International Journal of Soft Computing 2 (4): 513-519, 2007
© Medwell Journals, 2007

Fuzzy Based False Feedback Mitigation in Broker Architecture Using Linear Interpolation

P. Varalakshmi, S. Thamarai Selvi, Sarah Wazirah, S. Ramakrishnan and S. David Paul Rufus
Department of Information Technology, MIT, Anna University, India

Abstract: In a grid environment where remote services/resources are accessed, the trust-index of an entity is of utmost importance. In the broker architecture of the grid environment, the grid is divided (logically) into multiple domains where in each domain; multiple entities i.e., Consumers and Resource Providers (RP) co exist along with a number of brokers. The broker acts as a middleman for the transactions committed between the entities. Each entity will be associated with more than one broker to ensure reliability and eliminate single point of failure. The brokers' task is to select an appropriate Resource Provider to provide a resource to the consumer's request. After the transaction is committed, the consumer will submit a feedback to the broker. This feedback will be evaluated by the broker for genuineness. By doing so, the trust-index of the RP is safe guarded against dishonest ratings by the consumer. If the feedback of a particular transaction is found to be dishonest, the consumer's trust index is reduced. In order to predict a rating to replace the dishonest feedback, we propose one-dimensional linear interpolation. This pseudo-feedback can be used to compute the new trust-index of the RP. We have used fuzzy logic to compare the dynamically varying trust-indices of both the RP and consumer. In our fuzzy model, the variations in the trust-indices of these entities are moderate than that of those obtained from our probability model when the consumer acts maliciously. Also, we have proposed an efficient method to effectively reduce message overhead that results when the feedback of every transaction is verified for genuineness.

Key words: Trust-index, feedback, interpolation, fuzzy logic, overhead

INTRODUCTION

Grid is a virtual resource framework where resources are being shared among autonomous domains which can be geographically distributed (Woodas *et al*., 2004). Each grid domain is associated with multiple entities namely consumers and RPs. The entities involved in a specific transaction may be located in different domains. A consumer, belonging to one domain may require executing jobs on a remote resource belonging to another domain. Some consumers may not want their applications mapped onto resources that are owned and/or managed by entities they do not trust. Hence it is necessary to have entities that trust each other (Farag and Muthucumaru, 2002).

Trust is the firm belief in the competence of an entity to act as expected such that this firm belief is not a fixed value associated with the entity but rather it is subject to the entity's behavior and applies only within a specific context at a given time (Farag and Muthucumaru, 2002). The consumers who have used the resources of the RP submit feedback which is aggregated over a period of time. This forms the reputation of the specific RP.

Our architecture is not just a consumer-friendly one. In our architecture, the trust-index of the RP is protected from malicious consumers. The feedback submitted by the consumer after each transaction must be verified for genuineness and only truthful ratings will contribute to the updating of the RP's trust-index. Interpolation techniques are employed to predict ratings in cases where dishonest feedbacks are detected.

In our model, we have employed fuzzy logic to compare the dynamically varying trust-indexes of both the RP and consumer. We deployed fuzzy logic because it deals with intuitive reasoning and represents membership in vaguely defined sets in contrary to the concept of probability. Also, trust-index is a continuous value and can be mapped by fuzzy rules to a given output. An efficient mechanism to effectively reduce message overhead is also proposed in our model.

In Varalakshmi *et al.* (2006) an environment in which only the consumer and RP explicitly associate themselves to brokers in a given domain is proposed. In our model the brokers will first initialize and associate themselves to some domain randomly. However, the association of the entities to the brokers will be done by comparing the trust-indices of the entities with that of the broker. Architecture with equal distribution of the entities to brokers is proposed in Varalakshmi *et al.*(2006).This aims at load-balancing at each broker's end. But our model with

Corresponding Author: P.Varalakshmi, Department of Information Technology, MIT, Anna University, India

such association of entities will be more efficient. In Varalakshmi *et al.*(2006) a grid environment is suggested where the initial trust-index to the entities has been assigned randomly. In our proposed model we evaluate the initial trust-index of the RP. Azzedin (2002, 2004) and Kwei-Jay *et al.* (2005) suggest having each entity associated with a single broker per domain, leading to a single point failure. In our model, multiple brokers exist in each domain, thus avoiding single-point failure and improving overall redundancy and reliability of the system. While only the trust-worthiness of the broker has been considered for the selection of RP in Azzedin and Maheswaran (2004). In our model, selection of the best suitable RP is based on the trust value of the RP, the queue length for that particular resource at the RP and criticality of the transaction. Both the participating entities of a transaction will submit a feedback value and both will be punished when a mismatch is detected between their ratings in Thanasis and George (2005). However, this is unfair to the honest entity and in our model the consumer alone will submit a feedback and this will be checked for genuineness by the broker. In Xiangli *et al.* (2006) interpolation is merely used to bias tune the first-hand ratings. However, in our model, interpolation is employed to predict feedback values and update the trust-index of the RP.

PROPOSED ARCHITECTURE

The broker architecture has multiple domains. Each domain is associated with multiple brokers and each broker with multiple entities, as shown in Fig. 1.

On joining the grid, each entity will request a random number of brokers in a given domain for association. And this entity will be associated to the brokers if and only the trust-index of the entity is greater than that of the brokers. Brokers maintain a database of all consumers and RPs under it. Hence the entities need not maintain information about each other. Each entity will have its own information only. The routing of the requests, results and

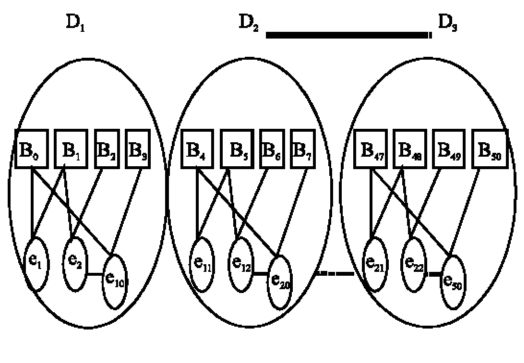

Fig. 1: Broker architecture

feedback values is done by the brokers alone. The entities will not be allowed to communicate with each other directly.

Since the entities are associated with more than a single broker, accumulation of too many requests at any broker site is avoided, leading to easing of network traffic at the broker site. The requests from the entities are distributed amongst the brokers and this balances the network load. Information about every entity should be available with at least two brokers to ensure that deletion of sensitive information of the entities at any one broker's site does not falter the entire architecture. However, the presence of multiple brokers raises other issues such as distribution of entities among brokers and maintenance of consistency of information about entities at different brokers. Failing to deal with these issues will negate the advantages of having multiple brokers. The RP will have its unique id, the details of the resources it offers, its trust-index and its parameters related to its existence in the domain. These parameters are its domain id, the id of the brokers to which it is associated and some Boolean attributes which include presence-index, adherence to standards and punishment state

Similarly the consumer will have its unique id, its trust-index and its parameters related to its existence in the domain. These parameters are similar to that of the RP. When the consumer requires a resource, it will send its request to the broker to which it is associated. This request will contain details like resource name, budget, deadline and criticality rate. This request will trigger a searching_match task at the broker's end. The broker will first perform a local task to see if any RP within the local domain is appropriate for servicing the consumer's request. If the local task fails to return satisfactory results, the forward task is executed to forward the consumer's request to brokers in other domains. So, a search for an appropriate RP is done in multiple domains. Although this increases the response time of the given request, this method ensures the best RP available to the consumer. Once an appropriate RP is found, the resource is provided to the consumer and the databases at the entities and the broker's end are updated.

PROTOCOLS IMPLEMENTED

The proposed architecture involves 4 protocols which are: User-request protocol, broker-broker protocol, RP-resource protocol and the feedback protocol.

User-request protocol: This protocol deals with the consumer's requests for a specified service to a broker in the same domain as the consumer. A user, taking the role of a consumer, C sends a request for a service to a broker Bi, in the domain Di (Varalakshmi *et al.*, 2006).

Broker-Broker protocol: This protocol handles the interactions between the brokers to collate information about the RPs in different domains in accordance with the defined set of rules. This aids in the choice of a RP suitable for a user-request (Varalakshmi *et al.*, 2006).

RP-Resource protocol: The RP provides the resource to the consumer, C for its purpose. And it also monitors its resource at periodic intervals to ensure no intrusion by unauthorized entities takes place.

Feedback protocol: After utilizing the resource of the selected RP, the Consumer (C) sends a feedback to the Broker (Bi) who recommended the RP. This Feedback (FDB) is calculated based on the parameters mentioned and the feedback is checked for genuineness using the algorithm in the study.

FEEDBACK AND TRUST-INDEX

At the end of any transaction, the consumer is expected to return a feedback value to the broker. Let this feedback value be a floating point number between 0 and 1 whereby 0 indicates 0% satisfaction and 1.0 indicates 100% satisfaction. This feedback value will determine if the consumer was satisfied with the service that he was offered.

The feedback submitted by a consumer, C to a RP is calculated as follows:

$$\mathrm{FDB(C,RP)} = \sum (w_j * p_j) \tag{1}$$

Where, P_1 denotes the percentage of completion of assigned work, P_2 denotes whether the job was completed within the budget, P_3 denotes the level of commitment of the RP to keep to the schedules, w_j denotes the weightage values given to the above parameters(P_j) and j takes the values from 1 to 4.

These feedback values from the consumer are used to compute the trust-index of the RP. And at the end of every transaction, the trust-index of the RP is updated. The trust-index of the RP is calculated as follows:

$$\mathrm{TIX\ (RP)} = \frac{a * \sum(\mathrm{FDB\ (C,RP)} * C_r)}{t_1} \frac{+ b * \sum(\mathrm{FDB(C,RP)} * C_r)}{t_2} \tag{2}$$

Where, TV(RP) denotes trust-index of the RP, FDB(C,RP) denotes feedback submitted by the consumer, C for a transaction with this RP, C_r denotes criticality rate of the transaction, t_1 denotes number of recent transactions, t_2 denotes number of past transactions and a, b are weightage values for recent and past transactions, respectively where a>b.

A consumer can purposely give a bad feedback even when the transaction was satisfactory i.e. negative discrimination. A consumer can purposely give a good feedback even when the transaction was not satisfactory i.e. positive discrimination. Now it is the responsibility of the broker to identify the badmouthers and the ballot stuffers, respectively (Jason and Jon, 2005). If the feedback is identified to be false, the consumer will be punished. His trust-index will be reduced by a value, r_1 which depends upon the criticality of the transaction and the trust-index of the consumer.

We propose this reduction factor, r_1 which is calculated as follows:

$$r_1 = \mathrm{FDB_{curr}(C,RP)} * \frac{\sum \mathrm{FDB_{neg}(C,RP)} * C_r}{n_{neg}} \tag{3}$$

Where, n_{neg} denotes number of negative ratings, $\mathrm{FDB_{neg}}$ (C,RP) denotes past negative feedbacks from consumer regarding RP, $\mathrm{FDB_{curr}}$ (C,RP) denotes current feedback from consumer regarding RP and C_r denotes the criticality of the given transaction.

The new Trust-Index of the consumer, $\mathrm{TIX(C)_{new}}$ will be:

$$\mathrm{TIX(C)_{new}} = \mathrm{TIX_{old}(C)} - r_1 \tag{4}$$

Where, $\mathrm{TIX_{old}(C)}$ denotes the old trust-index of the consumer and r_1 denotes the reduction factor.

The trust value of the RP and the consumer needs to be updated at all the brokers' site. Trust-index of the brokers, information about all RPs and the feedback information about the RPs are stored at the broker's site securely.

Identification of dishonest feedback/rating: In our model we assume the brokers to be trustworthy since they are responsible for managing the trust indices of the entities. The way to identify false feedbacks can be by trying to find the inconsistency between a rater's actual ratings and his rating habit (Xiangli *et al.*, 2006). Here, by dishonest ratings we denote those ratings as intentionally exaggerating or badmouthing the facts. But for those unintentionally introduced deviations such as ratings from lenient raters tend to be a bit higher while from strict raters tend to be a bit lower, we name this as forgivable errors and will not filter them out. We will refer to these forgivable errors in detail.

We will utilize the model of a pre evaluating set suggested in Xiangli *et al.* (2006). A pre-evaluating set is a set of scenarios describing an entity's behavior in a transaction, including trustworthy, untrustworthy and in-between behaviors. There is a pre-evaluating set in each domain in the whole Grid infrastructure.

There are two phases for detecting false feedback namely, Credibility Filtering and On-Spot Filtering.

The algorithm for Credibility Filtering is as follows:

- First get 4 rating sets: Ratings given by e_i(consumer) and the current evaluator(broker) to the common pre-evaluating set and the intersection set.
- Then get 2 deviation sets D_{pre} and D_{trans}: D_{pre} is got bycomparing e_i's ratings to the pre-evaluating set with the current evaluator's corresponding ratings and D_{trans} is got by comparing e_i's ratings to the intersection set with the current evaluator's corresponding ratings.
- Form a deviation space of [0,1] and divide this deviation space into 5 intervals of:
  [0-0.2], [0.2-0.5], [0.5-0.7], [0.7-0.9], [0.9-1.0] and calculate the probability that elements in D_{trans} and D_{pre} falling into the above n intervals, respectively, denoted by $p_{i,trans}$ and $p_{i,pre}$ (i=1,2...n).
- Find the difference $diff(p_i)=| p_{i,trans} - p_{i,pre}|$
- Calculate the final Distribution Deviation Degree (DDD) between D_{trans} and D_{pre}, that is DDD= $\sum w_i * diff(p_i)$ where w_i are customized weights for the 5 intervals.
- The credibility is calculated as 1-DDD

The algorithm for On-Spot Filtering is as follows:

- Let there be a set of entities $e_1,e_2,\ldots\ldots e_n$ and let the entity to be evaluated be e_k
- Now assume a set of ratings given by these entities to a common pre-evaluating set,
  $R_1= \{e_{<1,pre>},e_{<2,pre>},\ldots\ldots.e_{<n,pre>}\}$
- The set of ratings given by these entities to e_k will be $R_2= \{e_{<1,k>},e_{<2,k>},\ldots\ldots.e_{<n,k>}\}$
- Calculate the average of the ratings in set R_1 as avg(PRE).
- Now calculate the difference between each entity's rating in R_1 and the avg(PRE) as diff(PRE).
- Form a deviation space of [0,1] and divide this deviation space into 5 intervals of:
  [0-0.2], [0.2-0.5], [0.5-0.7], [0.7-0.9], [0.9-1.0]
- Now calculate the probability that each entity's diff(PRE) falls into the 5 intervals.
- Modify ratings given by the n raters to e_k by multiply their first-hand ratings by the credibility got in credibility filtering.
- Calculate the average of the modified ratings in set R_2 as avg (EVA).
- Calculate the difference between each entity's rating in R_2 and the avg(EVA) as diff(EVA).
- Calculate the probability that each entity's diff(EVA) falls into the 5 intervals. If the values, diff(PRE) and dif(EVA) do not fall in the same interval, then the rating given by that entity to e_k can be termed as dishonest.

Now after the genuineness of the feedback is checked, the trust-indices of the consumer and the RP are updated. And the consumer's trust-index will be incremented as an incentive if his feedback is found to be genuine. The incement factor will promote the submission of genuine feedbacks by the consumers.

We propose this increment factor, I which is calculated as follows:

$$I=FDB_{curr}(C,RP)*\frac{\sum FDB_{pos}(C,RP)}{n_{pos}}*C_r \qquad (5)$$

Where, n_{pos} denotes number of positive feedbacks, FDB_{pos} (C,RP) denotes past positive feedbacks from consumer regarding RP, FDB_{curr} (C,RP) denotes current feedback from consumer regarding RP and C_r denotes the criticality of the given transaction.

Interpolation of true first-hand rating: The false feedbacks will not contribute to the computation of the trust-index of the RP. Hence, transactions which yield false ratings can be rendered invalid and trust-index of the participating RP after such transactions can be left unmodified. However, this can be unfair to the RP who had acted honestly in the transaction committed. So in cases where false feedbacks are detected, we implement one-dimensional linear interpolation techniques to predict a pseudo-feedback for the committed transaction. Interpolation will generate an output when a given set of discrete values are given as input. Again the pre-evaluating set plays a role here.

Suppose a consumer, e_i gives a false feedback to RP, e_j who has had a true first-hand rating from consumer, e_k in the past. The ratings given by the consumers, e_i and e_k to the pre-evaluating set will be the two input vectors to the linear interpolation method. Also the true first-hand rating given by e_k will be the third input to this method. We have used Matlab to perform the interpolation and also compared the RP's trust-index in the presence and absence of interpolation.

Also, we will employ interpolation techniques on the forgivable errors mentioned in study to bias such ratings. This is done to reduce the deviation between such error ratings and the actual ratings, had there been one. We will also compare the RP's trust computed by forgivable errors with that computed by biased forgivable errors i.e., biased errors.

MINIMISING MESSAGE OVERHEAD

When the feedback of every single transaction is verified, extra time is needed for the execution of the algorithm. The workload at the broker's end increases. The number of messages transferred increases and the total number of transactions committed in the grid over a period of time also decreases. Hence, to avoid this problem, the rate of verification of feedback is also related to the trust value of the consumer.

The feedbacks of every transaction committed will be verified for genuineness only for those consumers with trust values less than 0.5. However, for those consumers whose trust values lie between 0.5 and 0.8 and for those whose trust values lie above 0.8, the feedbacks will be checked every t_1^{th} and t_2^{th} transaction, respectively where $t_2>t_1$ since more trustworthy consumers deserve an incentive. This is called the feedback-checking mechanism to effectively reduce message overhead. In our model, we have simulated this proposal with $t_1 = 50$ and $t_2 = 80$ and obtained the results.

This message overhead or traffic due to large number of messages transferred will reduce further if this rate of feedback-checking is increased. But the risk of having false feedback going undetected is there and this risk increases as the rate of feedback-checking increases, which is unadvisable.

EMPLOYING FUZZY LOGIC FOR TRUST VERIFICATION

In this study, we have deployed fuzzy logic to implement memberships of the trust-index parameter in vaguely defined sets. In its linguistic form, fuzzy logic has imprecise concepts like low, slight, medium, etc... Specifically, it allows partial membership in a set. We have used Mamdani-type inference as defined for fuzzy-logic toolbox as our fuzzy inference method because of its widespread use and ease of implementation. Its fuzzy inference rules are some if-then rules to define output behavior from the inputs.

By using the Membership Function (MF), the input values are mapped onto an output fuzzy set. Defuzzification is then done to obtain a single discrete value from the range of values obtained via fuzzification. A Membership Function (MF) is a curve that defines how each point in the input space is mapped to a membership value (or degree of membership) between 0 and 1. Gaussian function (gaussmf) is chosen for representing the membership since Gaussian distribution is identical to normal distribution.

SIMULATION

We simulated the architecture with 25 domains, 80 to 120 brokers, 160 consumers and 160 RPs. We maintained five different resources with fixed criticality rates (ranging from 0.1-0.9) for the purpose of simplicity; however this can be easily extended without losing generality.

Broker simulator: This module simulates the work of a broker.

```
While (true)
{The broker has a thread running for accepting the requests from the consumers and RPs.
The request for joining a domain can come from a RP/consumer.
The request for a service can come from a consumer.
And the appropriate RP is selected and the resource is given to the consumer.
It then accepts the feedback from the consumer, checks it for genuineness.
And it recalculates the trust of RP and consumer.
The trust values are updated to all the brokers' databases to which the RP is associated.}
```

Consumer simulator: This module simulates the work of the consumer.

```
While (true)
{There is one thread (for each consumer) running.
It sends a request to join a domain.
After this, it sends its service requests to the broker to which to which it's attached.
After the transaction, it again sends the feedback of the RP to the broker}.
```

Resource provider simulator: This module simulates the work of the RP.

```
While (true)
{There is one thread (for each RP) running.
It sends a request to join a domain.
After this, it waits continually to accept the request of a consumer.
It services the request and returns the result back to the broker.}
```

RESULTS

The varying trust-indices of the consumer are plotted against the no. of transactions in Fig. 2.

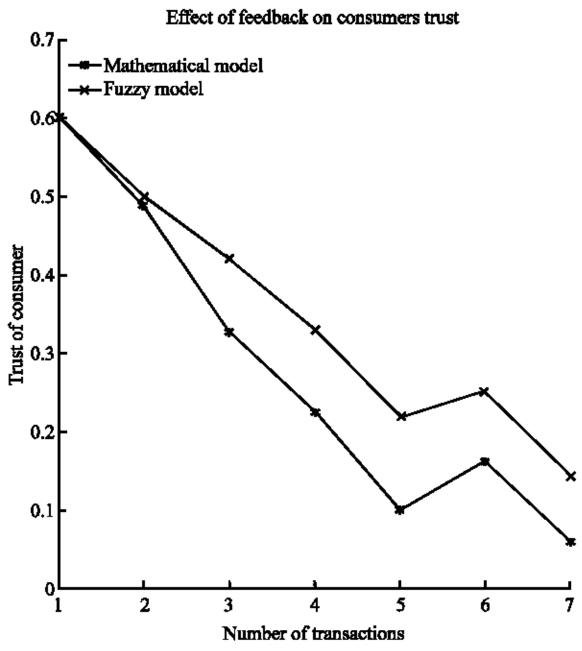

Fig. 2: Effect of feedback on consumer's trust-index

The change in the trust-index (after the genuineness of the feedback is verified) is calculated with both the mathematical formulae (probability-based) and the fuzzy rules of fuzzy logic model. And the variations in the trust-indices are moderate in case of our fuzzy model which is beneficial to the consumer. The dynamically varying RP's trust-index is plotted against the no. of transactions in Fig. 3. Here we have shown the differences in the RP's trust with 3 cases namely: RP's trust if false feedbacks are not detected and eliminated, RP's trust if false feedbacks are eliminated and the resulting transaction rendered invalid and RP's trust if interpolation techniques are implemented to predict some feedback (highly close to the true feedback, had there been one). And the observation made is that using interpolation is more efficient to safeguard the RP's trust-index and award the RP with its most deserving trust-index even when the consumer acts maliciously.

The trust value of the RP is plotted against the number of transactions in Fig. 4. Here we have considered a stress test model where the consumer is a lenient one. So his lenient ratings will not be termed as false feedback. Instead it will be "forgivable errors". And the effect of these errors on the RP's trust is plotted in comparison with the effect of biasing of these errors on the RP's trust. This biasing is also done with interpolation. The number of transactions is plotted against the time scale in Fig. 5.

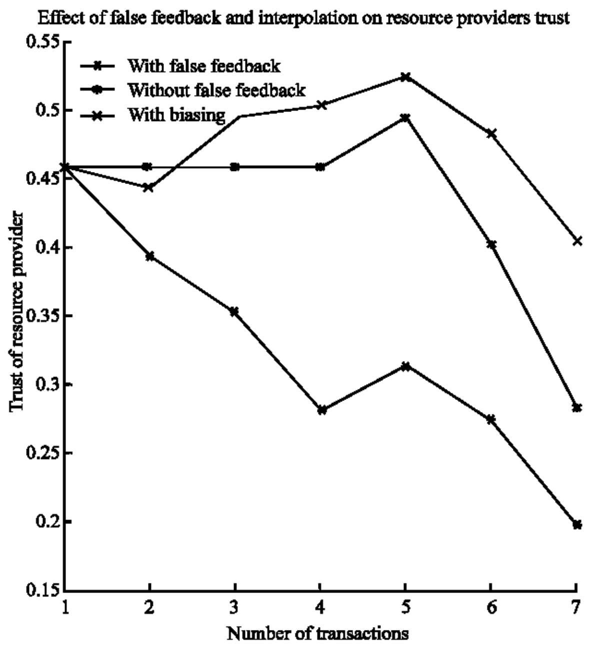

Fig. 3: Effect of feedback and interpolation on RP's trust value

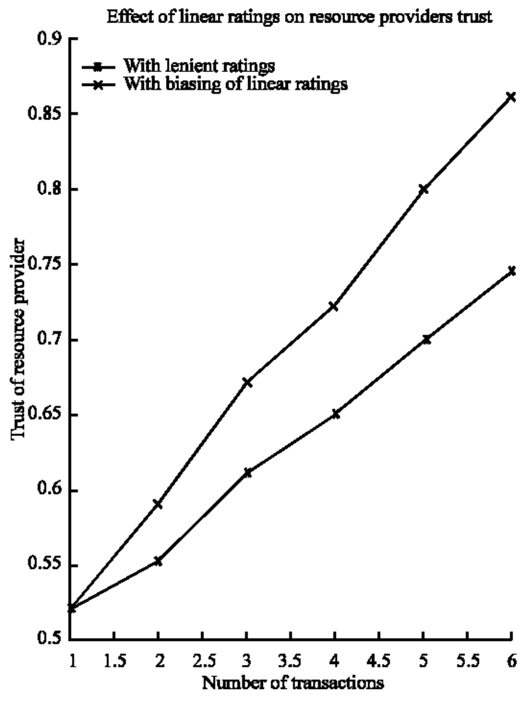

Fig. 4: Effect of linear ratings on RP's trust value

Here we have simulated how message overhead can be effectively reduced with our proposal. And it can be clearly seen that our proposal helps increase the number of committed transactions in a given time.

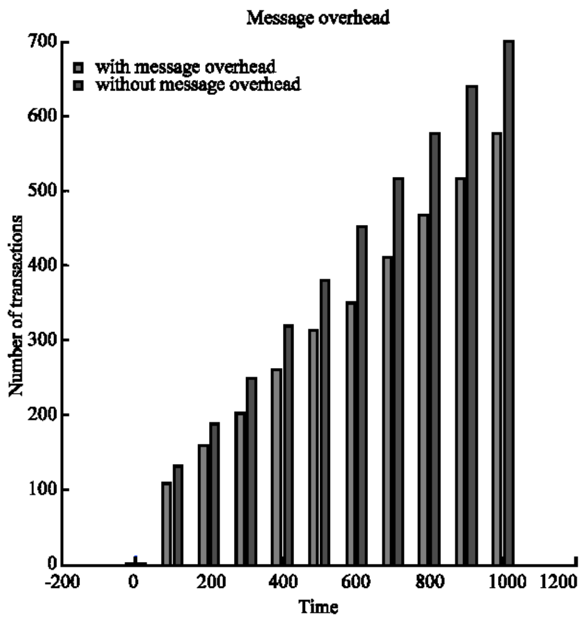

Fig. 5 Effect of our proposal on message overhead

CONCLUSION

We have implemented the algorithm suggested by Qu *et al*. (2006) for false feedback identification. But we have extended it by showing how deploying fuzzy logic helps in the determination of trust-indices of the entities. This is because the fuzzy model does not react very sharply to the input values as our mathematical model (probability-based).We have also proposed the reduction factor and increment factor to determine the punishment and incentive deviation deserving to the candidates irrespective of their trust-index. Also we have employed linear interpolation to predict feedback values that helped update the RP's trust-index correctly instead of considering transactions with false feedback as invalid. Linear interpolation has been also deployed to bias-tune the lenient ratings. Hence, biasing has improved the final trust-index of the RP. Also, we have proposed a mechanism to effectively reduce message overhead and simulated our proposal to prove the same.

REFERENCES

Azzedin F. and M. Maheswaran, 2004. A Trust Brokering System and Its Application to Resource Management in Public Resource Grid, Int. Parallel and Distributed Computing Symposium (IPDPS).

Farag Azzedin and Muthucumaru Maheswaran, 2002. Integrating Trust into Grid Resource Management Systems, in International Conference on Parallel Processing.

Jason, D. Sonnek and B. Jon Weissman, A Quantitative, 2005. Comparison of Reputation Systems in the Grid, 6th IEEE/ACM International Workshop on Grid Computing.

Kwei-Jay Lin, Haiyin Lu, Tao Yu, Chia-en Tai and Jane Yung-jen Hsu, 2005. A Reputation and Trust Management Broker Framework for Web Applications, Proceedings of the IEEE International Conference on Etechnology, E-Commerce and E-Service. Hong Kong.

Thanasis G. Papaioannou and D. George Stamoulis, 2005. An Incentives Mechanism Promoting Truthful Feedback in Peer-to-Peer Systems, Proceedings of the 5th IEEE International Symposium on Cluster Computing and the Grid (CCGrid).

Varalakshmi, P., S. Thamarai Selvi and K. Narmatha, 2006. A Broker Architecture Framework for Reputation-based Trust Management in Grid, Proceedings of International Conference on Information Security ICIS.

Woodas, W.K., Lai, N.G. Kam-Wing and R. Michael Lyu, 2004. Integrating Trust in Grid Computing Systems, in Grid and Cooperative Computing-GCC.

Xiangli Qu, Xuejun Yang and Jingwei Zhong, 2006. Towards Reliable Trust Establishment in Grid: A Pre-evaluating Set based Reputation Evaluation Approach , Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid (CCGRID).