Designing Incentives for Peer-to-Peer Routing

Alberto Blanc
Electrical and Computer Engineering Dept.
University of California, San Diego

Yi-Kai Liu, Amin Vahdat
Computer Science and Engineering Dept.
University of California, San Diego

***Abstract*— In a peer-to-peer network, each node is typically required to route packets on behalf of other nodes. We study the problem of designing incentives to ensure that nodes carry out this responsibility. We model the interactions between nodes as a "random matching game," and describe a simple reputation system that can provide incentives for good behavior. Using simulations, we investigate the robustness of this scheme in the presence of noise and malicious nodes, and we attempt to quantify some of the design trade-offs.**

I. Introduction

Peer-to-peer networks suffer from the problem of free-loaders, users who consume resources on the network without contributing anything in return. Originally it was hoped that users would be altruistic, "from each according to his abilities, to each according to his needs." In practice, however, altruism breaks down as networks grow larger and include more diverse users. This situation can lead to a "tragedy of the commons," where the individual players' self-interest causes the system to collapse.

This paper focuses on a special version of the free-loader problem which arises in peer-to-peer routing. Each node in the network relies on other nodes to forward its requests, and it in turn is expected to forward the requests sent by other nodes. However, a self-interested user might choose to free-load by refusing to forward requests, conserving local bandwidth.

To reduce free-loading, the system as a whole must provide incentives for behavior that maximizes aggregate utility while delivering acceptable payoffs to individual users. This paper investigates one such scheme, using tools from game theory. First, we model a peer-to-peer network using a random-matching game. This game was previously studied by Kandori, who showed that, if there is a simple reputation system, then cooperation can be sustained as a robust and subgame-perfect equilibrium [1]. We extend Kandori's result by showing that this equilibrium can tolerate malicious nodes and noise in the system.

We then consider a more complicated version of the random matching game that models peer-to-peer routing. Under certain assumptions, we again get a subgame-perfect equilibrium. We use simulations to show that cooperation in this game can be sustained in the presence of malicious nodes and noise. We also attempt to quantify some of the design trade-offs; in particular we demonstrate that the reputation system can still be effective even if it only monitors a small fraction of routing events. A key contribution of this paper is quantifying the necessary effectiveness of a reputation scheme to enforce the incentives required to deliver high levels of global aggregate utility. Our initial results are encouraging, with implications for the design of such a practical system.

Section II briefly describes the basic random-matching game and the equilibrium strategy; for more details, see the companion technical report [2]. Section III describes the peer-to-peer routing game, and section IV presents simulation results for that game. Section V discusses some open issues and related work. Section VI concludes the paper.

II. The Random Matching Game

A. The Random Matching Game

Consider a generic peer-to-peer network. We use the Prisoner's Dilemma to model the exchange of resources between two nodes: each node makes a request, and cooperation consists of servicing the other node's request. (This describes a symmetric exchange; later, in the peer-to-peer routing game, we will allow asymmetric exchanges.)

The main difficulty with peer-to-peer networks is that users do not form long-lived relationships with other users, so strategies like "tit-for-tat" do not work. The common case is to interact with a stranger, with no prior history and no expectation of meeting again in the future. We model this using Kandori's random matching game [1]: In each round, nodes are randomly matched, and then each pair plays a (single-round) Prisoner's Dilemma. For simplicity, we will do matchings between the left and right vertices in a complete bipartite graph. We do allow non-uniform random matching.

B. A Simple Equilibrium Strategy

We now describe a simple strategy for the random-matching game and prove that it is a subgame-perfect equilibrium. This result is due to Kandori [1]. In this paper we refer to it as the "social norm" strategy.

Each node has a reputation consisting of a number in the range $\{0, 1, ..., \tau\}$; 0 means innocent, nonzero means guilty. The reputations are maintained by a trusted authority, who observes the players' actions and updates their reputations accordingly. Essentially, the reputations ensure that a node who defects will be punished in the next round, even though

E-mail: alberto@cwc.ucsd.edu
E-mail: y9liu@cs.ucsd.edu. Supported by the National Security Agency (NSA) and Advanced Research and Development Activity (ARDA) under Army Research Office (ARO) Grant No. DAAD19-01-1-0520.
E-mail: vahdat@cs.ucsd.edu. Supported by NSF grant ITR-0082912, NSF CAREER award CCR-9984-328, Hewlett-Packard, IBM, Intel and Microsoft.

it plays a different opponent in each round. The strategy is as follows:

- If the two players are innocent, they both cooperate.
- If the two players are guilty, they both defect.
- If one is innocent and one is guilty, then the guilty player cooperates, and the innocent player defects.

Any deviation from the above strategy triggers a punishment that lasts for τ rounds. That is, the offending node is marked "guilty," causing it to be punished by its opponents. After τ rounds, the node becomes innocent again, provided it has followed its assigned strategy. If a node deviates during the τ-round punishment phase, the punishment is re-started from the beginning.

Kandori showed that the social norm strategy is a subgame-perfect equilibrium, provided that we set the punishment length τ and the discount factor δ correctly. (Roughly speaking, the discount factor measures the "patience" of the players.) The proof that this strategy is an equilibrium can be found in [1].

C. Tolerating Malicious Nodes

We were able to extend Kandori's analysis to look at the effect of malicious nodes, i.e., nodes that always defect. Provided that we use uniform random matching, we find that the incentives and the global efficiency decrease gradually, as the fraction of malicious nodes increases. This work is described in the technical report [2].

D. Simulations of the Random Matching Game

We ran simulations to measure the effects of malicious nodes and noise in the random matching game. We found that the social norm equilibrium is robust to small fractions of malicious nodes and low levels of noise. In particular, there is a trade-off between using harsher punishments so as to tolerate malicious nodes, and using milder punishments to tolerate noise. We also tried to test the stability of the social norm in an evolutionary game, but got mixed results. These results are described in the technical report [2].

III. The Peer-to-Peer Routing Game

To study the problem of peer-to-peer routing, we constructed a new kind of random matching game, and we defined an analogous "social norm" strategy for this game. We then ran simulations to measure the performance of the social norm strategy under varying conditions.

A. Peer-to-Peer Routing

We consider networks where each node has a routing table containing the addresses of a small number of nodes, and requests are forwarded through multiple hops until they reach their destinations. We use Chord [3] as an example, although this basic structure is found in many peer-to-peer networks. To allow the sender to determine the identity of the node that dropped its request, we adopt iterative rather than recursive routing for our game.

B. The Game

We define the peer-to-peer routing game as follows: We have N nodes, with routing tables as described above; the routing tables are filled in with randomly chosen neighbors before the start of the game. The game runs in continuous time, rather than discrete rounds: at any time, a node can send a request to be routed by the network. (The routing process is described below.) We assume that nodes do not control the generation of requests, but can only choose whether to route requests sent by other nodes. (Later, we will revisit this issue of how requests are generated.)

When a node sends a request, it is matched with a sequence of opponents, in a way that simulates the routing of a request to a destination chosen uniformly at random. For the first hop, the sender s is randomly matched with one of its fingers, choosing the j'th longest finger with probability $1/2^j$. In the case where none of the fingers is chosen (which happens with probability $1/N$), we match node s with its shortest finger.

Say that node s ends up matched with node t. The two nodes then play an asymmetric game: s does nothing, while t can either cooperate (forward the request) or defect (drop the request). At this stage, s does not receive any payoff, while t gets a payoff of -2 if it cooperates and 0 if it defects.

If t defects, then s is finished and gets payoff 0, since its request has been dropped. But if t cooperates, then s goes on to play another game—its request has been forwarded one hop, and it is now ready to make another hop. s can be matched with any of t's fingers that are *shorter* than t is as a finger of s. (In other words, the next hop must be shorter than the last hop.) We choose the j'th longest such finger with probability $1/2^j$.

Thus the game repeats, until either one of s's opponents defects, or s is matched with a finger of length 1 (which means there are no shorter fingers). Node s now plays the asymmetric game with this final opponent. If the opponent cooperates, s receives a large payoff of $+40$, because its request has reached its destination.

This completes the description of the game. We point out the following facts: First, this game uses non-uniform random matching. For the first hop, the matching is highly non-uniform, since there are only $\lg(N)$ possible choices (and one of them has probability $1/2$); but for later hops, the matching becomes more uniform.

Second, if we ignore the actual choices of the intermediate nodes, and simply look at the lengths of the hops, we observe that, for each $\ell = 1, 2, \ldots, \lg(N) - 1$, the probability of at some point taking a hop of length 2^ℓ is 1/2; for $\ell = 0$, the probability of taking a hop of length $2^\ell = 1$ is 1, but this is really a quirk of the game. So the expected number of hops per request is $(\lg(N) - 1)/2 + 1 = \lg(N)/2 + 1/2$.

Finally, we think it is realistic that the sender receives a large payoff when its request reaches its destination, and nothing when its request gets dropped. A successfully delivered request presumably has a fairly high value to the sender, much higher than the cost of forwarding someone else's request; whereas, when a request gets dropped, the sender may learn some

routing information, but it only amounts to a partial (and unreliable) route. Thus routing is a positive-sum game, but it is brittle, since a node that drops a request completely wipes out the sender's payoff. (Also note that as the network grows, the number of hops per request slowly increases. In order for the incentives to work, the final payoff must also increase, to balance out the cost of routing.)

C. The "Social Norm" Strategy

We would like to find an analogue of Kandori's "social norm" strategy, that will work in the peer-to-peer routing game. The routing game differs from Kandori's game in that it is asymmetric: in each round, we have node A requesting a service from node B and node B requesting a service from node C, where A and C are different.

$$A \longrightarrow B \longrightarrow C$$

However, it turns out that the social norm still makes sense in this situation. Using the social norm, what B should give to A depends only on A's reputation, and what B should receive from C depends only on B's reputation. So B only has to know about its own reputation and about A's reputation; it does not care if A and C are not the same entity.

So we can simply state the social norm strategy for the asymmetric game. Let A make a request to B. Then:

- If A is innocent, B cooperates; if A is guilty, B defects.

As before, we assume that there is a reputation scheme which marks nodes as innocent or guilty. We still assume that the reputation scheme is secure against tampering. However, we allow the reputation scheme to be unreliable in the following sense: If a node behaves properly, its reputation will always be updated correctly; but if a node misbehaves, the incident will only be detected with probability p_{rel}. This would describe a system that computes reputations based on random sampling. One of the goals of our simulations was to determine the amount of sampling, and the severity of punishment, that are needed to provide incentives that will deter cheating.

Finally, we need to specify what kinds of punishments will be enforced by the reputation scheme. We use a "time-based" punishment: when a node deviates from the social norm, it is punished for a period of time τ; if the node deviates again while it is being punished, the punishment period is re-started. During the punishment period, all requests sent by this node will be dropped; but this node will still be required to forward the requests of other innocent nodes. This is a natural way to do punishment in the continuous-time game, and it would not be hard to implement in a real system.

In certain cases, we can show that the social norm is a subgame-perfect equilibrium for the routing game. Specifically, if each node's requests are generated by a Poisson process with the same rate, then Kandori's original proof goes through with minor modifications. The intuition is that requests are generated at a smooth rate, so over the course of one punishment period, a node will ask other nodes to route its requests, and it will route requests for other nodes, roughly the same number of times. This situation is similar to the original (symmetric) random-matching game. The proof of this result is given in the technical report [2].

Unfortunately, if requests are bursty, or if nodes can manipulate the timing of their requests, then the social norm may not be an equilibrium. If a node receives a very large burst of requests, it might be cheaper to drop the requests and undergo punishment. Also, a node can cheat by defecting while it accumulates a large number of requests, then cooperating just long enough to rebuild its reputation and send off all of the requests in one burst.

Finally, even when an equilibrium can be achieved, the routing game is not as robust in the presence of malicious nodes. This is due to the non-uniform matching. The burden of the malicious nodes falls disproportionally on a small group of honest nodes—namely, those nodes who have a malicious node as one of their frequently-used "long" fingers. For instance, a node whose longest finger is a malicious node will lose half of its requests. For these unlucky nodes, the incentives break down very quickly.

IV. Simulations

We ran simulations to measure the performance of the peer-to-peer routing game. For simplicity, we simulated a game with discrete rounds, where each node sends one request per round, and the nodes are randomly shuffled before each round (so the order of moves is random). This approximates a continuous-time game where each node's requests are generated by a Poisson process with the same rate.

We ran simulations with 1024 nodes and 1000 rounds. Each request took 5.5 hops on average, and the expected payoff per request (assuming 100% cooperation) was $40 - 5.5 \cdot 2 = 29$. (Recall that a node earns 40 points when its request reaches its destination, and pays 2 points each time it forwards a request for another node.)

Punishment was measured in terms of rounds, with the reputations of the guilty nodes being decremented at the end of each round. We used punishment periods $\tau = 1$, $\tau = 2$ and $\tau = 5$. Note that with $\tau = 1$, every guilty node will be automatically forgiven at the end of the round; whereas with $\tau = 2$, a guilty node must cooperate for at least one full round before it is forgiven.

Error bars on the graphs show the 99% confidence intervals.

A. Simulation Results

1) Malicious Nodes: We show how the presence of varying fractions of malicious nodes affects the performance of the system. We model malicious nodes as nodes that always defect ("defect unconditionally" or "Du" nodes). Nodes that follow the social norm strategy are denoted "Sn" nodes. Figure 1 shows the payoff per request for the Sn and Du nodes, as the number of Du nodes increases.

Observe that the payoffs of the Sn nodes drop significantly as the number of Du nodes increases, while the payoffs of the Du nodes remain small, because they are marked guilty by the reputation system. The cross-over occurs when the population

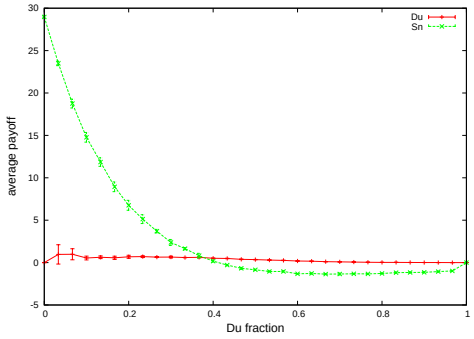


Fig. 1. Payoff per request for Sn and Du nodes, as the number of Du nodes increases (punishment $\tau = 2$)

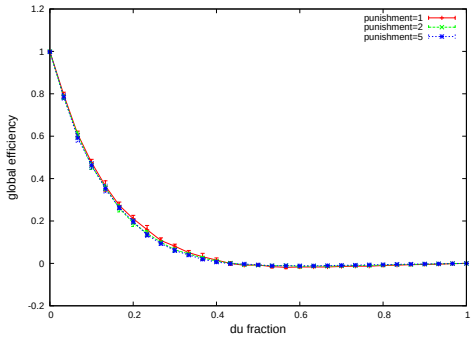


Fig. 2. Global efficiency as the number of Du nodes increases

is roughly 40% Du nodes. Note that there is a kink in the Sn curve at the right side of the graph; this is because when there are no Sn players, the average payoff for the Sn strategy is 0 by default.

Figure 2 shows the global efficiency as the number of Du nodes increases. The results are similar with punishment periods $\tau = 1$, $\tau = 2$ and $\tau = 5$.

2) Noise: We next consider the effect of noise or errors in the system. We set p_{noise} to be the probability that a node who tries to cooperate will end up defecting instead (if, for instance, the request gets dropped due to a network failure). In these simulations, we use a network of all Sn nodes.

Observe in figure 3 that as the noise level rises, the overall efficiency remains higher when we use the shorter punishment period. This illustrates a trade-off in choosing the punishment: longer punishments let us tolerate a higher fraction of malicious nodes, but shorter punishments let us tolerate a higher error rate.

3) Reliability of the Reputation Scheme: Finally, we look at how the reliability of the reputation scheme together with the length of the punishment period affects the behavior of the system. Here we simulate a network with a 20% fraction of Du nodes. We plot the payoff per request for the Sn and Du nodes (figure 4).

Observe that the reputation scheme is effective even when the reliability is quite low; this is partly because the Du nodes are easy to catch. In this particular scenario, to ensure that the Du nodes earn less than the Sn nodes, the reputation

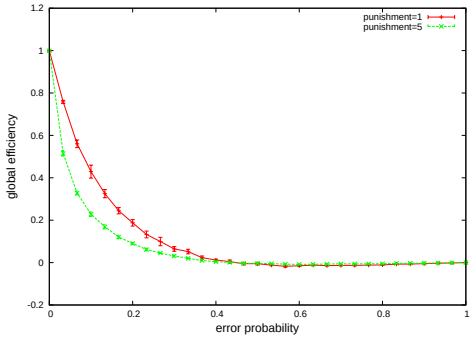
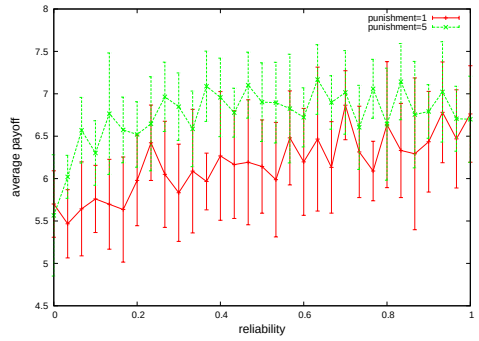
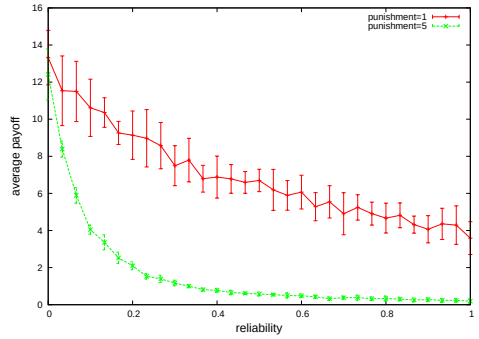


Fig. 3. Global efficiency with varying levels of noise



(a) Sn nodes

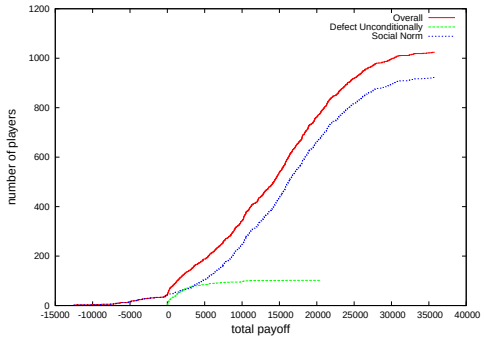


(b) Du nodes

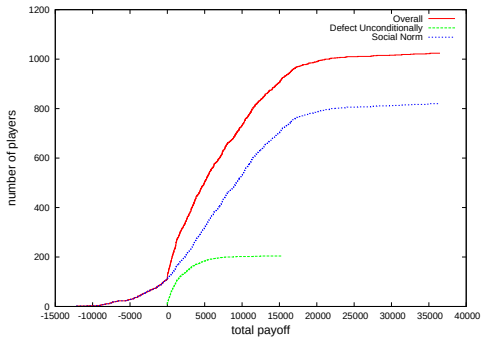
Fig. 4. Payoff per request for Sn and Du nodes, varying the reliability of the reputation scheme. Population is 20% Du nodes.

system must be at least 50% reliable when using a 1-round punishment, and at least 10% reliable when using a 5-round punishment.

4) Distribution of Payoffs among the Nodes: To gain further insight into the effectiveness of the reputation system, we look at the distribution of the payoffs among the nodes. We assume a reputation system with fairly low reliability ($p_{rel} =$ 20%), but a fairly severe punishment ($\tau = 5$). We then vary the number of Du nodes, and plot the cumulative distribution function (CDF) of the total payoffs of the Sn and Du nodes (figure 5).



(a) 10% Du nodes



(b) 20% Du nodes

Fig. 5. CDF of the total payoffs of the Sn and Du nodes, with reliability $p_{rel} = 20\%$ and punishment $\tau = 5$

There is a fairly large variance in the payoffs of the Sn nodes. This is due to the random choices of the routing tables: if a node appears in many other nodes' finger tables, it will have to route many requests, reducing its own payoff. The presence of Du nodes also contributes to the variance: an Sn node whose longest finger happens to be a Du node will do very poorly, as half of its requests will be dropped. The payoffs of the Du nodes, on the other hand, are concentrated close to zero. (Note that, unlike Sn nodes, Du nodes never have negative payoffs.) This shows that the reputation system is effective.

B. Improved Strategies

1) Social Norm with Random Defections: We also considered a variant of the social norm strategy that defects with some probability p_{def}. Our simulations showed that a punishment period of $\tau = 5$ is sufficiently severe that it wipes out any gains from occasional defections. These results are presented in the technical report [2].

V. Discussion

In this section we discuss some open issues in our understanding of the routing game. We finish by describing some of the related work.

A. Open Issues in the Routing Game

1) Timing of Requests: As mentioned earlier, a node can cheat by sending its requests in batches, so that it only needs to maintain a good reputation for the time needed to send a single batch. This kind of "timing" attack would not be practical in some applications, since it delays the servicing of requests. One area of future work is to quantify this trade-off: How much delay must be incurred, in order to get significant savings with this strategy? What kinds of monitoring and punishments would be needed to prevent this sort of cheating?

One idea is to punish a guilty node by dropping a certain number of its requests, instead of dropping its requests for a period of time. This turns out not to work; a guilty node can "fake" its punishment by sending a string of worthless requests that it knows will be dropped. Time-based punishment still seems to be the most effective.

Another idea is to make the reputations "sticky," so that if a node is consistently good or consistently bad, then it takes a sustained change in behavior to cause a change in its reputation. This is a little bit like a monetary scheme, except that it doesn't do exact bookkeeping. The advantage of this scheme is that it punishes bad nodes that defect most of the time and cooperate only when they have a bunch of requests to make; but it tolerates good nodes that suffer from occasional bursts of noise. However, this scheme would be harder to implement in a practical system.

On a related note, our simple cost function (counting the number of requests) may not be realistic if the traffic is bursty. Five requests spread out over time may not incur the same cost as five requests in a single burst.

2) Tampering with Reputations: In a real network, reputations cannot be implemented entirely by a trusted third party. At the very least, one would probably have to rely on the nodes to report when their requests are dropped. This creates some difficult incentive problems: for instance, a self-interested node might falsely accuse other nodes, especially those who are using it as one of their fingers, so that it can drop their requests.

This is an example of a more general concern, that nodes may manipulate an untrusted reputation system as part of their strategy. Preventing these attacks may require a combination of incentives (to encourage nodes to report truthfully), and a reputation system that resists tampering by a single node or a small group of nodes.

3) Other Limitations: The social norm strategy works best when the network is more-or-less homogenous: all nodes send the same number of requests per round, and all nodes choose destinations uniformly at random. But there is nothing in our present scheme that limits the number of requests that a node can send; so long as it continues to forward other nodes' requests, a node is free to send as many requests as it likes, thus maximizing its payoff. Our scheme works when all the nodes need to make roughly the same number of requests. In a highly heterogenous network, one may have to use some kind of monetary scheme instead, to obtain the right incentives.

Also, in the game, nodes are only allowed to choose between cooperating and defecting; whereas in real life, a node

can do other things, such as forwarding a request incorrectly. And, in a real system, the utility function of each node may be more complicated than in our simple model.

Finally, in the routing game we do not consider collusion between nodes. But there is nothing that prevents a subset of the nodes from attempting to build their own overlay network on top of the peer-to-peer system. This could be a significant issue.

4) Retrying Dropped Requests: One could modify the routing game to allow a node to retry a request that has been dropped, or to give nodes greater freedom in choosing their fingers. This would make routing much more robust, which would benefit the innocent nodes. Guilty nodes would not be able to take advantage of this, because the reputation system causes them to lose their requests regardless of what route they choose.

We have not implemented these changes, because of their complexity. However, the fact that such improvements are possible suggests that our current simulation results are fairly conservative.

B. Related Work

There have been many studies of incentive and reputation schemes in peer-to-peer networks; here we mention two papers that are similar in spirit to our work. Lai et al [4] use the evolutionary Prisoner's Dilemma as a model for file sharing, and study strategies based on private and shared history, as well as strategies that "adapt" to the behavior of strangers. Ranganathan et al [5] use the multiple-player Prisoner's Dilemma as a model for file sharing, and investigate two different reputation-based schemes and one monetary scheme.

Also, a number of systems have been built that incorporate incentives. Two examples are Karma [6], which uses a monetary scheme with distributed bookkeeping, and Samsara [7], which allows nodes to trade "claims" to resources. Also, Castro et al [8] studied a variety of attacks on peer-to-peer routing, and proposed some solutions using techniques from cryptography and security.

VI. Conclusions

In this paper we used a random-matching game to model routing in peer-to-peer networks. We defined an analogue of Kandori's "social norm" strategy, which uses a simple reputation system to provide incentives for cooperation. Our simulation results show that this scheme is robust in the presence of malicious nodes and noise. Furthermore, we showed that an unreliable reputation system which monitors only a fraction of the routing events can still be effective, provided that the punishments are sufficiently severe. Although our model does not capture all aspects of a real network, we feel that it is a useful starting point for understanding the incentive problems that arise in peer-to-peer systems.

One area of future work is to develop more realistic games which model different aspects of peer-to-peer systems. Some of these issues, such as the timing of requests, have been discussed in this paper.

Another area of work is to implement reputation systems in real networks. A real reputation system cannot use a trusted third party, but must be distributed over the nodes themselves. It is a serious engineering challenge to build a reputation system that is secure against tampering, provides the proper incentives, and has good performance and scalability.

References

[1] M. Kandori. "Social Norms and Community Enforcement." *Review of Economic Studies*, Vol. 59, No. 1 (Jan. 1992), pp.63-80.

[2] A. Blanc, Y. Liu, A. Vahdat. "Designing Incentives for Peer-to-Peer Routing." Technical report, available at http://www.cs.ucsd.edu/ vahdat/papers/routing-game-tr.pdf.

[3] I. Stoica, R. Morris, D. Karger, F. Kaashoek, H. Balakrishnan. "Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications." *Proc. ACM Sigcomm*, August 2001.

[4] K. Lai, M. Feldman, I. Stoica, J. Chuang, "Incentives for Cooperation in Peer-to-Peer Networks." *Workshop on Economics of Peer-to-Peer Systems*, Berkeley, 2003.

[5] K. Ranganathan, M. Ripeanu, A. Sarin, I. Foster, "To Share or Not to Share: An Analysis of Incentives to Contribute in Collaborative File Sharing Environments." *Workshop on Economics of Peer-to-Peer Systems*, Berkeley, 2003.

[6] V. Vishnumurthy, S. Chandrakumar and E. Gun Sirer, "KARMA: A Secure Economic Framework for P2P Resource Sharing." *Workshop on the Economics of Peer-to-Peer Systems*, Berkeley, California, June 2003.

[7] L. Cox and B. Noble, "Samsara: Honor Among Thieves in Peer-to-Peer Storage", *Proc. SOSP 2003*, Lake George, NY, October, 2002.

[8] M. Castro, P. Druschel, A. Ganesh, A. Rowstron and D.S. Wallach. "Secure routing for structured peer-to-peer overlay networks." *Proc. OSDI 2002*, Boston, MA, December 2002.