

# An Adaptive Service Strategy Based on User Rating in P2P\*

Jianming Fu<sup>1,2</sup>, Lei Zhang<sup>1</sup>, Weinan Li<sup>1</sup>, and Huanguo Zhang<sup>1</sup>

<sup>1</sup> School of Computer, Wuhan University, Wuhan 430072, China

<sup>2</sup> State Key Laboratory of Software Engineering, Wuhan University,  
Wuhan 430072, China  
fujms@public.wh.hb.cn

**Abstract.** In order to deal with free riding in P2P system, incentive mechanism or rating system is presented, and each user rating is computed by itself, which causes that some users exaggerate their ratings. In this paper, three aspects are developed for free riding and for improving service performance in P2P. First of all, it is useful to keep away fake rating that each user rating is calculated by the responders. Secondly, three kinds of user ratings including single user rating and two group ratings are exploited to get better service performance. The last aspect is to provide a service strategy that is adaptive to the responder workload. Finally, the experimental results demonstrate that this strategy can effectively handle the problem of free riding, and especially, group ratings can greatly boost response rate of the whole P2P system.

## 1 Introduction

Peer-to-Peer (P2P) technology can mask the implementation of low layer in computing system, and provide open and shared computing environment. Hence this technology has been widely applied and studied [1,2,3]. P2P systems are generally designed on the assumption that all peers will be altruists voluntarily contributing the resources to a global pool. However, the behavior of peers is quite different, some peers might provide services benevolently, while other peers might be malicious or stingy and might not provide services. The behavior of participants consuming many resources while contributing little is regarded as free riding and such participants are called free-riders. P2P system may gradually degrade into a model of client/server due to the free riding. The studies [1] have showed that most participants engage in freeloading: almost 70% of Gnutella peers share little or no files and top 1% hosts provide about 50% of all services. Therefore, incentive mechanism is introduced to ensure that some hosts with high contribution get high quality of service, while other hosts with little contribution have to get low quality of service, even are rejected.

---

\* Supported by the National Natural Science Foundations of China under Grant No.90104005 and No.66973034, and by the Hubei Natural Science Foundations of China under Grant No.2005AA101C44

In fact, a query message in Gnutella carries the information about sharing the disk space and uptime of the client peer, according to which the server peer provides different services. In addition, Gnutella can limit the search scope according to the client's contribution. The more contribution it has, the larger its TTL is. User rating in KaZaA [2] is defined as the ratio of uploads to downloads. However, these ratings are calculated by the requestors, hosts with little contribution may juggle their contributions in order to get better service. If there is a centralized directory service for storing and searching the user rating, juggling the contribution could be avoided. However, it is unfeasible due to the bottleneck and single-point failure of directory server. Hence, how to design a distributed rating system is open issue.

We state our motivations as follows. Firstly, user ratings are computed by service providers to avoid fake ones. Secondly, three kinds of user ratings are exploited to enhance the response rate of whole system. Finally, the service capacity of service providers should be considered, and whether the requests could be satisfied not only depends on the requestor's rating but also the service capacity of service providers. That is, the provider's serving is sensitive to its workload. On the basis of these motivations, an adaptive service strategy based on user rating is presented to deal with free riding in unstructured P2P.

The rest of this paper is organized as follows. Section 2 reviews some related work in a brief. Section 3 states our strategy besides some definitions. We evaluate our strategy in section 4, and give conclusion in section 5.

## 2 Related Work

Vivek Vishnumurthy[4] provides an economic framework using Karma. Karma represents the user rating, and it increases when uploading files, while decreases when downloading. Any peer can't download any file until its Karma achieves a certain value. The Karma of each user is stored at a Bank-set, which also take part in the dealing. Chiranjeeb Buragohain[5] studies a game theoretic framework for incentives, and each peer is rational and politic to pursue its maximum utility, which is based on its rating, so that the Nash equilibrium point can be obtained. Debojyoti Dutta[6] provides two verification schemes for checking the requestor' rating. Qixiang Sun[7] proposes SLIC mechanism based on link, which calculates a peer's rating by the neighbor's history records. There are many researches on peer trust and reputation[8,9,10], and each user is assigned a reputation by system that reflects its contribution and its participation in the system. In addition, XU FEI designs a P2P system based on peer group[11], which is useful to group peers, and Hai Zhuge presents query routing mechanism using semantic links among neighbors[12] to speed up queries and the future interconnection environment including P2P[13].

Three kinds of ratings are used in our strategy, and are different from SLIC, in which only the single user rating between neighbors is used.

### 3 Adaptive Service Strategy

In P2P system, there are two roles for each peer: Service Providers (*SP*) which providing others with service and Service Requestors (*SR*). Although each peer has two roles, we just pay attention to one role when we state our strategy. Before a *SP* makes a decision for a query, it must calculate the rating of the query requestor. There are three key points in our strategy: how to collect information for rating, how to rate a requestor, and how to provide service on the ratings. We will discuss these issues as below.

#### 3.1 Definition

Each peer will rate other peers based on the service it receives from them and will store its own rating. This service includes factors like the number of successful requests it makes, response delay, download speed of file transferring transaction, or length of downloaded file, etc. Here, three tables are used to store basic information and logs: *MI*, *GT* and *ST*.

The all peers in P2P are divided into different groups. Each peer has an identifier *PID*, and it must belong to a group. At the same time, each group has an identifier *GID*. In order to compute ratings, each peer keeps the following.  $MI = \{PID, TimeStamp\}$ , records all members of a group, here *TimeStamp* indicates the time when a member join this group.  $GT = \{PID, GID, FID, FS, TimeStamp\}$ , records logs describing a peer's getting service from the system, here *PID* is an identifier of a *SP*, *GID* is a group identifier of the *SP*, *FID* is an identifier of file to be downloaded, *FS* is the length of this file, and *TimeStamp* indicates the time when this file is downloaded.  $ST = \{PID, GID, FID, FS, TimeStamp\}$ , records logs containing a peer's providing files, here *PID* is an identifier of a *SR*, *GID* is a group identifier of the *SR*, and *FID*, *FS*, *TimeStamp* is the same as the one of *GT* respectively.

The records in *GT* and *ST* come from transactions in the past. After a peer  $N_{sr}$  downloads a file with *FID* from a peer  $N_{sp}$ ,  $N_{sr}$  will insert a record  $\{N_{sp}.PID, N_{sp}.GID, FID, FS, T1\}$  into its *GT*, meanwhile  $N_{sp}$  will insert a record  $\{N_{sr}.PID, N_{sr}.GID, FID, FS, T1\}$  into its *ST*, where *T1* is a timestamp.

In traditional rating system, a service provider decides its service based on the rating which the requestor claims. The rating  $R_i$  of user  $u_i$  is defined by the set of all users it served within some sliding window, so it reflects all contribution of  $u_i$  to the P2P system within some sliding window. However, it is easy to fake this rating, because this rating is computed by the requestor. Hence, this rating system requires some verification scheme to check the rating[6].

In our strategy, the rating  $R_i$  of user  $u_i$  on user  $u_j$  is defined by  $u_j$  whom  $u_i$  served within some sliding window, and it is computed by  $u_j$  without any verification schemes. This rating is also called single user rating defined as *Definition1*. Nevertheless, this rating reflects partial contribution of  $u_i$  to the P2P system, even has very small value that reduces the response rate of whole P2P system. In order to improve the response rate, two group ratings are introduced

in *Definition2* and *Definition3*. One group rating  $R_i$  of user  $u_i$  on user  $u_j$  is defined by  $u_j$ 's group whom  $u_i$  served. The other group rating  $R_i$  of user  $u_i$  on user  $u_j$  is defined by  $u_j$ 's group whom  $u_i$ 's group served. And two group ratings are calculated by the group of  $u_j$  without any verification schemes.

When  $N_{sp}$  receives a query from  $N_{sr}$ ,  $N_{sp}$  must choose its strategy for service, and  $N_{sp}$  decides how to rate  $N_{sr}$ . And three kinds of rating of  $N_{sr}$  vs.  $N_{sp}$  are defined as below.

*Definition1*: Rating of Peer vs. Peer (*RPP*) is defined as:

$$RPP(N_{sr}, N_{sp}) = \frac{\sum_{i \in N_{sp}.GT \text{ and } i.PID = N_{sr}.PID} i.FS}{\sum_{i \in N_{sp}.ST \text{ and } i.PID = N_{sr}.PID} i.FS} + \delta \tag{1}$$

*RPP* is a ratio of the service level to the consumption level of  $N_{sr}$  vs.  $N_{sp}$ .

*Definition2*: Rating of Peer vs. peer's Group (*RGP*) is defined as:

$$RGP(N_{sr}, N_{sp}) = \frac{\sum_{j \in N_{sp}.MI} \sum_{i \in j.GT \text{ and } i.PID = N_{sr}.PID} i.FS}{\sum_{j \in N_{sp}.MI} (\sum_{i \in j.ST \text{ and } i.PID = N_{sr}.PID} i.FS)} + \delta \tag{2}$$

*RGP* is a ratio of the service level to the consumption level of  $N_{sr}$  vs.  $N_{sp}$ 'group.

*Definition3*: Rating of peer'Group vs. peer's Group (*RGG*) is defined as:

$$RGP(N_{sr}, N_{sp}) = \frac{\sum_{j \in N_{sp}.MI} \sum_{i \in j.GT \text{ and } i.GID = N_{sr}.GID} i.FS}{\sum_{j \in N_{sp}.MI} (\sum_{i \in j.ST \text{ and } i.GID = N_{sr}.GID} i.FS)} + \delta \tag{3}$$

*RGG* is a ratio of the service level to the consumption level of  $N_{sr}$ 'group vs.  $N_{sp}$ 'group.

$\delta$  is the constant bias, which is useful when the consumption level is zero in above equations. *RGP* and *RGG* are group ratings, and are introduced to improve performance of P2P system, because many queries may be rejected thanks to low *RPPs* that are used to decide the service level by providers.

Without loss of generality, the size of *ST* and *GT* is limited, so each record in any tables must expire after the interval of *T\_Period*, which is time constant. In other words, when a period of *T\_Period* elapses after a record is inserted, this record ought to be deleted. In addition, we will introduce sliding windows as weights reflecting impacts of timestamps on user rating, which motivates users to contribute to the community continuously. During a period of valid time, the more recent the activity happens, the larger the weight is. Assume current time is *Current\_time*, *i.FS* of the definitions above should be modified as the following.

$$i.FS = i.FS * \frac{i.Timestamp - Current\_time + T\_Period}{T\_Period} \tag{4}$$

The logs of all members in a group are needed to compute  $RGP$  and  $RGG$ . In order to decrease the number of communications and network traffic,  $N_{sp}$  may choose  $q$  members randomly from its group for computing group ratings, here  $q < |N_{sp} \cdot MI|$ .

### 3.2 Service Strategy

After a query is reached from  $N_{sr}$ ,  $N_{sp}$  must handle this query by certain strategy such as responding at once, responding with delay, or rejecting. There are two kinds of relationship between  $N_{sp}$  and  $N_{sr}$ . One is  $N_{sp}.GID = N_{sr}.GID$ , which indicates that the query comes from the same group, and let the proposition  $ingroup(N_{sp}, N_{sr})$  be true (denoted  $ingroup$ ). The other is  $N_{sp}.GID \neq N_{sr}.GID$ , which indicates that the query comes from other group, so  $RPP$ ,  $RGP$  and  $RGG$  need to be computed on demand.

Besides ratings, our strategy is adaptive to the workload of  $SP$ , so assume the service capacity of  $N_{sp}$  is  $N_{sp}.C$  per time unit, and current workload is  $(N_{sp}.RN + N_{sp}.SN)$ . Here,  $N_{sp}.RN$  is the number of queries per time unit originated by  $N_{sp}$ , and  $N_{sp}.SN$  is the number of responses per time unit for  $N_{sp}$ . For simplicity, we should unify the workload, as  $N_{sp}.LD = (N_{sp}.RN + N_{sp}.SN)/N_{sp}.C$ . For convenience of service decision, we import the thresholds of workload  $\mu_1, \mu_2, \mu_3$  and  $\mu_4$ , which follow an order  $\mu_1 < \mu_2 < \mu_3 < \mu_4$ . These thresholds may influence the service quality. At the same time, we introduce rating thresholds  $\alpha_1, \beta_1$  and  $\gamma_1$ , where  $\alpha_1$  is a threshold of  $RPP$ ,  $\beta_1$  is a threshold of  $RGP$ , and  $\gamma_1$  is a threshold of  $RGG$ . All thresholds will be subject to our strategy.

Suppose  $\alpha = RPP(N_{sr}, N_{sp}), \beta = RGP(N_{sr}, N_{sp}), \gamma = RGG(N_{sr}, N_{sp})$ , then the decision of  $SP$  is shown as follows according to our adaptive service strategy.

(1) if  $N_{sp}.LD < \mu_1$  then if  $ingroup$  or  $\alpha > \alpha_1$  or  $\beta > \beta_1$  or  $\gamma > \gamma_1$  then responds at once. else responds with probability  $\rho$  as well as delay of  $DELAY\_CONST$ .

(2) else if  $N_{sp}.LD < \mu_2$  then if  $ingroup$  or  $\alpha > \alpha_1$  or  $\beta > \beta_1$  then responds at once. else if  $\gamma > \gamma_1$  then responds with delay of  $DELAY\_CONST$ . else refuses this query.

(3) else if  $N_{sp}.LD < \mu_3$  then if  $ingroup$  or  $\alpha > \alpha_1$  then responds at once. else if  $\beta > \beta_1$  then responds with delay of  $DELAY\_CONST$ . else refuses this query.

(4) else if  $N_{sp}.LD < \mu_4$ , then refuses this query.

Here,  $DELAY\_CONST$  is time constant. Our strategy includes 4 parts:

- When the workload of  $SP$  is very light ( $N_{sp}.LD < \mu_1$ ),  $SP$  immediately serves  $SR$  who shares the same  $GID$ , or has certain ratings about  $\alpha, \beta, \gamma$ . Or else  $SP$  provides service with  $\rho$  besides delay. That is,  $SP$  refuses this query with  $1-\rho$ .

- When the workload of  $SP$  is lighter ( $\mu_1 \leq N_{sp}.LD < \mu_2$ ),  $SP$  immediately serves  $SR$  who shares the same  $GID$ , or has certain ratings about  $\alpha, \beta$ . Or else

*SP* provides service with delay while *SR* has certain rating of  $\gamma$ . For other cases, *SP* refuses this query.

- When the workload of *SP* is heavier ( $\mu_2 \leq N_{sp.LD} < \mu_3$ ), *SP* immediately serves *SR* who shares the same *GID*, or has certain rating of  $\alpha$ . Or else *SP* provides service with delay while *SR* has certain rating of  $\beta$ . For other cases, *SP* refuses this query.

- When the workload of *SP* is the heaviest ( $\mu_3 \leq N_{sp.LD} < \mu_4$ ), *SP* refuses this query.

For above strategy, what is the first considered is *RPP*, then is *RGP*, and the last is *RGG*. In order to shorten response time, *SP* firstly computes its workload, and determines where the query come from, then decides whether it is necessary to compute *RGP* and *RGG*.

## 4 Performance Analysis

For evaluating the efficiency of our strategy, we will introduce the experiment preparation, methodology and the experimental results.

### 4.1 Preparation

Our experimental network is similar to that of Gnutella except for file searching. There are 300 peers and 3000 files (objects) in an unstructured P2P system.

There are many ways to construct groups [14], among which we introduce a simple one. First we should fix the number of groups as 15. Then, we randomly choose 15 peers as seed peers representing 15 groups, and each peer of the rest joins the corresponding group in a certain probability in term of its distances to groups.

All ratings are related to file sizes, so it is necessary to choose size for each file. The main file type in Gnutella which people have interests in is multimedia, which takes 95% of all the files. Among all audio is 35%, video is 59% and others are 6%[15]. And the results [15] also show that the distribution of the file lengths is 10% of 0.01MB-2MB, 75% of 2MB-9MB, 10% of 9MB-100MB and 5% of 100MB-600MB. We assume that the distribution of file lengths is uniform within each interval, and the distribution of all file lengths in our experiment follows that in Gnutella.

In order to minimize the search cost and encourage sharing, users should share the files after they get them, and there are many replication strategies [16]. In Gnutella, files are shared immediately when they have been downloaded. Hence the copy of the files will increase, and eventually the copies of files will obey Zipf-like distribution. For simplicity, our experiment does not use any replication strategy.

### 4.2 Methodology

Generally speaking, let  $\alpha_1 = \beta_1 = \gamma_1 = 1$ , which means if the client peer contributes more than it consumes, that is, the client's user rating for the server

**Table 1.** Definition and values of symbols

Parameter	Value	Description
$\alpha_1, \beta_1, \gamma_1$	1	rating thresholds
$C$	2.5	service capacity
$\lambda$	5	mean number of queries per round for a peer
$\rho$	0.5	service probability
$\mu_1, \mu_2, \mu_3, \mu_4$	0.5, 0.7, 0.8, 0.9	workload thresholds
$\delta$	100	rating bias

is more than 1, this peer may get better services. The service probability for low rating's requestors is decided by  $\rho$ , let  $\rho=0.5$ , so that the rejection of query and delayed service have the same probability. We may enlarge its value to improve the response rate of whole system.

In order to get statistic performance of our model, assume that a round consisting of 100 time units is regarded as a period, and each performance is computed per round. The time window  $T\_Period$  has important influences on the calculation for user rating, and let it be 1000 time units. And a delay factor of  $DELAY\_CONST$  is 10 time units.

In each round, 12.5% of the whole peers are chosen randomly to send queries. Without loss of generality, the number of requests sent by each peer per round follows the Poisson distribution with  $\lambda$ , which is mean number of requests. And the requested files obey Zipf-like distribution. The various files in Gnutella system have different popularities. Therefore, the more popular the file is, the more frequently it is requested. And the research has proved that the requests in Gnutella, Napster and Web follow Zipf-like distribution[17], that is and  $q_i$  is the popularity of the  $i$ th file, and  $\alpha = 1.2$ [16]. In order to obtain  $RGP$  and  $RGG$ , several members should be asked for the information, and let  $q=5$  to get more information. The bigger  $q$  is, the more precise these ratings are. Finally, our experiment runs 80 rounds, and other parameters are listed in *Table1*.

The service capacity in *Table1* is assigned 2.5, which depends on average workload of whole system. Because we don't know this workload, we use estimated workload as 1.25. The workload per peer is not uniform, since the distribution of requested files is subject to zipf-like that. Therefore,  $C$  is assigned 2.5, two times of 1.25.

### 4.3 Results

We evaluate the efficiency of our strategy in two metrics. One is Successful Request Rate ( $SRR$ ) indicating that how many peers receive the answers after queries submission, and the other is Mean Latency ( $ML$ ) indicating the average delay of all responses. In addition, the reason of query failure is either lower rating of service requestors or overload of service providers, so we will give exact reason while a query fails. The whole experiment consists of 4 parts. Part A evaluates the impacts of rating thresholds on  $ML$  and  $SRR$ . Part B evaluates the impacts of service capacity on  $ML$  and  $SRR$ . Part C evaluates the impacts

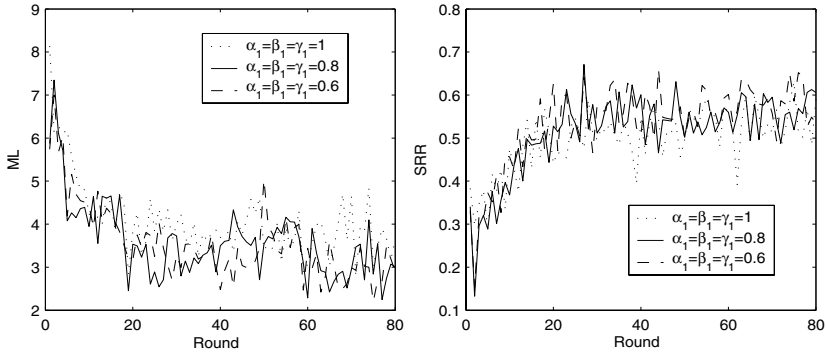


Fig. 1. *ML* and *SRR* for various rating thresholds

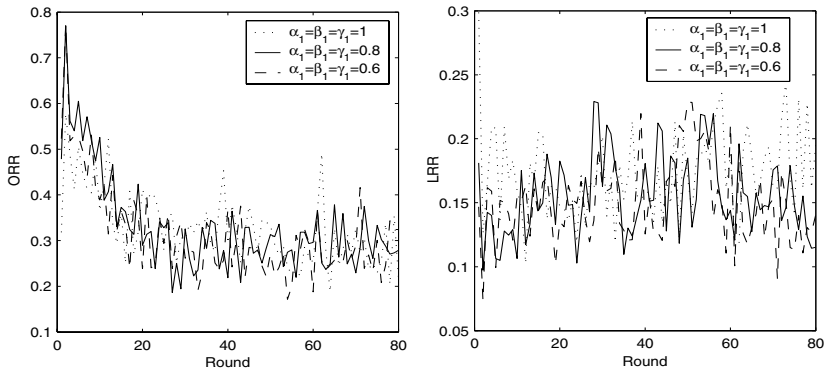


Fig. 2. The left denotes Refusal Rate for Overload(*ORR*)and the right is Refusal Rate for Low-rating(*LRR*)

of query loads on *ML* and *SRR*. Part D evaluates the impacts of three ratings on *ML* and *SRR*. The parameters are assigned the same as that in *Table1* except for the variational parameter annotated in the figures.

In Fig.5, the symbol of ' $\alpha$ ' represents that only *RPP* is employed in our strategy, the symbol of ' $\alpha + \beta$ ' indicates both *RPP* and *RGP* are employed, and the symbol of ' $\alpha + \beta + \gamma$ ' means three ratings are used. From all figures, we obtain some observations as follows:

- For whole system, we can improve response rate and reduce average latency via a way of reducing the rating thresholds of  $\alpha_1, \beta_1$  and  $\gamma_1$ . Moreover, the rate of query failure due to overload is close to two times of that owing to lower rating. The lower the rating thresholds are, the lower two rates (*ORR* and *LRR*) are.
- For whole system, we can improve response rate and reduce average latency via a way of enlarging the service capacities of service providers.

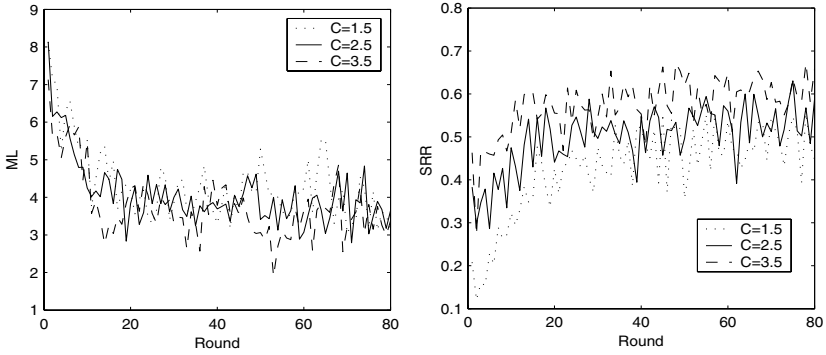


Fig. 3. ML and SRR for various service capacities

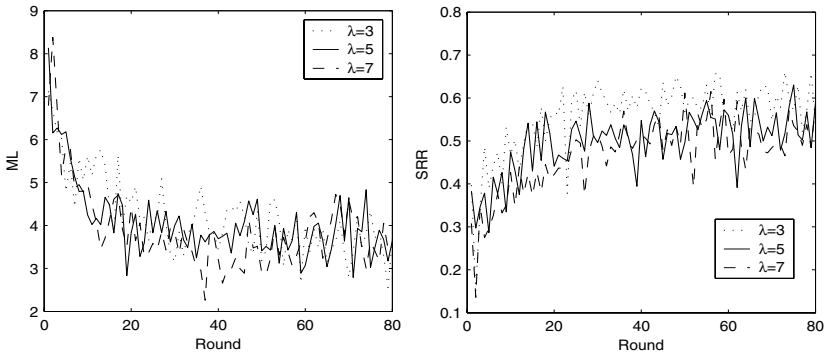


Fig. 4. ML and SRR for various query loads

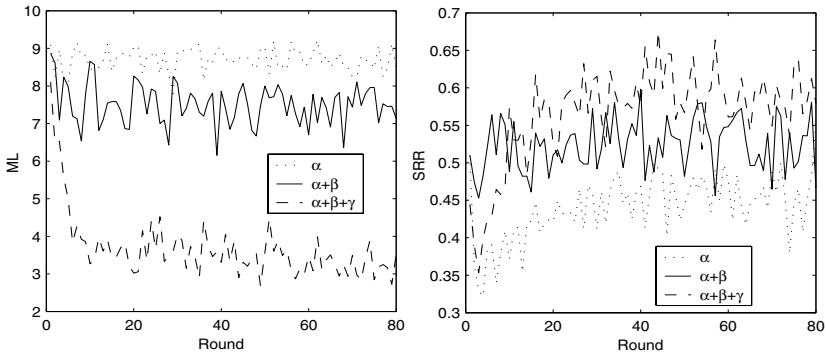


Fig. 5. ML and SRR for various rating

- If the mean number of queries is larger, then the latency is smaller, and response rate is smaller. This is because large queries may result in overload of *SP*.

- In terms of implementation of our strategy, the more the ratings employed are, the smaller the latency is and the larger the response rate is. Specially, *RGP* and *RGG* can greatly improve response rate compared with only *RPP*.

## 5 Conclusion

This paper presents an adaptive service strategy based on user ratings to deal with free riding in unstructured P2P. The user ratings include single user rating and two group ratings, and are computed by the service providers in light of the historical logs to avoid masquerading ratings. This rating mechanism does not need any verification schemes, and is collusion free in nature. Moreover, the strategy is adaptive to the workload of service providers, and provides the different services according to these ratings. So the host with little contribution can hardly get services from others. Finally, the experimental results demonstrate that this strategy can effectively handle the problem of free riding, and the group ratings can improve the response rate of whole P2P system. Although our strategy can make the free-rider get bad service or no service, how to identify and eliminate a greedy peer is our future research.

## References

1. E. Adar and B. A. Huberman. Free riding in Gnutella, Xerox PARC report, Oct 2000, avail-able at [www.parc.xerox.com/istl/groups/iea/papers/gnutella](http://www.parc.xerox.com/istl/groups/iea/papers/gnutella).
2. KaZaA, "http:// www.kazaa.com".
3. Aoying Zhou, Bo Ling, Zhiguo Lu et al. Efficient Semantic Search in Peer-to-Peer Systems. WAIM 2003, LNCS 2762, pp. 278-289, 2003.
4. Vivek Vishnumurthy, Sangeeth Chandrakumar, and Emin Gun Sirer. KARMA: A Secure Economic Framework for P2P Resource Sharing. In Proceedings of the Workshop on the Economics of Peer-to-Peer Systems, Berkeley, California, June 2003.
5. Chiranjeeb Buragohain, Divyakant Agrawal, Subhash Suri. A Game Theoretic Framework for Incentives in P2P Systems. Proc. of the Third International Conference on P2P Computing (P2P2003), Linkoping Sweden, 2003.
6. D. Dutta, A. Goel, R. Govindan, and H. Zhang. The Design of A Distributed Rating Scheme for Peer-to-peer Systems. In Workshop on Economics of Peer-to-Peer Systems, 2003.
7. Qixiang Sun, Hector Garcia-Molina. SLIC: A Selfish Link-Based Incentive Mechanism for Unstructured Peer-to-Peer Networks. 24th International Conference on Distributed Computing Systems (ICDCS'04), March, 2004. Hachioji, Tokyo, Japan.
8. Dou W, Wang HM, Jia Y, Zou P. A recommendation-based peer-to-peer trust model. Journal of Software, 2004,15(4): 571 583.
9. Seungjoon Lee, Rob Sherwood, Bobby Bhattacharjee. Cooperative Peer Groups in NICE. IEEE Infocom, April 2003.

10. Yao Wang, Julita Vassileva, Bayesian Network Trust Model in Peer-to-Peer Networks. The Second International Workshop on Agents and Peer-to-Peer Computing (P2PC 2003).
11. XU Fei, YANG Guang-Wen, JU Da-Peng. Design of Distributed Storage System on Peer-to-Peer Structure. *Journal of Software*, 2004, 15(2): 2-9.
12. H. Zhuge, J. Liu, L. Feng, X. Sun and C. He. Query Routing in a Peer-to-Peer Semantic Link Network. *Computational Intelligence*, 21(2)(2005)197-216.
13. H. Zhuge, The Future Interconnection Environment, *IEEE Computer*, 38 (4)(2005) 27-33.
14. S. Ratnaswamy, M. Handley, K. Karp, and S. Shenker. Topologically-aware overlay construction and server selection. In *Proceedings of INFOCOM*, June 2002.
15. Stefan Saroiu, Krishna P. Gummadi, Richard J. Dunn, Steven D. Gribble, and HeNsry M. Levy, An Analysis of Internet Content Delivery Systems. *Proceedings of the 5th Symposium on Operating Systems Design and Implementation Boston, MA, December 9-11, 2002*.
16. Qin Lv Pei Cao Edith Cohen Kai Li Scott Shenker, Search and Replication in Unstructured Peer-to-Peer Networks. *Proceedings of the 16th international conference on Supercomputing*, June, 2002, New York, USA.
17. Kunwadee Sripanidkulchai. The popularity of gnutella queries and its implications on scalability. In O'Reilly's *www.openp2p.com*, February 2001.