On Threshold Behavior in Query Incentive Networks

Esteban Arcaute[*†]
Stanford University
Stanford, CA 94305
arcaute@stanford.edu

Adam Kirsch[*‡]
Harvard University
Cambridge, MA 02138
kirsch@eecs.harvard.edu

Ravi Kumar
Yahoo! Research
Sunnyvale, CA 94089
ravikumar@yahoo-inc.com

David Liben-Nowell[*]
Carleton College
Northfield, MN 55057
dlibenno@carleton.edu

Sergei Vassilvitskii[*§]
Stanford University
Stanford, CA 94305
sergei@cs.stanford.edu

ABSTRACT

Motivated by the role of incentives in large-scale information systems, Kleinberg and Raghavan (FOCS 2005) studied strategic games in decentralized information networks. Given a branching process that specifies the network, the rarity of answers to a specific question, and a desired probability of success, how much reward does the root node need to offer so that it receives an answer with this probability, when all of the nodes are playing strategically? For a specific family of branching processes and a constant failure probability, they showed that the reward function exhibited a threshold behavior that depends on the branching parameter b.

In this paper we study two factors that can contribute to this transition behavior, namely, the branching process itself and the failure probability. On one hand we show that the threshold behavior is robust with respect to the branching process: for *all* branching processes and any constant failure probability, if $b > 2$ then the required reward is linear in the expected depth of the search tree, and if $b < 2$ then the required reward is exponential in that depth. On the other hand we show that the threshold behavior is fragile with respect to the failure probability σ: if σ is inversely polynomial in the rarity of the answer, then *all* branching processes require rewards exponential in the depth of the search tree.

Categories and Subject Descriptors

G.2 [**Mathematics of Computing**]: Discrete Mathematics; G.3 [**Mathematics of Computing**]: Probability and Statistics; J.4 [**Computer Applications**]: Social and Behavioral Sciences—*Economics*

General Terms

Economics, Theory

Keywords

query incentive networks, threshold phenomena, branching processes

[*]Work done in part while visiting Yahoo! Research.
[†]Supported in part by NSF grant CMMI-0620811.
[‡]Supported in part by an NSF Graduate Research Fellowship.
[§]Supported in part by a Microsoft Research Live Labs fellowship and NSF Grant ITR-033164.

1. INTRODUCTION

In a world where billions of web pages are at one's fingertips, the quest for information is no longer constrained by a lack of accessible data. Instead, one of the new prevailing limitations is that of *reliability.* Why would a web-page author carefully write a thoroughly investigated essay when there is little recourse if its facts turn out to have been fabricated? Why would a stranger give an honest appraisal of a product when he might be paid by the manufacturer to speak highly of it? Why should a user trust the information that she receives via a web page?

One solution to this issue is to rely on the implicit and trusted connections between friends to "endorse" the information, rather than trusting a stranger. Examples of this approach include *word-of-mouth* assessment of products, where consumers weigh their friends' opinions disproportionately heavily in choosing among products, and the more intentional *viral marketing*, in which companies encourage consumers to convince their friends to purchase the same products (e.g., [3,4,10,11,13,15]). Seeking information only from one's immediate friends vastly limits the pool of candidate experts, but one might trust one's friends to answer a query by asking their friends, who might in turn ask *their* friends, and so on. The social network serves as an implicit "web of trust" for endorsing acquired information: an answer to a question comes equipped with a path of trusted relationships in the social network connecting the querier to the answer (e.g., [1, 5–8, 14, 16]). In fact, explicit systems to connect information seekers with domain experts through a chain of trusted relationships have been introduced both in research (e.g., [9,17,18]) and commercial (e.g., LinkedIn, Visible Path) settings. Similarly motivated applications arise within the peer-to-peer domain, in which participants look to find a short chain to a peer with a sought piece of information, such as a music file.

Of course, unlike one's direct friends, one's friends' friends' friends have only the mildest of social incentives to answer one's questions. In the peer-to-peer domain, for example, this lack of incentive leads to the "free-rider" problem, which is potentially debilitating for these systems: if many users join a peer-to-peer system looking only to acquire information, without sharing any of their own knowledge, then there will be essentially no information to exchange. An explicitly economic perspective has proven extremely valuable to combat this problem, by directly incentivizing participation. In our information-seeking context, one can imagine offering a reward to one's friends for an answer to a query. For a sufficiently high offered reward, if one's friends cannot directly answer the question, they can choose to act as middlemen, seeking to acquire an answer by offering their friends a cut of the reward (and saving the rest for themselves).

In such a setting, it becomes useful to think of the agents involved in the system as buyers and sellers of information, who act strategically to try to maximize their profit. Suppose that Alice is offered a "finder's fee" of r dollars for information that she does not possess. There is a natural tradeoff in this context: she can choose to offer a small amount ($\ll r$) to her friends, hoping that they will provide the sought information at this relatively cheap price (allowing Alice to pocket the large difference between her receipt and outlay), or she can make a larger offer ($\approx r$) to her friends, taking a lower profit margin in exchange for the presumably increased odds of her friends being willing and able to produce the sought information.

Query incentive networks.

Motivated by these types of scenarios, Kleinberg and Raghavan introduced *query incentive networks* as a model of information acquisition in a strategic networked environment [12]. We give a brief overview of that work here, deferring the technical details until Section 2. Consider a *root* node v_{root} in a large network who wishes to learn an *answer* to a particular *query* (e.g., answer a particular question, find a person who can provide a particular service, etc.), and is willing to pay for such an answer. We seek an analytically tractable model that captures the tradeoff between the amount that v_{root} is willing to pay for an answer and the likelihood that she obtains an answer.

Kleinberg and Raghavan [12] introduce such a model. For analytical convenience, they first model the network as an infinite tree with v_{root} as the root. To capture the key tradeoff described above, they introduce a natural game-theoretic foundation, with players in the game representing nodes in the network. Then they examine the Nash equilibrium behavior of this game.

The model is as follows. We assume that every node has an answer to the query independently with probability $1 - p = 1/n$, and that the root node v_{root} would like to acquire an answer with probability at least $\sigma \in (0, 1)$. Here n is the *rarity* of an answer; it is the expected number of nodes that the root must query to obtain an answer. The root v_{root} offers each of its neighbors in the network (that is, children in the tree) a *reward* r for an answer. (Note that the quantities r and σ are fixed in advance, and are not determined by any strategic behavior of the root; in fact, the root has no strategic behavior and exists only for the purpose of formulating the model.) We assume that rewards are scaled so that there is a unit cost for sending an answer to one's parent. Thus if a child of the root has an answer, it can simply return the answer to v_{root}, pocketing the reward r less the unit cost of returning the answer. (For the purposes of tie breaking, if more than one answer is found v_{root} chooses arbitrarily among all received answers. The cost of participation in the offer process is 0 and thus having an answer that is not selected results in zero payoff.) A child v that does not have an answer can propagate the query to its children in turn, offering some integral reward $0 \leq f_v(r) \leq r - 1$. If node v receives an answer from one of its children, then v will propagate that answer to its parent. If that answer is selected by the root, then v will receive a payoff of $r - f_v(r) - 1$: node v receives r from its parent, pays $f_v(r)$ to the child that returned an answer, and pays unit cost to propagate that child's answer to its parent.

Under this setup, nodes choose $f_v(\cdot)$ strategically to maximize their expected payoffs. Note that a node that is offered a reward of zero will refuse to participate, for its utility would decrease by returning an answer. Furthermore, as a concrete tie-breaking rule, we assume that a node that is offered a reward of one will always forward an answer to its parent, even though the one unit of reward it receives would be spent on passing the answer.

We have now defined the network model and the corresponding game, and so it would appear that we are ready to move to analyzing the Nash equilibrium properties of this system. However, as pointed out by Kleinberg and Raghavan, the use of a deterministic network model has a major deficiency. Indeed, a node in a real network is typically aware only of the local structure of the network (e.g., its neighbors, and possibly its neighbor's neighbors). However, if we are interested in the Nash equilibrium behavior of a game played by the nodes in a deterministic network, then there is a hidden assumption that every node is able to adjust its strategy with respect to any other node; in effect, every node knows the entire network.

To deal with this technical issue, Kleinberg and Raghavan introduce some randomness into the network model. To specify the network, consider a full d-ary tree where every node is *active* independently with some probability q. The network consists of the subgraph induced by all active nodes in the root's connected component. Nodes must form their strategies before the set of active nodes is revealed. Thus, the information that any node has about its connectivity to any other node decreases rapidly with the original distance between them in the tree, which is a natural property for the model to have. (Of course, in the Nash equilibrium setting, a node can still adjust its strategy in response to the strategy of every other node, but now it must do so under some uncertainty about which nodes actually participate in the game.)

Kleinberg and Raghavan show that, under some mild technical restrictions, there is a unique pure strategy Nash equilibrium, which is symmetric. Let $g(\cdot) = g_v(\cdot)$ denote the settings of the $f_v(\cdot)$'s that form this Nash equilibrium. The key tradeoff described above can now be phrased as follows: for a fixed $\sigma \in (0, 1)$, how large must r be (as a function of n) in order for v_{root} to receive an answer to its query with probability at least σ, assuming that all nodes are playing according to the Nash equilibrium? That is, if v_{root} wants to receive an answer with probability at least σ, how much must it pay as answers become increasingly rare ($n \to \infty$)?

	constant σ	$\sigma \geq 1 - 1/n$
$b < 1$	dies w.p. 1	
$b = 1$	$\Omega(n!)$ if $c_1 = 1$ (Thm. 5); otherwise dies w.p. 1	
$1 < b < 2$	$n^{\Theta(1)}$ (Thm. 4 & 8)	$n^{\Theta(1)}$
$b > 2$	$O(\log n)$ (Thm. 4)	(Thms. 9 & 10)

Table 1: Statement of results for arbitrary branching processes.

Recall that every node in the tree is present independently with probability q. Then the number of active children of any particular node in the original d-ary tree follows a binomial distribution $\mathsf{Bin}(d, q)$—i.e., the probability that a node has exactly i children is given by $\binom{d}{i} \cdot q^i \cdot (1-q)^{d-i}$. Thus, the network can be thought of as a branching process, starting with v_{root}, where each node has a number of offspring nodes chosen independently from the distribution $\mathsf{Bin}(d, q)$. The expected number of offspring of a node is then $b := q \cdot d$. It is a classical result in the theory of branching processes that there is a phase transition at $b = 1$: if $b < 1$, then the process almost surely dies with only finitely many nodes, while if $b > 1$ then infinitely many nodes are produced with constant (non-zero) probability [2]. Thus for $b < 1$ there is no amount that the root can offer to guarantee an answer with probability $\geq \sigma$, because for sufficiently large n, with probability exceeding $1-\sigma$ there simply will not be any node connected to v_{root} that has the answer. But the remarkable and unexpected result of Kleinberg and Raghavan is that there is a *second* phase transition in the reward that the root has to offer to receive an answer with probability $\geq \sigma$, for a constant σ not exceeding the extinction probability of the branching process:

- for $q \cdot d < 2$, the root must offer a reward $r = \Omega(n^{\varepsilon})$, for some constant $\varepsilon > 0$, to receive an answer with probability $\geq \sigma$.
- for $q \cdot d > 2$, the root receives an answer with probability $\geq \sigma$ by offering reward $r = O(\log n)$.

The unexpected result here is that, for $1 < b < 2$, with constant probability the network is an infinite tree, in which case there is an answer within the first $O(\log n)$ levels of the tree with high probability. But in this range, the root must offer a reward *exponentially* larger than the length of the path from v_{root} to the nearest answer. The required reward then decreases drastically for $b > 2$, and there does not appear to be a phase transition in the underlying branching process that can easily explain this threshold. Thus, there appears to be a fundamental difference between networks that support efficient answering of queries in principle and those that support efficient answering of queries in the presence of incentives.

Our results: Arbitrary branching processes.

The class of network models studied by Kleinberg and Raghavan is natural but highly restricted. In particular, their assumption that every node is active independently is convenient, in that it allows them to think of the network as being generated by a branching process with a specific offspring distribution ($\mathsf{Bin}(d, q)$). However, since their main result is ultimately interesting largely because of a contrast with the standard theory of branching processes, one might wonder whether it can be generalized to other types of branching processes.

We show that this is indeed the case for *arbitrary* branching processes. As above, let d denote the (finite) maximum number of children of a node. Let c_i denote the probability of a node having exactly i children for $0 \leq i \leq d$, and let $b := \sum_{i=0}^{d} i \cdot c_i$ denote the expected number of children of a node. We show that the threshold behavior observed by Kleinberg and Raghavan—that is, a phase transition at $b = 2$ in the required payment that the root must make to receive an answer with constant probability $\sigma \in (0, 1)$—is robust with respect to the branching process. Specifically, we show that for $b > 2$, the required reward is linear in the expected depth of the search tree (and hence $O(\log n)$); and for $b < 2$, the required reward is exponential in that depth and hence $n^{\Theta(1)}$. We also show that when $b = 1$—that is, right at the cusp of the phase transition in the branching process—the required payment is superexponential in the (linear in n) expected depth of the search "tree": when the branching process is just a ray (i.e., $c_1 = 1$), the root must pay $[\Omega(n)]!$ to receive an answer with constant probability.

These results are interesting for a variety of reasons. Perhaps most potently, they tell us that the threshold behavior discovered by Kleinberg and Raghavan is a much more general phenomenon than it may seem at first glance. In particular, since the threshold condition is so simple, our results suggest that these sorts of phenomena may have a more intuitive probabilistic interpretation. Such an interpretation may well prove crucial to the further development of the theory of query incentive networks, as more insights are needed to make quantitative statements for more general classes of network models.

Our results: Decreased tolerance for failure.

In both the original results of Kleinberg and Raghavan and in the results described above, it is assumed that the lower bound σ on the probability that a root receives an answer is held constant as $n \to \infty$. In this paper we also consider the case where the $\sigma = 1 - o(1)$. We show that, surprisingly, the $b = 2$ threshold described above completely vanishes when we demand a success probability $\sigma \geq 1 - 1/n$. In particular, for *any* branching process, the root must offer a reward of $n^{\Omega(1)}$ in order to acquire an answer with probability $\geq 1 - 1/n$. We also show that this result is tight: for any branching process that is guaranteed not to die (i.e., $c_0 = 0$) and is not a ray (i.e., $b > 1$), the root can offer a reward of $n^{O(1)}$ to achieve success probability $\geq 1 - 1/n$. (The case of a ray, as described above, requires a reward exponential in n.) Thus, although it is robust to arbitrary changes to the branching process, the threshold is fragile with respect to the failure probability: if the desired success probability σ is inversely polynomial in n, then *all* branching processes require rewards exponential in the depth of the search tree.

Our results are summarized in Table 1.

2. TECHNICAL PRELIMINARIES

In this section we formalize some of the more technical aspects of query incentive networks in order to define the main parameters and concepts that we use throughout the paper.

n	*rarity*; each node has an answer w.p. $1/n$
p	probability of *not* having answer $= 1 - 1/n$
d	max # of children of a node
c_i	prob. that # of children $= i$ $\quad 0 \le i \le d$
q	in [12], each of d children is *active* w.p. q
b	expected # children of a node $b = \sum_{i=1}^{d} i \cdot c_i$ (in [12], $b = qd$)
σ	required probability root receives an answer
σ_0, κ_0	sufficiently small constants < 1
$g_v(r)$	in equilibrium, when v is offered reward r and v has no answer, v offers $g_v(r)$ to its children
$g(r)$	for a symmetric equilibrium, write $g(r)$ for $g_v(r)$
$\hat{\phi}_j$	prob. no node at level $\le j$ of tree has an answer
$\delta(r)$	min # of iterations of $g(\cdot)$ to go from r to 0
u_j	jth "reward breakpoint": $u_j = \min_r\{\delta(r) \ge j\}$
Δ'_j	point of indifference in expected payoff between setting $g(u_j + \Delta'_j) = u_{j-1}$ and $g(u_j + \Delta'_j) = u_j$
Δ_j	gap between breakpoints; $\Delta_j = \lceil \Delta'_j \rceil = u_j - u_{j-1}$
$R_\sigma(n, b)$	reward necessary to find answer w.p. σ given n, b
$\overline{T}$	generating function for the branching process
$z_{\overline{T}}$	prob. that the branching process becomes extinct

Table 2: Various notation used throughout.

2.1 The reward structure in query incentive networks

Recall that the query incentive network model is characterized by a branching process representing the friendship or trust network on which the queries are propagated. We have explained the game in the previous section, and following the proof in [12], we can show that subject to minor technical conditions, a pure Nash equilibrium always exists and is unique.

Let $\hat{\phi}_j$ be the probability that no node in the first j levels of the tree (excluding the root) has the answer. We show that the amount of reward the root needs to offer is tightly connected to the rate of decay of the $\hat{\phi}_j$'s. To understand this connection, we delve deeper into the Nash equilibrium conditions for each node.

To understand the strategy space of a node v, fix the actions of all other nodes. Node v is offered a reward r by its parent. If v has the answer, it keeps the reward, and returns the answer to the parent. If v does not have the answer then it must make some offer $g_v(r)$ to its children for an answer. Since the Nash equilibrium is unique, and the information set of all nodes is identical, the Nash equilibrium is symmetric and $g_v(r)$ is independent of v; we refer to this function of r as $g(\cdot)$.

Nodes being rational, $g(r)$ is non-decreasing, and we can define $\delta(r)$ as the minimum number of times we need to iterate $g(\cdot)$ starting at r to reach 0. Observe that $\delta(r)$ is precisely the maximum number of levels in the search tree the query will reach if the root offers reward r. We define u_j to be the minimum r such that $\delta(r)$ is at least j.

We are now ready to state the Nash condition that occurs at each node. The node is offered some reward r, and offers a value s to its children. Remembering the transaction cost of 1, the expected payoff is:

$$(r - 1 - s) \cdot (1 - \hat{\phi}_{\delta(s)}).$$

Since every node intends to maximize its payoff, the value for s should come from the set $\{u_1, u_2, \ldots, u_{\delta(r)}\}$. If not, the value for s can be reduced without affecting the success probability, thereby increasing the total payoff.

We can build up the value of u_j inductively. We define $u_1 = 1$. By the way we have set up the problem, a node receiving a reward offer of u_j will propagate the query j levels into the tree, and achieve a success probability of $1 - \hat{\phi}_j$. Consider what happens to the expected payoff as we increase the reward offered to the node from u_j. At u_j a node will offer u_{j-1} to its children, thereby obtaining an expected reward of:

$$(u_j - u_{j-1} - 1)(1 - \hat{\phi}_{j-1}).$$

This will assure v that the query is propagated j more levels into the tree. However, at some point, there will be a value c such that

$$(u_j + c - u_{j-1} - 1)(1 - \hat{\phi}_{j-1}) = (u_j + c - u_j - 1)(1 - \hat{\phi}_j);$$

this can be seen by comparing the growth rates of the two quantities. In other words, it will be more advantageous for v to keep less money for itself—namely, $(c-1)$ as opposed to $(u_j - u_{j-1} + c - 1)$—because the probability of success has increased from $1 - \hat{\phi}_{j-1}$ to $1 - \hat{\phi}_j$. But this value of $u_j + \lceil c \rceil$ (since all money is integral) is by definition u_{j+1}.

Let $\Delta'_j = c$ and $\Delta_j = \lceil \Delta'_j \rceil = u_j - u_{j-1}$. Rewriting, we have

$$\begin{aligned}(u_j + c - u_{j-1} - 1)(1 - \hat{\phi}_{j-1}) &= (c - 1)(1 - \hat{\phi}_j)\\ (\Delta'_{j+1} + \Delta_j - 1)(1 - \hat{\phi}_{j-1}) &= (\Delta'_{j+1} - 1)(1 - \hat{\phi}_j)\\ \frac{\Delta_j}{\Delta'_{j+1} - 1} + 1 &= \frac{1 - \hat{\phi}_j}{1 - \hat{\phi}_{j-1}}.\end{aligned}$$

This leads us to an important equation on the growth of rewards:

$$1 + \frac{\Delta_j}{\Delta_{j+1} - 1} \le \frac{1 - \hat{\phi}_j}{1 - \hat{\phi}_{j-1}} \le 1 + \frac{\Delta_j}{\Delta_{j+1} - 2},$$

implying that (assuming $\Delta_{j+1} \ge 2$):

$$\frac{1}{\frac{1-\hat{\phi}_j}{1-\hat{\phi}_{j+1}} - 1} \le \frac{\Delta_{j+1}}{\Delta_j} \le \frac{1}{\frac{1-\hat{\phi}_j}{1-\hat{\phi}_{j+1}} - 1} + 1. \qquad (1)$$

In the rest of the paper, we focus on bounding the reward needed for the root to achieve success with a certain probability. Let $R_\sigma(n, b)$ denote the reward necessary to find the answer with probability at least σ given that the rarity is n and that the expected number of children is b. (Recall that the rarity $n = 1/(1-p)$ is the expected number of nodes that the root must query to obtain an answer.) From the discussion above it is clear that we can write

$$R_\sigma(n, b) = \min_{j : 1 - \hat{\phi}_j > \sigma} u_j.$$

This equation allows us to study the asymptotic growth of $R_\sigma(n, b)$ by examining (1), which is our approach in the rest of the paper.

2.2 The general branching process model

We now focus our attention on the randomness used to generate the network. In their original model, Kleinberg and Raghavan begin with an infinite d-ary tree where each node is present independently with probability q. One can view this network as a result of a branching process where the number of children of each node follows a binomial distribution $\mathsf{Bin}(d, q)$—i.e., the probability c_i that a node has exactly i children is given by $\binom{d}{i} \cdot q^i \cdot (1-q)^{d-i}$. Note that the expected branching factor $b = d \cdot q$ is the crucial parameter for the branching process.

In this work, we begin again with a full d-ary tree, but instead of deleting nodes independently, we allow for dependencies between nodes lying at the same depth. In our analysis we assume that d is a constant, with $d \geq 2$ to allow the network to be something other than a ray (that is, a directed path from the root). We model the network as a result of an *arbitrary* branching process [2]. More formally, we assume that any particular node has exactly i active children with probability c_i, and that the numbers of active children for different nodes are independent.

The generating function of the branching process is then given by

$$\overline{T}(x) = \sum_{i=0}^{d} c_i x^i.$$

We let b denote the expected number of (active) children of a node:

$$b = \sum_{i=0}^{d} i c_i.$$

Since a node has a non-zero probability of having no children, the overall branching process has a constant probability of dying out. We denote this probability by $z_{\overline{T}}$. Recall the classical result that, when $b < 1$, or $b = 1$ with $c_1 < 1$, the branching process dies out almost surely, and when $b > 1$ the branching process survives indefinitely with non-zero probability [2].

Now, each node u in the network possesses an answer independently with probability $1 - p = 1/n$. We seek to analyze the asymptotics of the reward as a function n, the rarity of an answer.

Consider a given node u in the tree. We define the function

$$t(x) = \overline{T}(px) = \sum_{i=0}^{d} c_i x^i p^i,$$

so that

$$1 - t(1) = 1 - \overline{T}(p) = 1 - \sum_{i=0}^{d} c_i p^i$$

is the probability of getting an answer by querying a single layer of the tree. Furthermore, recalling that $\hat{\phi}_j$ is the probability that no node in first j levels has an answer, we have that

$$\hat{\phi}_j = t\left(\hat{\phi}_{j-1}\right) = t\left(t\left(\hat{\phi}_{j-2}\right)\right) = \cdots = t^{(j)}(1),$$

where $t^{(j)}(x)$ is the value obtained by iterating the function t at value x a total of j times.

3. PERSISTING THRESHOLD: ARBITRARY BRANCHING PROCESSES

In this section we explore the growth rate of rewards necessary to achieve a constant probability of success in arbitrary branching processes. From (1), we know that these are fully determined by the decay rate of $\hat{\phi}_j$'s.

We show that the threshold observed by Kleinberg and Raghavan for binomial branching processes is part of a much more general phenomenon that occurs at $b = 2$ in *all* branching processes. Specifically, when the average degree of a node is between 1 and 2—that is, when $1 < b < 2$—we show the rewards to be polynomial in n. When $b > 2$ and thus the $\hat{\phi}_j$'s decay quickly, the reward will grow logarithmically in n.

Following the proof approach used in [12], we begin by bounding the value of $t(x)$ for sufficiently small x close to 1. Recall the mean value theorem: for any function t, there exists a value $y \in [x, 1]$ such that

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

To get a bound on $t(x)$, and therefore on the rate of decay of the $\hat{\phi}_j$'s, we first bound $t(1)$ and $t'(x)$ for x sufficiently close to 1.

Lemma 1. $1 - \frac{b}{n} \leq t(1) \leq 1 - \frac{b}{4n}$.

Proof. We have

$$\begin{aligned} t(1) - 1 &= \sum_{i=0}^{d} c_i (p^i - 1) \\ &= (p-1) \sum_{i=1}^{d} c_i \sum_{j=0}^{i-1} p^j \\ &= -\frac{1}{n} \sum_{i=1}^{d} c_i \sum_{j=0}^{i-1} p^j \\ &\geq -\frac{1}{n} \sum_{i=1}^{d} i c_i \\ &= -\frac{b}{n}. \qquad\qquad \text{(since } p < 1\text{)} \end{aligned}$$

Conversely,

$$\begin{aligned} t(1) - 1 &= -\frac{1}{n} \sum_{i=1}^{d} c_i \sum_{j=0}^{i-1} p^j \\ &\leq -\frac{1}{n} \left(\sum_{i=1}^{d} i c_i \right) p^d \\ &= -\frac{p^d b}{n} \\ &\leq \frac{b}{4n}, \end{aligned}$$

where the last inequality follows if $n > d$ which implies $p^d = (1 - 1/n)^d \geq 1/4$ so long as $d \geq 2$. □

Lemma 2. *Let $1 > \epsilon > b/(4n)$. Then, we have*

$$\frac{t'(x)}{pb} \in [1 - 5\epsilon d, 1],$$

when $x \in [1 - \epsilon, 1]$.

Proof. First, we claim that $f(\delta) = (1 - \delta)^x \geq 1 - \delta x$ for $\delta \in [0, 1]$ and $x > 1$. To see this, note that $f(\delta)$ is a convex function since $f''(\delta) = x(x-1)(1-\delta)^{x-2} \geq 0$ for $x > 1$. Now, the claim follows from the convexity of $f(\delta)$ and the fact that $1 - \delta x$ is just the sum of the first two terms of the Taylor expansion of $f(\delta)$.

Using this, we bound $(p(1-\epsilon))^d$: we have

$$(p(1-\epsilon))^d = \left(1 - \frac{1}{n} - p\epsilon\right)^d \geq 1 - d\left(\frac{1}{n} + p\epsilon\right) \geq 1 - 5d\epsilon,$$

since $\epsilon > 1/(4n)$.

Finally, we have $t'(x) = p\overline{T}'(px)$ and

$$\overline{T}'(px) = \sum_{i=0}^{d} ic_i(px)^{i-1} \leq \sum_{i=0}^{d} ic_i = b.$$

It follows that $t'(x) \leq pb$ for any $x < 1$. Since for any branching process $t'(\cdot)$ is a monotone function, we also have $t'(x) \geq t'(1-\epsilon)$, and so it suffices to bound $t'(1-\epsilon)$. Since $(p(1-\epsilon))^i$ is a decreasing function of i,

$$\begin{aligned} t'(1-\epsilon) &= p\overline{T}'(p(1-\epsilon)) \\ &= p\sum_{i=0}^{d} ic_i\,(p(1-\epsilon)))^{i-1} \\ &\geq p(p(1-\epsilon))^d \sum_{i=0}^{d} ic_i \\ &\geq pb(1-5d\epsilon), \end{aligned}$$

from above. □

We have placed bounds on the decay rate of $t(x)$ only for x sufficiently close to 1. When these conditions are met, we can show a logarithmic bound on the number of iterations of $t(\cdot)$ necessary to reduce the error probability from $1 - \kappa_0/n$ to $1-\gamma$ for κ_0 and γ sufficiently small. This fact follows from Lemma 1 and Lemma 2 for the same reasons as in [12].

LEMMA 3 ([12]). *Suppose p, b, ϵ are such that $pb(1 - 5d\epsilon) > 1$ and let $0 < \gamma_0 < \gamma_1 \leq \epsilon$. Let $N(\gamma_0, \gamma_1)$ denote the number of iterations of the function $t(\cdot)$ needed to reduce the probability of failure from $1-\gamma_0$ to at most $1-\gamma_1$. Then, $N(\gamma_0, \gamma_1) = \Theta(\log(\gamma_1/\gamma_0))$.*

We now show that the threshold behavior persists at $b = 2$ for any branching process.

THEOREM 4. *Let $\sigma < 1 - z_{\overline{T}}$ be a constant. For any branching process, if $b > 2$, then the utility required for the root to find the answer with probability σ is $O(\log n)$ and if $b < 2$, then this utility is $n^{\Omega(1)}$.*

PROOF SKETCH. The proof structure mirrors [12]. For the case when $b < 2$, the crucial step is to obtain an upper bound on the decay rate of the $\hat{\phi}_j$'s for sufficiently small values of j, which we achieve using Lemma 2. Together with Lemma 3, this shows that we need $n^{\Omega(1)}$ reward to achieve even a small but constant probability of success, σ_0.

For the case when $b > 2$, we can again use Lemma 2 together with Lemma 3 to show it is sufficient to offer $O(\log n)$ reward to achieve a success probability of σ_0. It remains to show how to boost this success probability to σ with only constant additional reward. Let $i*$ be such that $\forall i < i*$, we have $c_i = 0$ and $c_{i*} > 0$. Then we have $t(x) = (px)^{i*}t_0(x)$ and $\overline{T}(x) = x^{i*}\overline{T}_0(x)$. From this definition, $t_0(x)$ defines a branching process with non-zero extinction probability and clearly $t_0(x) \geq t(x)$. Since the process has non-zero extinction, we require only $O(1)$ steps to improve the success probability to σ. □

Thus, we have shown that the threshold behavior observed by Kleinberg and Raghavan persists at $b = 2$, even for the case of general branching processes.

4. TIGHT UPPER BOUNDS FOR $b < 2$

In the next section we consider how the rewards required behave when we increase the probability of success even further. To that end we would like to have a tighter characterization on the growth structure of rewards when $b < 2$.

We show that when $1 < b < 2$ the rewards grow no faster than polynomial in n. This is not the case for $b = 1$, and in fact a separate threshold exists at $b = 1$. In the case of the deterministic ray when every node has exactly one child, the rewards grow superexponentially in the length of the path: to push the query h hops into the network the offered reward must be $[\Omega(h)]!$.

Recall that when $b = 1$ and $c_1 < 1$, i.e., the branching process has a non-zero extinction probability, then it becomes extinct almost surely [2], and thus for $b = 1$ the deterministic case is the only case where the asymptotic growth rate of rewards is well defined.

THEOREM 5. *Consider a query incentive network when the branching process is a ray ($c_1 = 1, c_{\neq 1} = 0$). Then $u_j \geq (j-3)!$ and therefore $R_\sigma(n, 1) = [\Omega(n)]!$.*

PROOF. For the case of the deterministic ray, we can write the closed-form solution $\hat{\phi}_j = p^j$. Then

$$\begin{aligned} \frac{\Delta_{j+1}}{\Delta_j} &\geq \frac{1}{\frac{1-\hat{\phi}_j}{1-\hat{\phi}_{j+1}} - 1} \\ &= \frac{1-p^{j-1}}{p^{j-1}-p^j} \\ &= \frac{(1-p)(1+p+\ldots+p^{j-2})}{p^{j-1}(1-p)} \\ &= \frac{1}{p} + \frac{1}{p^2} + \cdots + \frac{1}{p^{j-2}} \\ &\geq j-2. \end{aligned}$$

By induction, $\Delta_{j+1} \geq (j-2)\Delta_j = (j-2)!\Delta_3$. Since $u_j > \Delta_j$, we have $u_j > (j-3)!$. To conclude that $R_\sigma(n,1) = [\Omega(n)]!$, observe that the query needs to propagate $\Omega(n)$ steps to be successful with constant probability. □

However, when $b > 1$, the reward is polynomial in n, as we show below. Let $\kappa_0 > b/(2-b)$ be a small constant. Then we have:

LEMMA 6. *Let $\sigma_0 < \sigma$ be such that $pb(1-6d\sigma_0) > 1$ and $x \in [1-\sigma_0, 1-\kappa_0/n]$. Assume $p > 1/2$. Then,*

$$\frac{1-t(x)}{1-x} \geq \frac{bd\sigma_0}{2} + 1.$$

PROOF. From Lemma 2, we have that $t'(y) \geq pb(1-5d\sigma_0)$ and, by assumption, $pb(1-6d\sigma_0) > 1$. Thus we have that $t'(y) \geq pb(1-5d\sigma_0) = pb(1-6d\sigma_0) + pbd\sigma_0 > 1 + pbd\sigma_0$.

Now, from Lemma 1, we have that $1 - t(1) \geq b/(4n) > 0$.

By the mean value theorem, we have that

$$\begin{aligned}\frac{1-t(x)}{1-x} &= \frac{1-t(1)}{1-x} + t'(y) \\ &\geq \frac{b}{4n(1-x)} + pb(1-5d\sigma_0) \\ &\geq 1 + pbd\sigma_0 \\ &\geq 1 + \frac{bd\sigma_0}{2}.\end{aligned}$$

□

We can now place a bound on the growth of rewards.

LEMMA 7. *Define κ_0 and σ_0 as above, and let j be such that $\hat{\phi}_j \in [1-\sigma_0, 1-\kappa_0/n]$. Then there exists a constant $\beta > 1$ such that*

$$\frac{\Delta_{j+1}}{\Delta_j} \leq \beta.$$

PROOF. By (1),

$$\begin{aligned}\frac{\Delta_{j+1}}{\Delta_j} &\leq \frac{1}{\frac{1-\hat{\phi}_j}{1-\hat{\phi}_{j+1}-1}} + 1 \\ &\leq \frac{1}{\frac{bd\sigma}{2}} + 1 \qquad \text{(from Lemma 6)} \\ &= \beta.\end{aligned}$$

□

We are now ready to state and prove the main theorem:

THEOREM 8 (UPPER BOUND ON THE REWARDS). *Let σ_0 be as above. Then for any branching process with $b > 1$, if the root desires an answer with constant probability $\sigma < \sigma_0$, then the reward necessary is at most $n^{O(1)}$.*

PROOF. We have that, by definition, $r_j = r_{j-1} + \Delta_j = r_{j-2} + \Delta_{j-1} + \Delta_j$. If we let $\sigma_0 > \sigma$, we have that for all j's as in the lemma above,

$$\begin{aligned}r_j &\leq r_{j-2} + \Delta_{j-1}(1+\beta) \\ &\leq r_1 + \Delta_1 [1 + \beta [1 + \beta [\cdots [1 + \beta(1+\beta)] \cdots]]] \\ &= r_1 + \Delta_1 \left[\sum_{i=0}^{j} \beta^i\right] \\ &= r_1 + \Delta_1 \frac{\beta^{j+1}-1}{\beta - 1}.\end{aligned}$$

Lemma 3 tells us that $j = O(\log n)$. Therefore:

$$R_\sigma(n,b) = O(n^c),$$

for some constant c. □

While we have proven the theorem above for any probability of success less than some small constant σ_0, we shall see in the next section that this is true for all $\sigma < 1 - 1/n$.

5. VANISHING THRESHOLD: THE HIGH PROBABILITY CASE

In this section we show that as we decrease the probability of failure to below a constant, the threshold behavior disappears. We show that in order to achieve an arbitrary failure probability of $1 - \sigma = 1 - \sigma(n)$, the reward depends on two quantities. The first is how much we need to achieve a constant probability of failure. Theorem 4 together with Theorem 8 show that this is $O(\log n)$ when $b > 2$ and $n^{O(1)}$ when $b \leq 2$. The second is, given that the probability of failure is already a constant, how much more is needed to decrease the probability to $1 - \sigma$; we show the latter is polynomial in $1/(1-\sigma)$. Note that the second quantity begins to overtake the first quantity (for $b > 2$) as soon as $\sigma > 1 - 1/\log n$. In particular, if $\sigma = 1 - 1/n$, the threshold behavior vanishes since the reward for all values of b becomes polynomial in n. For simplicity, we state and prove the result in terms of this specific value of σ.

First we show the lower bound on the growth of rewards. Note that it suffices to consider the case $b > 2$ since the reward is at least polynomial in n when $b < 2$.

THEOREM 9. *For any branching process, $R_{1-1/n}(n,b) \geq n^{\Omega(1)}$.*

PROOF. We begin by solving the problem for the full d-ary tree and then argue that this is the "best" case for the growth of costs. Thus, our goal is to show that for the full d-ary tree, the cost of obtaining a solution with probability $1 - 1/n$ is $n^{\Omega(1)}$.

As we stated earlier, we assume $b > 2$. Let d be a constant with $d > 2$. Since $b > 2$, the rewards required to achieve a probability of failure of $1/2$ is $O(\log n)$. Hence, from now on we assume that the probability of failure is below $1/2$ and proceed with the rest of the analysis.

Consider the branching process corresponding to the full d-ary tree. Every node deterministically has d children. Effectively, $t(px) = p^d x^d$. Recall from (1):

$$\frac{\Delta_{j+1}}{\Delta_j} \geq \frac{1}{\frac{1-t(px)}{1-x} - 1}, \tag{2}$$

where x is the probability of failure after exploring j levels into the tree. Now,

$$\begin{aligned}&\frac{1-t(px)}{1-x} - 1 \\ &= \frac{1-px}{1-x} - 1 + \frac{1-px}{1-x}\sum_{i=1}^{d-1}(px)^i \\ &\leq \frac{x}{1-x}\cdot\frac{1}{n} + 2\sum_{i=1}^{d-1}(px)^i \qquad \text{(since } (1-px)/(1-x) \leq 2) \\ &\leq \frac{2x}{n} + 2(d-1)px \qquad \text{(since } px \leq 1) \\ &= 2x((d-1)p + 1/n) \\ &\leq 2xdp.\end{aligned}$$

Thus from (2) we have

$$\frac{\Delta_{j+1}}{\Delta_j} \geq \frac{1}{2xdp}, \tag{3}$$

i.e., we have bounded the growth ratio in terms of the probability of failure and showed it to be inversely proportional to x.

Next, recall that the probability of failure is $(1 - 1/n)^N$, where N is the total number of nodes. Since the total number of nodes in a d-ary tree of height j is $\frac{d^{j+1}-1}{d-1}$, to achieve a success probability of $1 - 1/n$, we need to explore $k = \log_d(n \log n) + \Theta(1)$ levels of the tree. Let ℓ denote the depth necessary to achieve a probability of failure of $1/2$; note that $\ell = \Theta(1) + \log_d n$. From this,

$$k - \ell = \log_d \ln n + \Theta(1). \tag{4}$$

Let x_i be the probability of failure after exploring the tree to depth i. Then

$$(x_\ell)^{(d^{i-\ell-1})} \le x_{\ell+i} \le (x_\ell)^{(d^{i-\ell})}. \tag{5}$$

Lower bounding Δ_k is enough to complete the proof: we have

$$\begin{aligned}
&R_{1-\frac{1}{n}}(n,b) \\
&\ge \Delta_k \\
&\ge \frac{\Delta_{k-1}}{2x_{k-1}dp} && \text{(using (3))} \\
&\ge \Delta_\ell \prod_{i=\ell}^{k-1} \frac{1}{2x_i dp} && \text{(inductively)} \\
&= \left(\frac{1}{2dp}\right)^{k-\ell} \exp\left(\sum_{i=\ell}^{k-1} \ln \frac{1}{x_i}\right) \\
&\ge \left(\frac{1}{2dp}\right)^{k-\ell} \exp\left(-\sum_{i=\ell}^{k-1} d^{i-\ell} \ln x_\ell\right) && \text{(using (5))} \\
&\ge \left(\frac{1}{2dp}\right)^{k-\ell} \exp\left(\sum_{i=\ell}^{k-1} d^{i-\ell}\right) && \text{(since } x_\ell = 1/2\text{)} \\
&\ge \left(\frac{1}{2dp}\right)^{k-\ell} \exp\left(d^{k-\ell-1}\right) \\
&\ge \left(\frac{1}{2dp}\right)^{k-\ell} n^{\Omega(1)} && \text{(using (4))} \\
&\ge n^{\Omega(1)},
\end{aligned}$$

where the last inequality follows since $d^{k-\ell} = \ln n$ implies $(2dp)^{k-\ell} \le O(\ln^2 n)$.

It is easy to see that there is nothing special about the full d-ary tree, and in fact the same transition occurs for any branching process that runs for $\log_d n + \Omega(\log_d \log n)$ steps. Consider any such branching process S with maximum degree d. The function s for S corresponding to the function t for the full d-ary tree satisfies $s(px) \ge (px)^d = t(px)$. Thus,

$$\frac{1}{\frac{1-s(px)}{1-x} - 1} \le \frac{1}{\frac{1-t(px)}{1-x} - 1},$$

and so the growth rate of rewards for S is only larger than that for the full d-ary tree. □

We now proceed to prove upper bounds on the growth of rewards. Here we assume that the branching process can never die out, i.e., $c_0 = 0$.

THEOREM 10. *For any branching process with $c_0 = 0$ and $b > 1$, $R_{1-1/n}(n,b) = n^{O(1)}$.*

PROOF. It suffices to upper bound Δ_k, as we can always crudely bound $R_{1-1/n}(n,b) \le k\Delta_k$. We have from (1)

$$\frac{\Delta_{j+1}}{\Delta_j} \le \frac{1}{\frac{1-t(px)}{1-x} - 1} + 1. \tag{6}$$

We focus again on the denominator in (6).

$$\begin{aligned}
\frac{1-t(px)}{1-x} - 1 &= \frac{1-\sum_{i=0}^d c_i(px)^i}{1-x} - 1 \\
&= \frac{x - \sum_{i=0}^d c_i(px)^i}{1-x}.
\end{aligned} \tag{7}$$

By assumption, $c_0 = 0$. Furthermore, since $b > 1$, it must be that $c_1 < 1$. Let σ_0 be defined as before, and let y_0 be such that

$$\frac{1-t(p\sigma_0)}{1-\sigma_0} - 1 = \frac{\sigma_0}{y_0}.$$

Observe that because t is a bounded-degree polynomial with nonnegative coefficients $c_i < 1$, for any $x < 1 - \sigma_0$:

$$\frac{1-t(px))}{1-x} - 1 \ge \frac{x}{y_0}.$$

We let ℓ be such that $\hat{\phi}_\ell \le \sigma_0$; note that

$$k - \ell = O(\log_d \log n). \tag{8}$$

Now for $k > \ell$,

$$\begin{aligned}
&\Delta_k \\
&\le \Delta_\ell \prod_{i=\ell}^{k-1} \frac{y_0}{x_i} + 1 && \text{(using (6))} \\
&\le \Delta_\ell (y_0+1)^{k-\ell} \prod_{i=\ell}^{k-1} \frac{1}{x_i} && \text{(since } x_i \le 1\text{)} \\
&\le \Delta_\ell (y_0+1)^{k-\ell} \prod_{i=0}^{k-\ell} x_\ell^{-d^i} && \text{(using (5))} \\
&= \Delta_\ell (y_0+1)^{k-\ell} \exp\left(\sum_{i=0}^{k-\ell} -d^i \log(x_\ell)\right) \\
&\le \Delta_\ell (y_0+1)^{k-\ell} \exp\left(O(d^{k-\ell+1})\right) \\
&\le \Delta_\ell (y_0+1)^{O(\log_d \log n)} \exp\left(O(\log n)\right) && \text{(using (8))} \\
&\le \Delta_\ell n^{O(1)}.
\end{aligned}$$

Using this, we obtain the bound on the rewards as follows.

$$\begin{aligned}
&R_{1-1/n}(n,b) \\
&\le R_{1/e}(n,b) + \sum_{i=\ell}^{k} \Delta_i && \text{(using Theorem 8)} \\
&\le n^{O(1)} + (k-\ell)n^{O(1)} \\
&= n^{O(1)}.
\end{aligned}$$

□

6. CONCLUSIONS AND FUTURE WORK

We have taken a closer look at the threshold behavior in query incentive networks introduced by Kleinberg and Raghavan [12]. In particular, we have shown that the threshold they observe when the average node degree crosses 2 is present in the case of arbitrary branching processes. This result indicates that the basic phenomena behind the threshold may be generalizable even further, to a much wider range of network models. (Indeed, as a first step along these lines, we can show that the threshold persists when the network is a layered graph that can be nicely embedded into a tree.) Any results along these lines would be extremely interesting, and would undoubtedly reveal new insights into the problem.

On the other hand, we have shown that the threshold behavior disappears as the failure probability is reduced to be inversely polynomial with the rarity of the answer. This result demonstrates another intriguing behavior in query incentive networks. As the model encodes many practical scenarios further study is necessary to fully understand the effect of selfish behavior on information systems.

Acknowledgments

We thank Jon Kleinberg and Prabhakar Raghavan for many useful discussions.

7. REFERENCES

[1] Z. Abrams, R. McGrew, and S. Plotkin. Keeping peers honest in eigentrust. In *Proc. Workshop on the Economics of Peer-to-Peer Systems*, 2004.

[2] K. B. Athreya and P. E. Ney. *Branching Processes.* Dover Publications, Inc., New York, 2004.

[3] J. J. Brown and P. H. Reingen. Social ties and word-of-mouth referral behavior. *J. Consumer Research*, 14:350–362, 1987.

[4] P. Domingos and M. Richardson. Mining the network value of customers. In *Proc. Conference on Knowledge Discovery and Data Mining (KDD)*, pages 57–66, 2001.

[5] D. Dutta, A. Goel, R. Govindan, and H. Zhang. The design of a distributed rating scheme for peer-to-peer systems. In *Proc. Workshop on the Economics of Peer-to-Peer Systems*, 2003.

[6] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proc. International Conference on the World Wide Web (WWW)*, pages 403–412, 2004.

[7] Z. Gyöngyi, H. Garcia-Molina, and J. Pedersen. Combating web spam with trustrank. In *Proc. Conference on Very Large Data Bases (VLDB)*, pages 576–587, 2004.

[8] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The EigenTrust algorithm for reputation management in P2P networks. In *Proc. International Conference on the World Wide Web (WWW)*, pages 640–651, 2003.

[9] H. Kautz, B. Selman, and M. Shah. ReferralWeb: Combining social networks and collaborative filtering. *Commun. ACM*, 30(3):63–65, 1997.

[10] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *Proc. Conference on Knowledge Discovery and Data Mining (KDD)*, pages 137–146, 2003.

[11] D. Kempe, J. Kleinberg, and E. Tardos. Influential nodes in a diffusion model for social networks. In *Proc. Colloquium on Automata, Languages and Programming (ICALP)*, pages 1127–1138, 2005.

[12] J. Kleinberg and P. Raghavan. Query incentive networks. In *Proc. Symposium on Foundations of Computer Science (FOCS)*, pages 132–141, 2005.

[13] J. Leskovec, L. A. Adamic, and B. A. Huberman. The dynamics of viral marketing. In *Proc. Conference on Electronic Commerce (EC)*, pages 228–237, 2006.

[14] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *Proc. 2nd International Semantic Web Conference*, pages 351–368, 2003.

[15] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *Proc. Conference on Knowledge Discovery and Data Mining (KDD)*, pages 61–70, 2002.

[16] B. Yu and M. P. Singh. A social mechanism of reputation management in electronic communities. In *Proc. Workshop on Cooperative Information Agents*, pages 154–165. Springer-Verlag, 2000.

[17] B. Yu and M. P. Singh. Searching social networks. In *Proc. Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 65–72, 2003.

[18] J. Zhang and M. V. Alstyne. SWIM: fostering social network based information search. In *Proc. Conference on Human factors in Computing Systems (CHI)*, pages 1568–1568, 2004.