Scheduling Remote Access to Scientific Instruments in Cyberinfrastructure for Education and Research

Jie Yin[1], Junwei Cao[2,3,*], Yuexuan Wang[4], Lianchen Liu[1,3] and Cheng Wu[1,3]
[1]National CIMS Engineering and Research Center, Tsinghua University, Beijing, 100084, P. R. China
[2]Research Institute of Information Technology, Tsinghua University, Beijing, 100084, P. R. China
[3]Tsinghua National Laboratory for Information Science and Technology, Beijing, 100084, P. R. China
[4]Institute for Theoretical Computer Science, Tsinghua University, Beijing, 100084, P. R. China
*Corresponding email: jcao@tsinghua.edu.cn

Abstract

While a grid represents a computing infrastructure for cross domain sharing of computational resources, the cyberinfrastructure, proposed by the US NSF Blue – Ribbon advisory panel, is expected to revolutionizing science and engineering by including more computer integrated resources, e.g. telescopes and observatories. As a part of the China national cyberinfrastructure for education and research, resource sharing of expensive scientific instruments is discussed in this work. A layered model of instrument pools is introduced and the process from submitting a job to instrument pools to obtaining results is analyzed. Fuzzy random scheduling algorithms are proposed in instrument pools when a job is submitted to one of instruments within a pool. The randomness lies in the probability which instrument could be chosen for an experiment and the fuzziness origins from vagueness of users' feedback opinions on experimental results. Users' feedback information is utilized to improve overall quality of service (QoS) of an instrument cyberinfrastructure. Several algorithms are provided to increase utilization of instruments providing higher QoS and decrease utilization of those with poor QoS. This is demonstrated in details using quantitative simulation results included in this paper.

1. Introduction

Grid computing, originally motivated by wide-area sharing of computational resources [4], has evolved to be mainstream technologies for enabling large-scale virtual organizations [5]. Especially, entering the new century, the cyberinfrastructure vision [1], proposed by the US NSF Blue – Ribbon advisory panel, provides a blueprint of future infrastructure in cyberspace for revolutionizing science and engineering. Grid technologies are potential to be utilized for cross-domain sharing of much more computer integrated resources, e.g. telescopes and observatories [11].

Current situation in China is that on one hand some organizations have expensive scientific instruments with high maintenance costs but low utilization ratios. On the other hand, many universities cannot obtain necessary experimental facilities supporting academic experiments and activities. As a part of the China national cyberinfrastructure for education and research, remote manipulation of geographically distributed scientific instruments and cross-organization sharing of high-quality education resources using grid technologies was discussed in [16, 17].

This work focuses on scheduling remote access of scientific instruments with consideration of quality of service (QoS) issues. A layered model of instrument pools is introduced. In our previous work the role of human was not taken into account and there was no QoS feedback mechanism to reflect whether users are satisfied with experimental results. In this paper the feedback information regarding instrument QoS is considered to be a fuzzy variable with one of the following linguistic values, *terrible*, *bad*, *normal*, *good* and *excellent*. The probability whether an instrument could be chosen for a job is dynamically adjusted according to users' QoS feedback information. As a result, utilization of instruments providing higher QoS according to users' feedback is increased so that QoS of an instrument cyberinfrastructure as a whole is dramatically improved. This is quantitatively illustrated using detailed modeling and simulation results included in this paper.

Resource scheduling issues for clusters and grids has been discussed for many years. Especially, using historical QoS data to improve scheduling performance has been proved to be very effective. In a parallel and distributed computing environment, QoS data can be defined easily using quantitative values, e.g. job execution time [13], queue waiting time [14], data

transfer time [3], CPU workloads [18], which can be modeled and analyzed using performance prediction technologies [15] and utilized to improve resource scheduling performance. However, it is difficult for users to characterize instrument performance quantitatively since various criteria (e.g. time, cost and precision) may play different roles in different experiments. In general, users can only provide an overall impression of instrument QoS. The fuzzy random theory is adopted here, which is suitable and straightforward when applied to the scheduling scenarios involved in an instrument cyberinfrastructure, though not necessarily providing the best scheduling solution. A similar work using fuzzy methods for grid scheduling can be found in [2], but the exact model and algorithms are different.

The rest of this paper is organized as follows. In Section 2, a layered model of instrument pools is introduced. In Section 3, we present the fuzzy random scheduling model and algorithms with consideration of users' QoS feedback information. Simulation results are given in Section 4 and the paper concludes in Section 5.

2. Scientific Instrument Sharing

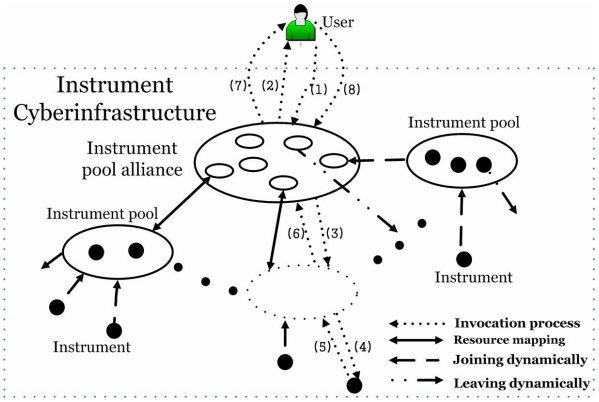

Figure 1. The process of invoking a service in an instrument cyberinfrastructure

As shown in Figure 1, in an instrument cyberinfrastructure, similar instruments are organized into an instrument pool and different instrument pools constitute an instrument pool alliance. When a user wants to do experiment via the instrument cyberinfrastructure, he submits the job to the instrument pool alliance, which analyses the job and verifies whether it can be accomplished with existing pools within it. If the job can be fulfilled, the instrument pool alliance will submit it to the required instrument pools by order of the job's inherent requirements. When an instrument pool receives a job, it will find an available instrument to do it.

Every instrument in Figure 1 belongs to a certain instrument pool and can join and leave the pool dynamically. All instrument pools have their images in the instrument pool alliance and can also join and leave the pool alliance dynamically. When a user wants to do an experiment and submits it to the instrument pool alliance in Step 1, the instrument pool alliance will check whether the instrument cyberinfrastructure has the required instruments needed to fulfill the experiment. If not all resources needed are presented, the pool alliance will reply the user with refusal information in Step 2. Otherwise the alliance will decompose the experiment into parts and submit the related parts to corresponding pools in Step 3. All the related pools will find suitable resources and submit job parts to chosen instruments in Step 4. In Step 5, chosen instruments return results of the experiment to pools after the experiment was done and the pools return results to the pool alliance in Step 6. The pool alliance composes all middle results and returns a final result to the user in Step 7. In Step 8, the user feed back his opinion about the experimental result, which is important to improve QoS of the instrument cyberinfrastructure as discussed later.

3. Fuzzy Random Scheduling

As we mentioned before, instrument QoS can be hardly described using explicit parameters. In this section, we introduce a fuzzy random theory to characterize users' feedback QoS information.

3.1 Fuzzy random theory

The fuzzy random theory is an emerging field in the uncertain theory, a branch of modern mathematics. It takes two aspects of uncertain factors, randomness and fuzziness, respectively, into account and has attracted many research interests. Some key concepts of the fuzzy random theory are given in this section. A detailed introduction can be found in [9].

A fuzzy random variable is a measurable function from a probability space to the set of fuzzy variables. In other words, a fuzzy random variable is a random variable taking fuzzy values. The notion of fuzzy random variable was first introduced by Kwakernaak in [7] and [8]. This concept was developed in [6], [10] and [12] by different requirements of measurability. Definition of fuzzy random variable is as follows [10]:

A fuzzy random variable is a function ξ from a probability space ($\boldsymbol{\Omega}$, $\boldsymbol{A}$, $\boldsymbol{Pr}$) to the set of fuzzy

variables such that $Cr\{\xi(\omega) \in \boldsymbol{B}\}$ is a measurable function of ω for any Borel set $\boldsymbol{B}$ of $\boldsymbol{R}$, the real number domain. $\boldsymbol{\Omega}$ is a nonempty set, $\boldsymbol{A}$ is an algebra over $\boldsymbol{\Omega}$ and $\boldsymbol{Pr}$ is a probability measure. Cr is the credit of a fuzzy variable, which is similar to the probability of a random variable. Definition of the expected value of a fuzzy random variable ξ introduced in [10] is as follows:

$$E[\xi] = \int_0^{+\infty} \Pr\{\omega \in \Omega \mid E[\xi(\omega)] \geq r\} dr - \int_{-\infty}^{0} \Pr\{\omega \in \Omega \mid E[\xi(\omega)] \leq r\} dr \quad (1)$$

, providing that at least one of the two integrals is finite.

From the definition above, expected value of a fuzzy random variable is a scalar value. In Equation (1), $\xi(\omega)$ is a fuzzy variable and E in the left of the equation is the expectation of a fuzzy random variable, while E on the right is the expected value of a fuzzy variable. In most real world instances, the expectation calculation of a fuzzy random variable can be simplified.

3.2 Scheduling models

The fuzzy random scheduling model refers to the schedule process of Step 4 in Figure 1, which is an essential step in an instrument cyberinfrastructure for resource sharing. The scheduling model described in this work take users' feedback information into account and try to satisfy user requirements better.

Consider an instrument pool with N instruments in it, as shown in Figure 2.

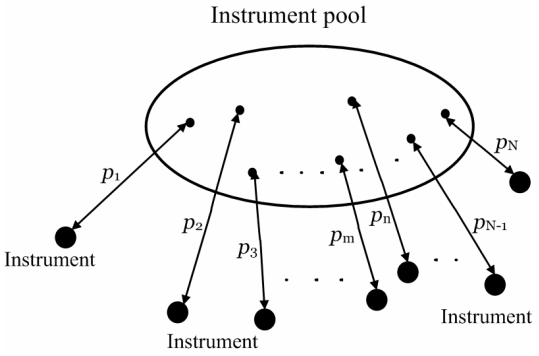

Figure 2. The job scheduling in an instrument pool

When a new experiment is submitted to an instrument pool, the probability that the experiment runs on each instrument is p_i ($i \in [1, N]$). It is obvious that the following equation holds:

$$\sum_{i=1}^{N} p_i = 1 \quad (2)$$

When an experiment is submitted to any chosen instrument, there are many factors which have influence on users' appraisals, for example the cost of experiment this instrument charges for, the execution time and waiting time, whether the result from this instrument is reliable and whether the precision of the instrument can satisfy the experiment requirement. All these factors differ with different instruments and can be looked as a virtual parameter of the instrument. In this paper, this parameter is named as QoS of instrument and denoted by q, and q_i means the QoS of the ith instrument in an instrument pool according to a specific experiment. The QoS of the same instrument will be different when the users' constrains changed. The pool adjusts the probability p_i according to the user's appraisal, Q, to the experiment after he received his result from the instrument cyberinfrastructure. Both variables q and Q are fuzzy variables because a user can not depict how he satisfied with a result accurately. Only vague linguistic values like *terrible*, *bad*, *normal*, *good* and *excellent* can express his appraisal towards the result from the instrument cyberinfrastructure.

When a user submits a job with detailed experiment specifications to an instrument cyberinfrastructure, the instrument pool alliance will pass this job to corresponding instrument pools. If the experiment is submitted to the ith instrument in instrument pool, q_i has the value as one of the following linguistic values, *very bad*, *bad*, *normal*, *good* and *very good*. In most cases, a q_i with *very good* value has a large probability to receive *excellent* value of Q , *good* to *good*, *normal* to *normal*, *bad* to *bad* and *very bad* to *terrible* of Q. Because the value of q to a specific experiment is not known by instrument pool and can only be reflected by the user's appraisal towards the total process of the experiment, the instrument pool will adjust p_i to make the instrument with *good* or *very good* appraisal higher utilization ratio to satisfy users. In some urgent experiments, users may attach more importance on the time constrain. In such case the instrument with shorter job execution time and waiting time will more satisfied the users. While in some experiments, users may care more about the cost.

This fuzzy random scheduling model is a close loop model, which takes the user's response into account and is believed to be able to provide higher instrument QoS. Figure 3 is the system block of the model.

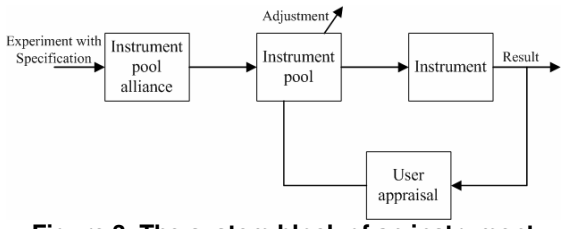

Figure 3. The system block of an instrument cyberinfrastructure

Many good scheduling strategies and algorithms can be designed on the basis of the fuzzy random model introduced above. Most importantly, users' QoS feedback information is used in this model thus it complies with the users' intention better.

3.3 Scheduling algorithms

The adjustment of p_i from users' appraisals is described in this section. In this work, the algorithm to adjust p_i is proportional to the expected value of fuzzy random variable $preq_i$, which is the prediction of the fuzzy value q_i, as shown in Equation (3). The reason why the $preq_i$ is used in Equation (3) instead of q_i is that the instrument pool has no information of q_i and has to predict what the value it is through users' appraisals.

$$p_i = E[preq_i] / \sum_{i=1}^{N} E[preq_i] \qquad (3)$$

In the following examples, the membership function of the fuzzy variable q_i is shown in Figure 4.

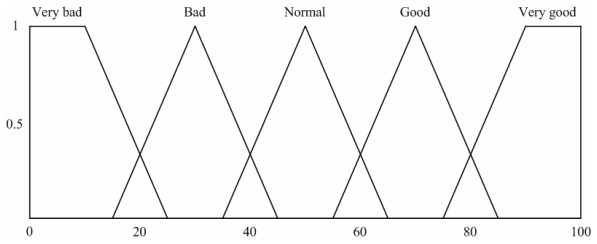

Figure 4. The QoS membership function

$$preq_i = \begin{cases} "Verygood" & with\ probability\ prep_i^1 \\ "Good" & with\ probability\ prep_i^2 \\ "Normal" & with\ probability\ prep_i^3 \\ "Bad" & with\ probability\ prep_i^4 \\ "VeryBad" & with\ probability\ prep_i^5 \end{cases} \qquad (4)$$

The distribution of $preq_i$ is as Equation (4). In Equation (4), $prep^k_i$ ($1 \le k \le 5$) means the probability that $prep_i$ equals to kth value in equation (4). The initial values of $prep^k_i$ are the same and equal to 0.2. Because $preq_i$ is a fuzzy random variable, we can calculate its expected value using Equation (1).

In this example the expectation of $preq_i$ can be simplified to Equation (5).

$$E[preq_i] = \sum_{k=1}^{5} prep_i^k \times c_i \qquad (5)$$

, in which c_i is defined to be the center of membership function and in this case, they are 17.5, 30, 50, 70 and 82.5 respectively. It should be noted that when the membership function changed, the expected value of $preq_i$ will also be different.

According to the users' feedback information, the $prep^k_i$ will be adjusted by Algorithm 1.

***Algorithm 1*:**

```
switch ( appraisal )
  {
    case "very good":
        for (i=1; i<=5; i++)
        { prep^k_i = prep^k_i * (1– 4 * increment); }
        prep^1_i = prep^1_i + 4 * increment;
        break;
    case "good":
        for (i=1; i<=5; i++)
        { prep^k_i = prep^k_i * (1– 2 * increment); }
        prep^2_i = prep^2_i + 2 * increment;
        break;
    case "normal":
        for (i=1; i<=5; i++)
        { prep^k_i = prep^k_i * (1– increment); }
        prep^3_i = prep^3_i + increment;
        break;
    case "bad":
        for (i=1; i<=5; i++)
        { prep^k_i = prep^k_i * (1– 2 * increment);}
        prep^4_i = prep^4_i + 2 * increment;
        break;
    case "very bad":
        for (i=1; i<=5; i++)
        { prep^k_i = prep^k_i * (1– 4 * increment); }
        prep^5_i = prep^5_i + 4 * increment;
        break;
  }
```

In the above algorithm, the instrument that can satisfy users will have a higher probability to be used according to Equations (3) and (5). Parameter *increment* is a constant number. If a new instrument joins into the instrument pool, the following algorithm works.

***Algorithm 2*:**

```
N = N + 1 ;
p_N = 1 / N ;
for ( i = 1; i < N; i++)
{ p_i = p_i * ( N – 1) / N ;}
```

In Algorithm 2, any new instrument joining into an instrument pool will have the average probability to be used. N is the existing number of instruments in a pool.

When an instrument wants to leave the pool, the probabilities are adjusted according to Algorithm 3. In Algorithm 3, the kth instrument in an instrument pool is supposed to leave the pool.

Algorithm 3:

$totalP$=0 ;
p_k=0 ;
for (i=1; i<=N; i++)
{ $totalP = p_i + totalP$; }
for (i=1; i<=N; i++)
{ $p_i = p_i / p$; }
for ($i = k$; i<N; i++)
{ $p_i = p_{i+1}$; }
$N = N - 1$;

Algorithm 3 only allows the instrument without any experiment running on it at that time to leave. Any instrument with job running on it is not permitted to leave. If it leaves by some inevitable reasons, the pool will record the instrument as unstable and it will have trouble when next time it wants to join the pool.

4. Performance Evaluation

In this section three case studies are given to illustrate the fuzzy random scheduling model and algorithms introduced in Section 3. The programming language of the simulation environment is Java.

4.1 Case study I

A simple experiment, which requires only one instrument, is submitted to the pool alliance. An instrument pool with N instruments, which can run the experiment, is chosen by the pool alliance. Every instrument has the same initial probability to run the experiment. In this example N equals to 50, and 10 of them have *very good* QoS and may receive users' feedback value of *excellent*, 10 *good*, 10 *normal*, 10 *bad* and 10 *terrible*. Figure 5 is the result when QoS feedback information is used to adjust probabilities of instruments in 100,000 such experiments. The vertical axis represents the number of jobs and the horizontal axis represents users' feedback information in terms of vague values. It is also the case in Figures 6, 8, 9, 10 and 11. For the purpose of comparison, the result without probability adjustment is also given in Figure 5. The parameter *increment* is a constant and in this example the values are 0.02% and 2%, respectively.

As shown in Figure 5, when feedback information from users' appraisals is considered, those instruments which can not satisfy users well will have fewer chances to be used. If the owners of these instruments want to have more chances for their instruments to be used, they should improve the QoS of their instruments, like decreasing the price their instruments charge for or shortening the execution time of their instruments.

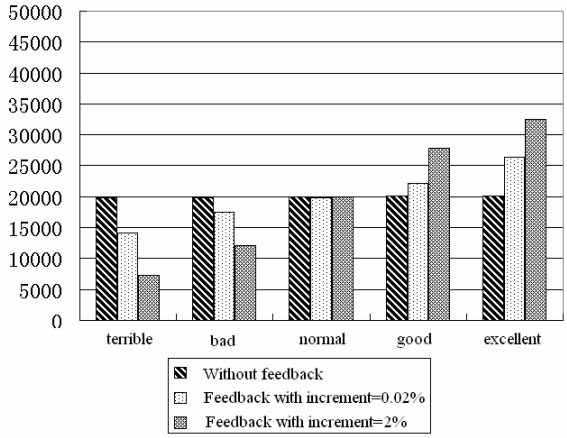

Figure 5. Results of users' appraisals for 100,000 experiments

When the probability adjustment strategy is improved on the basis of Equation (3), better results can be obtained and the occurrence of *excellent* experiments will increased.

There are more complicated scenarios for remote instrument access. For example, an experiment may involve multiple instruments, which is discussed in the case study II. Also serving more tasks will somehow decrease QoS levels of instruments, which is not considered in this case. For example, if task arrival rate is high enough to exceed processing capability of an instrument, responding delays will decrease QoS levels of users' feedback information. This is discussed in the case study III using detailed simulation results.

4.2 Case study II

In this example, two instruments in two different instrument pools are required to complete an experiment. The numbers of instruments in the two pools are N_1 and N_2, respectively. Every instrument in each pool has the same initial probability to be used. The number of instruments with different QoS values in each pool is the same. The final QoS value of an experiment is $q_i^1 \Lambda q_j^2$ when the ith instrument in one pool works coordinately with the jth instrument in another pool. This means the appraisal to the overall experiment is the worse one of the two instruments. In this example N_1 and N_2 are both 50. The user's

feedback information will have the same impact on the two instruments used.

Figure 6 includes simulation results when feedback information is used to adjust probabilities of both instrument pools. When feedback information is used, less *bad* or *terrible* experiments appeared. In comparison with Figure 5, no more *excellent* experiments are achieved and there is no obvious QoS improvement in this case. This is caused by that the two instruments are coupled in one experiment and the user can only provide feedback information on the whole experiment instead of each instrument.

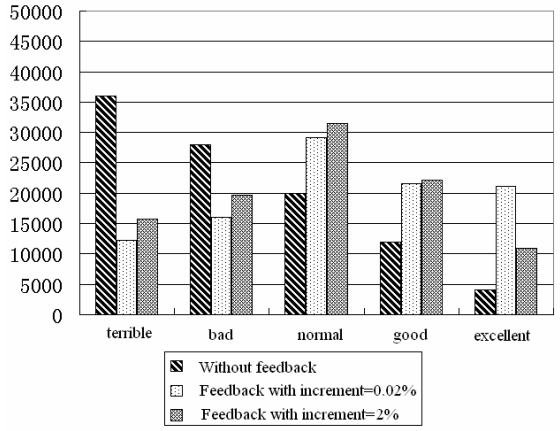

Figure 6. Results of users' appraisals for 100,000 experiments each involving 2 instruments

4.3 Case study III

In this example 100,000 similar experiment requests are submitted to the pool alliance and an instrument pool will be chose as the execution pool of these experiment requests. Similar to the case study I, there are 50 instruments in the chosen pool. Different from example 1, job execution times are taken into account. The execution time of the instruments with *very good* QoS complies with an exponential distribution and the expected value of the distribution is E_1, E_2 for *good*, E_3 for *normal*, E_4 for *bad* and E_5 for *very bad*. For the purpose of illustration and simulation the five expected values from E_1 to E_5 in this example are 1/250, 1/180, 1/150, 1/120 and 1/100, respectively. The request arrival time is supposed to be a possion distribution with λ, where λ is the average arrival rate in a possion distribution. In this case study, two situations are considered.

If experiment requests come beyond execution capabilities of an instrument pool, a queue is unavoidable. In this situation, instruments with high QoS feedback are chosen first and those with poor QoS feedback next. In the example, λ_1 and λ_2 are given according to this case. λ_1=5000 results in request arrivals far beyond a pool capability and λ_2=1000 corresponds to a situation that the request arrival is only slightly beyond a pool capability. The other situation is that experiment requests are within the capability of all instruments in a pool. Corresponding λ values are λ_3=600 and λ_4=100. The following simulation results are obtained using the flow chart described in Figure 7.

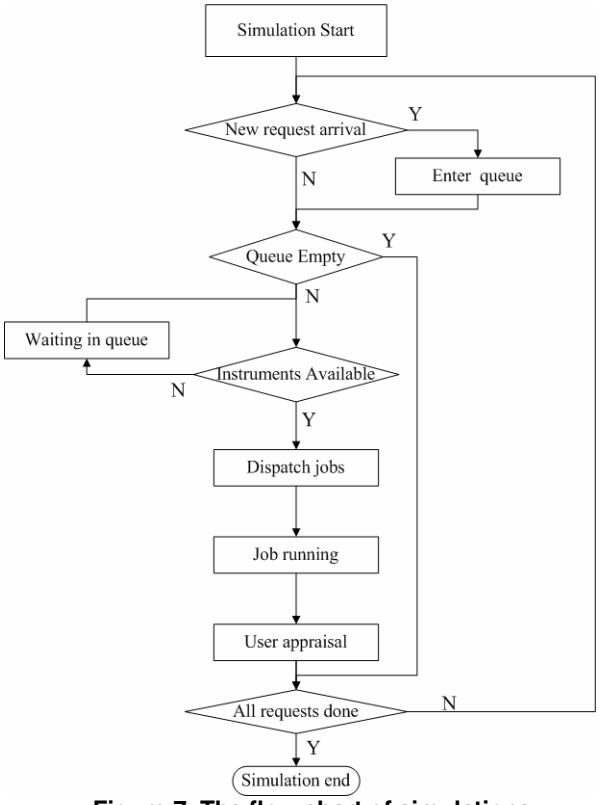

Figure 7. The flow chart of simulations

One thing we should bear in mind is that too long responding time, including waiting time in a queue and execution time on instruments, will degrade users' appraisals towards the results they got. With consideration of this situation, additional rules are applied.

- If $T_1 < RT < T_2$, the appraisal will degrade by one level.
- If $T_2 < RT$, the appraisal will degrade by two levels.

In above rules, RT represents the total responding time to an experiment request. T_1 and T_2 are two time limits that users can bear. We suppose that T_1 and T_2 are about ten to twenty times of execution time, thus T_1 =10 and T_2 =20 in this example.

Effects of these rules are also shown in Table 1, which describes relationships between users' appraisals and the responding time. For example, a *very good* experiment in a user's impression could be downgraded to be *good* if responding time is longer than T_1 and *normal* if responding beyond T_2.

Table 1. *RT* and corresponding users' appraisal

Level / RT	Very good	Good	Normal	Bad	Very bad
$< T_1$	Very good	Good	Normal	Bad	Very bad
$[T_1, T_2]$	Good	Normal	Bad	Very bad	Very bad
$> T_2$	Normal	Bad	Very bad	Very bad	Very bad

In Figures 8 to 11, simulation results of the case study III are illustrated. In each figure, simulation results with probability adjustments algorithms described in Section 3.3 and those without probability adjustments are all given for the purpose of comparison.

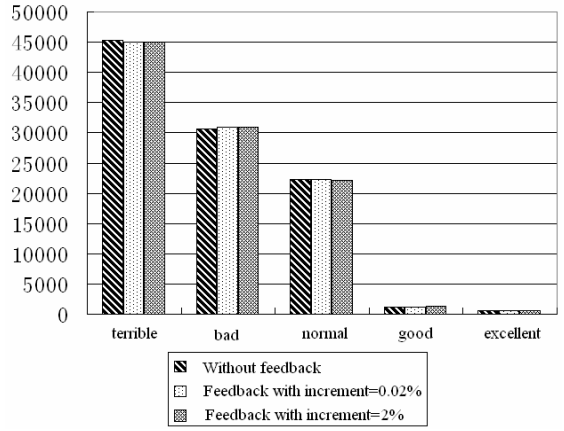

Figure 8. Results of users' appraisals under λ_1

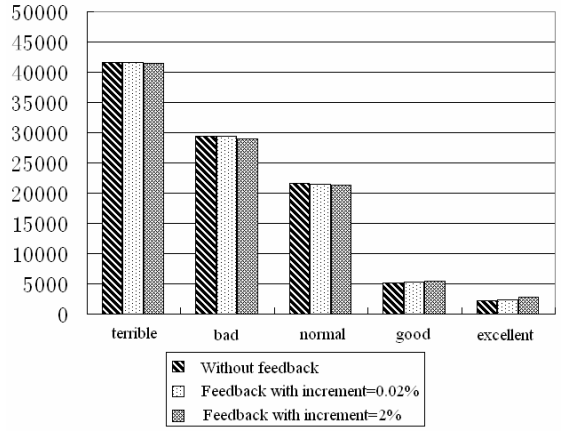

Figure 9. Results of users' appraisals under λ_2

As shown in Figures 8, when request arrival speed is far beyond a pool's processing capability, the probability adjustment algorithm does not work well to provide users with more *excellent* service, since *bad* services have to be utilized anyway. Also when requests arrive too fast and have to wait in a queue, a longer responding time will downgrade users' appraisals even if an *excellent* service is supposed to be provided. The only way to still ensure high QoS for users is to let more similar instruments join the pool to increase the pool's processing capability. The situation is improved when request arrival speed is lower in Figure 9.

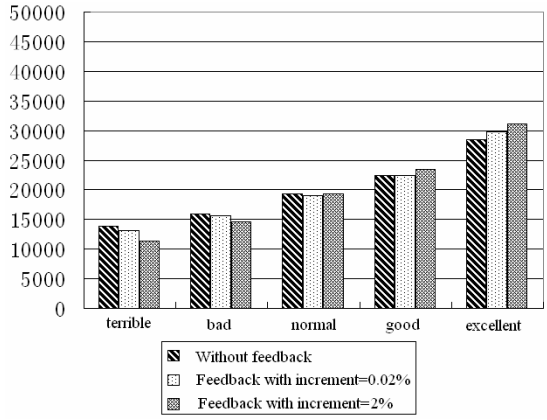

Figure 10. Results of users' appraisals under λ_3

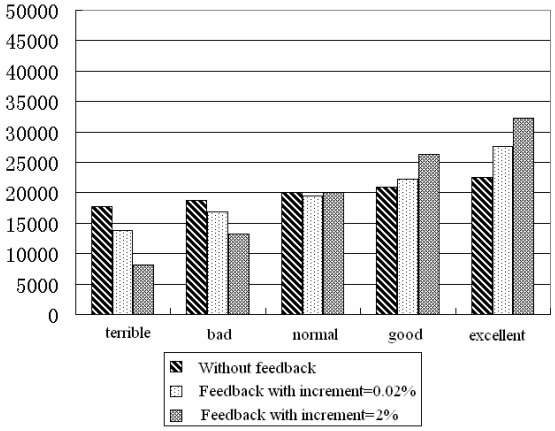

Figure 11. Results of users' appraisals under λ_4

As shown in Figures 10 and 11, arrival requests are within a pool's processing capability, more satisfactory appraisals will achieve through the adjustment of probability in an instrument pool. Since a queue seldom appears in these situations, requests do not have to be served with *bad* instruments and downgrade rules are not often applied. These results are conformed to those achieved in the case study I.

5. Conclusions

The contribution of this paper lies in the proposal of a fuzzy random scheduling model, which takes the users' QoS feedback information into account to provide more satisfactory services for users in an instrument cyberinfrastructure. The QoS appraisals from users can not be represented in an accurate and quantitative way, since there are many factors in instrument QoS that have effects on users' appraisals. In many real world scenarios, users' feedback information is fuzzy and the fuzzy random model is suitable and straightforward when applied to the scheduling scenarios described in this work.

The algorithms provided in this work to increase the utilization probability of some instruments with higher QoS and decrease usage of those with lower QoS, is proved to be effective in a cyberinfrastructure environment for scientific instrument sharing when pool capability is beyond experiment requests. In situations when request arrival speed is far beyond processing capability of an instrument pool, algorithms supposed to improve instrument QoS do not work, since long queuing time downgrades users' appraisals and instruments with low QoS feedback have to be used anyway.

When applying the work described in this paper into a real world situation, additional issues have to be considered besides resource management and scheduling. Ongoing work include an information service providing detailed instrument and experiment data, a workflow enactor to mange experiments involving multiple instruments, and a layered security mechanism for authentication and authorization of remote instrument access.

Acknowledgement

This work is supported by Ministry of Education of China under the 211/15 cyberinfrastructure project "National University Instrument and Resource Sharing Systems", Ministry of Science and Technology of China under the national 863 high-tech R&D program grants No. 2006AA10Z237 and No. 2006AA10Z216, and National Science Foundation of China under the grant No. 60604033.

References

[1]. D. E. Atkins, K. K. Droegemeier, S. I. Feldman, H. Garcia-Molina, M. L. Klein, D. G. Messerschmitt, P. Messina, et. al., *Revolutionizing Science and Engineering through Cyberinfrastructure*, National Science Foundation Blue – Ribbon Advisory Panel on Cyberinfrastructure, January 2003.

[2]. J. Cao, S. A. Jarvis, S. Saini and G. R. Nudd, "GridFlow: Workflow Management for Grid Computing", in *Proceedings of 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid*, Tokyo, Japan, pp. 198-205, 2003.

[3]. M. Faerman, A. Su, R. Wolski and F. Berman, "Adaptive Performance Prediction for Distributed Data-Intensive Applications", in *Proceedings of ACM/IEEE Supercomputing Conference*, 1999.

[4]. I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*, Morgan-Kaufmann, 1998.

[5]. I. Foster, C. Kesselman and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", *International Journal of Supercomputer Applications*, Vol. 15, No. 3, 2001.

[6]. R. Kruse and K. D. Meyer, *Statistics with Vague Data*, D. Reidel Publishing Company, Dordrecht, 1987.

[7]. H. Kwakernaak, "Fuzzy Random Variables-I. Definitions and Theorems", *Information Sciences*, Vol. 15, pp. 1-29, 1978.

[8]. H. Kwakernaak, "Fuzzy Random Variables-II. Algorithms and Examples for the Discrete Case", *Information Sciences*, Vol. 17, pp. 253-278, 1979.

[9]. B. Liu and J. Peng, *A Course in Uncertainty Theory*, Tsinghua University Press, Beijing, 2005.

[10]. Y. K. Liu and B. Liu, "Fuzzy Random Variables: a Scalar Expected Value Operator", *Fuzzy Optimization and Decision Making*, Vol. 2, No. 2, pp. 143-160, 2003.

[11]. NSF Cyberinfrastructure Council, *NSF's Cyberinfrastructure Vision for 21st Century Discovery*, Version 5.0, January 20, 2006.

[12]. M. L. Puri and D. Ralescu, "Fuzzy Random Variables", *Journal of Mathematical Analysis and Applications*, Vol. 114, pp. 409-422, 1986.

[13]. W. Smith, V. Taylor and I. Foster, "Predicting Application Run Times Using Historical Information", *Job Scheduling Strategies for Parallel Processing,* LNCS Vol. 1459, Springer Verlag, pp. 122-142, 1998.

[14]. W. Smith, V. Taylor and I. Foster, "Using Run-Time Predictions to Estimate Queue Wait Times and Improve Scheduler Performance", *Job Scheduling Strategies for Parallel Processing,* LNCS Vol. 1659, Springer Verlag, pp. 202-219, 1999.

[15]. D. P. Spooner, S. A. Jarvis, J. Cao, S. Saini and G. R. Nudd, "Local Grid Scheduling Techniques Using Performance Prediction", *IEE Proceedings – Computers and Digital Techniques*, Vol. 150, No. 2, pp. 87-96, 2003.

[16]. Y. Wang, L. Liu, C. Wu and W. Ni, "Research on Equipment Resource Scheduling in Grids", *Grid and Cooperative Computing,* LNCS Vol. 3251, Springer Verlag, pp. 927-930, 2004.

[17]. Y. Wang and C. Wu, "A Study on Education Resource Sharing Grid", *International Journal of Information Technology, Special Issue on Grid Computing I*, Vol. 11, No. 3, pp. 73-80, 2005.

[18]. L. Yang, J. M. Schopf and I. Foster, "Conservative Scheduling: Using Predicted Variance to Improve Scheduling Decisions in Dynamic Environments", in *Proceedings of ACM/IEEE Supercomputing Conference*, 2003.