An Incentive Mechanism for Message Relaying in Unstructured Peer-to-Peer Systems

Cuihong Li
School of Business
University of Connecticut
Storrs, CT 06269, USA

Bin Yu
Quantum Leap Innovations
3 Innovation Way, Suite 100
Newark, DE 19711, USA

Katia Sycara
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213, USA

ABSTRACT

Distributed message relaying is an important function of a peer-to-peer system to discover service providers. Existing search protocols in unstructured peer-to-peer systems either create huge burden on communications or cause long response time. Moreover, these systems are also vulnerable to the free riding problem. In this paper we present an incentive mechanism that not only mitigates the free riding problem, but also achieves good system efficiency in message relaying for peer discovery. In this mechanism promised rewards are passed along the message propagation process. A peer is rewarded if a service provider is found via a relaying path that includes this peer. We provide some analytic insights to the symmetric Nash equilibrium strategies of this game, and an approximate approach to calculate this equilibrium. Experiments show that this incentive mechanism brings a system utility generally higher than breadth-first search and random walks, based on both the estimated utility from our approximate equilibrium and the utility generated from learning in the incentive mechanism.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems

General Terms

peer-to-peer systems, message relaying, incentive mechanism

1. INTRODUCTION

Peer-to-peer (P2P) systems have recently gained a lot of attention in academic and industrial communities. One important challenge for P2P systems is how to find peers that provide certain information or services in an efficient way, so that peers can exploit the distributed resources owned by other peers in the system. To ensure the scalability and robustness of the system, as well as to avoid legal issues, a centralized database of content of each peer usually does not exist in a P2P system (an exception is Napster). Instead, a distributed catalog of content is favored in which each peer only maintains a list of resources/services, and may contain information about the acquaintances and neighbors. In this distributive model peer discovery is realized via peer-to-peer message relaying.

Traditionally there are two ways to perform distributive peer discovery in unstructured peer-to-peer systems [19, 24]: *breadth-first search* (BFS, used by Gnutella) and *depth-first search* (DFS, used by Freenet). With BFS the messages are flooded in the system. Therefore the consumption of bandwidth is enormous, although results can be found very quickly. With DFS searches can be terminated once a result is found, and therefore use less bandwidth. But the response time could be very long and is exponential in the depth limit. Recently *random walks* (RW) have been considered as a method for P2P search that significantly reduces the number of messages compared to BFS [11]. But the performance based on random walks is highly variable, and greatly depends on the network topology and the number of walkers [22].

Besides the system efficiency, another problem that requires attention is incentives. A P2P network is a highly decentralized system and each peer may represent a different self-interested entity. A peer may manipulate the local information to take advantage of other peers' resources [14, 16]. For example, a peer may simply drop a message that is sent from other peers for relaying, for the purpose of saving communication bandwidth and energy. Therefore P2P systems are vulnerable to the *free riding* problem, i.e., a node relies on others' efforts to relay its own messages, but does not cost itself to relay messages for other nodes. Free riding can cause severe degradation of the system performance and prevent requesters from finding high quality providers efficiently [2, 16]. It is important to design an incentive mechanism that motivates each peer to behave rationally, and results in good system efficiency.

In this paper we present an incentive mechanism of message relaying for peer discovery that overcomes the flooding problem of BFS search while preserving the quick response property and good reliability. Although both peer discovery and distributed routing are related to message relaying, they are different problems [5]. In distributed routing the destination of a message is known, but in peer discovery there is no guidance about who the message should be sent to. The unknown destination implies more uncertainty and less control in message relaying for peer discovery, and prohibits applying the pricing mechanisms for distributed routing that requires prior knowledge of routing paths [15].

In our mechanism, the source peer sends the query to some neighbors and promises some payment to each receiver if the resource provider is found via a transmission route that includes the receiver. Depending on the offer, each receiver decides the number of neighbors it relays the message to and also the promised payment to its immediate downstream peers. Each of the new receivers again makes similar decisions, until the maximum number of hops (time-

AAMAS'07, May 14–18, 2007, Honolulu, Hawai'i, USA.

978-81-904262-7-5 (RPS)

to-live) is reached. One feature of this mechanism is that it does not price the relaying activities, but instead prices the relaying result, which influences the relaying activities. To the best of our knowledge, this paper is the first on incentive mechanisms of *query propagation* processes in P2P systems. Most recent work on incentive mechanisms in P2P systems studies resource sharing and does not consider the message relaying process, such as [10, 12, 20], while our mechanism tackles not only incentive problems, but also communication efficiency and reliability in a P2P system.

2. MESSAGE RELAYING MECHANISM

2.1 Motivation

The design of our incentive mechanism is motivated by the following three requirements:

(1) Communication efficiency: A system with high communication efficiency is featured by high marginal values of message relaying. A significant part of inefficiency in message propagation is caused by the *overlapping* or *saturation* issue. By *overlapping* we mean that a peer may forward the message to some peers that have received the same message from other peers, and these actions only waste the communication resources. Since the number of times that the message is transmitted in the system increases exponentially with respect to each peer's transmission effort (the number of neighbors to forward the message to), the probability of overlapping will soon get close to 1 if each peer makes significant relaying efforts, in other words, the system becomes *saturated* very quickly. To reduce the communication inefficiency caused by overlapping, *a peer should explicitly consider the overlapping probability, and be able to adjust the transmission effort with the progress of propagation*, or the saturation status of the network.

(2) Reliability: Communication efficiency and reliability are two conflicting goals. The intensity of message relaying is positively correlated with the reliability of peer discovery. Therefore an efficient message relaying scheme should tradeoff the communication efficiency and reliability. *A peer should decide the optimal relaying effort by considering both the cost and the expected payoff of finding a service provider.* The expected payoff of finding a provider not only depends on the value of finding the service to the requestor, but also on the reliability of finding a provider, which increases with the coverage of the peers that are exposed to the query.

(3) Information locality: Pricing the scarce resource and charging for the usage of the resource via a micro-payment system is a common approach to provide incentive compatibility [10, 25]. Such a mechanism is not applicable to a P2P discovery system. In message relaying, a micro-payment mechanism would require the requestor to "buy" relaying *actions* of other peers. This means that the source peer can identify all the intermediate peers and their transmission efforts, which is not feasible in a decentralized P2P system. On the other hand, it is not easy either for the requestor or the mechanism designer to decide the right price to charge for each relaying action as the local environment, such as the number of neighbors, of a peer is not known by the mechanism designer or by a third party. Revelation of such local information is called *non-private value revelation* in [18]. One way to avoid this revelation problem in mechanism design is to ask a peer to *price "items" provided by immediate downstream nodes based on only its own local information*. In our mechanism the immediate downstream nodes and their responses, and the input incentive are all local information of a peer. The item that is priced is the search result.

2.2 Incentive mechanism for message relaying

A P2P search process based on our incentive mechanism can be decomposed into two phases: message relaying and rewarding.

Message relaying phase: The message relaying phase is initiated by the requestor. The requestor sends the query message to some neighbors, along with a promised reward to each receiver. A node who receives this message relays the message further to other nodes, also with a promised reward. We call a peer that has received the message a *knower*. Otherwise if a peer has not been exposed to the message, it is called an *ignorant*. The number of knowers increases while the number of ignorants decreases along with the propagation. The requestor is initially a knower.

The propagation builds a family tree between nodes that are covered in the process. If node i receives the message from node j, then node i is a *downstream* node of j, and node j is the *upstream* node of i. A node i is called a *descendant* of a node j if i is a downstream node of j or j's descendant. If a node receives the message from multiple senders, it can choose to be a descendant of the sender with the highest reward. The *hop* number of a node in the family tree is defined as its distance (the number of generations) away from the requestor. The requestor is the only node in hop 0. In the propagation process, the hop number is attached to the message, and is automatically increased by one each time it is relayed. The maximum hop number allowed in the process is defined by time-to-live (TTL) [24]. The relaying process ends if the hop number reaches TTL.

A node will relay a message only once, although the message may be forwarded to multiple peers. This avoids repeated relaying and reduces repeated queries, and is used in the Gnutella Protocol v0.4. We assume that nodes in an earlier hop conduct relaying earlier than nodes in a later hop. Therefore when a node receives a message, it assumes that all nodes in earlier hops have completed their relaying efforts of this message.

Figure 1 shows an example of message relaying with TTL=4, where each circle represents a node, indexed by the number inside the circle, and each arrow represents a transmission of the message between two nodes. The arrows in solid lines contribute to the identification of the hop number for the receivers. The arrows in dashed lines represent the situations where the message is sent to a knower, and that does not change the hop that the receiver belongs to. The numbers attached to each arrow are the promised rewards from the upstream to downstream nodes.

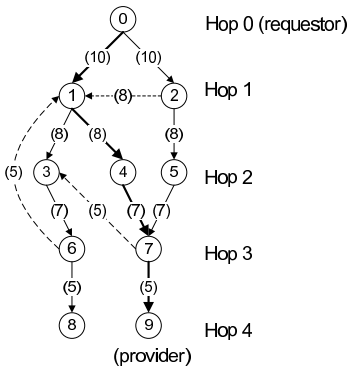

Figure 1: Illustration of hops in message relaying

Rewarding phase: A relaying path ends if the message reaches a provider or the hop number reaches TTL. If a provider is reached, a rewarding process in the opposite direction of the relaying process

is triggered. The provider first responds to the sender, who returns the identity/address of the provider to its own upstream node. Such information is reported backward along the relaying path all the way up to the requestor. Each node on the path, except the requestor, receives the promised reward from its own upstream node.

In Figure 1, the arrows in bold compose a path that reaches the provider, Node 9. The rewarding process traces backwards this path. As a result, Nodes 9, 7, 4 and 1 receive rewards of 5, 7, 8 and 10 from their upstream Nodes 7, 4, 1 and 0.

In this incentive mechanism, the incentive for a node to relay a message is the reward promised by its upstream node. A node can only receive the reward if a provider is discovered on a path extended by its relaying effort. The probability that a provider is found among a node's descendants depends not only on the node's own relaying effort (how many neighbors the node relays the message to), but also its downstream node's relaying efforts. Therefore, after a query message is received along with a promised reward, a node has to decide how many nodes to relay the message to, and what reward should be provided to its downstream nodes. A greater relaying effort enlarges the number of downstream nodes, and a larger promised reward induces a greater relaying effort of the downstream nodes, both leading to a higher probability of covering a provider in the descendants and hence receiving the reward from the upstream node. However, relaying a message to more peers incurs more costs as well due to the consumption of bandwidth and energy. Therefore, a node needs to trade off the cost and return in the decision of the relaying effort and output incentive.

We assume nodes are homogeneous, i.e., each node, except the requestor, has the same *ex ante* probability to be a provider. In this situation, a node does not discriminate its neighbors; the success rate (the number of providers discovered) only depends on how many neighbors a node relays the message to, but not on who the message is sent to. The assumption may be violated in a social network in which a peer acquires information from its past experience about the expertise and functionality of acquaintances and neighbors [26]. In that case, a node may incorporate the knowledge of other peers by relaying the message to selected neighbors. But the assumption of homogeneity is appropriate in a simple network such as a sensor network. Lemma 2.1 shows that in a homogeneous network the expected number of providers discovered is proportional to the number of peers covered in the propagation process.

LEMMA 2.1. *In a homogeneous network of N nodes, if the total number of providers is M, the expected number of providers covered among randomly selected L nodes is $L \cdot M/N$.*

The incentive mechanism constitutes a game in which a peer's utility depends on the decision of itself as well as other peers' decisions. In Section 3 we analyze and approximately calculate the equilibrium strategy of this game.

3. EQUILIBRIUM ANALYSIS AND APPROXIMATE SYSTEM UTILITY

In this section we analyze the equilibrium strategy of peers in the incentive mechanism for message relaying. After the notations are introduced, in Section 3.1 we formally define the the game and strategy equilibrium, and obtain some analytic insights to a peer's strategy. Then in Section 3.2 we proceed to provide an algorithm to calculate approximately the symmetric equilibrium.

Denote by $N+1$ the (estimated) number of peers in the network. Let the requestor be node 0, the other nodes are indexed by 1, 2, ..., N. Let $I \doteq \{0, 1, ..., N\}$ be the collection of all nodes. Denote by D_i the degree of peer i, i.e., the number of neighbors of peer i. In the network there are $M \leq N$ peers (providers) that can provide the information or service requested by the source node (requestor). The value of finding a provider to the requestor is v_0. Since the providers are substitutes, the requestor gains v_0 even if more than one providers are discovered. Denote by v_i, $i = 1, 2, ..., N$, the promised reward received by node i from its upstream node. For a node the cost of relaying a message to a neighbor is c. The hop number of node i in the propagation process is h_i. Note for the requestor, $h_0 = 0$. Given the incentive v_i and hop number h_i, a node i needs to decide the *transmission effort* k_i – the number of peers to relay the message to, and the *output incentive* u_i – the incentive to be provided to each downstream node.

3.1 Equilibrium analysis

In order to analyze the equilibrium, we shall define the strategy and utility of a peer, which constitute the P2P incentive message relaying game and a strategy equilibrium.

Strategy and Utility: Given the degree D_i of a node i, the collection of possible transmission efforts of node i is: $\mathcal{K}_i = \{0, 1, \ldots, D_i\}$. Let $\mathcal{H} = \{0, 1, \ldots, H\}$ be the set of possible hop numbers along the propagation, where H is the maximum hop number (TTL).

Definition The (pure) *strategy* s_i of a node i, $i \in I$, is a map from the hop number $h_i \in \mathcal{H}$ and input incentive $v_i \geq 0$ to the transmission effort $k_i \in \mathcal{K}_i$ and output incentive $u_i \geq 0$: $(k_i, u_i) = s_i(h_i, v_i) : \mathcal{H} \times \mathbb{R}^+ \to \mathcal{K}_i \times \mathbb{R}^+$.

Let $S_i = \{s_i : \mathcal{H} \times \mathbb{R}^+ \to \mathcal{K}_i \times \mathbb{R}^+\}$ be the *strategy space* of node i, and $s = (s_0, s_1, \ldots, s_N)$ be a *strategy profile* of all nodes, where $s_i \in S_i$. The strategy profile of all nodes except i is denoted by $s_{-i} \doteq (s_0, \ldots, s_{i-1}, s_{i+1}, \ldots, s_N)$.

The number of providers covered by descendants of node i, denoted by $\widetilde{\tau}_i$, is a random variable; the distribution of $\widetilde{\tau}_i$ depends on the strategies of all nodes. Note that if there is only one provider in the network, $M = 1$, then $\widetilde{\tau}_i$ is the probability that the provider is covered by node i's descendants. Given the strategy profile s, let $\tau_i(s) \doteq \mathbb{E}[\widetilde{\tau}_i|s]$ be the expected value of $\widetilde{\tau}_i$.

If a node i receives an input incentive v_i for each provider found among its descendants, and the node passes on an output incentive u_i to its immediate downstream nodes, then the node receives $(v_i - u_i)\mathbb{E}[\widetilde{\tau}_i|s]$ on expectation. On the other hand, if the node relays the message to k_i nodes, the cost of relaying is $c \cdot k_i$. Therefore, given the strategy profile s of all nodes, the *expected utility* of node i, $i = 1, 2, ..., N$, $U_i(s)$ can be characterized by:

$$u_i(s) = (v_i - u_i)\mathbb{E}[\widetilde{\tau}_i|s] - c \cdot k_i. \quad (1)$$

From Equation 1 we can see that for a node who is not the requestor, the utility is a linear function of the number of providers found through its relaying. This is because a node receives the reward from its upstream node for each provider that is discovered. However, this is not the case for the requestor, who obtains the same information from all providers and hence benefits from one single provider. The requestor obtains a value v_0 for any $\widetilde{\tau}_0 \geq 1$, but pays u_0 for each provider discovered. Therefore, the *expected utility of the requestor* is:

$$u_0(s) = v_0 Pr(\widetilde{\tau}_0 \geq 1|s) - u_0\mathbb{E}[\widetilde{\tau}_0|s] - c \cdot k_i. \quad (2)$$

Note that the number of providers covered by the descendants of node i, $\widetilde{\tau}_i$, not only depends on node i's transmission effort k_i, but also on transmission efforts of other nodes'. The dependence on other nodes' efforts is for two reasons. First, a node competes with other nodes in the same hop in searching for providers; the greater transmission efforts or other nodes, the less chance a node

covers a service provider among its descendants. Second, the total number of providers reached, directly or indirectly by a node, is higher if the node has a larger population of descendants. But the total number of descendants of a node depends on the transmission efforts of its descendants, which again are impacted by the input incentives passed on from their upstream nodes.

Game and Equilibrium: Having described the strategies and utility functions of nodes, we are now ready to define the P2P incentive message relaying game, and the strategy equilibrium in the game.

Definition The *P2P incentive message relaying game* is defined by the following elements:

Players: the players of the game are the peers in the network, $I = \{0, 1, 2, ...N\}$

Strategies: the strategy space of each peer i is S_i, with the pure strategy of a peer i defined as $(k_i, u_i) = s_i(h_i, v_i) : \mathcal{H} \times \mathbb{R}^+ \rightarrow \mathcal{K}_i \times \mathbb{R}^+, i = 1, ..., N$, and $(k_0, u_0) = s_0(0, v(\cdot))$.

Payoffs: The utility function of peer i, given a strategy profile s, is $u_i(s)$ as defined in Equations 1 and 2. The objective of each node is to maximize its utility.

A pure strategy Nash equilibrium is a strategy profile such that for each node, given the strategies of the others, its strategy maximizes its expected utility.

Definition A pure strategy *Nash equilibrium (NE)* is a profile s of all nodes' strategies such that for all nodes i,

$$u_i(s_i|s_{-i}) \geq u_i\left(s_i'|s_{-i}\right) \text{ for all } s_i' \in S_i.$$

PROPOSITION 3.1. *In a pure strategy Nash equilibrium, for a node i, $i = 1, 2, ..., N$,*

(i) when v_i decreases, k_i or u_i decreases and it is not possible that both increase;

(ii) the expected number of descendants increases with v_i.

Proposition 3.1 says that in the equilibrium strategy, a node that receives a lower input incentive will decrease the output incentive or transmission effort. Note that a lower output incentive further reduces the output incentive or transmission effort of the downstream nodes, causing less *future* transmission efforts. As a result, a node develops a smaller population of descendants when facing less input incentive. As the input incentive decays by hops along a relaying path, it means that with this incentive mechanism the flooding problem and pyramid effect are automatically avoided with peers' individual self-interested actions.

3.2 Approximation of utility in a symmetric network

Calculating the equilibrium for the incentive message relaying game is difficult. This is because message relaying is a stochastic process: the number of downstream nodes developed by a peer is a random variable with the distribution depending on the network status (the number of knowers) and the transmission efforts of the node itself as well as of other peers. Therefore one cannot have a close form expression of the expected number of descendants as a function of the hop number and the strategy profile of all peers. In this section, we develop an algorithm based on symmetric networks to approximately calculate the population of descendants of a node. We call a P2P network *symmetric* if all nodes are homogeneous with the same degree and equal probability of being a provider.

In a symmetric game, since all nodes are homogeneous, we can drop the node index; two nodes have equal expected utility if their strategies are the same. Therefore, in a symmetric game, we can restrict to *symmetric* strategy profiles in which all nodes have the same strategy, with the output incentive and transmission effort only depending on the hop number and input incentive, but not on the node index. Denote such a strategy by $(k, u) = s_h(v)$, where h represents the hop number, v the input incentive, k the transmission effort, and u the output incentive.

Our approximation of the descendant population follows the *certainty equivalence* estimate [3]. In this approximation, the expected number of immediate downstream nodes developed by each node is treated as certainty; this allows estimating the expected number of descendants of a node, as a function of its transmission effort, without enumerating all possible paths of system states along the development of hops.

In order to estimate the number of descendants of a node, we need to know the expansion of the family tree by each hop. Proposition 3.2 characterizes such expected expansion by the relaying actions of one hop, given the situation of the system before the relaying actions of the hop.

PROPOSITION 3.2. *If there are n_h knowers in the system up to (including) hop h, the number of nodes in hop h is m_h, and each node in hop h relays the message to k_h peers, then*

(1) the expected number of ignorants $\overline{d}_h$ reached by each node in hop h, is:

$$\overline{d}_h = \begin{cases} 0 & \text{if } k_h = 0 \text{ or } m_h = 0 \\ \frac{N-n_h}{m_h}[1 - (1 - \frac{k_h}{N-1})^{m_h}] & \text{else} \end{cases},$$

(2) the expected number of nodes in hop $h+1$ is:

$$\overline{m}_{h+1} = m_h \overline{d}_h = (N - n_{h-1} - m_h)[1 - (1 - \frac{k_h}{N-1})^{m_h}], \quad (3)$$

(3) the expected number of nodes up to hop $h+1$ is:

$$\overline{n}_{h+1} = n_h + m_h \overline{d}_h. \quad (4)$$

Based on Proposition 3.2, given the number of knowers n_h up to hop h and the number of nodes m_h in hop h, we can calculate the expected number of knowers $\overline{n}_{h+1}$ after the relaying actions of hop h, and the expected number of nodes $\overline{m}_{h+1}$ in the next hop $h+1$, with the known relaying effort . With the certainty equivalence estimate, we take the expectations $\overline{m}_{h+1}$ and $\overline{n}_{h+1}$ as certainty: $m_{h+1} = \overline{m}_{h+1}, n_{h+1} = \overline{n}_{h+1}$. This allows us applying Equations 3 and 4 recursively starting from hop 0, and obtaining an estimate of the expected numbers of nodes in all hops. Initially, the only knower is the requestor: $n_0 = m_0 = 1$.

In a symmetric system, the family tree is developed evenly, nodes in the same hop having the same number of descendants on expectation. Therefore given the expected numbers of nodes in all hops, the expected number of descendants of a node in hop h can be estimated by:

$$\overline{L}_h \approx \frac{1}{\overline{m}_h} \sum_{l=h+1}^{H} \overline{m}_l,$$

where $\sum_{l=h+1}^{H} m_l$ is the total expected number of descendants of nodes in hop h.

Based on Lemma 2.1, given $\overline{L}_h$, the expected number of providers covered among the descendants of a node in hop h is then equal to: $\overline{\tau}_h = \overline{L}_h \cdot M/N$, where M/N is the probability of a node being a provider.

Given $\overline{\tau}_h$, the expected utility U_h of a node in hop h, $h \geq 1$, can be approximated by:

$$U_h \approx (v_h - u_h)\overline{\tau}_h - c \cdot k_h. \quad (5)$$

For the requestor, taking $\overline{\tau}_0$ as certainty, the expected utility U_0 can be approximated by:

$$U_0 \approx v_0 \min(\overline{\tau}_0, 1) - u_0 \cdot \overline{\tau}_0 - c \cdot k_0 \quad (6)$$

Given a symmetric strategy profile of all hops, the process of predicting (approximately) the expected utility of a node in each hop, based on certainty equivalence, is summarized in Algorithm 3.2.

Algorithm 1 Approximate the utility given a symmetric strategy profile

Step 0: $m_0 = 1$, $n_0 = 1$. $h = 0$.

Step 1: Calculate $\overline{m}_{h+1}$ and $\overline{n}_{h+1}$ following Equations 3 and 4.

Step 2: Let $m_{h+1} = \overline{m}_{h+1}$, and $n_{h+1} = \overline{n}_{h+1}$.

Let $h = h + 1$. If $h < H + 1$, go to Step 1; otherwise go to Step 3.

Step 3: $\overline{L}_h \approx \frac{1}{\overline{m}_h} \sum_{l=h+1}^{H} \overline{m}_l$, $\overline{\tau}_h \approx \overline{L}_h \cdot M/N$, for all hops h.

Step 4: Calculate U_h for all hops h following Equation 6 ($h = 0$) or 5 ($h \geq 1$).

4. SIMULATION

In this section we provide some experimental results to benchmark the system performance of the distributed search based on our incentive mechanism, compared to breadth-first-search (BFS) and random walks, two commonly used schemes for searching [24, 11]. We do not consider depth-first-search (DFS). This is because without considering the response time, DFS undoubtedly outperforms the others in terms of the system utility – in DFS a message is only further relayed, sequentially between peers, when no result has been found. But this efficiency of DFS is in the cost of long response time. The response time of BFS, random walks and our incentive mechanism is bounded by $O(TTL)$. But the response time of DFS is in the order of the total number of relaying activities, since they are conducted sequentially.

4.1 Search Algorithms in Experiments

We consider three classes of search algorithms in the experiments: distributed search with incentive mechanisms, BFS and random walks. There are two strategies for distributed search. One is based on the approximate symmetric Nash equilibrium calculated following the approach in Section 3.2. Another is the strategy learned in the message relaying processes using reinforcement learning.

For example, in the learning algorithm, a peer has a strategy table $\mathcal{H} \times \mathcal{V} \times \mathcal{K} \times \mathcal{U} = 4 \times 10 \times 4 \times 10$, where the hops of the message $\mathcal{H}$ can be $\{0, 1, \ldots, 4\}$, the input $\mathcal{U}$ and output $\mathcal{V}$ incentives are bounded by 10, and the transmission effort $\mathcal{K}$ is bounded by 4. The initial incentive v_0 is 10 for each query. Peers are randomly chosen as service providers. A peer in the system queries other peers through message relaying.

In reinforcement learning each peer uses a standard α-greedy algorithm to explore the discretized strategy space, where $\alpha = 0.1$ [21]. Specifically, the estimated value of state a after t plays is denote by $Q_t(a)$ and r_{k_a} is the reward from choosing price a at the k_a-th time, where state a is a combination of $\mathcal{V}$, $\mathcal{H}$, $\mathcal{U}$, $\mathcal{K}$. A peer updates the estimated value of a state, if the state is chosen, based on the current reward and the previous estimated value. If a state gives a higher reward in this period, accordingly the estimated value of the price will be higher. The update of the estimated value is based on the following rule:

$$Q_{k_a+1}(a) = Q_{k_a}(a) + \alpha(r_{k_a+1} - Q_{k_a}(a)) \quad (7)$$

where step size α is a constant, $0 < \alpha \leq 1$. Based on this rule the recent rewards are weighted more heavily than long past ones. This is necessary in a non-stationary environment in which the mean reward of a state changes over time. The recent environment is more similar to the environment today and therefore gives more information, than the environment of long past. With a constant step size α, the estimates never completely converge but continue to vary in response to the most recently received rewards [21]. This is actually desirable in a non-stationary environment faced by each peer in peer-to-peer systems.

We find the approximate optimal strategies of most peers converge to the followings,

- Given the initial incentive $v_0 = 10$ and hops of the message $h = 0$, the peer will ask 3 neighbors with output incentive 7.
- Given the input incentive $v_1 = 7$ and hops of the message $h = 1$, the peer will ask 3 neighbors with output incentive 2.

Breadth-first search (BFS) is a simple searching strategy in peer-to-peer systems. Here we consider a variant of BFS, in which the requesting peer randomly chooses a ratio of their neighbors to forward the query. The ratio is also called branching factor and is six in this paper. In random walks, the requesting peer sends out k query messages (or walkers) to randomly chosen neighbors ($k = 5$ 10, or 15 in this paper if not specified). Each walker follows its own path, having nodes forward it to a randomly chosen neighbor at each step. The random walks are restricted to the same number of hops used in other mechanisms.

4.2 Experiments Setup

The topology of the system is initialized as a directed random graph. We use a random graph with N nodes, and approximate 10 out-edges per node (to its neighbors) as a starting point for the experiment. Other parameters are defined as follows, (1) The total number of nodes N is in the range from 50 to 250; (2) The total number of service providers in the network is M. $M = 1$ denotes that there is one service provider in the system; (3) The initial incentive v_0 changes from 10 to 30; (4) The maximal hop number of message is $H = 2$; in other words, each query is propagated for three hops; (5) The branching factor D for BFS and distributed search is 6.

We are interested in (1) the total utility of the system; (2) the total coverage, i.e., the number of peers exposed to the message, at the end of the propagation, which measures the reliability. Note that we show the average system utilities and coverage for all search algorithms over 10 iterations. In each iteration, each peer in the system is allowed to query other peers once. For distributed search using reinforcement learning, we compute its average system utility and coverage after 100 iterations.

We first consider only one service provider for a given unit cost c. Then we examine the results for multiple service providers and effects of different unit costs and network sizes.

4.3 Single Service Provider

Figure 2 shows that, for linear cost functions $0.15k$, the incentive mechanism generally achieves a higher system utility than BFS and Random walks. When v_0, the value of finding the service, is low, BFS walkers could bring a negative utility.

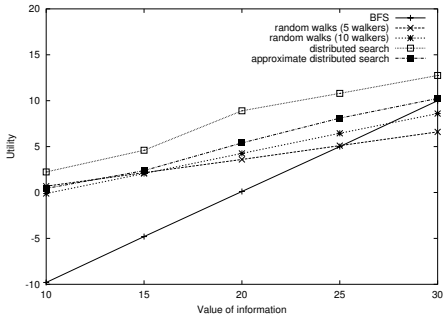

Figure 2: Comparing the system utility in experiments for a single provider ($M = 1, N = 50, c = 0.15$)

With the increase of v_0, the utility advantage of the incentive mechanism over BFS decreases. The experimental results show that distributed search is close to BFS when v_0 is equal to 30. This is because BFS generates very high coverage, which is close to the optimal when the query value is high. In other words, given the low communication costs, sometimes distributed search cannot find the service provider, while BFS can find due to its message flooding. One hypotheses is that if we impose a different cost function, e.g., $0.5k$, distributed search will win. We will examine the effects of cost functions on system utilities in section 4.5.

Figure 2 also shows the utilities for strategies of propagating messages beyond 2 hops are negative when the input incentive is low. This is due to communication costs and saturation of the network. The strategies of peers show that the transmission effort using our model is efficient than flooding (breadth-first search), where a peer in our incentive mechanism only queries a subset of its neighbors. The system utility is close to the predication of the system performances of relaying mechanisms for the linear cost function cases.

The total number of peers that receive the query message during the propagation process based on each mechanism, or the *coverage*, is recorded in Figure 3.

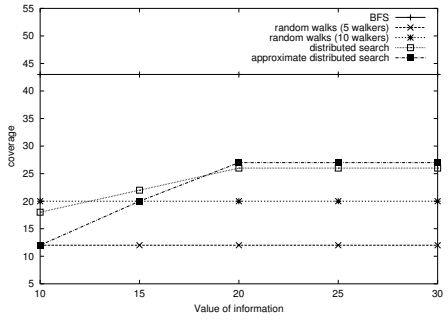

Figure 3: Comparing the coverage in experiments for a single provider ($M = 1, N = 50, c = 0.15$)

We can find that BFS always covers more peers than other mechanisms, which implies a higher reliability (but its utility may be lower as shown in Figure 2). With the given policy parameters, the coverage generated by BFS and random walks is independent of v_0, the value of finding the service. The coverage in the incentive and approximate search mechanisms increases with v_0, as a peer adapts its searching strategies in the message relaying processes. However, in random walks, walkers may visit the same nodes multiple times, and results in lower coverage compared with other search algorithms such as BFS and distributed search.

4.4 Multiple Service Providers

We have assumed in the above experiment that there is only one service provider. Next we study the model with multiple service providers.

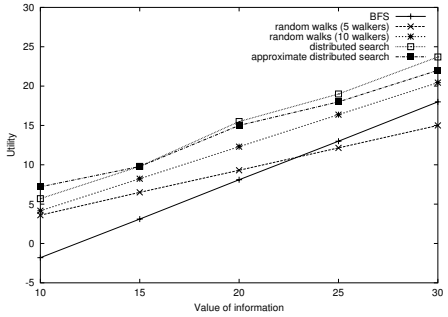

Figure 4: Comparing the system utility in experiments for three service providers ($M = 3, N = 50, c = 0.15$)

Figure 4 shows the system utility when totally three service providers are available in the system. A peer will get the reward if it finds any of the service providers. We can see that the utility of random walks is very close to distributed search when the incentive v_0 is low, but distributed search becomes better when v_0 is high. This indicates that random walks can have a good chance to cover at lease one service provider when multiple ones are available in the systems. Also, when the incentive v_0 is high, BFS gains more utility than distributed search. The reason is, BFS can guarantee to find at least one service provider; this high reliability leads to a high utility if the communication cost is low. We will study the impact of communication costs in Section 4.5.

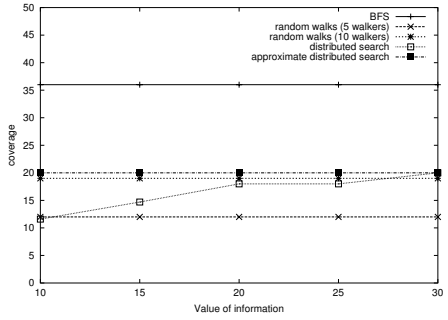

Figure 5: Comparing the coverage in experiments for multiple provider ($M = 3, N = 50, c = 0.15$)

Figure 5 shows the coverage of different searching mechanisms. We can find that, just like the case with a single provider, BFS

always covers more peers than the other mechanisms. When there are multiple providers in the systems, the coverage in distributed search is close to the random walks with 10 walkers. However, random walks still get lower system utility even though they might have similar coverage as distributed search. This indicates that in random walks peers may send more messages than necessary to find a service provider.

4.5 Effect of Communication Cost

From the experiments above we notice that BFS may gain more utilities than distributed search for a communication cost $c = 0.15$. In this section we study the impacts of the unit communication cost on system utilities by changing c, where $c = 0.1, 0.2, 0.3, 0.4, 0.5$. For simplicity, we only consider the case of one service provider.

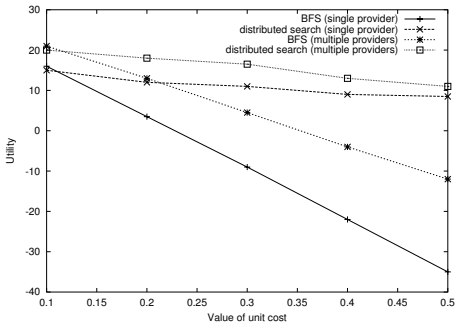

Figure 6: Comparing the system utility in experiments for different unit costs ($M = 1, N = 50, c = 0.1, \ldots, 0.5, v_0 = 30$)

Figure 6 shows the system utility of BFS and distributed search for the general linear cost function. We can find that the utility of BFS drops dramatically when the value of β increases. Distributed search is not that sensitive to the cost functions. The reason is that in distributed search, a peer can dynamically adjust its transmission effort to balance its cost and the probability of success.

4.6 Effect of Network Size

In the last experiment we study the effects of network size on different search schemes. We have a P2P network with about 50 to 250 nodes. We only consider a fixed value of v_0, where $v_0 = 30$. In order to be comparable, we assume that every fifty nodes have three providers and each message is allowed to be propagated for three hops.

Figure 7 shows the results for BFS, random walks and distributed search. We can find that distributed search is better than BFS and random walks when we scale the network size from 50 to 250. Also, note that BFS gets much lower utility when we scale up the network. One reason might be that some intermediate peers may find the (same) provider multiple times. Another reason is that in a large network the probability of overlapped message relaying is low, which causes an enormous amount of relaying activities. The coverage of BFS is about 80 in a network of size 250, which is much larger than the one for a smaller network with 50 nodes.

One interesting question is whether random walks will outperform distributed search if we increase the number of walkers. Figure 7 describes the utilities of random walks algorithms with three different numbers of walkers, 5, 10 and 15. The figure shows that even if we increase the number of walkers to fifteen, random walks still perform worse than distributed search. The results indicate that

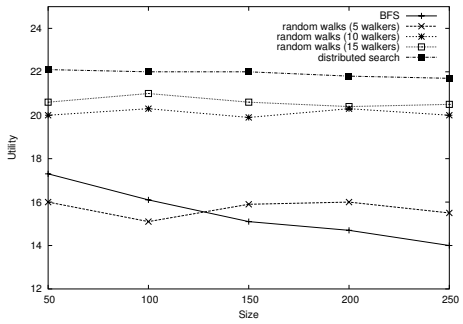

Figure 7: Comparing the system utility in experiments for multiple providers ($M = 3 * N/50, N = 50, 100, \ldots, 250, v_0 = 30, c = 0.15$)

the search algorithm based on incentive mechanism can achieve better system performance than other existing search algorithms in peer-to-peer systems.

5. RELATED WORK

Peer-to-peer (P2P) systems have received increasing attention for benefits such as improving scalability, eliminating the need for costly infrastructure, and enabling resource aggregation [19]. With all these benefits, P2P systems also create challenges in discovering information efficiently in the network.

Some research work for search techniques in unstructured P2P systems aims at reducing the number of nodes that receive and process each query with little sacrifice of the quality of results. These approaches include: adaptively deepening the search based on the responses [24], selectively querying neighbors based on their quality or reputation [24], building local indices that allow nodes to process query on behalf of nodes in a local range [24, 1], maintaining "hints" as to the possible information location by learning from the history [7], and random walks [11]. Although these techniques improve the efficiency of searching P2P networks, they are based on the assumption that nodes are cooperative and can be programmed to follow these protocols.

P2P systems are often composed of nodes governed by self-interested parties, each acting to better its own outcome. The rational behavior of nodes creates the free riding situations in peer-to-peer settings. Several papers have helped to advance the understanding of disincentives of cooperation in P2P systems. For example, [8] quantifies disincentives in file sharing P2P networks; [6] proposes a cost-based model to assess the resources that each overlay node has to contribute for being part of the overlay; [17] discusses the notions of rationality and self-interest in P2P systems. In [18] they advocate mechanism design for P2P systems in which peers are expected to be rational and self-interested and may deviate from a suggested protocol.

Generally, the P2P incentive mechanism is implemented via a micropayment system. In the micropayment system, a peer is rewarded with virtual currency or credit for each action by the system or the peer who benefits from the action. In a P2P system, the action can be forwarding a message, uploading a file, answering a query, etc. Based on game theoretic analysis, [10] and [9] evaluate the effectiveness of different micropayment mechanisms to motivate file sharing in P2P systems. [4] and [27] apply micropayment mechanisms to stimulate packet forwarding actions in a P2P net-

work. Most existing micropayment schemes require a trusted centralized broker (server), which is responsible to distribute and cash credits. More recently, several researchers propose some micropayment mechanisms that may reduce the load of the broker [13, 25]. In these above mentioned mechanisms, the reward that a peer receives for each action is independent of the resources, the work load, or the value of the service. In other words, the payment is based on a static and uniform price. In our mechanism, the payment a peer promises or receives depends on the value of information and its position (hop number) in the propagation process. Therefore, it can be regarded as a "dynamic pricing" mechanism that allows the price to change with the micro-situation of the "market".

6. DISCUSSION AND CONCLUSION

In this paper we present an incentive mechanism for message relaying in peer-to-peer discovery. In this problem the common micro-payment protocol based on the relaying actions is not feasible for an anonymous message relaying process. By pricing the search result but not the relaying action, our mechanism provides appropriate incentives for distributed message relaying that induce efficient tradeoffs between communication efficiency and reliability, while satisfying information locality.

In the paper, we focus on the cost and benefit of message relaying for information discovery, without considering the network heterogeneity. In the future work we plan to study the mechanism for heterogeneous networks such as small-world networks and scale-free networks [23]. We believe the approximate strategy equilibrium developed under the assumption of symmetric homogeneous networks will help us understand the influences of some important system parameters such as degree distribution and network connectivity on the performance of different distributed search algorithms.

Acknowledgements

This research was supported by the AFOSR under grant No. F49620-01-1-0542, by AFRL/MNK under grant No. F08630-03-1-0005 and by NSF under grant No. IIS-0205526

7. REFERENCES

[1] Lada Adamic, Rajan Lukose, Amit Puniyani, and Bernardo Huberman. Search in power-law networks. *Physical Review E*, 64:046135, 2001.

[2] Eytan Adar and Bernardo Huberman. Free riding on gnutella. *First Monday*, 5(10), 2000.

[3] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 2005.

[4] Levente Buttyán and Jean-Pierre Hubaux. Stimulating cooperation in self-organizing mobile ad hoc networks. Technical Report NO. DSC/2001/046, Swiss Federal Institute of Technology, 2001.

[5] Sam Chandan and Christiaan Hogendorn. the bucket brigade: Pricing and network externalities in peer-to-peer communications networks. In *Telecommunications Policy Research Conference*, 2001.

[6] Nicolas Christin and John Chuang. A cost-based analysis of overlay routing geometries. In *Proceedings of IEEE INFOCOM'05*, 2005.

[7] A. Crespo and H. Garcia-Molina. Routing indices for peer-to-peer systems. In *Proceedings of the 28th Conference on Distributed Computing Systems (ICDCS)*, July 2002.

[8] Michal Feldman, Kevin Lai, John Chuang, and Ion Stoica. Quantify disincentives in peer-to-peer networks. In *Workshop on Economics of Peer-toPeer Systems*, 2003.

[9] Daniel Figueiredo, Jonathan Shapiro, and Don Towsley. Incentives to promote availability in peer-to-peer anonymity systems. In *Proceedings of IEEE ICNP*, 2005.

[10] Philippe Golle, Kevin Leyton-Brown, Ilya Mironov, and Mark Lillibridge. Incentives for sharing in peer-to-peer networks. In *Proceedings of the Second International Workshop on Electronic Commerce*, 2001.

[11] Qin Lv, Pei Cao, Edith Cohen, Kai Li, and Scott Shenker. Search and replication in unstructured peer-to-peer networks. In *ICS*, New York, 2002.

[12] Richard T. B. Ma, Sam C. M. Lee, John C. S. Lui, and David K. Y. Yau. An incentive mechanism for P2P networks. In *ICDCS*, Tokyo, Japan, 2004.

[13] S. Micali and R. Rivest. Micropayments revisted. In *Proceedings of CTRSA*, 2002.

[14] Chaki Ng, David Parkes, and Margo Seltzer. Strategyproof computing: Systems infrastructures for self-interested parties. In *The First Workshop on Economics of Peer-to-Peer Systems*, Berkeley, CA, June 2003.

[15] C. H. Papadimitriou. Algorithms, games, and the internet. In *Annual ACM Symposium on Theory of Computing (STOC)*, 2001.

[16] Jeff Shneidman and David Parkes. Rationality and self-interest in peer-to-peer networks. In *The Second International Workshop on Peer-to-Peer Systems (IPTPS'03)*, Berkeley, CA, February 2003.

[17] Jeff Shneidman and David Parkes. Rationality and self-interest in peer-to-peer networks. In *2nd International Workshop on Peer-to-Peer Systems (IPTPS)*, 2003.

[18] Jeff Shneidman, David Parkes, and Margo Seltzer. Overcoming rational manipulation in distributed mechanism implementations. Technical Report TR-12-03, Harvard University, 2003.

[19] Dejan S.Milogicic, Vana Kalogeraki, Rajan Lukose, and etc. Peer-to-peer computing. Technical Report 2002-57RI, Hewlett-Packard Company, 2002.

[20] Qixiang Sun and Hector Garcia-Molina. Slic: A selfish link-based incentive mechanism for unstructured peer-to-peer networks. In *ICDCS*, Tokyo, Japan, 2004.

[21] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[22] Dimitrios Tsoumakos and Nick Roussopoulos. A comparison of peer-to-peer search methods. In *International Workshop on the Web and Databases (WebDB)*, 2003.

[23] D. Watts and S. Strogatz. Collective dynamics of small-world networks. *Nature*, 393:440, 1998.

[24] Beverly Yang and Hector Garcia-Molina. Improving search in peer-to-peer systems. In *ICDCS*, Vienna, Austria, 2002.

[25] Beverly Yang and Hector Garcia-Molina. PPay: Micropayments for peer-to-peer systems. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS)*, October 2003.

[26] Bin Yu and Munindar P. Singh. Searching social networks. In *Proceedings of Second International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 65–72, 2003.

[27] S. Zhong, J. Chen, and Y. Yang. Sprite: a simple, cheat-proof, credit-based system for model ad-hoc netowrks. In *Proceedings of INFOCOM*, 2003.