Query Incentive Networks

Jon Kleinberg [*]   Prabhakar Raghavan [†]

Abstract

The concurrent growth of on-line communities exhibiting large-scale social structure, and of large decentralized peer-to-peer file-sharing systems, has stimulated new interest in understanding networks of interacting agents as economic systems. Here we formulate a model for *query incentive networks*, motivated by such systems: users seeking information or services can pose queries, together with incentives for answering them, that are propagated along paths in a network. This type of information-seeking process can be formulated as a game among the nodes in the network, and this game has a natural Nash equilibrium.

In such systems, it is a fundamental question to understand how much incentive is needed in order for a node to achieve a reasonable probability of obtaining an answer to a query from the network. We study the size of query incentives as a function both of the rarity of the answer and the structure of the underlying network. This leads to natural questions related to strategic behavior in branching processes. Whereas the classically studied criticality of branching processes is centered around the region where the branching parameter is 1, we show in contrast that strategic interaction in incentive propagation exhibits critical behavior when the branching parameter is 2.

1   Introduction

A growing number of on-line information systems are adopting a model in which, rather than posing queries to a centralized index of the system, users in effect pose queries to the network itself. Requests for information are propagated along paths through the network, connecting those with information needs to those with relevant answers. Two recent settings in which this model has arisen are decentralized peer-to-peer file-sharing systems, and the current generation of social-networking services. While these types of systems differ in their specifics, the fundamental issues underlying them are in a number of respects similar.

Mechanisms to propagate queries through peer-to-peer systems have been the subject of considerable recent interest. Early systems such as Gnutella and Freenet, as well as research systems including Chord, CAN, Tapestry, and Pastry, proposed algorithms for distributing information across a network of peers, and then allowing decentralized searches to proceed by routing queries to the appropriate content [16].

In a different direction, *social-networking systems* are among the recent arrivals on the landscape of Internet media and information systems [1, 27]. Systems such as Friendster, Orkut, LinkedIn,

initial utility 9

v* rew. 5

b rew. 2 c rew. 2

answer d e rew. 1 f answer

g rew. 0

**If root chooses to pass
reward to node d, then
b receives 5–2 = 3.
d receives 2.
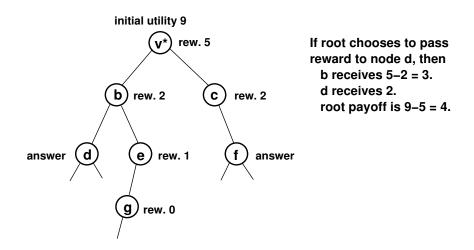root payoff is 9–5 = 4.**

Figure 1: The propagation of a query with rewards. Each node is offered a reward by its parent, and offers a (smaller) reward to its children for continued propagation. The search down each branch stops when the reward reaches 0, or when a node with an answer is reached.

and Ryze are designed to maintain an explicit social network on their members. New users join by linking to friends who already belong to the system, and this leads over time to a potentially large on-line community built around friendships in the off-line world. The basic motivation behind such systems is to use the underlying network as a way to help find information and services; tasks such as finding a new job, or a new apartment, can be facilitated by seeking help from one's friends, who in turn can ask their friends, and so forth, providing indirect access to the large set of people connected through multi-step chains of acquaintances. A number of research systems have sought to use social networks for referral in similar ways (e.g., [13, 28, 30]).

In this way, social-networking services are also based on a model of posing queries that are propagated through a network — in this case, a network defined by friendships. The propagation along chains of friends is crucial in this context, because it provides a powerful way of *vetting* the answers one gets. You trust your friends, who in turn trust their friends, and so on; thus, when an answer to a question emerges from a chain of referrals in the network, it is endowed with a corresponding chain of trust.

Networks as marketplaces. As these types of systems grow in scale and complexity, their potential as marketplaces for information and services becomes more compelling. As one article in the popular press described the situation for social-networking services, "Essentially, [Friendster is] an attempt to apply the economic theory behind eBay – bringing buyers and sellers together to create commerce – to a far more complex social phenomenon" [21].

The attempt to model networks of information resources as marketplaces raises a range of issues. In particular, how do members of such a network extract utility from their interactions with other members, and what is the overall system's behavior when individual members interact strategically so as to maximize their utilities? While early work on peer-to-peer and social-networking systems was motivated primarily by the structural considerations underlying large networks — in particular, that such networks should have short, easily findable paths joining queriers and answerers — the considerations above highlight the fundamental incentive issues at work in the operation of such

2

networks. A growing line of work in the peer-to-peer community has begun investigating such issues (e.g., [3, 12, 14, 15, 19, 25, 29]). In turn the analysis of incentive propagation in such networks speaks to the process by which people discover tangible goods and services, beyond information.

Here we consider the effect of incentive issues on the performance of these types of systems, in the spirit of recent work on the *price of anarchy* and *price of stability* [18, 23], but with somewhat different types of guarantees. We formulate a simple model of query propagation on a random network, and our main result has the following general form (to be made precise below). Suppose that n is the *rarity* of the answer to a query: on average, one out of n nodes possesses the answer. When the "effective branching factor" of the network — essentially the average number of new neighbors per node encountered in a breadth-first search – is greater than 1, then (by well-known results) there is a giant component in which the node v^* originating the query has a path of length $O(\log n)$ to a node w with the answer. However, if nodes must be compensated for their participation in propagating the query, then under strategic behavior, the originating node v^* must invest an enormous amount in order to receive an answer with constant probability when the effective branching factor is less than 2, while it need only invest $O(\log n)$ when this branching factor is greater than 2. Thus, while an effective branching factor of at least 1 is the threshold for achieving purely *structural robustness* (in the form of short paths to the answer), an effective branching factor of at least 2 is needed to make searching feasible in the presence of *incentives*. The region between 1 and 2 represents a case in which the short paths that exist are unusable due to the large investments required.

We now proceed to formulate a basic model that captures this phenomenon in a clean and tractable manner. Before doing so, we note that questions like this also belong to a broader context concerned with the economic properties of networks, and we refer the reader to [10] for work on network formation, and [11] for work on buyer-seller networks.

1.1 Formulating a Model

Since there are a number of issues that should be taken into account in the development of a model for these phenomena, we undertake it via a sequence of steps.

1. General Considerations. The basic type of operation we study is the following: a node v^* belonging to the network is seeking a particular service or piece of information held by certain other nodes; abstractly, we think of v^* has posing a *query* that it wishes to have *answered.* v^* offers a reward to its neighbors, which it will pay when it receives the answer; each neighbor of v^*, if it does not have the answer, can offer a (smaller) reward to *its* neighbors (hoping to pocket the difference), effectively using them as "sub-contractors" in producing the answer. In this way, the query propagates along paths, eventually reaching (possibly several) nodes that hold the answer.

2. A Tree Model. A model of this general form was proposed by Li et al. [15] in the setting of peer-to-peer systems. As the model in [15] has a number of features that are not important for us here, we begin by describing a variant that is related, but somewhat different in its details.

We model the underlying network as a (finite or infinite) tree T. The root v^* of T has a query for which it has a utility of r^*, and each node holds the answer with some probability $1-p > 0$. The node v^* offers a reward for the answer, and the query thus propagates down the tree as outlined above. Specifically, each node v has an integer-valued function f_v; if v is offered a reward of r by its

parent, and v does not possess the answer to the query, then it in turn offers a reward of $f_v(r) < r$ to its children. (The root follows the analogous rule at the outset, offering an initial reward of $f_{v^*}(r^*)$.) The propagation of the query stops along a particular path in T when the offered reward shrinks to 0 or when a node w is reached who holds the answer. In the latter case, the identity of w is propagated back up to the root. From among all answer-holders that are discovered by this process, v^* selects one, say w, by an arbitrary (potentially randomized) procedure, and propagates the reward down to w. Each intermediate node on the path to w keeps its "share" of this reward, paying out the amount it offered to its child on the path to w. An example of this process is illustrated in Figure 1.

If nodes are behaving strategically, then each will choose how to offer a reward so as to maximize its payoff. We thus view each node v as a player in a game, selecting a strategy in the form of a function f_v. As a useful device in defining the payoffs, we assume that the root's utility r^* is drawn from an arbitrary distribution that puts positive probability on each positive integer. The form of this distribution is immaterial; its purpose is simply to make the formulation cleaner, encoding the notion that any utility can arise at the root. With respect to a set of strategies $\{f_v\}$, the payoff to a node v is the expected reward it retains over the random choice of r^* and the random selection of which nodes possess the answer.

It is not difficult to prove that this game possesses a Nash equilibrium that is (subject to some technical conditions) unique. Such a result was shown by Li et al. [15] in the context of their related model. Our interest here is in going beyond the existence question to investigate how the behavior of nodes under a Nash equilibrium affects the performance of the system — in other words, how it affects nodes' abilities to cheaply find answers to queries.

3. A Branching Process Model. For studying the dynamics of query propagation and the structure of its Nash equilibria, the general tree model has two drawbacks. First, computing the Nash equilibrium requires knowledge of the tree, so positing that nodes play Nash equilibrium strategies comes with a tacit assumption that they know the entire tree. It would be preferable to assume that each node simply had an estimate of the average "branching factor" of the tree, and based its choice of strategy only on this. Second, it is hard to discuss how the behavior of query propagation varies over the space of all possible trees; for such an investigation, it would be more useful to have a single tunable parameter, like the "branching factor," that characterizes the network topology. In this way, we could study the behavior in a rich family of network structures that arise as a function of this parameter.

For both these reasons, it becomes very natural to consider the following variant of the tree model, based on branching processes. We start with an infinite complete d-ary tree T rooted at node v^*; this tree represents the set of all nodes who *might* take part in the propagation of the query. Each node in T is declared to be *active* independently with probability q. The set of all nodes reachable from the root using paths consisting entirely of active nodes is a random subtree T' of T. We now run the tree model on T'; the payoff to each node, as a function of the set of strategies $\{f_v\}$, is the expected amount of reward it retains under the random choice of r^*, the random choice of tree T', and the random outcome of which nodes possess the answer. Note that for a particular choice of T', nodes $v \in T \setminus T'$ receive no reward.

In this stylized model, one could interpret T as the full peer-to-peer or social-networking system, and the active nodes as the subset that are "on-line" and able to participate when the query is issued. But for our purposes, the main point is that the model forms a very clean way to generate

4

a family of random networks whose structure varies based only on the two parameters q and d. We can show that there exists a Nash equilibrium for this model (again, essentially unique, subject to some technical conditions), and that nodes can compute their appropriate strategy using knowledge of just q and d.

In the terminology of branching processes [2], T' is a *Galton-Watson tree* generated from an offspring distribution that produces j children with probability $\binom{d}{j}q^j(1-q)^{d-j}$. For such branching processes, the crucial parameter is the mean number of offspring b per node: if $b < 1$ then the tree T' is almost surely finite, while if $b > 1$ then there is a positive probability of obtaining an infinite tree T'. We will focus on the case $b > 1$; note that in our model $b = qd$, and we will have an infinite tree with probability $1 - e_{q,d}$ for some constant $e_{q,d} \in (0,1)$.

1.2   Results

We are interested in studying the initial investment needed in order to obtain an answer to a query with reasonable probability. In terms of the branching process model we have just defined, we can phrase this as follows: how large a utility r^* must the root v^* have in order to achieve a fixed constant probability σ of obtaining an answer from the network? We consider the utility needed as a function of the rarity n of the answer, defined as $n = (1-p)^{-1}$; in other words, one of out of every n nodes has the answer.

We seek to understand how this relationship between r^* and n is affected by the underlying network topology. The branching process model allows us to naturally study this simply by varying b. Our main results are that a transition in this dependence happens when the branching factor b passes 2.

- When $b < 2$, the utility required for v^* to find the answer with constant probability σ is $\Omega(n^c)$, where c depends on $2 - b$. (Theorem 4.6.)

- When $b > 2$, the utility required for v^* to find the answer with constant probability $\sigma < 1 - e_{q,d}$ is $O(\log n)$, where the constant inside the $O(\cdot)$ depends on $b - 2$ and $1 - e_{q,d} - \sigma$. (Recall that $e_{q,d}$ is the extinction probability of the branching process.) (Theorem 4.10.)

So when $b > 2$, the utility is within a constant factor of a simple structural lower bound: assuming we get an infinite random tree T', the distance in T' to the nearest answer is $O(\log n)$ with high probability, and each node on the path will retain at least one unit of reward. Thus, for a large branching factor, the propagation of queries is very efficient in its use of reward. On the other hand, when $1 < b < 2$, the distance to the nearest answer (for T' infinite) is still $O(\log n)$, but now the utility needed by the root is exponential in the distance; thus query propagation has become bogged down by the friction of multiple intermediaries.

We find this transition surprising, because the value $b = 2$ is *not* a critical point for the underlying branching process itself. Rather, for all $b > 1$, there is a positive probability of getting an infinite tree, and this probability increases continuously in b. Moreover, in these infinite trees the answer to the query is $O(\log n)$ steps from the root with high probability, again for all $b > 1$. It is only once incentives are taken into account that $b = 2$ stands out as a critical value. This, then, is the precise formulation of the qualitative finding that we described earlier: the network achieves structural robustness (in the form of a positive probability of an infinite, exponentially growing tree below the root) once $b > 1$, but it only becomes usable for efficient incentive-based queries once b reaches 2.

5

Extensions. There are clearly many directions in which the model could be generalized; our goal here is to explore some of the surprising phenomena that emerge already in a very simple model of such processes. One can easily generalize the framework to assume that the branching factors and probabilities of holding the answer are non-uniform, and the results on existence of Nash equilibria extend to this setting. It is less clear how to find a comparably simple parametrization of this class of models that, like the branching factor, exposes the sharp transition we have seen. For some generalizations of the model, we have found through simulation that a transition in the necessary reward is evident when a parameter based on the growth rate of balls in the graph metric passes the analogue of the branching factor 2; we will describe this in more detail in the full version of the paper.

Another significant direction lies in considering network models not based on trees. This requires enhancing the game-theoretic basis of the model, since nodes may now be offered (possibly different) rewards from multiple neighbors, and they must decide how to act on these. We discuss a more general model for this setting briefly in the final section.

Placing a Value on Effort. Finally, we discuss an issue that will arise in working out the payoff functions in the next section: the value nodes place on their own *effort* in taking part in the protocol.

If nodes placed no value on their effort, then it would be hard to avoid the following type of degenerate solution: the root offers an arbitrarily small initial reward ε and succeeds in getting an answer anyway because intermediate nodes simply skim off a positive fraction of what they are offered and pass the rest on. (This is a common type of "Zeno's paradox" pathology.) Our condition that rewards be integer-valued prevents this from happening; it requires each intermediate node to retain at least one unit of reward, and it is motivated by the following model for valuing effort.

The detailed query propagation process can be viewed as consisting of three stages: (i) the query is propagated outward from the root; (ii) the identities of nodes holding answers are propagated inward to the root; and (iii) a single answerer w is chosen, and the desired "connection" from v^* to w is established along the path from v^* to w. In keeping with the bulk of the recent work discussed above (see e.g. [13, 28, 30]), we assume nodes treat stages (i) and (ii) as having negligible cost, modeled as 0 (a node will forward a query even if there is only a small probability that it will lie on the selected path to the answer); but nodes value their effort in stage (iii) at unit cost (a node retains at least one unit of reward for the event in which it is on the path to the answer).

This distinction is a very natural one in the context of both peer-to-peer and social-networking systems. For example, many proposals for social-network referral systems view steps (i) and (ii) as being of negligible cost — and generally automatable by software agents — for people in the system, since they involve simply forwarding search requests to lists of contacts [13, 28, 30]. On the other hand, step (iii) – establishing the connection between root and answer – is where the role of the social network for *vetting* comes into play; there is an underlying protocol in which people on the path validate the relationship between the endpoints, and it is this effort that demands compensation [30]. A closely analogous distinction arises in many peer-to-peer systems; in Freenet [7], for example, step (iii) corresponds to the transfer of a file in which it is cached at every step along the path, and so step (iii) involves considerably more effort than (i) and (ii). When the query seeks a tangible good or service (as opposed to an information need), it is step (iii) that delivers the "fulfillment" of the order placed at the root.[1]

---

[1]There is no technical obstacle to modeling settings in which nodes balance each of steps (i), (ii), and (iii) against a

2 The Nash Equilibrium

We begin by constructing the Nash equilibrium for query propagation in the branching process model, argue why it is (essentially) unique, and then in subsequent sections study the growth rate of rewards when players use this equilibrium strategy.

We first define the following auxiliary functions. Let $\mathbf{f}$ denote the set of all reward functions, $\{f_v : v \in T\}$. We define $\alpha_v(\mathbf{f}, x)$ to be the probability the subtree of T' below v yields the answer, given that v offers reward x and that v does not possess the answer. Let $\beta_v(\mathbf{f}, x) = 1 - \alpha_v(\mathbf{f}, x)$. These quantities can be defined using a recurrence relation as follows. Each child w of v is endowed with a utility of x for the answer by the offered reward, so the subtree rooted at w fails to produce the answer with probability $p\beta_w(\mathbf{f}, f_w(x))$. Node w passes the answer to v if this failure does not occur, *and* if it is active, for a probability of $q(1 - p\beta_w(\mathbf{f}, f_w(x)))$. Finally, v fails to get the answer if none of these successes occurs:

$$\beta_v(\mathbf{f}, x) = \prod_{w \text{ child of } v} [1 - q(1 - p\beta_w(\mathbf{f}, f_w(x)))]. \tag{1}$$

We now construct a Nash equilibrium $\mathbf{g}$ for this game, defining the functions in $\mathbf{g}$ inductively. By definition we have $g_v(1) = 0$ for all nodes v. Now, assume that $g_v(x)$ has been defined for all v and all $x < r$. The crucial observation is that from these values, we have enough information to determine the values of $\alpha_v(\mathbf{g}, x)$ for all v and all $x < r$. If v does not possess the answer, and it offers a reward of x to its children, then if an answer emerges from the subtree below v (with probability $\alpha_v(\mathbf{g}, x)$), and if a node in its subtree is selected by the root for the reward, v will retain a reward of $r - x$, minus a cost of 1 for its effort in establishing the connection to the root. We thus define $g_v(r)$ to be the value $x < r$ maximizing

$$(r - x - 1)\alpha_v(\mathbf{g}, x). \tag{2}$$

Note that by this construction, all functions g_v are the same; we denote this common function by g.

We now claim that $\mathbf{g}$ is a Nash equilibrium. The proof is somewhat technical, and provided in the appendix; roughly, it expands the brief argument in the previous paragraph, relating the expression in (2) to the payoff of node v.

Theorem 2.1 *The set of functions $\mathbf{g}$ is a Nash equilibrium.*

For the uniqueness of $\mathbf{g}$, we note two caveats that prevent us from claiming this directly. First, in the expression (2) defining $g_v(x)$, there might be two values of x that both achieve the maximum. Second, based on the way $\mathbf{g}$ is defined, there are certain values r so that nodes v far down in the tree will never be offered a reward of r. In these cases, we can define $g_v(r)$ arbitrarily and still have a Nash equilibrium.

However, these turn out to be the only two issues to address, and so we formulate the uniqueness result as follows. First, we note that a node receives a payoff of 0 regardless of how it sets the value

---

unit effort cost, but the resulting solutions are pathological. This is much like the "Zeno's paradox" case, but extreme in the opposite direction: in the Nash equilibrium the root needs to offer a reward of at least roughly n^k to get the query to travel k steps, which dwarfs the payments even for the case of $b < 2$ discussed above. This suggest that the whole underlying framework for incentive-based query propagation is much more suitable to settings in which steps (i) and (ii) are viewed as having negligible effort cost.

$g_v(2)$: if it defines $g_v(2) = 0$, then the query is not propagated at all, and if it defines $g_v(2) = 1$ then its retained reward of 1 is offset by its effort cost. As a tie-breaker (say, for example, that the effort cost is really very slightly less than 1), we will assume that all nodes define $g_v(2) = 1$. Second, we say that p is *generic* with respect to q if (p, q) is not a zero of any bivariate polynomial with rational coefficients, other than the polynomial that is identically zero. (For example, if q is rational this is equivalent to saying that p is not a root of any non-trivial polynomial equation with rational coefficients.) Finally, we say that a reward r is *reachable* at node v, with respect to a set of functions $\mathbf{f}$, if there is some initial utility the root may have with positive probability, so that v will offer reward r if the query is propagated down to it. (Recall that the distribution on root rewards is immaterial to our results; its function is to facilitate a game-theoretic formulation.)

Theorem 2.2 *Let p be generic with respect to q, and $\mathbf{f}$ a Nash equilibrium in which $f_v(2) = 1$ for all nodes v. Then for all nodes v, and rewards r that are reachable at v with respect to f, we have $f_v(r) = g_v(r)$.*

The proof, which appears in the appendix, is by induction on r: the functions f_v and g_v must agree for $r = 1, 2$, and if they first differed on some larger reachable r, the fact that they both maximize expression (2) would define a polynomial identity that contradicts the assumption that p is generic with respect to q.

3 The Breakpoint Structure of Rewards

For an answer rarity $n = (1 - p)^{-1}$, a target success probability σ, and a branching factor b, we let $R_\sigma(n, b)$ denote the minimum utility needed by v^* in order for the query process to yield an answer with probability at least σ. The basic question raised in Section 1 can be phrased as follows: how does $R_\sigma(n, b)$ grow asymptotically in n, b, and σ? In this section, we fix values for $n > 1$ and $b > 1$, and study how $R_\sigma(n, b)$ depends on σ. In the next section we explore the dependence on n and b, using the results of this section.

When n and b are fixed, $R_\sigma(n, b)$ is characterized by a sequence of discrete *breakpoint* values: the set of possible values for the success probability σ is partitioned into intervals, within each of which $R_\sigma(n, b)$ is constant. This follows simply because $R_\sigma(n, b)$ takes integer values, but we will find in fact that something stronger and less obvious is going on: between successive breakpoints, $R_\sigma(n, b)$ will in general increase by an integer difference that is greater than one. Intuitively, a utility of r^* at the root (and the resulting reward $g_{v^*}(r^*)$) propagates the query, under the strategic behavior of nodes, out to some depth in the tree. As we increase the root's utility beyond r^*, nodes closer to the root have a growing incentive to "push reward" deeper into the tree, in the hope of increasing the probability of finding the answer and thus compensating for the loss of the reward that they push out. We will argue that the breakpoints occur precisely when the iteration of the function g results in propagating the query down an extra level of the tree.

We define some quantities related to $\alpha_v(\mathbf{g}, r)$. First, let $\delta(r)$ denote the number of times we have to iterate the function g to reduce an initial reward of r down to 0; the quantity $\delta(r)$ is thus the number of levels the query would visit if the root had a utility of r, all nodes were active, and no node possessed the answer. Then, an equivalent way to view the search process with initial utility r is as follows: run the branching process until it either reaches level $\delta(r)$, dies out early, or reaches a node with the answer.

In order to analyze this equivalent view, it is useful to consider the question: if we run the branching process indefinitely (without rewards), what is the probability that no node in the first j levels has the answer, given that the root does not?[2] We denote this probability by $\hat{\phi}_j$. There is a simple connection between $\hat{\phi}_j$ and $\beta_{v^*}(\mathbf{g}, r)$, the probability the root will not obtain the answer given utility r, conditioned on v^* not possessing the answer itself: since a reward of r will push the query $\delta(r)$ levels into the tree, we have $\beta_{v^*}(\mathbf{g}, r) = \hat{\phi}_{\delta(r)}$. For purposes of analysis later, it useful to have a recurrence for $\hat{\phi}_j$, directly parallel to Equation (1)), and justified by an analogous argument: $\hat{\phi}_{j+1} = (1 - q(1 - p\hat{\phi}_j))^d$.

Let u_j be the minimum r for which $\delta(r) > j - 1$. Note that if u_j exists, then $\delta(u_j) = j$. Indeed, $\delta(u_j) > j - 1$, but by the minimality of u_j, we have $\delta(f(u_j)) \leq j - 1$ and so $\delta(u_j) = 1 + \delta(f(u_j)) \leq j$.

We can prove the existence of u_j for all j, and determine the values of all u_j, by induction. We define auxiliary quantities y_j; the role of these will become clear below. We start with $u_1 = 1$ and $u_2 = 2$. Now, suppose that $u_1, u_2, \ldots, u_j$ have been defined. For a given initial utility r, the only possible optimal rewards have the form u_i for some i; any other reward could be reduced to the next largest u_i without affecting the probability of success. For a reward r, define the following linear functions of r: $\ell_i(r) = (r - u_i - 1)(1 - \hat{\phi}_i)$. This is the payoff to the root when it offers reward u_i with utility r. The functions ℓ_i can be viewed as a family of lines in the reward-payoff plane. Thus at utility r, reward u_i is preferable to u_k if and only if $\ell_i(r) > \ell_k(r)$.

Suppose by induction that the following holds for a given j: for all $r \geq u_{j-1}$, we have $\ell_{j-1}(r) > \ell_{j-2}(r) > \cdots > \ell_1(r)$. We now define u_{j+1} as follows. At $r = 1 + u_j$, we have $\ell_j(r) = 0$, and hence $\ell_j(r) < \ell_{j-1}(r)$. Thus, the point at which lines ℓ_j and ℓ_{j-1} cross is to the right of $1 + u_j$, since the slope of ℓ_j is greater than that of ℓ_{j-1}. This crossing point is at the value y for which

$$(y - u_j - 1)(1 - \hat{\phi}_j) = (y - u_{j-1} - 1)(1 - \hat{\phi}_{j-1}). \tag{3}$$

Define y_{j+1} to be this value of y. Once a node's utility exceeds y_{j+1}, it is preferable to offer a reward that sends the query j more steps rather than $j - 1$. We define $u_{j+1} = \lceil y_{j+1} \rceil$; then u_{j+1} is the first integer for which $\delta(u_{j+1}) > j$. Note that for all $r \geq u_{j+1}$, we have $\ell_j(r) > \ell_{j-1}(r) > \cdots > \ell_1(r)$.

Now, we write $\Delta'_j = y_j - u_{j-1}$ and $\Delta_j = u_j - u_{j-1}$. By Equation (3), we have

$$1 + \frac{\Delta_j}{\Delta'_{j+1} - 1} = \frac{1 - \hat{\phi}_j}{1 - \hat{\phi}_{j-1}}. \tag{4}$$

This equation captures the growth rates of the breakpoints u_j through the ratio of their successive differences, and it will be important in the analysis in the next section.

4 The Growth Rate of Rewards

We now come to the central point, which is the asymptotic dependence of $R_\sigma(n, b)$ on n and b. We will see that the dependence on σ will tend to be hidden inside the asymptotic bounds, provided σ is bounded away from the non-extinction probability of the underlying branching process, $\sigma < 1 - e_{q,d}$. In general, it is also helpful to think of $\sigma \gg n^{-1}$; the exact relationship among these quantities will be made precise below.

---

[2]One can in fact describe this as a multi-type Galton-Watson process with an early stopping condition [22], but it does not appear that these results are relevant to what we need here.

We also define some additional notation, for convenience: we let $b = qd$, and we define the function $t(x) = (1 - q(1 - px))^d$ so that the recurrence defining $\hat{\phi}_j$ can be written $\hat{\phi}_j = t(\hat{\phi}_{j-1})$.

Our main result — the dichotomy between the cases where $b < 2$ and $b > 2$ — will hinge on the rate at which $\hat{\phi}_j$, the probability of failing to find an answer, decreases as a function of j. Due to the way in which the search for an answer unfolds down the tree, aspects of the analysis are similar in flavor to results on the evaluation of Boolean formulas with random inputs [4, 5, 17, 24], although the technical issues here are sufficiently different that the results in this body of work do not seem to be directly usable for our analysis.

In this section we obtain three generic bounds that are useful for estimating the decay rate of $\hat{\phi}_j$; we then apply these to the two cases of interest in Sections 4.1 and 4.2. The first claim gives bounds on $t'(x)$, the second gives bounds on $t(1)$, and the third combines these (together with the Mean Value Theorem) to show that the gap between a number in $[1 - \gamma, 1)$ and 1 (for small γ) widens by a constant factor each time t is iterated. In particular, this will be used to show that $\Theta(\log n)$ iterations of t are needed to reduce a quantity that is $1 - \Theta(n^{-1})$ to a quantity that is $1 - \Theta(1)$.

Claim 4.1 *Fix ε such that $\frac{1}{dn} < \varepsilon < 1$. If $x \in [1 - \varepsilon, 1]$ then $t'(x) \in [pb(1 - 2bd\varepsilon), pb]$.*

Proof. We first observe that $t'(x)$ is monotone in x over $[0, 1]$, so

$$t'(x) \le t'(1) = pqd(1 - q(1 - p))^{d-1} = pb(1 - q(1 - p))^{d-1} \le pb.$$

For the lower bound,

$$
\begin{aligned}
t'(x) &\ge t'(1 - \varepsilon) = pb(1 - q(1 - p(1 - \varepsilon)))^{d-1} = pb(1 - q(1 - (1 - \tfrac{1}{n})(1 - \varepsilon)))^{d-1} \\
&\ge pb(1 - q(\tfrac{1}{n} + \varepsilon))^{d-1} \ge pb(1 - 2qd\varepsilon)^{d-1} \ge pb(1 - 2qd^2\varepsilon) = pb(1 - 2bd\varepsilon). \qquad \blacksquare
\end{aligned}
$$

Claim 4.2 $1 - \frac{b}{n} \le t(1) \le 1 - \frac{1}{dn}$.

Proof. For the upper bound,

$$t(1) = (1 - q(1 - p))^d = (1 - \frac{q}{n})^d \ge 1 - \frac{qd}{n} = 1 - \frac{b}{n}.$$

For the lower bound,
$$t(1) = (1 - q(1 - p))^d = (1 - \frac{q}{n})^d \le 1 - \frac{q}{n},$$

and then we use the fact that $qd > 1$, and hence $q > 1/d$, to get $t(1) \le 1 - 1/dn$. $\blacksquare$

Claim 4.3 *Suppose p, b, and ε are such that $pb(1 - 2bd\varepsilon) > 1$, and let $0 < \gamma_0 < \gamma_1 \le \varepsilon$. Let $N(\gamma_0, \gamma_1)$ denote the number of iterations of the function t needed to reduce $1 - \gamma_0$ to a quantity that is $\le 1 - \gamma_1$. Then $N(\gamma_0, \gamma_1) = \Theta(\log(\gamma_1/\gamma_0))$.*

Proof. Since $t(1) \leq 1 - 1/dn$, we may assume that $\gamma_0 \geq 1/dn$, since this may be achieved with a single iteration of t. Consider some $x = 1 - \gamma$, where $1/dn \leq \gamma \leq \varepsilon$. By the Mean Value Theorem, we have $t(1) - t(x) = t'(y)(1 - x)$ for some $y \in [1 - \varepsilon, 1]$; rearranging terms, we get $t(x) = t(1) - t'(y)\gamma$. By Claim 4.1, we have $t'(y) \in [pb(1 - 2bd\varepsilon), pb]$. Now,

$$t(x) = t(1) - t'(y)\gamma \leq 1 - t'(y)\gamma \leq 1 - [pb(1 - 2bd\varepsilon)]\gamma$$

and

$$t(x) = 1 - t'(y)\gamma + (t(1) - 1) \geq 1 - t'(y)\gamma - \frac{b}{n} \geq 1 - (t'(y) + db)\gamma \geq 1 - (pb + db)\gamma.$$

Hence, each iteration of t widens the gap between the current value and 1 by at least $[pb(1-2bd\varepsilon)] > 1$ and at most $pb + db$. Thus, after k iterations of the function t, $1 - \gamma_0$ has been mapped to a number that is at most $1 - [pb(1 - 2bd\varepsilon)]^k\gamma_0$ and at least $1 - (pb + db)^k\gamma_0$. For this to drop below $1 - \gamma_1$, we have

$$\log(\gamma_1/\gamma_0)/\log(pb(1 - 2bd\varepsilon)) \leq N(\gamma_0, \gamma_1) \leq \log(\gamma_1/\gamma_0)/\log(pb + db). \qquad \blacksquare$$

4.1 The case when $b < 2$

For a fixed $b < 2$, we consider the sequence of $\hat{\phi}_j$ values up to the point at which it drops below $1 - \sigma_0$, for a small constant $\sigma_0 < \sigma$. We show that in this sequence, there is a first phase of constant length in which the ratio from the right-hand-side of Equation (4) may exceed 2; however, this is followed by a second phase of length $\Theta(\log n)$ in which the ratio is at most a constant $b_1 < 2$. We use this to show that the gaps between successive breakpoints grow geometrically for $\Theta(\log n)$ steps, yielding the lower bound in Theorem 4.6.

We choose a constant $\sigma_0 < \sigma$ small enough, and (since we are only concerned with asymptotics in n) n large enough, so that $pb(1 - 2bd\sigma_0) > 1$. We define the *first segment* of the sequence of $\hat{\phi}_j$ to be the set I_1 of indices j for which $\hat{\phi}_j \geq 1 - \kappa_0/n$ for a constant $\kappa_0 > b/(2 - b)$. (The need for this bound on κ_0 comes from the proof of Lemma 4.5.) We define the *second segment* to be the set I_2 of indices j for which $1 - \kappa_0/n > \hat{\phi}_j \geq 1 - \sigma_0$. The *third segment* is everything else. Claim 4.3 implies

Lemma 4.4 *The first segment has length $O(1)$ and the second segment has length $\Theta(\log n)$.*

We will argue that for each of the $\Theta(\log n)$ steps in the second segment, the gaps between successive utility values increase by at least a constant factor greater than 1. To this end we establish the following.

Lemma 4.5 *There is a constant $b_1 < 2$ such that for all j in the second segment we have* $\frac{1 - \hat{\phi}_{j+1}}{1 - \hat{\phi}_j} \leq b_1.$

Proof. Choose an index j in the second segment, and let $x = \hat{\phi}_j \in [1 - \sigma_0, 1 - \kappa_0/n]$. Then

$$\frac{1 - \hat{\phi}_{j+1}}{1 - \hat{\phi}_j} = \frac{1 - t(x)}{1 - x} = \frac{1 - t(1)}{1 - x} + \frac{t(1) - t(x)}{1 - x}$$

$$= \frac{1 - t(1)}{1 - x} + t'(y)$$

for some $y \in [x, 1]$ by the Mean Value Theorem. Now, by Claim 4.1, we have $t'(y) \le pb$. Also,

$$\frac{1 - t(1)}{1 - x} \le \frac{1 - (1 - \frac{b}{n})}{1 - (1 - \frac{\kappa_0}{n})} = \frac{b}{\kappa_0}.$$

Now, define $b_1 = pb + \frac{b}{\kappa_0}$. Since $b/\kappa_0 < 2 - b$, we have

$$\frac{1 - \hat{\phi}_{j+1}}{1 - \hat{\phi}_j} \le pb + \frac{b}{\kappa_0} = b_1$$

and

$$b_1 = pb + \frac{b}{\kappa_0} < b + (2 - b) = 2. \qquad \blacksquare$$

Since the gaps between reward breakpoints grow multiplicatively for $\Theta(\log n)$ steps, we have

Theorem 4.6 *There is a constant $c > 1$, depending on b, so that if $\hat{\phi}_j < 1 - \sigma$, then $u_j \ge n^c$. Hence $R_\sigma(n, b) \ge n^c$.*

Proof. Recall that $\Delta_j = u_j - u_{j-1}$. We have

$$\Delta_j = \frac{\Delta_j}{\Delta_{j-1}} \cdot \frac{\Delta_{j-1}}{\Delta_{j-2}} \cdots \cdots \frac{\Delta_3}{\Delta_2} \cdot u_1.$$

By Equation (4), if j belongs to the second segment of indices, then

$$\frac{\Delta_{j+1}}{\Delta_j} \ge \frac{\Delta'_{j+1}}{\Delta_j} \ge \frac{\Delta'_{j+1} - 1}{\Delta_j} = \frac{1}{\frac{1 - \hat{\phi}_{j+1}}{1 - \hat{\phi}_j} - 1} \ge \frac{1}{b_1 - 1} > 1.$$

Let $c_0 = \frac{1}{b_1 - 1}$. Now since the second segment has length $\ge \tau \log n$ for a constant $\tau > 0$, we have

$$
\begin{aligned}
u_j &\ge \Delta_j = u_1 \prod_{i=3}^{j} \frac{\Delta_i}{\Delta_{i-1}} \ge \prod_{i \in I_2} \frac{\Delta_i}{\Delta_{i-1}} \\
&\ge c_0^{|I_2|} \ge c_0^{\tau \log n} = n^{\tau \log c_0}. \qquad \blacksquare
\end{aligned}
$$

4.2 The case when $b > 2$

For a fixed $b > 2$, we consider the sequence of $\hat{\phi}_j$ values up to the point at which it drops below $1 - \sigma$. In this case, we show that there is a first phase of length $O(\log n)$ in which the utilities increase by additive increments of constant size. Following this, there is a second phase of constant length in which the gaps may grow geometrically. However, since this second phase has constant length, it does not prevent us from obtaining an $O(\log n)$ bound on $R_\sigma(n, b)$.

More concretely, we choose a constant $\sigma_0 < \sigma$ small enough, and n large enough, so that $pb(1 - 2bd\sigma_0) > 2$. We take the sequence of $\hat{\phi}_j$ (for $j > 1$) and divide the indices into three segments: a first segment J_1 of indices j for which $\hat{\phi}_j \geq 1 - \sigma_0$, a second segment J_2 of indices j for which $1 - \sigma_0 > \hat{\phi}_j \geq 1 - \sigma$, and a third segment J_3 consisting of everything else. By Claim 4.3, we have

Lemma 4.7 *The first segment has length* $\Theta(\log n)$.

Next we use the Mean Value Theorem to show

Lemma 4.8 *There is a constant $b_2 > 2$ such that for all j in the first segment we have $\dfrac{1 - \hat{\phi}_{j+1}}{1 - \hat{\phi}_j} \geq b_2$.*

Proof. Choose an index $j \geq 2$ in the first segment, and let $x = \hat{\phi}_j \in [1 - \sigma_0, 1 - \frac{1}{dn}]$. We observe that

$$\frac{1 - \hat{\phi}_{j+1}}{1 - \hat{\phi}_j} = \frac{1 - t(x)}{1 - x} \geq \frac{t(1) - t(x)}{1 - x} = t'(y)$$

for some $y \in [x, 1]$ by the Mean Value Theorem. Now, by Claim 4.1, we have $t'(y) \geq pb(1 - 2bd\sigma_0) > 2$. The claim now follows with $b_2 = pb(1 - 2bd\sigma_0)$. ∎

We bound the length of the second segment by showing that the number of iterations of t needed to reduce $1 - \sigma_0$ to $1 - \sigma$ is dominated by the convergence time of a certain branching process parameter, which in turn can be bounded as a constant in terms of $1 - \sigma_0$ and $1 - e_{q,d} - \sigma$, where we recall that $e_{q,d}$ is the extinction probability.

Lemma 4.9 *The second segment has length $O(1)$.*

Proof. We first define a function closely related to $t(x)$:

$$\bar{t}(x) = (1 - q(1 - x))^d = (qx + 1 - q)^d.$$

(It differs from $t(x)$ only in the lack of a coefficient of p on x.) $\bar{t}(x)$ is the generating function for the distribution of the number of children of a node in the branching process, and iterating it from any $x \in [0, 1)$ converges to the unique $e_{q,d} \in [0, 1)$ satisfying the fixed-point equation $\bar{t}(x) = x$. (Note that $x = 1$ is also a fixed point of $\bar{t}$.) We have the following simple chain of inequalities for $x \in (e_{q,d}, 1)$:

$$t(x) < \bar{t}(x) < x. \tag{5}$$

Now we define the following constants. For any $x < 1$, repeated iteration of $\bar{t}$ starting from x converges to $e_{q,d}$, and so for any $\varepsilon, \varepsilon' > 0$, the value $1 - \varepsilon'$ will be reduced to $\leq e_{q,d} + \varepsilon$ in a constant number of iterations, where the constant depends on ε and ε'. Let $\eta(\varepsilon, \varepsilon')$ denote this constant.

The length of the second segment is the number of iterations of t required to reduce a number $x \le 1 - \sigma_0$ to a quantity that is $\le 1 - \sigma$. By Inequality (5), this is no more than the number of steps required to reduce x to a quantity $\le 1 - \sigma$ by repeated iterations of $\bar{t}$. This latter number of steps is at most $\eta(1 - e_{q,d} - \sigma, \sigma_0)$. $\blacksquare$

Since the reward grows additively for $\Theta(\log n)$ steps, and then multiplicatively for $O(1)$ steps, we have

Theorem 4.10 *There is a constant c', depending on b and σ, so that for some j with $\hat{\phi}_j < 1 - \sigma$, we have $u_j \le c' \log n$. Hence $R_\sigma(n, b) = O(\log n)$.*

Proof. Let j_1 denote the index on which the first segment ends, and let j_2 denote the index on which the second segment ends. We first claim by induction that for all j in the first segment, we have $\Delta_j \le 2(b_2 - 1)/(b_2 - 2) = O(1)$. This holds for Δ_2; and supposing by induction that it holds up to Δ_j, we have

$$\frac{\Delta'_{j+1} - 1}{\Delta_j} = \frac{1}{\frac{1 - \hat{\phi}_{j+1}}{1 - \hat{\phi}_j} - 1} \le \frac{1}{b_2 - 1},$$

whence

$$\Delta_{j+1} \le \Delta'_{j+1} + 1 \le \frac{\Delta_j}{b_2 - 1} + 2 \le \frac{2(b_2 - 1)}{b_2 - 2}.$$

From Lemma 4.7, it follows that $\hat{\phi}_{j_1} < 1 - \sigma_0$ and $u_{j_1} = O(\log n)$. Now, for the second segment, $x - t(x)$ is positive on the closed interval $[1 - \sigma, 1 - \sigma_0]$, so by compactness it achieves a minimum that is a positive constant μ (depending on σ and σ_0). The second segment consists of a constant number of iterations, in each of which Δ_j can increase by at most a factor of

$$1 + \frac{1}{\frac{1 - \hat{\phi}_{j+1}}{1 - \hat{\phi}_j} - 1} \quad \le \quad 1 + \frac{1}{\hat{\phi}_{j_2} - \hat{\phi}_{j_2+1}}$$

$$= \quad 1 + \frac{1}{\hat{\phi}_{j_2} - t(\hat{\phi}_{j_2})} \le 1 + \mu^{-1}.$$

Thus, in the second segment, the gap between successive quantities u_j is multiplied by a constant for at most a constant number of iterations. Hence $u_{j_2} = O(\log n)$ as well. $\blacksquare$

5 Extensions and Further Directions

Further Directions. By simulation, we have found that the transition at $b = 2$ is already apparent for rarities n of moderate value. Analyzing the dependence of the reward on the rarity in the neighborhood of the critical value $b = 2$ is an interesting open question. In the special case of $b = 2$ when each node deterministically produces two offspring, we can show that utility $O(\log n)$ is sufficient (as in the case $b > 2$), but this does not immediately suggest a conjecture for the general case. Another question is the behavior of the lower bound as b approaches 1 from above. We can show that for query propagation on an infinite path (the deterministic version of $b = 1$), the reward needed grows as $\Omega(n!)$, but we do not have a detailed picture of the transition between this lower bound and the one in Theorem 4.6.

There are a number of promising further directions for investigation, including the analysis of query incentives for more complex queries (as Charikar et al. did for a single-player case [6]), and models of interaction that incorporate response time (as Etzioni et al. did for a non-network setting [8]). It is natural to conjecture that the transitional behavior at $b = 2$ in our model hints at a more general phenomenon related to the growth rate of the underlying network.

Directed Acyclic Graphs and a Model of Competition. Finally, we briefly discuss the issue of modeling query propagation on networks that are not trees. When queries are propagated on a tree, there is an implicit form of competition among nodes in different parts of the tree to be selected to receive the reward. For more general graphs, there is a further notion of competition, due to the existence of multiple paths from the node that poses to the query to each node that possesses the answer. We conclude by outlining a model to capture the effects of this type of competition.

Let G be a (finite or infinite) directed acyclic graph with a single *root node* v^*: there are no edges into v^*, and each other node is reachable by a path from v^*. By analogy with the terminology for trees, for a given node w, we refer to the nodes with an edge to w as its *parents*, and the nodes to which w has an edge as its *children*.

Suppose the root v^* offers an initial reward for the answer to a query, where the answer is held by each node independently with probability $1 - p$. As before, the query and accompanying rewards are propagated outward on paths from v^*. Now, in a general directed acyclic graph, a node w may be offered (possibly different) rewards by multiple parents. Thus, in addition to deciding what reward to offer, w must also decide which parent(s) to pass the answer back to in the event that it receives an answer from a child. Moreover, in setting the reward it offers, w must take into account that its children may be receiving offers of rewards from other parents as well, not just from w. If we suppose that v^* selects from among the responding answer-holders based purely on their identities, and not based on the paths by which these identities were propagated back to v^*, then it is not hard to argue by induction that, if there is any pure Nash equilibrium, then there is one in which every node only replies to a single parent offering the highest reward (breaking ties according to some fixed rule), rather than replying to multiple parents that offer rewards.

For this model, however, we do not know whether there exists a pure Nash equilibrium set of strategies in every directed acyclic graph. While we can show the existence of pure Nash equilibria for special cases, via specialized arguments, we believe that understanding the nature of equilibria for this model in general — and the consequences for efficiency of query propagation — is an interesting direction for further work.

References

[1] L. A. Adamic, O. Buyukkokten and E. Adar. A social network caught in the Web. *First Monday* 8:6(2003).

[2] K.B. Athreya and P.E. Ney. Branching Processes. Springer, 1972.

[3] A. Blanc, Y-K. Liu and A. Vahdat. Designing Incentives for Peer-to-Peer Routing. *2nd Workshop on Economics of Peer-to-peer systems*, 2004.

[4] A. Broder, R. Krauthgamer and M. Mitzenmacher. Improved Classification via Connectivity Information. *Proc. 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2000.

[5] A. Brodsky and N. Pippenger. The Boolean Functions Computed by Random Boolean Formulas OR How to Grow the Right Function. UBC Computer Science Technical Report, TR-2003-02, 2003.

[6] M. Charikar, R. Fagin, V. Guruswami, J. Kleinberg, P. Raghavan and A. Sahai. Query Strategies for Priced Information. *Proc. 32nd ACM Symposium on Theory of Computing*, 2000.

[7] I. Clarke, O. Sandberg, B. Wiley and T. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. *International Workshop on Design Issues in Anonymity and Unobservability*, 2000.

[8] O. Etzioni, S. Hanks, T. Jiang, R.M. Karp, O. Madani and O. Waarts. Efficient information gathering on the Internet. *Proc. IEEE Symposium on Foundations of Computer Science*, 1996.

[9] M. Granovetter. The strength of weak ties. *American Journal of Sociology*, **78**(6):1360-1380, 1973.

[10] M. Jackson. A Survey of Models of Network Formation: Stability and Efficiency. In *Group Formation in Economics: Networks, Clubs and Coalitions*. G. Demange and M. Wooders, eds. Cambridge, 2004.

[11] S. Kakade, M. Kearns, L. Ortiz, R. Pemantle and S. Suri. Economic Properties of Social Networks. *Proc. NIPS*, 2004.

[12] S. Kamvar, B. Yang and H. Garcia-Molina. Addressing the Non-Cooperation Problem in Competitive P2P Systems. *1st Workshop on Economics of Peer-to-peer systems*, 2003.

[13] H. Kautz, B. Selman and M. Shah. ReferralWeb: Combining Social Networks and Collaborative Filtering. *Communications of the ACM*, 1997.

[14] R. Krishnan, M. Smith and R. Telang. The Economics of Peer-to-Peer Networks. SSRN Working Paper, September 2003.

[15] C. Li, B. Yu and K. Sycara. An Incentive Mechanism for Message Relaying in Peer-to-Peer Discovery. *2nd Workshop on Economics of Peer-to-peer systems*, 2004.

[16] E-K Lua, J. Crowcroft, M. Pias, R. Sharma and S. Lim. A Survey and Comparison of Peer-to-Peer Overlay Network Schemes, *IEEE Communications Survey and Tutorial*, March 2004

[17] M. Luby, M. Mitzenmacher and A. Shokrollahi. Analysis of Random Processes via And-Or Tree Evaluation. *Proc. 9th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1998.

[18] C. H. Papadimitriou. Algorithms, Games, and the Internet. *Proc. 33rd ACM Symposium on Theory of Computing*, 2001.

[19] B. Raghavan and A. Snoeren. Priority Forwarding in Ad Hoc Networks with Self-Interested Parties. *1st Workshop on Economics of Peer-to-peer systems*, 2003.

[20] P. Resnick, R. Zeckhauser, E. Friedman and K. Kuwabara. Reputation Systems. *Communications of the ACM*, **43**(12):45-48, 2000.

[21] A. Salkever, in *BusinessWeek*, 28 October 2003.

[22] B.A. Sevastyanov. Asymptotic Behavior of the Extinction Probabilities for Stopped Branching Processes. *Theory Prob. Appl.* **43**(2):315-322, 1999.

[23] É. Tardos. Network Games. *Proc. 36th ACM Symposium on Theory of Computing*, 2004.

[24] L.G. Valiant. Short monotone formulae for the majority function. *Journal of Algorithms,* **5**:363-366, 1984.

[25] V. Vishnumurthy, S. Chandrakumar and E.G. Sirer. KARMA : A Secure Economic Framework for P2P Resource Sharing. *1st Workshop on Economics of Peer-to-peer systems*, 2003.

[26] A. Vivacqua and H. Lieberman. Agents to Assist in Finding Help. *Proc. ACM SIGCHI Conf. on Human Factors in Computing Systems*, 2000.

[27] B. Wellman. The Three Ages of Internet Studies: Ten, Five and Zero Years Ago. *New Media and Society*, **6**(1):123-129, 2004.

[28] B. Yu and M. P. Singh. Searching Social Networks. *Proc. 2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 2003.

[29] B.Yu and M.P. Singh. Incentive Mechanisms for Peer-to-Peer Systems. *Proc. 2nd International Workshop on Agents and Peer-to-Peer Computing*, 2003.

[30] J. Zhang and M. Van Alstyne. SWIM: fostering social network based information search. *Proc. ACM SIGCHI Conf. on Human Factors in Computing Systems.* 2004.

Appendix: Existence and Uniqueness Results for Equilibrium

Proof of Theorem 2.1.   For arguing that this is a Nash equilibrium, we seek to establish that the quantity $(r - x - 1)\alpha_v(\mathbf{g}, x)$ is essentially the payoff to node v when it chooses to offer a reward of x. The proof is complicated by the fact that this is not quite true — the query might not make it to v; the reward might not be given to v even when it finds an answer; and v receives a value of $r - 1$ rather than $r - x - 1$ if it in fact holds the answer. The crux of the proof lies in the fact that $(r - x - 1)\alpha_v(\mathbf{g}, x)$ is proportional to the portion of the payoff over which v "has control," in a certain sense.

Given the initial utility r^* at the root, and a choice of functions $\mathbf{f}$ at each node, we can define three events: C, that the query reaches v; B, that an answer is found in the subtree below v (including v); and A, that the reward is propagated down to v. We also define an event D, that v holds the answer. If $Y_{\mathbf{f},r^*}$ is a random variable denoting the payoff to v, given $\mathbf{f}$ and r^*, then $E\left[Y_{\mathbf{f},r^*}\right]$ can be computed by conditioning on all 2^4 Boolean combinations of outcomes for A, B, C, and D, and taking the sum of these weighted by the probability of this combination. That is,

$$E\left[Y_{\mathbf{f},r^*}\right] = \sum_{(A',B',C',D')} E\left[Y_{\mathbf{f},r^*} \mid A' \cap B' \cap C' \cap D'\right] \cdot \Pr\left[A' \cap B' \cap C' \cap D'\right],$$

17

where A' denotes one of A or $\overline{A}$, and likewise for B', C', and D'. Of these terms, we know that

$$E\left[Y_{\mathbf{f},r^*} \mid \overline{A}\right] = E\left[Y_{\mathbf{f},r^*} \mid \overline{C}\right] = 0,$$

and $\Pr\left[A \cap \overline{B}\right] = 0$, so the only terms for which $E\left[Y_{\mathbf{f},r^*} \mid A' \cap B' \cap C' \cap D'\right] \cdot \Pr\left[A' \cap B' \cap C' \cap D'\right]$ is non-zero are the Boolean combinations $A \cap B \cap C \cap D$ and $A \cap B \cap C \cap \overline{D}$. Thus we have

$$\begin{aligned}
E\left[Y_{\mathbf{f},r^*}\right] &= E\left[Y_{\mathbf{f},r^*} \mid A \cap B \cap C \cap D\right] \cdot \Pr\left[A \cap B \cap C \cap D\right] \\
&\quad + E\left[Y_{\mathbf{f},r^*} \mid A \cap B \cap C \cap \overline{D}\right] \cdot \Pr\left[A \cap B \cap C \cap \overline{D}\right].
\end{aligned}$$

Now,

$$\Pr\left[A \cap B \cap C \cap \overline{D}\right] = \Pr\left[A \mid B \cap C \cap \overline{D}\right] \cdot \Pr\left[B \mid C \cap \overline{D}\right] \cdot \Pr\left[C \cap \overline{D}\right],$$

and hence

$$\begin{aligned}
E\left[Y_{\mathbf{f},r^*}\right] &= E\left[Y_{\mathbf{f},r^*} \mid A \cap B \cap C \cap D\right] \cdot \Pr\left[A \cap B \cap C \cap D\right] \\
&\quad + E\left[Y_{\mathbf{f},r^*} \mid A \cap B \cap C \cap \overline{D}\right] \cdot \Pr\left[A \mid B \cap C \cap \overline{D}\right] \cdot \Pr\left[B \mid C \cap \overline{D}\right] \cdot \Pr\left[C \cap \overline{D}\right].
\end{aligned}$$

Let $\rho_v(\mathbf{f}, r^*)$ denote the reward that is offered to v if the set of reward functions is $\mathbf{f}$, the root's utility is r^*, and the reward is propagated all the way to v. Suppose $\rho_v(\mathbf{f}, r^*) = r$. We then have

$$E\left[Y_{\mathbf{f},r^*} \mid A \cap B \cap C \cap D\right] = r - 1,$$

and

$$E\left[Y_{\mathbf{f},r^*} \mid A \cap B \cap C \cap \overline{D}\right] = r - f_v(r) - 1.$$

Also, by definition,

$$\Pr\left[B \mid C \cap \overline{D}\right] = \alpha_v(\mathbf{f}, f_v(r)).$$

It follows that our definition of the functions $\mathbf{g}$ above chooses $g_v(r)$ so as to maximize

$$E\left[Y_{\mathbf{g},r^*} \mid A \cap B \cap C \cap \overline{D}\right] \cdot \Pr\left[B \mid C \cap \overline{D}\right].$$

In the expression for $E\left[Y_{\mathbf{g},r^*}\right]$, the choice of $f_v(r)$ does not affect any of the remaining terms, each of which is positive, and hence offering the reward $g_v(r)$ in fact maximizes $E\left[Y_{\mathbf{g},r^*}\right]$. Thus, any deviation by v from reward $g_v(r)$ to some other offered reward $h_i(r)$ cannot increase the payoff, and hence $\mathbf{g}$ is a Nash equilibrium. $\blacksquare$

Proof of Theorem 2.2. By definition we must have $f_v(1) = 0 = g_v(1)$, and by assumption we have $f_v(2) = 1 = g_v(2)$. Now, suppose by way of contradiction that the statement of the theorem does not hold for $\mathbf{f}$, and let $r > 2$ be the smallest reward that is reachable at v with respect to $\mathbf{f}$, and for which $f_v(r) \neq g_v(r)$.

In the proof of Theorem 2.1, we establish that the payoff to v from offering x in response to r has the form $\psi + \xi(r - x - 1)\alpha_v(\mathbf{f}, x)$, where ψ and ξ are expressions that do not depend on x. Since all functions in $\mathbf{f}$ agree with $\mathbf{g}$ on values below r, we can write this as $\psi + \xi(r - x - 1)\alpha_v(\mathbf{g}, x)$.

We know that setting $x = g_v(r)$ maximizes this expression, so if $\mathbf{f}$ is also a Nash equilibrium then $f_v(r)$ must also maximize this expression, since r is reachable. Thus if we view

$$(r - g_v(r) - 1)\alpha_v(\mathbf{g}, g_v(r)) - (r - f_v(r) - 1)\alpha_v(\mathbf{g}, f_v(r))$$

as a bivariate polynomial in variables p and q, then it has rational coefficients, it is not identically zero (since $f_v(r) \neq g_v(r)$, and neither is 0 for $r > 2$), and it has (p, q) as a zero. This contradicts our assumption that p is generic with respect to q. $\blacksquare$