Delegation Protocols Founded on Trust

Rachil Chandran[a], Henry Hexmoor[b]
[a] Department of Computer Science and Computer Engineering
University of Arkansas, Fayetteville, AR 72701
chandra@uark.edu
[b] Department of Computer Science
Southern Illinois University at Carbondale, Carbondale, IL 62901
hexmoor@cs.siu.edu

Abstract- *Delegations are the basic mechanism through which agents in a multiagent system coordinate. Delegations are also a major source of insecurity in a multiagent system. In this paper, we outline delegation protocols based on trust. Through these protocols, we address the problem associated with delegation decisions in a multiagent system.*

1. INTRODUCTION

Any computer program that is capable of autonomous behavior is termed as an agent [1]. Such an agent is said to represent a user and operate on the user's behalf to achieve certain goals. A multitude of such agents coexisting together gives rise to a multiagent system. The primary aim of such a multiagent system is to achieve, through collaboration among its members, goals and tasks that are difficult for a single agent to attain. The collaboration in a multiagent system usually takes the form of delegation from one agent to another.

The act of handing over an object by one agent to another is known as delegation [2, 3, 13]. The object that is being delegated can be a task, a role or a goal which, in turn, is composed of multiple tasks [4, 11]. The entity that initiates the transfer is known as the delegator and the entity to which it is transferred is termed as the delegatee [5]. The delegatee agent is expected to complete the delegated object to the satisfaction of the delegator agent.

In a multiagent system composed of disparate self-interested agents, delegation is the primary mechanism of inter-agent collaboration and cooperation. By disparate, we mean that the

agents are constituently different. They are designed by and act on behalf of different users. In other, words these agents do not form part of a team. Unlike a team of agents that have a common goal, each of these agents has its own individual goal that they were designed to achieve. Hence, we classify the agents as self-interested i.e. the agents are primarily interested in their own goal. In such an environment, an agent acts as a delegatee and cooperates with delegator only in lieu of an incentive which is termed as the cost of delegation [6]. Since the agents are disparate and self-interested, there can be no assumption made about the benevolence of such delegatee agents towards delegator agents [6].

Delegations in environments described above can be a major source of insecurity. Wong and Sycara identified delegations as one of the four main sources of insecurity in a multiagent system [7]. A malicious delegatee can breach information and resources associated with a delegated object. Even a benevolent agent may fail to complete a delegated object satisfactorily due to lack of ability to do the same. Hence, the problems associated with delegations in multiagent systems with disparate self-interested agents are two fold. One is that of potential insecurity and the other, the probability of failure of delegation. In either case, we consider the delegation as one that was not completed to the satisfaction of the delegator.

Having defined the difficulties associated with delegations among disparate self-interested agents, we now define the *delegation decision problem*. In an open environment such as the one discussed in the above paragraphs, the entities are customarily alien to one another. In other words, a given agent is usually not acquainted with another's ability (to complete a delegated object) or reliability (in terms of benevolence). In such an environment, how does a delegator choose a delegatee from a given set of potential options? Hence the *delegation decision problem* is a selection problem: A problem of selecting a suitable delegatee with respect to ability and reliability.

We propose to solve the *delegation decision problem* by using a rating system where the delegatee for a particular delegation is chosen based on some parameter. If the

1-4244-0945-4/07/$25.00 ©2007 IEEE

rating system could indicate the delegatee's ability and reliability, the delegator could then select a delegatee who has a high probability of completing the delegated object in a reliable, satisfactory manner. Although there has been a lot of research in the area of delegations, none attempt to address the delegation decision problem [14, 15].

In this paper, we propose using the notion of *trust* as a basis for delegation decisions. Trust is defined as the quantified belief of one entity towards another with respect to honesty, competence and dependability within a specified context [8, 12]. Trust also signifies the trustor's estimate of risk involved when transacting with a trustee [6]. Trust, when used as the determining factor for delegation decisions, can alleviate the problems related to delegations in multiagent systems. A delegatee that is trusted to a greater degree by the delegator would have a higher probability of completing the delegation to the latter's satisfaction than a delegatee with a lower trust. Hence, we opine that trust, to a great extent de-alienates potential delegatees and delegators i.e. delegtors gets more acquainted with the delegatee's benevolence and ability with respect to delegations. The value of trust we employ in our work is primarily experienced based trust [6]. However, we also propound the notion of pseudo-experience based trust as well.

We present our work as a model of delegation based on trust. It should be noted that there can be any number of factors influencing delegations decisions. Some important ones are the cost of delegation, availability of resources and knowledge (on the part of the delegatee) required to complete the delegation etc. For the length of our work, however, we do not consider these other factors. We focus on delegations based solely on trust. The model presented herein can be easily extended to include other aspects associated with delegation. In the next section we define and describe the various ideas and protocols that constitute our model for delegation decisions.

2. TRUST BASED DELEGATION PROTOCOLS

As delineated earlier, we propose to base our delegation decisions on trust. Hence, the trust value should be sufficiently accurate in indicating the ability and benevolence of the concerned agent. An efficient trust updating technique would result in more delegations being successful. In this section, in addition to defining the various ideas we propose through our model, we outline various trust updating techniques which in our opinion augments to the accuracy of trust value as far as delegations are concerned.

Delegation Group: A *delegation group* is a collection of agents that are interested in either delegating or receiving delegations. In other words the member agents of a delegation group are potential delegators, delegatees or both.

The primary purpose for the existence of a delegation group is to provide a forum for the agents to get acquainted with each other by building trust. Trust values thus acquired help make delegation decisions. It should be noted that the term *group* has special significance in the context of a multiagent system. The cohesive force between the agents of a group is neither as strong as that in a team of agents nor as weak as that in a crowd [9]. The cohesive force in a delegation group arises from the common interest of its member agents in delegation. A delegation group can contain any number of agents that are delegators/delegatees or both.

Delegation Harmony: *Delegation harmony* or DelHarmony is a novel idea we present through this work. DelHarmony indicates the degree or extent of accord between two agents with delegations in context. It reflects the consistency of a delegatee agent in completing delegations from a given delegator's perspective. Hence, DelHarmony is a quantified notion that a given agent (a delegator) has towards another (delegatee).

DelHarmony is expressed as a real number with a minimum of zero. It is not bounded by a maximum value. We represent the value of DelHarmony by using *DH (A, B)*, which denotes the DelHarmony that agent A has towards agent B. Hence DelHarmony is always specified with respect to two specific agents. It should also be noted the DelHarmony is an asymmetrical notion i.e. DH (A, B) need not be the same as DH (B, A).

We formulate DelHarmony as follows:

$$DH(A, B) = \frac{HD(A, B)}{DR(A, B)} \times \log_{10} HD(A, B)$$

Where, HD (A, B) is the number of honored delegations with agent A as the delegator and B as the delegatee and DR (A, B) is the total number of delegation requests from A (again as delegator) to B (delegatee). By number of honored delegations, we mean the number of delegations where the delegator, in this case A, was satisfied with the outcome of the delegation both with respect to the outcome of the delegation itself and the absence of events that compromise security of information and resources associated with the delegation.

The term *HD (A, B) / DR (A, B)* represents the fraction of delegations from agent A to B that was successful. This fraction, however, would have the same value for a delegatee that has satisfactorily completed only 2 delegations and a delegatee that has satisfactorily

completed 100 delegations (the fraction in both cases is 1.0 as all delegations are deemed to be completed satisfactorily). Hence, the fraction does not reflect any consistency on the part of the delegatee. To take account of consistency, we introduce the $log_{10} HD (A, B)$ expression in our formula of DelHarmony. We identify this expression as the *longevity factor*. The longevity factor serves to indicate the length of delegation interactions between the given delegator and delegatee. Hence, an agent that has satisfactorily completed a certain fraction of a higher number of delegations would have a higher value of DelHarmony associated with it than an agent with the same fraction of successful delegations over a relatively lesser number of delegations. For instance, citing the above comparison of delegatees with 2 out of 2 and 100 out of 100 successful delegations, the former would have a DelHarmony value of .6021 and the latter would have a value of 200 using the aforementioned formula.

It should be noted that any suitable function can be used in the longevity factor expression in place of the log base 10 proposed by us. We found the log base 10 function to have the right degree of sensitivity in reflecting the consistency of a delegatee. By suitable, we imply that the function used should increase at a reasonably slow rate. This would ensure that the value of DelHarmony does not radically change for each successful delegation. As we will discover in the following paragraphs, the innovative concept of DelHarmony is used to lend a dynamic approach to trust updating techniques. A drastic difference in the DelHarmony values of agents slightly differing in performance would, in turn, have a drastic effect on the values of trust as well.

Trust: The trust value is the primary dynamic, based on which delegation decisions are predicated in our model. Since trust value indicates probability of success in a transaction, it is natural to the base our delegation decisions on this notion [10].

We denote trust value that an agent A has towards agent B by $T (A, B)$. Trust, like DelHarmony is quantified as a real number and does not have a maximum limit for its value. However, unlike DelHarmony, the value of trust can also take a negative value. Trust is also an asymmetric entity. T (A, B) need not be the same as T (B, A).

The trust value we use in our model is primarily experience based i.e. the trust value that one agent has in another reflects the experience that the former had while interacting with the latter. To reflect the experience in delegations that a delegator has with respect to a delegatee, we update the value of trust (towards a delegatee) after every delegation (again, to that particular delegatee). Delegations completed to the satisfaction of the delegator are treated as a positive experience and conversely, negative experience comprises of delegations that were not completed at all or completed with some breach in security. Accordingly, trust values are incremented for every successful delegation and decremented for delegation failure. The values by which the trust values are updated after a delegation is termed as *trust incentives* and *penalties*. Below, we explain trust incentives and penalties in greater detail.

Trust Incentive and Penalty: The value by which a delegator increments its trust value towards a delegatee (for successful completion of delegation by the latter) is known as the *trust incentive*. In case of a delegation failure, the delegator decrements its trust towards the involved delegatee by a value termed as *trust penalty*.

The trust incentive and penalty are generic terms and do not represent a value. Instead, the actual values of incentive and penalty are dependant on the two agents involved and we adopt a dynamic approach to compute the same. This is one instance where the notion of DelHarmony lends a dynamic nature to our protocol. An agent with a high value of DelHarmony can be identified as one that has been more consistent than an agent with comparatively lower value of DelHarmony. We contend that the incentive for a more consistent agent should be higher and its penalty lower. The reasoning behind the statement is the fact that a rare failure in an otherwise consistent agent should be given a lesser influence on the trust value of that particular agent. On the other hand the incentive for a less consistent agent should be lower than that of a highly consistent agent. Again, the reasoning is parallel to the one above. Trust value of an agent who has not been very consistent should not be increased by a high value for a relatively less frequent delegation success. Since DelHarmony represents consistency, we propose that trust incentive be directly proportional to DelHarmony and penalty be inversely proportional to DelHarmony.

Using the aforementioned ideas and the notion of DelHarmony, we formulate trust incentive (denoted by $\alpha (A, B)$) and penalty ($\beta (A, B)$) as follows:

$$\alpha (A,B) = \omega \times DH (A,B)$$

and

$$\beta (A,B) = \omega / DH (A,B)$$

Where DH (A, B) is the DelHarmony between agent A and agent B, ω is the weight of delegation. The weight of delegation ω is nothing but the importance of a delegated object. We take a stratified approach to delegation

importance. Hence, ω can take values of 0.25, 0.5, 0.75 and 1.0. A value of 0.25 indicates delegation that is not very important or one that does not require information or resources that can be breached. A value of 1.0 indicates a crucial delegation. For instance, if the object being delegated is a goal comprising of multiple tasks and requiring access to multiple resources and information, a weight of 1.0 can be assigned to it. As can be perceived from the above formulae, an agent performing well and not breaching security for a very high risk delegation gets more incentive. Conversely, an agent failing in a similar scenario would incur a heavier penalty.

Once the values of incentive or penalty have been determined, the trust value for a delegatee can be updated to reflect the outcome of a delegation. For a successful completion of delegation by agent B, the delegator A updates the trust as

$$T(A, B) = T(A, B) + \alpha(A, B)$$

In case of a failure, the penalty value is deducted from trust value as follows

$$T(A, B) = T(A, B) - \beta(A, B)$$

Depreciation of Trust: Trust is a dynamic, temporally changing notion. For the same reason, an old value of trust may be outdated in the sense that there might be changes in an agent's ability, benevolence or both. To maintain the accuracy of trust in situations where there has been a period of inactivity between and given delegator and delegatee, we depreciate the trust value. Depreciation of trust constitutes a decrease in the value of trust.

The value by which trust depreciates is dependent on the rate of depreciation $\tau(A, B)$, and the time that has elapsed since the last interaction. Here again, we adopt a dynamic DelHarmony based approach to computing the rate of depreciation. The probability in the change of benevolence and ability of an agent that has been consistent for a long period of time is lesser than the probability of a less consistent agent's benevolence and ability changing. To justify the same, the decrease in trust of former type of agent should be lesser for a given interval of time and the decrease in trust for the latter type should be more for same amount of inactivity time. Since, for a certain period of time, the rate of depreciation determines the decrease in trust, and DelHarmony indicates consistency, the rate of depreciation is made inversely proportional to the DelHarmony.

We formulate the rate of depreciation as:

$$\tau(A, B) = 1 / DH(A, B)$$

Where DH (A, B) is the DelHarmony that agent A has towards B. Hence the rate of depreciation is specified with respect to a delegator and delegatee.

Once the rate of depreciation has been calculated, the trust value can be decreased by the *depreciated trust*. The depreciated trust can be computed as follows

$$DT(A, B) = \tau(A, B) \times \text{time units elapsed since the last interaction between A and B}$$

The trust value $T(A, B)$ is then decreased by $DT(A, B)$ and the value thus updated gives the new value of trust.

$$T(A, B) = T(A, B) - DT(A, B)$$

The Subscriber Model: As mentioned earlier, the trust we employ to predicate our delegation decisions is primarily an experience based one. In this section we introduce a pseudo-experience based trust approach in addition to the experience based trust. In the subscriber model, an agent is able to update its trust towards another agent without actually delegating to that delegatee.

Whenever an agent's trust towards another agent is greater than what is termed as the *publisher threshold*, the former sends a request to the latter requesting to be its subscriber. We denote publisher threshold as δP. The publisher threshold is an agent level value. In other words, each agent has its own publisher threshold. The actual value that an agent chooses for its publisher threshold depends on the cautiousness of the agent. As an illustration of the subscriber model, consider an agent A with a certain value for publisher threshold δP. Let us say A's trust towards another agent B, T (A, B) increases above its δP. A then subscribes to B and the latter is called as the publisher.

Each time an agent delegates and has an outcome for that delegation, the agent (if it has any subscribers), in addition to updating its own DelHarmony and trust values, multicasts the outcome of the delegation and its importance to all of its subscribers. The subscribers then, in turn, update their own DelHarmony towards the delegatee involved. They then update their trust towards the delegatee based on the new DelHarmony.

The subscriber model presents a pseudo-experience trust approach as opposed to a recommendation based trust approach because the publisher just multicasts a plain fact (the result, importance and delegatee involved in a delegation) and the subscriber updates the trust value based on its own DelHarmony towards the concerned delegatee. We posit that this innovative approach to updating trust is desirable as the subscribers are able to

update their trust towards a delegatee without delegating objects themselves and possibly experiencing failure through it.

Delegation protocols: We now describe the series of steps that constitute a delegation cycle. Each agent in a delegation group is required to perform these steps sequentially. The algorithm for a delegation cycle is depicted in the below.

1: Compute depreciated trust and update trust for each agent.
2: Choose a suitable delegate.
3: Assign weight of delegation, delegate.
4: Update DelHarmony based on the outcome of delegation.
5: Update trust towards the delegatee based on latest DelHarmony.
6: Publish result.
7: Check if updated trust < elimination threshold, if yes initiate elimination against the delegatee.

Figure 1 - Algorithm for delegation cycle.

The algorithm outlined above is simple and intuitive. A potential delegator first computes the depreciated trust for all agents in the delegation group that it is considering for the delegation. It then updates the trust value of each to obtain the new value of trust based on which it predicates the delegation decision. It then delegates the object by assigning it a weight according to the object's importance. Once the delegation's outcome is available, the delegator updates the DelHarmony and trust for the delgatee to reflect the outcome of the delegation. The outcome and the importance of delegation is then published to all of the delegator's subscribers. Finally, the delegator agent checks the trust value of the delegatee to check if it is below the delegation group's elimination threshold, in which case, the delegator initiates elimination against the delegatee. We now describe the notion of elimination threshold and the elimination protocol.

Elimination Threshold: As elucidated in the delegation cycle algorithm, an agent whose trust value has fallen below a certain value is eligible to be eliminated from the delegation group. By doing this, a minimum level of quality can be maintained within a delegation group. This is another manner in which the idea of delegation group can be employed.

The minimum value of trust that all the agents in a group are expected to maintain is known as the *elimination threshold*. We represent elimination threshold by the symbol δE. It should be noted that the elimination threshold value, unlike the publisher threshold described earlier, is a group level value. Here again, our model espouses a dynamic approach. By a dynamic approach, we mean the value of elimination threshold is not a static, fixed value. The elimination threshold is dependent on the number of delegations or delegation cycles that an agent has performed. For instance the elimination threshold for a delegatee that has faced 500 cycles of delegation would not be the same as that for a delegatee that has performed only 10 delegations. We term this approach as a *cycle based elimination threshold*. The reason for accepting such an approach arises from the fact that an agent that has faced a large number of delegations should have necessarily have a higher value of trust than an agent that has performed a relatively lower number of delegations. Hence, the cycle based elimination threshold considers the opportunity that a delegatee has had, in terms of the delegations it has faced, to prove its ability and reliability. An illustration and detailed explanation of the elimination threshold is presented in the next section. Once the trust value of an agent towards other has decreased to a point below the elimination threshold, the former initiates elimination against the latter. In such a scenario, the former is called the *initiator of elimination* and the latter is termed as the *candidate of elimination*. Once the elimination has been initiated, all the agents in the delegation group except the initiator and candidate of elimination vote on the initiation. If majority of the group vote for the elimination, the agent in question is eliminated, else, it is reinstated in the group. The agents voting on elimination follow the elimination protocol described in the next paragraph.

Elimination Protocol: Each agent in the group decides to vote for the elimination or against it based on the *elimination protocol*. The protocol is outlined in figure 2.

1: Decide on the initiation based on the trust towards the candidate of elimination. If a low value of trust is perceived, vote in the affirmative.
2: If no trust has been established with the candidate, check the trust towards the initiator, vote in affirmative if a high trust value is detected.
3: If no trust value has been established even with the initiator, vote in accordance to the majority.

Figure 2 – Elimination Protocol

A decision on the elimination of the concerned agent is then made based upon the majority decision.

3. SIMULATIONS

A series of simulations were conducted as part of our investigation. The primary goal of the simulations was to illustrate the variation of the values of the different ideas presented herein. The graphs resulting from our simulations confirm our view on the working of the model. The simulations were coded using the Java 1.4.2 language. To understand the experiments, it is imperative to understand the idea of success rate.

Success rate corresponds to the ratio of the number of successful delegation to the total number of delegation performed by a given delegatee expressed as a percentage. We use this notion in our simulations to depict the variation in values for agents with different success rates. The success rates in the experiments are assumed to reflect the ability and dependability of agents and hence remain constant for the length of the experiment.

Our first experiment illustrates the building of DelHarmony for three agents (with respect to a common delegator). These three agents are assumed to have success rates of 100%, 80% and 50%. In other words the agents never fail, fail once in every five delegations and every other delegation respectively. Figure 3 shows the variation in the DelHarmony value for each of these three agents.

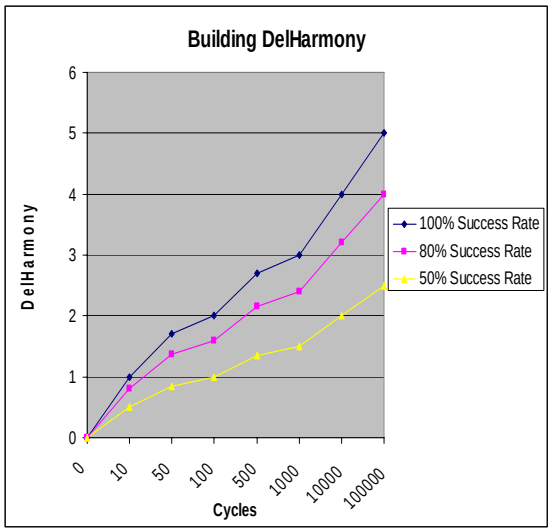

Figure 3 – Building delegation harmony

In the second experiment we plot trust values for the same three agents as in the previous experiment. As can be perceived from figure 4, the values of trust are more sensitive than the DelHarmony values. For instance, although we ran this experiment for 250 cycles as opposed to 100000 cycles in the DelHarmony experiment, the trust values for agents with 100% and 80% success rates differed by as much as 200, as compared to a difference of 1.0 between the DelHarmony values of the same two agents.

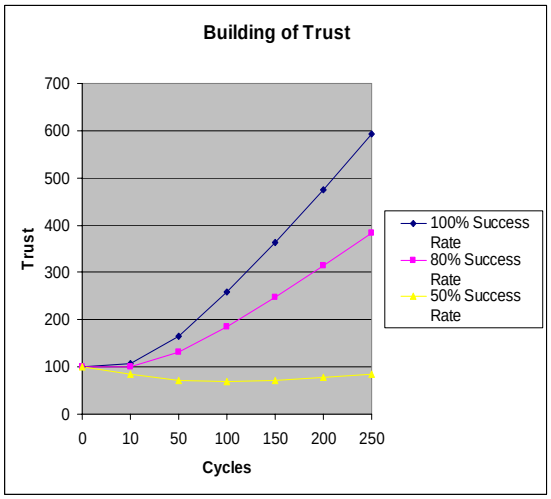

Figure 4 – Building trust

Hence, trust is sensitive enough to differentiate between two agents with different success rates even with a small number of delegations, whereas DelHarmony is a long term value and changes only slightly from one delegation to next. Therefore, we use trust to predicate our delegation decisions and use DelHarmony as the foundation for updating the trust.

In out next experiment, we show how the value of trust depreciates over time without interactions. Here again, we consider three agents that are assumed to have DelHarmonies of 4.0, 3.2, and 1.6. The slopes of the three lines represent the rate of depreciation for the three agents. As can be observed, the trust for an agent with a DelHarmony of 1.6 depreciates more than that of an agent with a DelHarmony of 4.0 for the same given inactivity period. The figure below depicts the results obtained.

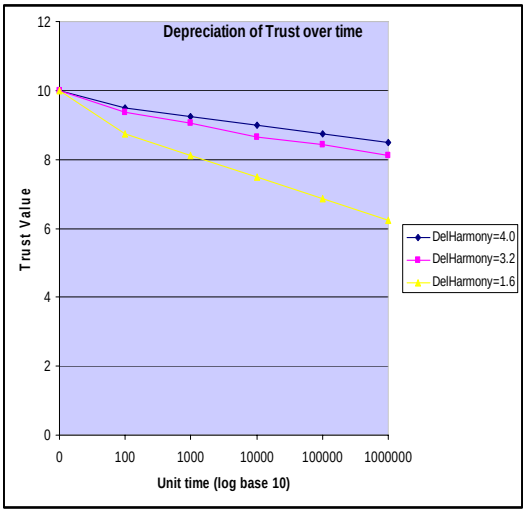

Figure 5 – Trust depreciation

We explore the agent elimination threshold in the next simulation. Figure 6 shows the values of trust that have to be chosen as elimination threshold (at a given cycle, since the threshold is a cycle based value) to maintain a 50% success rate in the delegation group. For example, the values of elimination threshold for an agent that has performed only 10 delegations would be around 82 whereas that for an agent exposed to 500 delegations would be 142. This graph illustrates the significance of the cycle based elimination threshold concept. To maintain a minimum success rate of 50% in the delegation group, an agent that has faced 500 delegations and has failed to attain a trust of 142, should be eliminated from the group.

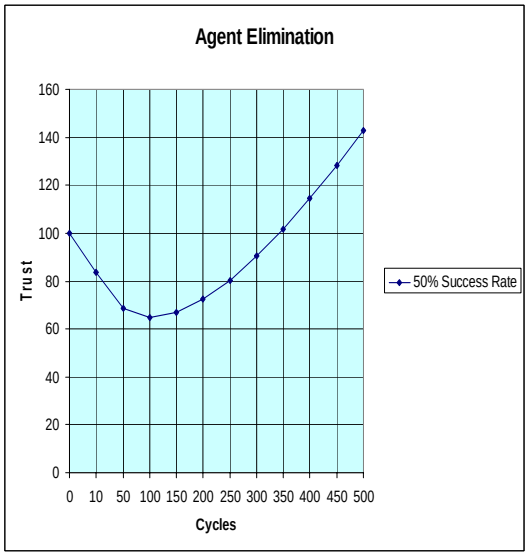

Figure 6 – Cycle based elimination thresholds for 50% success rate

Figure 7 shows the cycle based elimination threshold values for maintaining a minimum of 20% success rate in a group. Agents that perform poorly have a higher chance of getting eliminated from the group than those agents that have an average success rate (around 50%). As can be observed from the figure below, an agent with 20% success rate has a negative value for its trust even before the 25th delegation, even though the initial trust was assumed to be 100.

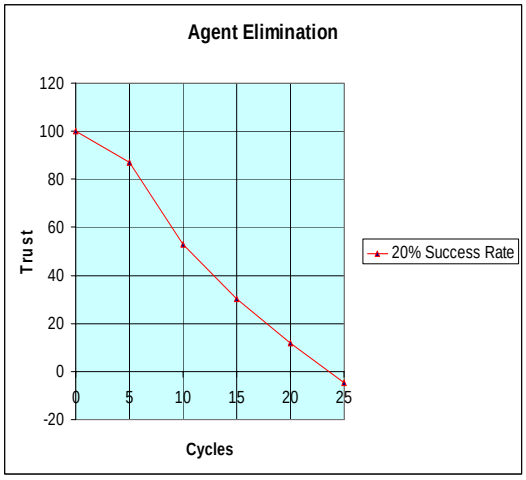

Figure 7 - Cycle based elimination thresholds for 20% success rate

4 CONCLUSIONS and FUTURE WORK

Ideas presented in this work augment trust values employed in rating based system involving delegations and beyond. Trust updating techniques we introduced constitute a dynamic approach. Updating procedures increase the precision of trust assessment and hence increase its utility and dependability.

The work presented herein can be extended in a number of different ways. There could be consideration of the cost of delegation, resource and knowledge availability on the part of the delegatee etc. in addition to the trust in predicating delegation decisions. The work could be extended to delegation chains and networks and inter-group delegation i.e. delegation that transcend the delegation group boundaries. We hope to address some of these areas in the future.

ACKNOWLEDGEMENTS

This work was supported by AFOSR grant FA9550-04-1-0429.

REFERENCES

[1] Stan Franklin and Arthur C. Graesser. "Is it an Agent, or Just a Program? A Taxonomy for Autonomous Agents", In the *Proceedings of the Workshop on Intelligent Agents III, Agent Theories, Architectures, and Languages*, pp 21-35, 1996.

[2] Abadi M., Burrows, M.,Lampson, B., and Plotkin, G. "A Calculus for Access Control in Distributed Systems." *Technical Report 70*, Digital Systems Research Center, February, 1991.

[3] Castelfranchi, C. and Falcone, R. "Towards a Theory of Delegation for Agent-based Systems." *Robotics and Autonomous Systems* 24 , pp: 141-157, 1999.

[4] Rachil Chandran. "Delegation Protocols Founded on Trust." *Masters Thesis*, Department of Computer Science and Computer Engineering, University of Arkansas, 2006.

[5] Ahsant, M., Basney, J. and Mulmo, O. "Grid Delegation Protocol", *UK Workshop on Grid Security Experiences*, Oxford, 2004.

[6] Griffiths, N. "Task Delegation using Experience-Based Multi-Dimensional Trust." In *Proceedings of the Fourth International Conference on Autonomous Agents and Multiagent Systems*, 489-496, ACM Press, 2005.

[7] Wong, H. C., Sycara, K. "Adding Security and Trust to Multi Agent Systems." In *Proceedings of Autonomous Agents'99 (Workshop on Deception, Fraud and Trust in Agent Societies)*, 1999.

[8] Grandison, T. and Sloman, M. "Specifying and Analysing Trust for Internet Applications." In *Proceedings of the 2nd IFIP Conference on E-Commerce, E-Business, and E-Government (I3e 2002)*, Lisbon, Portugal, pages 145-157, 2002.

[9] Silverman, B., Johns, M., Weaver, R., O'Brien, K., and Silverman, R. "Human Behavior Models for Game-Theoretic Agents: Case of Crowd Tipping." In *10th Conference on Computer SISO*, 2001.

[10] Castelfranchi, C. and Tan, Y. "Introduction: Why Trust and Deception are Essential for Virtual Societies." In C. Castelfranchi & Y.Tan (Eds.), *Trust and Deception in Virtual Societies*, Kluwer Academic Publishers, 2001.

[11] Castelfranchi, C. and Falcone, R. "From Task Delegation to Role Delegation." In *Proceedings of the 5th Congress of the Italian Association for Artificial Intelligence on Advances in Artificial Intelligence*, 1997.

[12] Beavers, G. and Hexmoor, H. "Understanding Agent Trust." In *Proceedings of The International Conference on Artificial Intelligence (IC-AI 2003)*. pp: 769-775, 2003.

[13] Hardjono, T., Chikaraishi, T. and Ohta, T. "Secure Delegation of Tasks in Distributed Systems." In *Proceedings of the 10th International Symposium on the TRON Project*, Los Alamitos, California, USA, 1993.

[14] Hu, J. H. "Some Thoughts on Agent Trust and Delegation." In *Proceedings of the Fifth International Conference on Autonomous Agents*, Montreal, Quebec, Canada, 2001, pp 489-496, 2001.

[15] Norman, T. J. and Reed, C. A. "A Model of Delegation for Multi Agent Systems." In M. d'Inverno, M. M. Luck, M. Fisher & C. Preist (editors), *Foundations and Applications of Multi Agent Systems*, volume 2403 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag, pages 185-204, 2002.