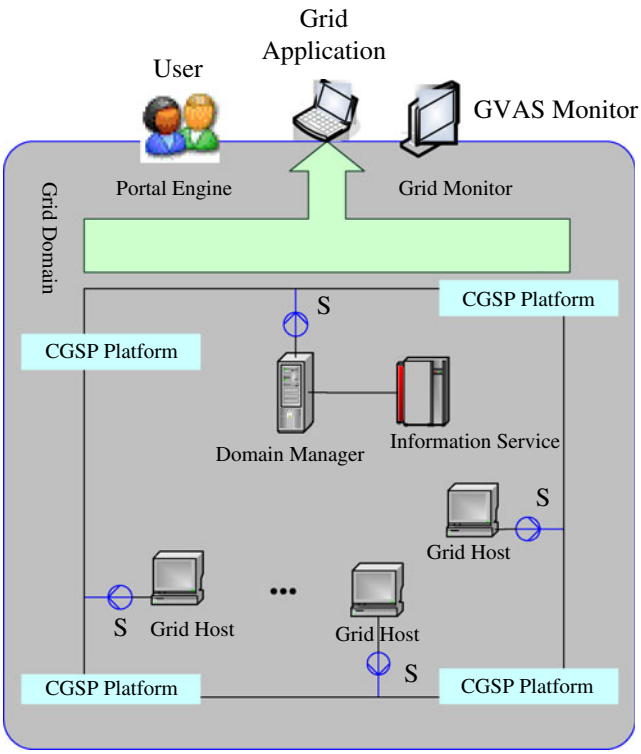


Fig. 2 Working environment of GVAS based on CGSP2

2. Calculate $D(S_i)$ of S_i according to definition 4, namely, $D(S_i)=(d_{i1}, d_{i2},\cdots, d_{im})$:

   for $j=1, 2,\cdots, m$ do $d_{ij} = 2P(q_{ij}^{*})$;

3. Calculate the comprehensive QoS satisfaction degree $\overline{D}(S_i)$ of S_i, that is $\overline{D}(S_i) = D(S_i) \times \overline{\omega}$, and to exit the algorithm with $\overline{D}(S_i)$.

Algorithm 3 describes the computing process of QoS satisfaction degree. In this algorithm, d_{ij} is a single QoS satisfaction degree, which is the satisfaction degree of each QoS attribute of candidate services computed by Definition 4. At the same time, $D(S_i)$ is a total QoS satisfaction degree of candidate services. However, $\overline{D}(S_i)$ represents a comprehensive one, which is the product of $D(S_i)$ and $\overline{\omega}$. Therefore, $\overline{D}(S_i)$ has combined with user QoS preferences and QoS satisfaction degree to be able to implement server on demand for users. Eventually, Algorithm 3 will get the vector $\overline{D}(S_i)$ in the light of results from Definition 4 and Algorithm 2.

During operations, Algorithm 2 and Algorithm 3 will be called by Algorithm 1 to implement service selections based on QoS preferences.

4 Experimental results and analysis

4.1 Experimental platform

Our laboratory designed and implemented a monitoring and analysis system of Grid activity, namely grid vision and analysis system (GVAS). We applied it to the Grid platform supported by CGSP. CGSP is a core Grid middleware developed for Chinagrid. It has consolidated various educational and scientific resources, shielded the heterogeneity and dynamicity, and provides transparent Grid services with high performance, high reliability, and security for a variety of scientific computations and engineering researches. CGSP is a server portal of Chinagrid, simultaneously also provides a set of complete development and deployment environments for a variety of Grid applications.

Chinagrid connected more than 20 universities and institutes and integrated their computational, storage, and software resources. We developed CGSP (Chinagrid Support Platform) [19]. So far, we have developed and implemented five typical Grid applications such as biological information, fluid mechanics, and so on. CGSP becomes gradually an important platform of scientific research for Chinese universities.

GVAS utilized the existing hardware, including two servers(AMD 64 optero, 2 GB memory, 200 GB hard disk) and 16 hosts (Intel Pentium 4.3 GHz, 512 MB memory), and installed parts of the CGSP2 system to constructed a Grid environment, as shown in Fig. 2.

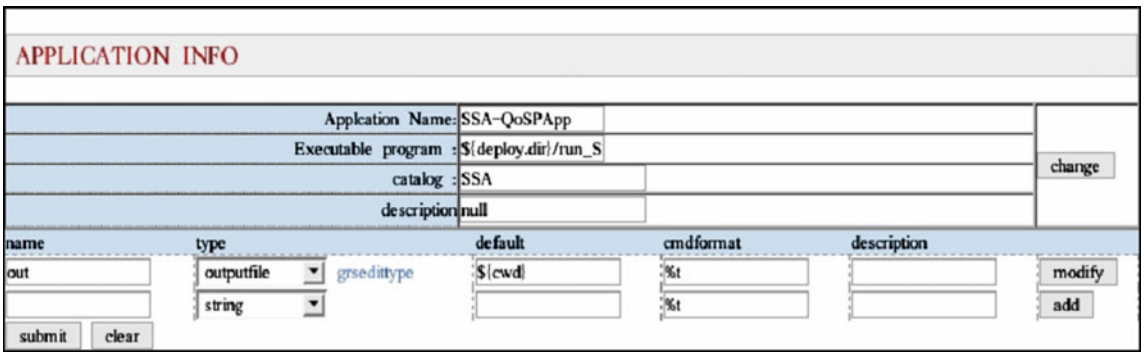
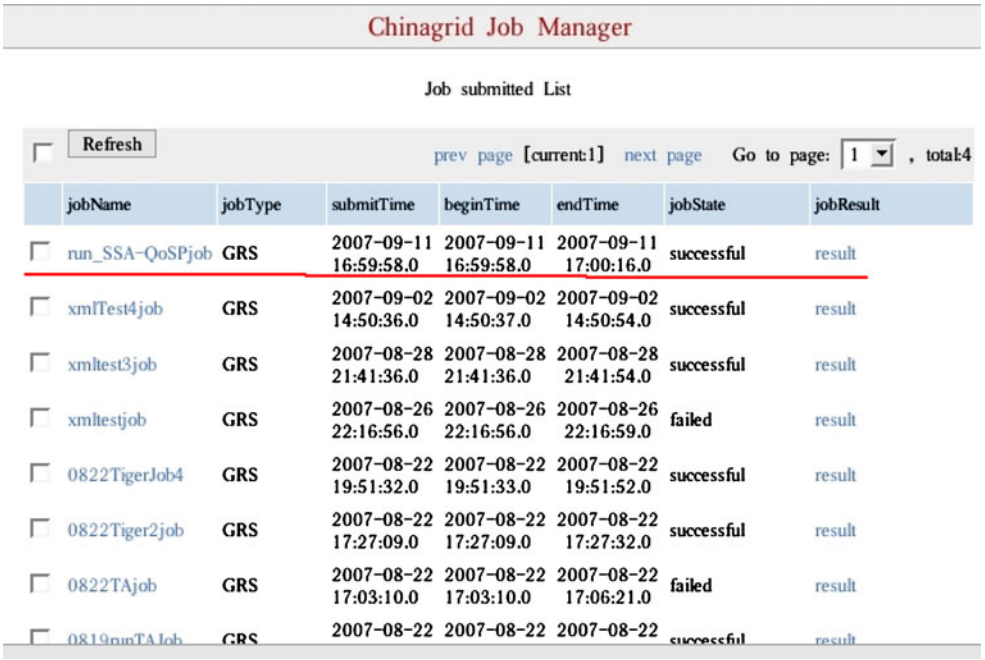


Fig. 3 Algorithm packaged as a GRS service

Fig. 4 SSA-QoSPApp running



Chinagrid Job Manager

Job submitted List

Refresh prev page [current:1] next page Go to page: 1 , total:4

jobName	jobType	submitTime	beginTime	endTime	jobState	jobResult
run_SSA-QoSPjob	GRS	2007-09-11 16:59:58.0	2007-09-11 16:59:58.0	2007-09-11 17:00:16.0	successful	result
xmlTest4job	GRS	2007-09-02 14:50:36.0	2007-09-02 14:50:37.0	2007-09-02 14:50:54.0	successful	result
xmltest3job	GRS	2007-08-28 21:41:36.0	2007-08-28 21:41:36.0	2007-08-28 21:41:54.0	successful	result
xmltestjob	GRS	2007-08-26 22:16:56.0	2007-08-26 22:16:56.0	2007-08-26 22:16:59.0	failed	result
0822TigerJob4	GRS	2007-08-22 19:51:32.0	2007-08-22 19:51:33.0	2007-08-22 19:51:52.0	successful	result
0822Tiger2job	GRS	2007-08-22 17:27:09.0	2007-08-22 17:27:09.0	2007-08-22 17:27:32.0	successful	result
0822TAjob	GRS	2007-08-22 17:03:10.0	2007-08-22 17:03:10.0	2007-08-22 17:06:21.0	failed	result
0819runTAJob	GRS	2007-08-22	2007-08-22	2007-08-22	successful	result

In order to monitor the Grid hosts, it is necessary to deploy corresponding sensors in each host to monitor hosts and collect information such as processing information. Cooperating with other monitoring tool, we can emphasize on target-oriented monitoring. System management routines can be done through Web pages. On one hand, it is convenient for Grid user to understand task allocation and the current progress, on the other hand, it can also facilitate the Grid manager obtain the resource information and the analysis result from a distance.

GVAS has the characteristics as high stability, a large number of nodes monitored, a big capacity of database and good scalability. For two servers currently, they can monitor hundreds of hosts with a database which capacity will reach 1~3T, and have a backup database to ensure the security of the data. If necessary, so long as to improve server performance or increase the number of servers, it is easy to expand scale of GVAS.

4.2 Experiment description and results analysis

Now, we use GVAS to test Algorithm 1. The experiment is divided into basic two stages. In the first stage, we package Algorithm 1 for a general running service (GRS) service, as shown in Fig. 3, and assign "SSA-QoSPApp" to "Application Name", which is the identifier of the GRS service. The executable program is associated with an executable program that the application actually refers to. When GRS service is called, it will invoke the executable program. At the same time, you can give a category in "Catalog" column, for example, it is designated as "SSA" in Fig. 3, and will make applications search expediently. In addition, you can also give an appropriate description in "Description" column for the application.

Then we deploy the GRS service to the appropriate Grid host in accordance with the experimental requirement, and run the GRS service. In Fig. 4, it shows the running state of SSA-QoSPApp as a GRS service invoked, which state indicates that the job was successfully implemented. What needs to be pointed out is that GRS services invoking do not limit to a particular node, however, it can complete through the Grid portal within or outside the territory.

In the second stage, the system gathers performance data, and analyzes corresponding results. Performance data mainly concern running time, CPU, and memory occupation. The goal of experiment is to test the stability and

Fig. 5 The monitoring data as service number changing

IP	time_start	time_end	cpu_speed	cpu_idel	cpu%	mem%	mem_cache
202.204.54.11	2007-09-20 14:36:10.377	2007-09-20 14:36:13.563	2800	49.933334	19.433659	5.68%	10.76%
202.204.54.49	2007-09-20 14:40:36.567	2007-09-20 14:40:41.358	2800	45.345636	20.434560	8.06%	11.87%
202.204.54.32	2007-09-20 14:55:19.435	2007-09-20 14:55:25.549	2800	41.567385	25.095625	15.87%	12.06%
202.204.54.77	2007-09-20 15:06:10.452	2007-09-20 15:06:19.534	2800	38.267184	26.795491	24.55%	12.79%
202.204.54.26	2007-09-20 15:25:37.571	2007-09-20 15:25:50.051	2800	36.107184	28.395491	31.32%	15.55%

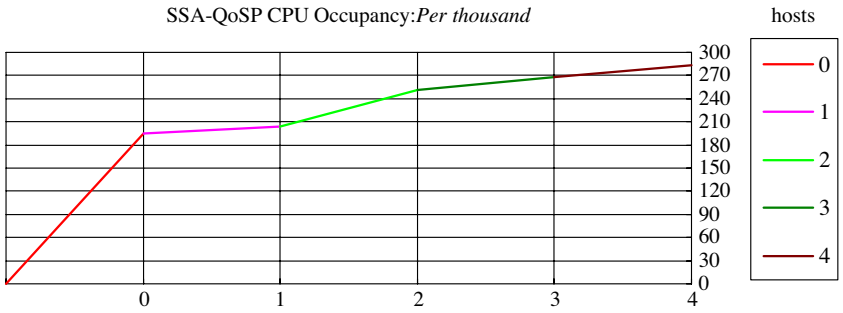


Fig. 6 The curve of CPU occupancy as service number changing

effectivity of the algorithm. In this experiment, the stability is the case when the candidate service increases, the CPU occupancy will also increase, but similar number of candidate services, no matter how to operate, the CPU occupancy is similar. The effectively is a case that the algorithm can always find a relatively good service satisfying requirements of user QoS preferences, regardless how many candidate services.

In order to reduce the complexity of programming, QoS satisfaction degree does not use probability distribution, but the expected value of relative difference, namely, $D(S_i) = \left(\sum_{j=1}^{n} \left|\frac{q_{ij}^{ref} - q_{ij}}{q_{ij}}\right|\right) \Big/ n$. The first experiment operated in five different Grid hosts, and the candidacy service number in each host is 20, 40, 80, 120, and 160, respectively. Different service numbers collect changing operating data, which will be the foundation for GVAS to analyze, as shown in Fig. 5. GVAS can analyze the data and plot curves related to running time, CPU, and memory occupation.

These curves reflects a general tendency, that is, as the service increases, the corresponding operating data also increases. We can take the curve of CPU occupancy as an example. For SSA-QoSPApp services of each node (i.e., on behalf of the GRS service of Algorithm 1), the CPU occupancy rate increases along with the candidate service number increases, as shown in Fig. 6. It is because the computation quantity of Algorithm 1 will increase as the candidate service number increased. But the CPU occupancy rate does not always increase, it will reach a stable value close to 1, which indicates the algorithm can always find a suitable service regardless of how many candidate services. The tendency had reflected that the algorithm is reliable and effective, namely, it faithfully processes the changing data according to corresponding strategies and algorithm.

The second experiment is to test stability of the algorithm. The experiment chooses four different Grid hosts to operate the algorithm repeatedly with different service number of 20, 40, 80, and 160, respectively, and obtained the 4×3 group data. Through analyzing these data, we could find that data changing is smaller in different Grid hosts on condition of the same number of services. In other words, operating data in the same group will obtain similar curve, which shows the algorithm is relatively stable.

For example, the CPU occupancy is shown in Fig. 7. Different column diagrams of three groups elevate along with increasing of candidate services, but in the same group, the difference between diagrams is very small. It is because the difference of computing quantity is not significant, after SSA-QoSApp processed the service set with a same number, which shows Algorithm 1 is stable. However, among different groups, the curve changes with a similar tendency like the first experiment. In additions, what needs to be pointed out is that, experimental results is very close to the simulation results we had accomplished, which also indicates that the algorithm is effective and stable.

5 Conclusion

This paper proposes three ways to describe the QoS preference information, SRV, AJM, and LEC, and puts forward corresponding conversion and computation methods. Subsequently, a method to determine the relative optimal service is presented based on the strategy of dealing with QoS preference information, which can reflect requirements of the user QoS preference. An algorithm of the service selection combined with satisfaction degree was designed. This algorithm can reflect the individual demand of user.

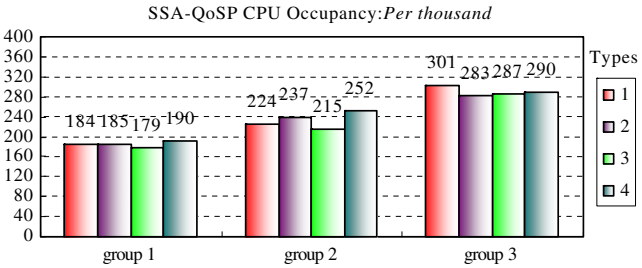


Fig. 7 The chart of CPU occupancy in case of repeatedly running

The description, conversion, and computation for user QoS preference is an important task to achieve "server on demand" and guarantee QoS. The method for description and computation, errors reduction, and computational results optimization has great influence on service selection; therefore, it is worth making more thorough research in the area.

Acknowledgment We will say thanks to our friends and classmates, they gave us many good ideas and suggestions, we cannot finish the work without their help. At the same time, thanks a lot for the support of National Natural Science Foundation of China (no. 60803123 and no. 60873193) and Science Research Projects of Fujian University of Technology (no. GY-Z09009).

References

1. Saaty TL (1980) The analytical hierarchy process. Mc Graw-Hill, New York
2. Liang Q, Yang Y, Liu L (2007) A service-oriented Grid model with quality of service provision. Inf Control 36(4):401–409
3. Joutsensaloa J, Luostarinenb K, Siltanenc J et al (2006) Adaptive scheduling method for maximizing revenue in flat pricing scenario. AEU Int J Electron Commun 60(2):159–167
4. Joutsensalo J, Viinikainen A, Wikstrom M et al (2007) Pricing based adaptive scheduling method for bandwidth allocation. AEUÉ Int J Electron Commun 61(2):118–126
5. Dai YS, Xie M, Poh KL (2006) Reliability of Grid service systems. Comput Ind Eng 50(1-2):130–147
6. Liu Y, Ngu AHH, Zeng L (2004) QoS computation and policing in dynamic Web service selection. ACM SIGecom Exchanges 5 (5):66–73
7. Amin K, von Laszewski G, Hategan M et al (2006) An abstraction model for a Grid execution framework. Journal of System Architecture 52(2):73–87
8. Zeng LZ, Benatallah B (2004) QoS-aware middleware for Web service composition. IEEE Trans Softw Eng 30(5):311–327
9. Lee H, Chung K, Chin S et al (2005) A resource management and fault tolerance services in Grid computing. Journal of Parallel and Distributed Computing 65(11):1305–1317
10. Chen YP, Li ZZh, Tang YZh (2006) A method satisfying Markov process of Web service composition under incomplete constrains. Chinese Journal of Computers 29(7):1076–1084
11. Sunjae L, Wonchul S, Dongwoo K et al (2007) A framework for supporting bottom-up ontology evolution for discovery and description of Grid services. Expert Systems with Applications 32(2):376–385
12. Mastroianni C, Talia D, Verta O (2005) A super-peer model for resource discovery services in large-scale Grids. Future Generation Computer Systems 21(8):1235–1248
13. Malhan MS, Shah RN (2006) Dynamic system activity profile forecasting for improved resource selection in quality of service based Grid computing model[C]. Proceeding of the 1st International Conference on Communication Systems Software and Middleware, IEEE press 1(2):308–312
14. Deora V, Shao J, Gray WA, et al. Supporting QoS based selection in service oriented architecture[C]. Proceeding of International Conference on Next Generation Web Services Practices, IEEE COMPUTER SOC, 2006:117–123.
15. Mei Lin, Zhangxi Lin (2006) A cost-effective critical path approach for service priority selections in Grid computing economy. Decision Support Systems 4(3):1628–1640
16. Doulamis N, Doulamis A, Litke A et al (2007) Adjusted fair scheduling and non-linear workload prediction for QoS guarantees in Grid computing[J]. Comput Commun 30 (3):499–515
17. Z. SH. Xu. The decision making method of Indefinite multi-objective and its application[M]. Beijing: Tsinghua University Publishing House, 2004.
18. Guo ChX, Guo HH (2005) Approach of multiple-attribute group decision making. Journal of Systems Engineering and Electronics 27(1):63–65
19. Chinagrid.Introduce.http://chinagrid.hust.edu.cn/index.php,Last visited:Dec,2008.