A Social Query Model for Decentralized Search

Arindam Banerjee
Dept of Computer Science & Engineering
University of Minnesota, Twin Cities
banerjee@cs.umn.edu

Sugato Basu*
Google
Mountain View, CA
sugato@google.com

ABSTRACT

Decentralized search by routing queries over a network is fast emerging as an important research problem, with potential applications in social search as well as peer-to-peer networks [17, 18]. In this paper, we introduce a novel Social Query Model (SQM) for decentralized search, which factors in realistic elements such as expertise levels and response rates of nodes, and has the Pagerank model and certain Markov Decision Processes as special cases. In the context of the model, we establish the existence of a query routing policy that is simultaneously optimal for all nodes, in that no subset of nodes will jointly have any incentive to use a different local routing policy. For computing the optimal policy, we present an efficient distributed approximation algorithm that is almost linear in the number of edges in the network. Extensive experiments on both simulated random graphs and real small-world networks demonstrate the potential of our model and the effectiveness of the proposed routing algorithm.

1. INTRODUCTION

Decentralized search by routing queries over a network is fast emerging as an important research problem cutting across several different areas, with potential applications in social search as well as peer-to-peer networks [17, 18, 1]. Recent years have seen both theoretical [16, 17, 18] and empirical studies [1, 32] on decentralized search, with particular emphasis on small-world networks. In its most basic form, decentralized search considers a network where any node can initiate a "query" for which one or more nodes in the network have a correct "response". The central task in decentralized search is to efficiently route the query to one of the nodes with the correct response, without making use of a central index of the entire network. In peer-to-peer file sharing networking, the query may take the form of a request for a certain audio or video file that may be present in only a set of live nodes. In a social search setting, the query may be a question that gets routed in the social network until some node in the network gives a correct response.

Existing models in decentralized search typically consider deterministic models for the nodes, while allowing randomized routing policies [17]. In several practical scenarios, including social networks as well as peer-to-peer systems, the nodes are far from deterministic. In a social search setting where queries are routed between actors in the network, several nodes may have the expertise to respond correctly, but may be too busy to respond. Further, the expertise levels of different nodes on the topic of query may vary, at times resulting in an incorrect response being generated.[1] Peer-to-peer networks face response rate issues for rather different reasons, e.g., the response rate may vary because of system/network load or when the node is down. Further, the type/amount of files shared by the nodes vary widely, having the same effect as expertise levels in a social network. Existing models for decentralized search were not designed to account for such practical factors.

In this paper, we propose a Social Query Model (SQM) for decentralized search that takes into account practical factors such as response rates and expertise levels of nodes. Unlike existing models, the SQM is a probabilistic model, where the effectiveness of a routing policy is measured in terms of the probability of getting an answer. Further, the Pagerank model as well as certain Markov Decision Processes (MDPs) can be seen as special cases of the SQM. In the context of the model, we establish the existence of a query routing policy that is simultaneously optimal for all nodes, in that no subset of nodes will jointly have any incentive to use a different local routing policy. For computing the optimal policy we present an efficient distributed approximation algorithm that is almost linear in the number of edges in the network (we describe "almost linear" clearly in Section 3). Extensive experiments on both simulated random graphs and real small-world networks demonstrate the potential of our model and the proposed routing algorithm.

The rest of the paper is organized as follows. In Section 2, we present the SQM along with its relationship with the Pagerank model and MDPs. The existence and computation of the optimal routing policy is discussed in Section 3. In Section 4, we present extensive experimental results on both simulated and real networks. We discuss related work in Section 5 and conclude with a discussion of future work in Section 6.

*Work mostly done while at SRI International

The 2nd SNA-KDD Workshop '08 (SNA-KDD'08) August 24, 2008, Las Vegas, Nevada, USA

[1]Such considerations are important for question-answer based social search engines, e.g., iLink (ilink.sri.com/ilink), Yahoo! Answers (answers.yahoo.com), Yedda (yedda.com).

2. THE SOCIAL QUERY MODEL

Let $S = \{x_1, \ldots, x_n\}$ be a set of nodes in a network, and let N_i denote the set of neighbors of x_i. We are interested in studying query routing policies of the nodes in the following setting: Any node x_i, upon receiving a query, can (i) drop the query, (ii) respond to the query, or (iii) forward it to other nodes. The model consists of the following components:

(i) **Expertise**: Each node has an expertise level $e_i \in [0, 1]$. The expertise level e_i indicates the probability with which x_i decides to respond to the query. With probability $(1 - e_i)$, x_i decides to forward the query to a neighbor in N_i. We denote the $n \times 1$ expertise vector across all nodes by $\mathbf{e}$.

(ii) **Correctness**: When a node x_i decides to respond to a query, which happens with probability e_i, let $w_i \in [0, 1]$ denote the probability that the answer is correct. We denote the $n \times 1$ correctness probability vector by $\mathbf{w}$.

(iii) **Response Rate**: A node $x_j \in N_i$, upon receiving a query from x_i, accepts the query with a rate $r_{ij} \in [0, 1]$, and decides to drop the query with probability $(1 - r_{ij})$. We denote the $n \times n$ response rate matrix by R.

(iv) **Policy**: Each node x_i has a policy π^i of forwarding the query to any other node in N_i, i.e., π^i is a probability distribution over N_i. Given that x_i decides to forward the query, which happens with probability $(1 - e_i)$, it will be forwarded to $x_j \in N_i, j \neq i$ with probability π^i_j. Node x_j can then decide to accept the forward with probability r_{ij} and ignore the question with probability $(1 - r_{ij})$. The overall policy for all nodes in the network is a $n \times n$ matrix denoted by Π, whose i^{th} row is π^i.

In such a query routing setup, the central object of interest for a node x_i is the probability P_i of getting a correct answer to a query initiated by x_i. Given a policy Π, a node can get a correct answer (i) if it decides to answer the query and knows the correct answer, which happens with probability $w_i e_i$, or (ii) if it decides to forward the query, with probability $(1-e_i)$, and sends the query to another expert, in particular to x_j with probability π^i_j. Then, the expert x_j can ignore the query, with probability $(1-r_{ij})$, or accept the query and try to get an answer, with probability r_{ij}. If x_j chooses to get a correct answer to the query, it gets the answer with probability P_j. Hence, $\forall i$, we have the following recursion:

$$\begin{aligned} P_i &= w_i e_i + (1 - e_i) \sum_{j \in N_i}^{n} \pi^i_j ((1 - r_{ij}) \times 0 + r_{ij} P_j) \\ &= w_i e_i + (1 - e_i) \sum_{j \in N_i}^{n} \pi^i_j r_{ij} P_j \ . \end{aligned} \tag{2.1}$$

If P is the vector of probabilities of getting a correct answer, then we first show that, under fairly general conditions on $(\mathbf{w}, \mathbf{e}, R)$, any valid policy matrix Π uniquely determines the probability vector P. Since self-forward is not allowed in a valid policy, we have $\pi^i_i = 0$, Further, let $C_i = w_i e_i$ and $a_{ij} = \pi^i_j r_{ij}$, so that $C = \mathbf{w} \circ \mathbf{e}$ and $A = \Pi \circ R$, where $\circ$ denotes a Hadamard product. Further, let $D = \text{diag}(1 - \mathbf{e})$. Then, in vector/matrix notation, we have

$$P = C + DAP \ \Rightarrow \ P = (\mathbb{I} - DA)^{-1} C \ .$$

The last expression gives a unique P if $(\mathbb{I} - DA)$ is non-singular, which is true if $\|DA\|_p < 1$ according to some matrix norm [11]. The following result establishes a sufficiency condition for this, and hence for P to be uniquely defined.

Proposition 1 *For a given $(\mathbf{e}, R, \Pi)$, the probability vector P is uniquely determined if $\forall i$, $e_i > 0$ or $\exists j', \pi^i_{j'} > 0$ and $r_{ij'} < 1$.*

The proof of Lemma 1, as well as all other technical proofs, is given in the Appendix. We denote the probability vector corresponding to a policy Π as P_Π and let $\lambda = \max_{i,j}(1 - e_i) r_{ij}$ so that $\lambda < 1$. Next, we show that the basic Pagerank/random walk model [8] can be seen as a special case of the social query model.

Proposition 2 *In the social query model, let $e_i = 0$ and $\forall i, \forall j, \in N_i, r_{ij} = 1$, where N_i denotes the neighbors of x_i. Then, if π^i is a uniform distribution over N_i, the SQM reduces to the basic Pagerank/random walk model.*

Finally, note that the proposed social query model has important similarities with certain classes of Markov Decision Processes (MDPs) [6, 7, 24]. In the context of MDPs, consider a query in SQM as an agent and the nodes in SQM as states. With the assumption that $\forall i, e_i = e, \forall i, \forall j, \in N_i, r_{ij} = 1$, SQM reduces to a deterministic MDP with reward $C_i = w_i e$, discount factor $\gamma = (1 - e)$, value P_i and policy π^i in state x_i, i.e.,

$$P_i = C_i + \gamma \sum_{j \in N_i} \pi^i_j P_j \ .$$

In general, SQM allows the possibility of different expertise levels e_i, and the possibility of rejection by the next state x_j with probability $(1 - r_{ij})$, which makes SQM somewhat different from a standard MDP. For example, the discount factor γ in a (deterministic) MDP is a constant across all states and has no dependency on the policy π^i, whereas in case of SQMs, the "discount factor" depends on the state x_i forwarding the query, the state x_j to which the query is being forwarded, and the policy π^i used for forwarding, which effectively determines how the r_{ij} terms gets weighed. However, due to the significant similarities between the two models, we derive equally useful results for SQMs by appropriately extending existing arguments for MDPs [7, 24].

3. ANALYSIS AND ALGORITHMS

We establish the existence of a policy that is simultaneously optimal for all nodes, and give a distributed approximation algorithm for computing the optimal policy.

3.1 Existence of Optimal Policy

One can view the SQM as a multi-party game, where the players are the nodes of the network and the payoff is P_Π, the probability values of nodes getting answers by following policy Π. Given any policy Π, there is another natural question one can ask: is Π the best policy, or is it possible to get a policy Π' which is better than Π? The notion of "best" needs a careful definition, since we are working with vector-valued payoffs (P_Π is a vector of probabilities). We introduce the following definition to quantify the goodness of a policy:

Definition 1 A policy Π is called a *Nash policy*, if no single node x_i has any incentive to change its policy π^i, given that all other nodes (other than i) are following their prescribed policy (indicated by Π^{-i}).

The above definition exactly follows the notion of Nash equilibrium in a multi-party game. However, since query routing a social network setting is not an adversarial game, it will be common for actors to collude and jointly change their policies to get higher payoffs, potentially at the cost of lowering the payoff for other actors. Such considerations motivate a stronger notion of goodness of a policy:

Definition 2 A policy Π is called a *social policy*, if no group of actors $\mathcal{X}_h \in 2^{\mathcal{X}}$ has any incentive to change their policies $\Pi^{\mathcal{X}_h}$, given that all other actors are following their prescribed policies $\Pi^{-\mathcal{X}_h}$.

A social policy follows the idea of social utility in welfare economics [10, 12]. It is a significantly stronger notion of goodness, since every possible subset of users is required to be "happy" using this policy, and there is no incentive for deviation for a single actor or any group of actors. Clearly, a social policy is a Nash policy, by definition. Based on the above definitions, two questions are natural to ask:

(i) Does a Nash policy always exist for any given SQM?

(ii) Does a social policy always exist for any given SQM?

The questions appear tricky to answer since the SQM cannot be converted to a multi-party normal form game, for which extensive literature exists [22]. Surprisingly, the answer to both questions is a "Yes." In particular, we can prove the existence of a social policy for arbitrary SQMs, and the existence of a Nash policy follow as a special case.

Let M_D be the set of all deterministic policies, which involve forwarding the query to one agent deterministically. Let $\mathcal{P}_D$ be the space of payoff vectors generated by $\Pi \in M_D$. Since, one can define only a partial order among elements in $\mathcal{P}_D$, we uniquely define $P^*_D = \sup_{\Pi \in M_D} P_\Pi$ as a pointwise supremum, noting that P^*_D may not be achievable by any $\Pi \in M_D$. Further, for any P, let

$$\mathcal{L}P = \sup_{\Pi \in M_D} \{C + DA_\Pi P\} ,$$

where $A_\Pi = \Pi \circ R$. Let M_R be the set of randomized policies, which involve forwarding the query to one of the neighboring agents following a distribution. Let $P^*_R = \sup_{\Pi \in M_R} P_\Pi$. Then, the following Lemma shows that the best deterministic policy is as good as the best randomized policy.

Lemma 1 *For all P, we have*

$$\sup_{\Pi \in M_D} \{C + DA_\Pi P\} = \sup_{\Pi \in M_R} \{C + DA_\Pi P\} .$$

Hence, $P^*_D = P^*_R = \sup_{\Pi \in M_R} P_\Pi$.

For convenience, we denote the optimal payoff corresponding to the social policy as P^* so that $P^* = P^*_D = P^*_R$. Next, we focus on a characterization of the optimal payoff P^* corresponding to the social policy. In particular, we present the following key result that shows that *if* there is a achievable payoff P that cannot be increased by any deterministic policy, then P is the optimal payoff P^*.

Theorem 1 *If $\exists P \in \mathcal{P}_D$ such that $P = \mathcal{L}P$, then $P = P^*$.*

Theorem 1 effectively shows that the optimal payoff, if it exists, will be the fixed point of the operator $\mathcal{L}$. In the following theorem, we show that $\mathcal{L}$ indeed has a fixed point, and the corresponding optimal payoff vector P^* determines the deterministic social policy.

Theorem 2 *There exists $P_0 \in \mathcal{P}_D$ such that $\mathcal{L}P_0 = P_0$. Further, $P_0 = P^*$ is the optimal payoff, and the corresponding Π_0 is the social policy.*

Thus, we have established the existence of a deterministic social policy that corresponds to the optimal payoff vector P^*, which can be alternatively characterized as the fixed point of the operator $\mathcal{L}$.

3.2 Computing the Optimal Policy

We focus on efficient algorithms for computing the optimal policy. One can always construct a naive algorithm that runs as follows: start with an arbitrary deterministic policy Π_0 with corresponding probability vector P_0, and go iteratively over all individual policies to check if making any changes leads to an improvement in P. Such an approach is akin to the greedy strategy used by the KMeans family of algorithm [5], as well as policy iteration algorithms in the MDP literature [19]. Unfortunately, such an approach requires that P_Π be evaluated at every step, which in itself is an expensive operation, and it is not clear how many iterations are needed for such an algorithm to converge. Instead, we give a distributed algorithm that efficiently computes a good approximation to the optimal policy.

We start by presenting a simple observation on the structure of deterministic routing paths emanating from any node, assuming infinitely many hops. Let σ_i denote the path $x_i, u_1, \ldots, u_k, v_1, \ldots, v_l, v_1$, where each node is unique until the first repetition of a node. Clearly, $k \leq (n-1)$ as there are n unique nodes in the network. Let $x_i \bar{u} \bar{v}$ denote the path σ_i without the repeating node v_1, and let $\sigma_i^\infty = x_i \bar{u} \bar{v} \bar{v} \cdots$ denote the infinite path that goes through $x_i \bar{u}$ and infinitely loops through $\bar{v}$. The following result shows that the paths corresponding to all deterministic policies are of the form of σ_i^∞.

Proposition 3 *For any deterministic policy Π, the infinite path of a query initiated by x_i will be of the form of σ_i^∞.*

The structure of the deterministic paths is quite straightforward, and it is apparent that a provably polynomial time algorithm can be designed to get the optimal routing policy.[2] However, the following two considerations about the practical applicability of the model motivate us to design a fast distributed approximation algorithm:

1. Any optimal algorithm will involve all-pairs-shortest path style computations [23, 13], leading to cubic or higher-order polynomial complexity. Real-world graphs, e.g., social networks, are often very large so that running such algorithms can be computationally prohibitive.

[2]The interested reader is referred to [19, 23, 13, 20] for ideas on designing the optimal algorithm.

2. In practice, because of computational, storage, as well as privacy constraints, it is highly desirable that the algorithm be run in a purely distributed fashion, where each node in the graph can perform local computations based on its own neighborhood.

Let Π^* be an optimal deterministic policy with probability vector P^*. Let Π^*_T be the optimal deterministic policy when the query is allowed to be forwarded for at most T hops, and let P^*_T be the corresponding probability vector. Note that $\lim_{T\to\infty}\ \Pi^*_T = \Pi^*$ and $\lim_{T\to\infty}\ P^*_T = P^*$. Now, consider using the policy Π^*_T in the general case, where the query is allowed to be forwarded an unbounded number of times. Clearly, the corresponding probability vector $P_{\Pi^*_T} \leq P^*$, since P^* is optimal in the general case. However, as the following result shows, $P_{\Pi^*_T}$ approaches P^* exponentially fast with T.

Theorem 3 *If $P_{\Pi^*_T}$ is the probability vector corresponding to Π^*_T in the general setting of potentially infinitely many forwards, then, with $c = \log(1/\lambda) > 0$,*

$$\|P_{\Pi^*_T} - P^*\| \leq \exp(-cT)\ ,$$

where $\|\cdot\|$ denotes the sup-norm,

In particular, for $T = n$, we note that $P_{\Pi^*_n}$ will be within $\epsilon' \leq \exp(-cn)$ of the optimal probability vector P^*. The result gives a strong justification for developing a fast algorithm for getting an approximation to Π^*_n, which will be provably close to the optimal solution. We propose the following value iteration style algorithm for obtaining such a policy, where the superscript (t) for any variable represents it's value at iteration t:

1. Let $v_i^{(0)} \leftarrow e_i, [i]_1^n$
2. For $t = 1, \ldots, n$
   (a) $j_i^* \leftarrow \text{argmax}_{j\in N_i}\, r_{ij} v_j^{(t-1)}, [i]_1^n$
   (b) $v_i^{(t)} \leftarrow w_i e_i + (1 - e_i) r_{ij_i^*} v_{j_i^*}^{(t-1)}$
3. $\forall i$, set $\pi^i_{j_i^*} = 1$, and $\pi^i_j = 0$ for $j \neq j_i^*$

The following result shows that the algorithm returns ϵ-optimal policy in almost linear time.

Theorem 4 *Let m_z be the maximum number of neighbors that any node has in the SQM. Then, the distributed algorithm returns ϵ-optimal policy in $O(nm_z)$ time, where $\epsilon \leq \frac{4\lambda}{1-\lambda}\exp(-cn)$ with $c = \log(1/\lambda) > 0$.*

Finally, we explain why the algorithm runs in "almost linear" time. Let E denote the number of edges in the network. If the degree distribution of the network is not very skewed, then E is $\Theta(nm_z)$. In such cases, the runtime complexity of the algorithm will be $O(E)$, i.e., linear in the number of edges in the network. If the degree distribution is skewed, $nm_z > E$, then the algorithm is super-linear, with a worst-case factor of $O(n^2/E)$ over a linear algorithm.

4. EXPERIMENTS

We performed detailed experiments on both simulated and real world networks to demonstrate that optimal routing of messages using the SQM model is better (as quantified by different metrics) than other routing policies, e.g., random policy, degree-based policy [1].

4.1 Datasets

We ran our experiments on four types of networks, two simulated and two real-world.

1. Simulated networks: We generated networks using two popular random graph models.

(i) ER model: This is a random graph model popularized by Erdos and Renyi [9], where one starts with n nodes and adds edges between each pair of nodes independently following a fixed probability p.

(ii) BA model: Barabasi and Albert [2] proposed a random graph model where a graph is grown according to the preferential attachment principle: when a new node joins the graph, it links to an existing node with a probability proportional to the existing in-degree of a node. The BA model is considered to be a more realistic model (compared to ER) of real-world networks, since it has some properties of small-world graphs, e.g. scale-free structure [16].

2. Real networks: We ran experiments on two real-world networks from the Internet domain that exhibit power-law degree distribution and small-world connectivity.[3]

(i) California query network (CA): The CA network comprises the top 10,000 webpages that are returned as a match to the search engine query "California", using a 2001 snapshot of the Internet. The webpages correspond to the nodes, and the hyperlinks in between the webpages are the links. After removing some dead links, the network comprises 9664 nodes with an edge density of 0.034% (i.e., the average number of neighbors per node is 3.28).

(ii) Autonomous System (AS) network: The AS network consists of AS-level connectivities from Oregon route-views of the Internet in 2002. Nodes correspond to autonomous peers, and links in the network signify direct peering relations between the nodes. There are 26,590 nodes, with an edge density of 0.0075% (i.e., the average number of neighbors per node is 1.99).

4.2 Methodology

As outlined in Section 2, the SQM model has three parameters that can be modified—expertise, response rate, and policy. In our experiments, we used different variants of these parameters.

1. Expertise: We consider two variants for expertise levels.

(i) Random: Expertise e_i, for the i^{th} node x_i, is sampled uniformly at random from [0,1].

(ii) Degree-based: Expertise e_i is proportional (up to a small random perturbation) to the in-degree of node x_i. This models the phenomenon that the expertise of a node in a real social network is often judged by how many other nodes link to it.

2. Response: We consider two variants for response rates.

(i) Random: Response rate r_{ij} of node x_j to node x_i is distributed uniformly in [0,1]. The response rate matrix R has the same sparsity structure as the graph G, implying that node x_j responds only to nodes that are linked to it in the underlying connection graph. The assumption can be useful in modeling scenarios where the response rates of

[3]http://www.cs.cornell.edu/courses/cs685/2002fa/

nodes in the network are not related to the expertise distribution of the nodes, e.g., some members in an organization have a higher propensity to prompt replying that may be unrelated to their expertise or that of the query originator.

(ii) Degree-based: Response rate r_{ij} is considered to be sigmoid(e_i/e_j), to model the phenomenon that the probability of node x_j responding to node x_i is directly proportional to the expertise e_i of node x_i (people tend not to ignore requests from experts), and inversely proportional to the expertise e_j of node x_j (experts tend to be often too busy to respond). A small random perturbation is added to the sigmoid function, and it is scaled to ensure that $r_{ij} \in [0,1]$.

3. Policy: We consider three types of policies.

(i) Random: Policy π_j^i, of node x_i forwarding to node x_j, is distributed uniformly in [0,1], while ensuring that the self-forwarding probability π_i^i for each node x_i is 0, and the probability of x_i forwarding a message to its neighbors sums to 1. The policy matrix R has the same sparsity structure as the graph G.

(ii) Degree-based: Policy π_j^i is proportional to the expertise e_j of node x_j (up to a small random perturbation), reflecting the phenomenon that node x_i would tend to forward messages to its neighbors in a social network with probability proportional to their relative expertise.

(iii) Optimal: Policy π^i is the optimal policy for the SQM model, calculated using the value iteration algorithm (described in Section 3).

4.3 Results

The main set of experiments, referred to as `Expt1`, compared the random, degree-based, and optimal policies for message routing on different combinations of expertise and referral rate parameters, on all four types of networks.

For the simulated networks, we created graphs with 10,000 nodes using the ER and the BA graph growth models, where each node was connected on an average to 10 neighbors. For every network, we simulated 10,000 queries being routed through the network for different settings of the expertise, response and policy parameters. The probability w_i of the i^{th} node responding correctly to a message was set to 0.9.

Figures 4.1-4.9 show some of the results of `Expt1`. In each plot, the x-axis shows buckets of probability values P of getting correct answers to a query, and the y-axis is the number of nodes in the network with probability P lying in that bucket. These plots show results on some combinations of expertise and response parameters from the choices mentioned above – the results on all the choice combinations could not be shown because of lack of space, but the omitted figures showed similar trends. The histogram in Figure 4.1 demonstrates that on the ER graph with random expertise and response rates, using the optimal policy gives much better probability values P, of nodes getting correct answers to their queries, than the other policies (random and degree-based). Similar improvement of the optimal policy over other policies is shown in small-world graphs, as demonstrated by Figures 4.2 and 4.3 for BA, Figures 4.4 and 4.5 for AS, and Figures 4.6 and 4.7 for CA.

In our experiments on simulated BA graphs with degree-based expertise and random response, we observed an interesting phenomenon—a large proportion of the nodes in the

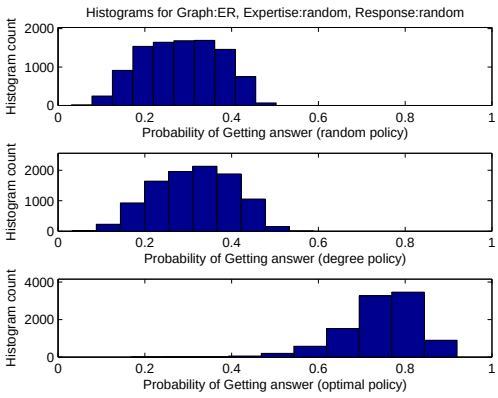

Figure 4.1: Histograms for ER in `Expt1` with random expertise and random response. The x-axis is the buckets of probability values of getting correct answers to a query, while the y-axis is the number of nodes in the network with probabilities in each bucket.

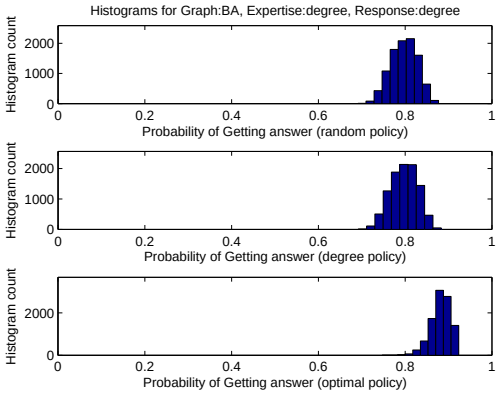

Figure 4.2: Histograms for BA in `Expt1` with degree-based expertise and degree-based response.

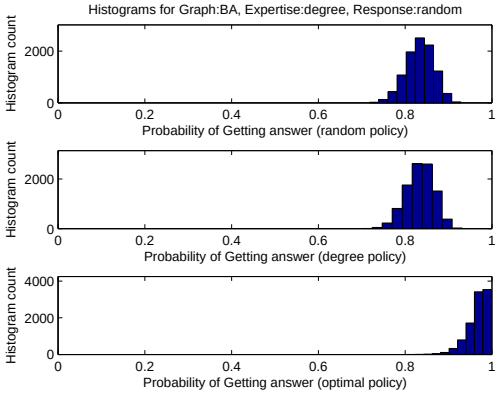

Figure 4.3: Histograms for BA in `Expt1` with degree-based expertise and random response

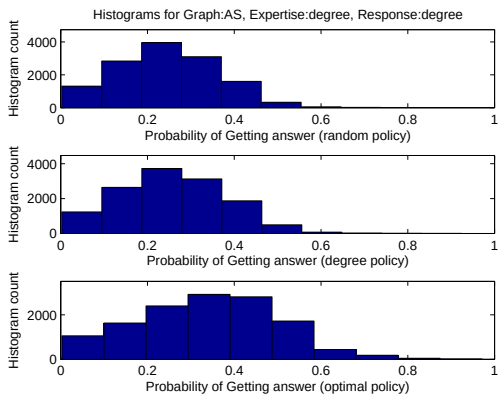

Figure 4.4: Histograms for AS in `Expt1` with degree-based expertise and degree-based response

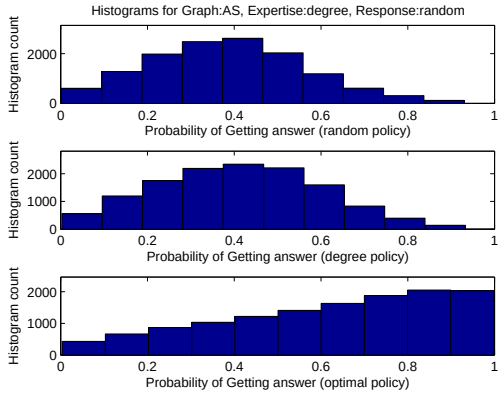

Figure 4.5: Histograms for AS in `Expt1` with degree-based expertise and random response

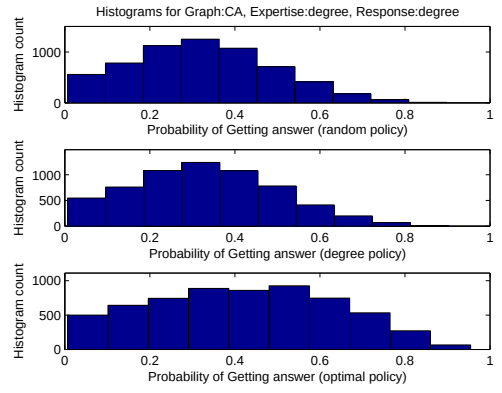

Figure 4.6: Histograms for CA in `Expt1` with degree-based expertise and degree-based response

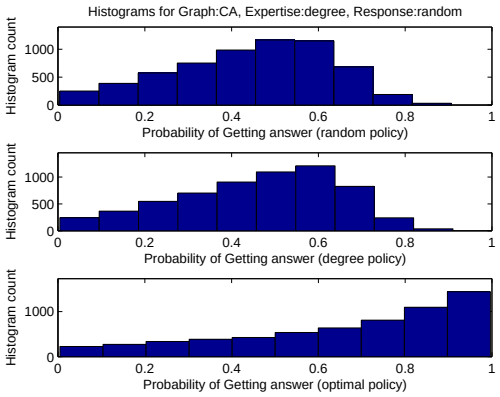

Figure 4.7: Histograms for CA in `Expt1` with degree-based expertise and random response

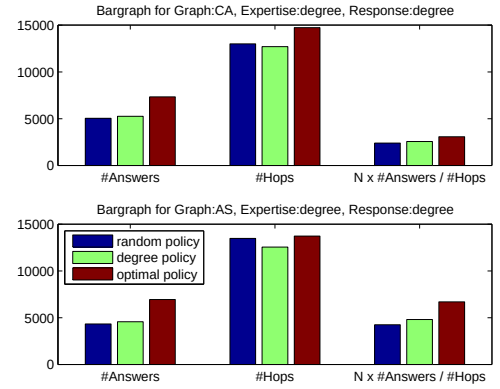

Figure 4.8: Bar-graphs for real networks (AS, CA) in `Expt1` with degree-based expertise and degree-based response

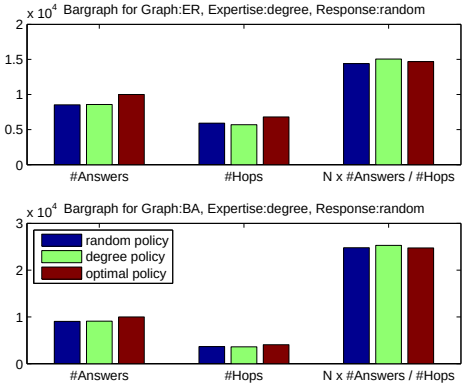

Figure 4.9: Bar-graphs for simulated networks (ER, BA) in `Expt1` with degree-based expertise and random response

network had a probability value of getting correct answers to their questions very close to 1 (Figure 4.3), much better than for the degree-based response (Figure 4.2). On the real network datasets, AS and CA, the improvement of optimal policy over the other policies was again more marked for random response rather than degree-based response, for degree-based expertise distribution—the mode of the probability histogram for the network was at 1 in both graphs (compare Figure 4.5 to Figure 4.4, Figure 4.7 to Figure 4.6).

We present one possible explanation of this phenomenon. BA, AS and CA are small-world graphs, and with degree-based expertise distribution most nodes are non-experts, i.e., have low expertise (in the long tail of the degree distribution). When the response rate is degree-based, the non-experts will get (i) low response rate from experts, since experts have a low probability of responding to non-experts in the degree-based response rate model; (ii) higher response rate from non-experts, but they will have a low probability of giving a correct response to the query, since their expertise is low. In both the cases, the probability of obtaining a correct response will be lower for non-experts, but this accounts for most of the nodes in the small-world networks. Therefore, the probability histogram shifts toward lower values for the degree-based response policy. Note that for networks that do not have a small-world structure and where the expertise is not degree-based, this effect of random response is not that pronounced (Figure 4.1).

As seen in Figures 4.8 and 4.9, the number of correct answers obtained by the optimal policy is also higher than that of the degree-based or random policies. This comes at a cost of higher number of hops in the network, which is not surprising since the policy is optimized to increase the number of questions answered correctly, but it does not explicitly decrease the number of hops. However, as seen from the bar plots, the average number of correct answers per hop (scaled by the network size, for the purpose of visualization) is in most cases best for the optimal policy. Note that the degree-based policy in most cases did not perform much better than random policy when evaluated using the probability histograms, an observation that is consistent with related previously reported results [1]. However, the degree-based policy gave better performance than random policy when compared along number of correct answers per hop, as shown in Figures 4.8 and 4.9. Due to lack of space, we could not show further analysis, e.g., showing the distribution of correct and incorrect answers as the expertise value is changed, number of queries dropped/ignored, etc.

4.4 Multicast Query Routing in SQM

While the analysis in Section 3 gives a near optimal policy for unicast forwarding, i.e., when every node can forward to only one other node, in practice, one may be interested in multicast forwarding, where any node is allowed to forward to $k \geq 1$ nodes. For a given k, one can derive a simple greedy strategy for multicast routing based on the optimal policy for unicast routing as follows:

1. Let G_1 be the given network. For $h = 1, \ldots, k$
   (a) Let $\Pi_h \leftarrow$ the optimal unicast policy.
   (b) Let $G_{h+1} \leftarrow G_h \setminus$ all edges that participate in Π_h.
2. Multicast policy is $\{\Pi_1, \ldots, \Pi_k\}$.

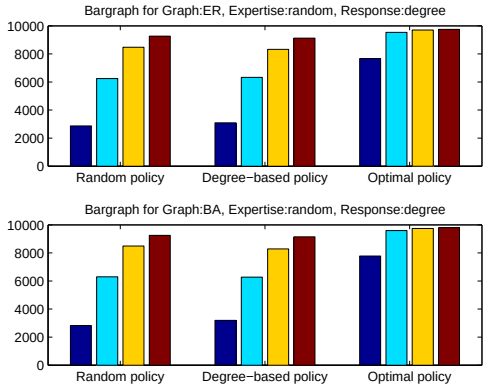

Figure 4.10: Bar-graphs for simulated networks (ER, BA) in `Expt2` with random expertise and degree-based response. X-axis is the type of routing policy, Y-axis is the number of questions correctly answered by the network.

Thus, the strategy is to simply find the next-best policy in each round. Such a multicast policy need not have any provable optimal policies. We performed another experiment, `Expt2`, where we compared the performance of this greedy multicast algorithm for different parameters of expertise, referral, and policy on the networks considered in `Expt1`. For the multicast scenario, we tested the greedy strategy for routing messages to as many as four neighbors.

Figure 4.10 show the result on simulated graphs for `Expt2`. The bar plots show the progressive increase in the number of questions correctly answered by the network, as the number of multicasts from each node is increased from one to four. A notable phenomenon is the diminishing returns from increasing the number of multicasts from each node, whereby the number of additional correct responses received with an increasing number of multicasts starts to progressively decrease (especially for the optimal policy). The multicast algorithm is a simple greedy strategy that gives good results in this experiment but is not guaranteed to give optimal results in all cases. Designing a multicast routing strategy for the SQM, which would have near optimal properties like the unicast case, is an interesting area of future research.

5. APPLICATIONS AND RELATED WORK

A very promising area of application and further development of SQM is social networks analysis, where the problem of searching in a social network has generated significant interest [29, 1], with particular attention to navigation in complex networks [17], as well as game theoretic incentive mechanisms for efficient query routing [18]. Another important emerging application of SQM-style models is in the domain of distributed resource discovery in desktop grids [14, 21]. In a distributed computing setting, a job inserted into the system by a node has to be routed to a node meeting the minimum requirements for processing the job, while doing load balancing at the same time [14, 15]. Another potential application area of SQM-style probabilistic models is in peer-to-peer search [28, 4]. While P2P indexing and search has been widely studied using distributed hash tables (DHT), resulting in several successful models, such as

CAN [25], Pastry [26], Chord [27], etc., most existing models do not have a natural way to handle uncertainties, and are not secure against attacks by malicious nodes. Since SQM has no restriction on the underlying graph on which it performs routing, it can be applied to the overlay network for a P2P service, where the greedy key-based routing will be replaced by the routing policy recommended by the SQM. Further, the social network structure provided by SQM can be leveraged to develop more secure services [31].

6. CONCLUSIONS AND FUTURE WORK

We have discussed a novel Social Query Model (SQM) for decentralized search by query routing, modeling realistic elements such as expertise levels and response rates of nodes, and has the Pagerank model and Markov decision processes as special cases. We introduce the notion of a social policy that follows the idea of social utility and generalizes the Nash policy, in which no subset of nodes have an incentive to use a different local routing policy. We demonstrate that an SQM always has a unique social policy, and we propose an efficient distributed algorithm for approximately computing this optimal query routing policy. Detailed experiments, on both simulated random graphs and real small-world networks, demonstrate the effectiveness of our model and the proposed routing algorithm, using different performance metrics.

Several extensions to SQM are of significant interest. The basic model assumes that the expertise and the response rate of any node can be effectively represented by single numbers. In practice, a node can have different expertise levels and response rates depending on the "topic" of the query. In the social network setting, extension of the model to add topics (e.g., topic-based response rates, expertise) or tags (e.g., tagged forwarding of messages) will be of practical interest. A known graphical correspondence structure between network nodes can be used to extend the model in different ways, e.g., using graphical game-theoretic cost-benefit analysis to decide if/when a node should attempt to establish a connection to a new node, performing exploration/exploitation tradeoff calculations (like in reinforcement learning) to decide whether to route a message to an existing neighbor or a new neighbor. In general, the routing policy has to be query dependent, which provides a strong motivation to merge the SQM formalism, which can naturally handle uncertainty, with DHT based systems, which are good at content driven routing. Another assumption of our current treatment is that the parameters, e.g., expertise level, response rates, etc., stay constant over time. In general, the parameters will change over time, e.g., expertise will gradually grow over time, response rate will vary depending on current load, etc. As a result, it will be necessary to develop an online version of SQM, where one tracks the optimal policy based on system dynamics. Further, since multicast routing is an option in certain application settings, it will be important to design such an algorithm for general SQMs with near-optimal properties like the unicast algorithm presented in the paper. Finally, since none of the existing routing approaches, including SQM, has an explicit safeguard against malicious nodes, it will be important to try to develop secure routing methods, possibly taking advantage of network reputation systems [31].

Acknowledgements: The authors would like to thank the iLink team at SRI for discussions and valuable feedback. The research was supported in part by a Grant-in-Aid (GIA) of Research, Artistry and Scholarship and the DTC Data Mining Consortium (DDMC) at the University of Minnesota, Twin Cities, and by DARPA under Contract No. NBCHD030010 (the Calo grant).

7. REFERENCES

[1] L. A. Adamic and E. Adar. How to search a social network. *Social Networks*, 27(3):187–203, July 2005.

[2] R. Albert and A. L. Barabási. Topology of complex networks: Local events and universality. *Physical Review Letters*, 85:5234–5237, 2000.

[3] G. A. Anderson, A. Granas, and J. Dugundji. *Fixed Point Theory*. Springer, 2006.

[4] J. Aspnes and G. Shah. *Theoretical and Algorithmic Aspects of Sensor, Ad Hoc Wireless and Peer-to-Peer Networks*, chapter Distributed data structures for P2P systems. CRC Press, 2005.

[5] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh. Clustering with Bregman divergences. *Journal of Machine Learning Research*, 6:1705–1749, 2005.

[6] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[7] D. Bertsekas. *Dynamic Programming: Deterministic and Stochastic Models*. Prentice-Hall, 1987.

[8] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[9] P. Erdos and A. Renyi. On random graphs I. *Publ. Math. Debrecen 6*, pages 290–297, 1959.

[10] A. N. Feldman and R. Serrano. *Welfare Economics and Social Choice Theory*. Springer, 2006.

[11] G. H. Golub and C. F. V. Loan. *Matrix Computations*. Johns Hopkins University Press, 1996.

[12] P. E. Johnson. *Social Choice: Theory and Research*. Sage, 1998.

[13] R. M. Karp. A characterization of the minimum cycle mean in a digraph. *Discrete Mathematics*, 23, 1978.

[14] J. Kim, P. Keleher, M. Marsh, B. Bhattacharjee, and A. Sussman. Using content-addressable networks for load balancing in desktop grids. In *IEEE Intl Symp High Performance Distributed Computing*, 2007.

[15] J. Kim, B. Nam, P. Keleher, M. Marsh, B. Bhattacharjee, and A. Sussman. Resource discovery techniques in distributed desktop grid environments. In *IEEE/ACM Intl Conf on Grid Computing*, 2006.

[16] J. Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proc. 32nd ACM Symposium on Theory of Computing*, 2000.

[17] J. Kleinberg. Complex networks and decentralized search. In *Proc. of the Intl. Congress of Mathematicians (ICM)*, 2006.

[18] J. Kleinberg and P. Raghavan. Query incentive networks. In *Proc. 46th Annual IEEE Symposium on Foundations of Computer Science*, 2005.

[19] M. L. Littman, T. L. Dean, and L. P. Kaelbling. On the complexity of solving Markov decision problems. In *Proc. 11th Annual Conf. on Uncertainty in Artificial Intelligence (UAI–95)*, pages 394–402, 1995.

[20] O. Madani. Polynomial value iteration algorithms for deterministic MDPs. In *Proceedings of 18th Conference on Uncertainty in Artificial Intelligence (UAI-02)*, 2002.

[21] D. Oppenheimer, J. Albrecht, D. Patterson, and A. Vahdat. Distributed resource discovery on planetlab with SWORD. In *First Workshop on Real, Large Distributed Systems*, 2004.

[22] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, 1994.

[23] C. H. Papadimitriou and J. N. Tsitsiklis. The complexity of

Markov decision processes. *Mathematics of Operations Research*, 12(3), 1987.
[24] M. L. Puterman. *Markov Decision Processes*. John Wiley & Sons, 1994.
[25] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *ACM SIGCOMM*, 2001.
[26] A. Rowstran and P. Druschel. Pastry: Scalable, distributed object location and routing for large peer-to-peer systems. In *IFIP/ACM Intl Conf on Distributed System Platforms*, 2001.
[27] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan. A scalable peer-to-peer lookup service for internet applications. In *ACM SIGCOMM*, 2001.
[28] D. Tsoumakos and N. Roussopoulos. Adaptive probabilistic search for peer-to-peer networks. In *3rd IEEE Intl. Conf. on P2P Computing*, 2003.
[29] D. J. Watts, P. S. Dodds, and M. E. J. Newman. Identity and search in social networks. *Science*, 296:1302–1305, 2002.
[30] R. J. Williams and L. C. I. Baird. Tight performance bounds on greedy policies based on imperfect value functions. Technical Report NU-CCS-93-13, Northeastern University, College of Computer Science, 1993.
[31] H. Yu, M. Kaminsky, P. Gibbons, and A. Flaxman. Sybilgaurd: Defendeing against sybil attacks via social networks. In *ACM SIGCOMM*, 2006.
[32] H. Zhang, A. Goel, and R. Govindan. Using the small-world model to improve freenet performance. In *IEEE Infocom*, 2002.

APPENDIX

Proof of Proposition 1

Let s_i denote the sum of the elements in the i^{th} row of DA. Since π^i is a probability distribution over the neighbors of x_i, we have

$$s_i = (1-e_i)E_{\pi^i}[r_{i\cdot}] = (1-e_i)\sum_{j=1}^{n}\pi_j^i r_{ij} \ .$$

Since each $r_{ij} \leq 1$, $E_{\pi^i}[r_{i\cdot}] \leq 1$. Now, if $e_i > 0$, $(1-e_i) < 1 \Rightarrow s_i < 1$. On the other hand, if $\exists j'$, with $\pi_{j'}^i > 0, r_{ij'} < 1$, then $E_{\pi^i}[r_{i\cdot}] < 1 \Rightarrow s_i < 1$. Thus, $s_i < 1$ if the conditions are satisfied. Then, we have

$$\|DA\|_\infty = \max_{1\leq i\leq n}\sum_{j=1}^{n}(1-e_i)\pi_j^i r_{ij} = \max_{1\leq i\leq n} s_i < 1 \ .$$

Hence, $(\mathbb{I}-DA)$ is non-singular [11], and $P = (\mathbb{I}-DA)^{-1}C$ is uniquely determined. □

Proof of Proposition 2

By definition, we have $P = C + DAP$. if $\mathbf{e} = \mathbf{0}$, then $C = \mathbf{0}$ and $D = \mathbb{I}$. Further, $\forall i,j, r_{ij} = 1$ implies $A = \Pi$. Hence, we get $P = \Pi P$, so that P is the primary eigenvector of Π. In particular, since the policy π^i is uniform over N_i, the corresponding random walk makes equally likely transitions to one of the neighbors. Thus P is the Pagerank of the directed graph with sparsity structure as Π. □

Proof of Lemma 1

Since $M_D \subset M_R$, it is sufficient to show that the left-hand side is at least as big as the right-hand side. Now, for any $\Pi_R \in M_R$, and each component i, we have

$$\begin{aligned}
\sup_j r_{ij}P_j &\geq E_{j\sim\pi^{(i)}}[r_{ij}P_j] \\
\sup_j \{C_i + (1-e_i)r_{ij}P_j\} &\geq C_i + (1-e_i)E_{j\sim\pi^{(i)}}[r_{ij}P_j] \\
\sup_{\Pi\in M_D}\{C + DAP\} &\geq C + DA_RP \ ,
\end{aligned}$$

where $A = \Pi \circ R, A_R = \Pi_R \circ R$. Observing that the inequality holds for all P and for arbitrary $\Pi \in M_R$ on the right-hand side completes the proof. □

Proof of Theorem 1

The statement follows if one can show that (i) if $\exists P \in \mathcal{P}_D$ such that $P \geq \mathcal{L}P$, then $P \geq P^*$, and (ii) if $\exists P \in \mathcal{P}_D$ such that $P \leq \mathcal{L}P$, then $P \leq P^*$. First, we prove (i). Consider a policy $\Pi = (d_1, d_2, \ldots) \in M_D$. Then, $P \geq \mathcal{L}P$ implies

$$\begin{aligned}
& P \geq \sup_{\Pi\in M_D}\{C + DA_\Pi P\} \\
\Rightarrow \quad & P \geq C_1 + D_1A_1P \\
& \geq C_1 + D_1A_1(C_2 + D_2A_2P) \\
& = C_1 + D_1A_1C_2 + D_1A_1D_2A_2P \ .
\end{aligned}$$

After T steps, the inequality can be compactly written as

$$P \ \geq \ C_1 + \sum_{\tau=1}^{T-1}\left(\prod_{t=1}^{\tau}(D_tA_t)\right)C_{\tau+1} + \left(\prod_{t=1}^{T}(D_tA_t)\right)P \ .$$

The payoff corresponding to the policy $\Pi \in M_D$ is given by

$$\begin{aligned}
P_\Pi &= C + DAP_\Pi \\
&= C_1 + D_1A_1C_2 + D_1A_1D_2A_2C_3 + \ldots \\
&= C_1 + \sum_{\tau=1}^{\infty}\left(\prod_{t=1}^{\tau}(D_tA_t)\right)C_{\tau+1} \ .
\end{aligned}$$

Then, we have

$$P - P_\Pi \geq \left(\prod_{t=1}^{T}(D_tA_t)\right)P - \sum_{\tau=T}^{\infty}\left(\prod_{t=1}^{\tau}(D_tA_t)\right)C_{\tau+1} \ = \ E_1 - E_2 \ ,$$

where E_1 and E_2 are the two terms in the expression. Next, we show that as $T \to \infty$, $E_1 - E_2 \to 0$ so that $P \geq P_\Pi$. First, note that $E_1 \geq \mathbf{0}$ and $E_2 \geq \mathbf{0}$. Let $\lambda = \max(1-e_i)$ so that with $D_\lambda = \mathrm{diag}(\lambda)$, $D - D_\lambda \leq \mathbf{0}$ so that $D_tA_t \leq D_\lambda A_t = \lambda A_t$. Since $e_i > 0$,[4] $\lambda < 1$, and we have

$$E_2 = \sum_{\tau=T}^{\infty}\left(\prod_{t=1}^{\tau}(D_tA_t)\right)C_{\tau+1} \leq \sum_{\tau=T}^{\infty}\lambda^\tau\left(\prod_{t=1}^{\tau}(A_t)\right)C_{\tau+1}$$

Now, note that since $\mathbf{1} \geq \Pi_tC_{\tau+1} \geq \Pi_{t'}\Pi_tC_{\tau+1}$ for any t, t', by repeated use of the argument

$$\begin{aligned}
\mathbf{1} \ &\geq \ \left(\prod_{t=1}^{\tau}\Pi_t\right)C_{\tau+1} \ = \ \left(\prod_{t=1}^{\tau}(\Pi_t\circ R_t + \Pi_t\circ(1-R_t))\right)C_{\tau+1} \\
&= \ \left(\prod_{t=1}^{\tau}(A_t + \Pi_t\circ(1-R_t))\right)C_{\tau+1} \geq \left(\prod_{t=1}^{\tau}(A_t)\right)C_{\tau+1} \ .
\end{aligned}$$

Then,

$$E_2 \ \leq \ \sum_{\tau=T}^{\infty}\lambda^\tau\mathbf{1} \ = \ \frac{\lambda^T}{1-\lambda}\mathbf{1} \ .$$

[4]Such a condition is easy to ensure by treating non-experts, i.e., $e_i = 0$, as non-neighbors.

Hence $E_1 - E_2 \geq 0 - \frac{\lambda^T}{1-\lambda}\mathbf{1} = -\frac{\lambda^T}{1-\lambda}\mathbf{1} \to 0^-$ as $T \to \infty$. As a result,

$$P \geq P_\Pi - \lim_{T\to\infty} \frac{\lambda^T}{1-\lambda}\mathbf{1} = P_\Pi + 0^- \ .$$

Since $\Pi \in M_D$ was an arbitrary policy, we have $P \geq \sup_{\Pi\in M_D} P_\Pi$ so that $P \geq P^*$. That completes the proof of (i). Next, we prove (ii). If $P \leq \mathcal{L}P = \sup_{\Pi\in M_D} \{C + DAP\}$, then there exists $\Pi \in M_D$ for which $P \leq C + DAP$ so that, by recursively expanding the inequality for P to an infinite series, we have $P \leq (\mathbb{I} - DA)^{-1}C = P_\Pi$. Hence $P \leq \sup_{\Pi\in M_D} P_\Pi$. Combining (i) and (ii) gives the proof of the main result. □

Proof of Theorem 2

First, we show that $\mathcal{L}$ is a contraction mapping on $\mathcal{P}_D$, i.e., for any $P_1, P_2 \in \mathcal{P}_D$, $\|\mathcal{L}P_1 - \mathcal{P}P_2\| \leq \lambda\|P_1 - P_2\|$, where $\lambda \in (0,1)$ and $\|\cdot\|$ denotes the sup-norm. Let P_1, P_2 be such that $\mathcal{L}P_1(i) \geq \mathcal{L}P_2(i)$. Further, let

$$j_1^* = \underset{j\in N_i}{\text{argmax}} \ \{C_i + (1-e_i)r_{ij}P_1(j)\} \ .$$

Then,

$$\begin{aligned} 0 &\leq \mathcal{L}P_1(i) - \mathcal{L}P_2(i) \\ &\leq C_i + (1-e_i)r_{ij_1^*}P_1(j_1^*) - (C_i + (1-e_i)r_{ij_1^*}P_2(j_1^*)) \\ &= (1-e_i)r_{ij_1^*}(P_1(j_1^*) - P_2(j_1^*)) \\ &\leq (1-e_i)r_{ij_1^*}\|P_1 - P_2\| \ . \end{aligned}$$

Conversely, if P_1, P_2 be such that $\mathcal{L}P_1(i) \leq P_2(i)$ a similar argument can be used to show that

$$0 \leq \mathcal{L}P_2(i) - \mathcal{L}P_1(i) \leq (1-e_i)r_{ij_2^*}\|P_1 - P_2\| \ ,$$

where

$$j_2^* = \underset{j\in N_i}{\text{argmax}} \ \{C_i + (1-e_i)r_{ij}P_2(j)\} \ .$$

If $\lambda = \max((1-e_i)r_{ij_1^*}, (1-e_i)r_{ij_2^*})$, then $\lambda < 1$ and

$$\|\mathcal{L}P_1 - \mathcal{L}P_2\| = \sup_i |\mathcal{L}P_1(i) - \mathcal{L}P_2(i)| \leq \lambda\|P_1 - P_2\| \ .$$

Hence, $\mathcal{L}$ is a contraction operator on $\mathcal{P}_D$. Then, from Banach's fixed point theorem [3], it follows that $\exists P_0 \in \mathcal{P}_D$ such that $\mathcal{L}P_0 = P_0$. Since P_0 satisfies the condition of Theorem 1, it follows that $P_0 = P^*$, the optimal payoff. Since $P_0 \in \mathcal{P}_D$, there is a $\Pi_0 \in M_D$ that has P_0 as its payoff. Π_0 is the optimal social policy, completing the proof. □

Proof of Proposition 3

Any infinite path on the graph will always have a first repetition of a node. For σ_i, that node is v_1. If the path is generated following a deterministic policy Π, the path can go to only one node from v_1. In particular, that node will always be v_2. Following the exact same argument for v_2 and all subsequent nodes, it follows that $\bar{v}$ will keep repeating, and the path will be of the form σ_i^∞. □

Proof of Theorem 3

For any deterministic policy Π, we have

$$P_\Pi = C_1 + \sum_{\tau=1}^{\infty}\left(\prod_{t=1}^{\tau}(D_tA_t)\right)C_{\tau+1} \ .$$

If $\alpha_t = (1-e_t)r_{t(t+1)}$, where the transitions are following the policy, any component of the probability vector can be written as

$$\begin{aligned} P_\Pi(i) &= C_1(i) + \alpha_1(C_2(i) + \alpha_2(C_3(i) + \alpha_4(C_4(i) + \cdots))) \\ &= C_1(i) + \sum_{\tau=1}^{\infty}\left(\prod_{t=1}^{\tau}\alpha_t\right)C_{\tau+1}(i) \\ &= \left(C_1(i) + \sum_{\tau=1}^{T}\left(\prod_{t=1}^{\tau}\alpha_t\right)C_{\tau+1}(i)\right) \\ &\quad + \prod_{s=1}^{T+1}\alpha_s\left(\sum_{\tau=0}^{\infty}\left(\prod_{t=1}^{\tau}\alpha_{t+T+1}\right)C_{\tau+T+2}(i)\right) \end{aligned}$$

Considering a path that starts from x_{T+2}, it follows that

$$\left(\sum_{\tau=0}^{\infty}\left(\prod_{t=1}^{\tau}\alpha_{t+T+1}\right)C_{\tau+T+2}(i)\right) < 1 \ .$$

Further, $\lambda = \max_{i,j}\ ((1-e_i)r_{ij}) < 1$ is the maximum possible value of α, so that

$$\prod_{s=1}^{T+1}\alpha_s\left(\sum_{\tau=0}^{\infty}\left(\prod_{t=1}^{\tau}\alpha_{t+T+1}\right)C_{\tau+T+2}(i)\right) \leq \lambda^T \ .$$

Instantiating the infinite series for P_Π using $\Pi = \Pi_T^*$ and $\Pi = \Pi^*$, letting P_Π^T denote the probability accumulated by Π in the first T hops, and taking the difference while using the fact the residual terms in the infinite summation from $(T+1)$ lie in $[0, \lambda^T]$, we have

$$\begin{aligned} P_{\Pi^*}(i) - P_{\Pi_T^*}(i) &\leq (P_{\Pi^*}^T(i) - P_{\Pi_T^*}^T(i)) + \lambda^T \\ &\leq \exp(-cT) \ , \end{aligned}$$

since Π_T^* is the optimal T-step policy implying that $(P_{\Pi^*}^T(i) - P_{\Pi_T^*}^T(i)) < 0$, and where $c = \log(1/\lambda) > 0$ since $\lambda < 1$. Since Π^* is the optimal policy, $\forall i, P_{\Pi^*}(i) \geq P_{\Pi_T^*}(i)$, so that

$$|P_{\Pi^*}(i) - P_{\Pi_T^*}(i)| \leq \exp(-cT) \ .$$

Noting that the inequality holds for all i completes the proof. □

Proof of Theorem 4

In the algorithm, the value $v_i^{(t)}$ is the maximum probability of getting an answer when a query initiates from node x_i and is allowed to be forwarded for at most t hops. Then, from Theorem 3 we have $\|v^{(n)} - P^*\| \leq \exp(-cn)$, so that the residual error

$$\|v^{(n+1)} - v^{(n)}\| \leq \|v^{(n+1)} - P^*\| + \|P^* - v^{(n)}\| \leq 2\exp(-cn) \ .$$

Then, by a direct extension of a result in [30], it follows that the policy extracted by value iteration will be within ϵ of the optimal, where $\epsilon = \frac{2\lambda}{1-\lambda}\|v^{(n+1)} - v^{(n)}\| \leq \frac{4\lambda}{1-\lambda}\exp(-cn)$.

Now, note the algorithm is fully distributed since at any iteration every node can do its computations based on the values of the neighboring nodes in the previous iteration, but independent of the computations of any node in the current iteration. Each iteration involves computing a maximum over its neighbors, which takes $O(m_z)$ time. Since there are n iterations, the policy is computed in $O(nm_z)$ time. □