Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 05-030

Network Size Estimation In A Peer-to-Peer Network

Sandeep Mane, Sandeep Mopuru, Kriti Mehra, and Jaideep Srivastava

September 15, 2005

Network Size Estimation In A Peer-to-Peer Network

Sandeep Mane, Sandeep Mopuru, Kriti Mehra and Jaideep Srivastava
Department of Computer Science
University of Minnesota
Minneapolis, Minnesota, USA
Email: {smane,mopuru,kmehra,srivasta}@cs.umn.edu

***Abstract*— The emergence of peer-to-peer networks over the last decade has changed user's perspective about information available on the Web. But, with thousand of nodes joining and leaving a peer-to-peer network within a short span of time, it has become practically impossible for a node (or peer) to keep track of complete network. Often times, however, a node needs to at least have an estimate of number of nodes in such a network. For example, in determining time-to-live for a search query packet, a node must have a good estimate of network size. Previous deterministic approaches require a complete walk on the network, since such networks usually lack a central authority. Such approaches hence do not scale well to large networks. A few approaches, which collect partial information about the network, have been proposed as an alternative to address the scalability issues. This paper presents a novel approach for size estimation of a peer-to-peer network. The basic idea is to sample nodes in the network and then using this partial information about the network, an estimate of the network size is obtained using capture-recapture method. The capture-recapture method is a statistical method, which has been widely used for estimation of size of a closed population in oceanography and epidemiology. For a better estimate, the capture-recapture method requires two or more random (independent) samplings (sets of detected nodes) of the network. In our case, for independent sampling, we use random walks on the peer-to-peer network, since a random walk can achieve same statistical properties as independent samplings for a peer-to-peer network (see Gkantsidis et al [1]). Experimental results show that the proposed random walk based capture-recapture approach gives a good estimate of network size. In addition, results of using proposed method as well as three other size estimation methods on scale-free and random networks shows that the former algorithm gives a better estimate (lesser error) with a slight overhead on computation. This research motivates further study of estimation techniques for open networks (i.e. networks whose size changes during the estimation process).**

***Index Terms*— Peer-to-peer network, network size, capture-recapture method, random walk.**

I. Introduction

The emergence of peer-to-peer over last decade has revolutionized the world of computer networks. With peer-to-peer networks having thousands of nodes (also called "peer"), the task of keeping track of size of the network is becoming more complex and challenging. However, a node in such a network often needs to know, at least approximately, the total number of nodes in the network. For example, to search whether a particular peer is present in the network, a node needs to send a query message into the network and wait for a query reply message. To set an appropriate time-to-live for such a query message and/or an appropriate time-out for query reply, the node needs to know the approximate size of network. Thus, an estimate of network size of the peer-to-peer network is useful. Network size estimate is also required for several other applications like for setting the size of routing tables in overlays, to decide subgroup sizes and to collect network statistics. Deterministic estimation of size of a peer-to-peer network is achieved by sending out broadcast packets that collect information about complete network. However, this method is practically not feasible for large-scale networks as the communication as well as time overheads are extremely large. In addition, the dynamic nature of peer-to-peer network (since nodes join and leave the network frequently) makes it difficult to do a complete walk on the network within a short time interval. The dynamic nature of the network also necessitates that a node must estimate size of the network frequently. Thus, the main problem of interest in this paper is to efficiently obtain a good estimate of network size.

In this paper, we address the problem of estimating the size of a peer-to-peer network using partial information about the network. The capture-recapture method is used successfully to estimate size of a closed population (i.e. a population whose size does not change during the estimation process) in many research fields like oceanography (e.g. to estimate number of fish) and epidemiology (e.g. to estimate size of patient groups in a population). This statistical method requires two or more samplings of the population. The individuals detected in these samplings are cross-tabulated and then an estimate of the population size is obtained using ML-estimation techniques (if independence holds among samplings) or log-linear models (otherwise). The former is a simpler method while the latter is computation intensive modeling technique. The capture-recapture method gives a good estimate using ML-estimation technique if the samplings are random (independent of each other). Gkantsidis et al showed in their work [1] that a random walk on a peer-to-peer network can achieve same statistical properties as independent sampling of nodes in the network. This motivates us to use random walks on a peer-to-peer network for sampling in our capture-recapture based approach. Simulation results using such a combined approach using random walks and capture-recapture shows that a good estimate of network size can be obtained with reasonably less amount of communication overhead. Since the random walk is carried out for a fixed number of steps, we also study the effect of the number of hops of a random walk on the accuracy of size estimate. It is found that the estimate stabilizes to approximately the network size after a threshold value for time-to-live. In addition, the effect of different parameters of network topology like preferential connectivity and number of neighbors of a node on the accuracy of estimate is also studied. We simulate three other size estimation methods and observe that the proposed approach performs as well as, or is some cases better than, other size-estimation methods on scale-free networks (Barabási-Albert model) and random networks (Waxman model).

The rest of the paper is organized as follows: Section 2 gives a background of related work for the problem of estimation of network size. Section 3 formally defines the problem of estimating size of a peer-to-peer network and then explains the theory behind techniques used in this paper, namely, capture-recapture method and random walk on peer-to-peer network. Section 4 first explains the proposed random walk-based capture-recapture algorithm for estimating size of a peer-to-peer network, then illustrates an example and finally analyzes the time complexity of algorithm. Section 5 shows simulation results of proposed approach and three other size estimation techniques on two network topologies. Section 6 summarizes the conclusions and provides future directions for research.

II. Related work

One deterministic approach to determine the size of a peer-to-peer network is to gather information about entire network by sending broadcast messages. Though such an approach gives exact size of the network, it does not scale well to very large networks. In addition, the overhead of communication costs as well as time required to gather information are too high. A few other approaches have been proposed previously to estimate the size of peer-to-peer networks, most of which use the partial information collected about the network to estimate the size of network. Horowitz and Malkhi [2] propose a ring-based algorithm for size estimation, which requires a logical ring to be maintained among the existing processes in the system. Bawa et al [3] propose three different approaches for network size estimation. The first, called "Randomized Report", is a broadcast protocol except that when a node receives the broadcast message, it may reply back to the sender (source node) with a probability p. If sender (of broadcast) sees k replies in a time interval ΔT (starting from the broadcast time), then this approach estimates the size of network as k/p. The second one is a "Birthday Paradox"-based approach for size estimation where the algorithm keeps sampling nodes randomly until a node is sampled twice for the first time. If X_r is the total number of nodes sampled, then the estimate of size of the network is $(X_r^2/2)$. The third approach, called "Random Increasing Walks", is based on a random walk, which always goes from a node with lower identifier to a node with higher identifier until the walk cannot go further. The length of walk is determined as l and averaged over a number of such walks as $\tilde{l}$. The network size is estimated as $e^{\tilde{l}}$. Dimitrios et al. [4] also propose two methods for estimating size of distributed systems based on gossip protocols. The first approach, called "Hops-sampling", is similar to the approach proposed by Bawa et al. It estimates the network size using Birthday Paradox and gossip algorithm. The other approach is called "Interval density". In this approach, each process identifier is hashed to a value in the range [0,1]. The total number of processes is estimated based on the number of processes identifiers that hash to a sub-interval of [0,1], assuming a uniform distribution for process identifiers. The latter approach requires a central authority which assigns node identifiers to nodes and hence is difficult to implement in a peer-to-peer network.

III. Size estimation of a peer-to-peer network: Theoretical background

Before proceeding further, we formally define the problem and explain the statistical theory behind the techniques used in the approach proposed in this paper.

A. Problem statement

Estimating the size of a peer-to-peer network is necessary for numerous applications like determining the size of routing tables and collecting network statistics. However, traditional, deterministic broadcast methods do not scale to large networks. Hence, this paper proposes a method to efficiently estimate network size using partial information collected about the network. The main problem thus addressed by this paper is – *"How can a node in a peer-to-peer network obtain an accurate estimate of size of the network efficiently ?"*

B. Capture-recapture method

To address the defined problem, we will present a novel approach for size estimation based on the capture-recapture method. This is a statistical technique which is widely employed in different research fields of like oceanography and epidemiology. It is used to estimate the size of closed populations. The method was initially used by Laplace to estimate the population of France in 1786. Later, in 1896, Petersen used it to estimate the harvestable stocks of Danish fish populations. In recent times, this method has been used effectively to estimate the size of patient groups among a defined human population.

1) The Approach: The basic capture-recapture method (see Darroch [5]) is based on a "two-sample model". It involves an initial random sampling of the population, marking the samples, releasing the marked samples back into the population and then recapturing another sample randomly from the population. Based on the number of individuals captures in both samples, it is possible to estimate the total population. This method can also be extended to allow multiple samplings.

An example of estimation of fish in a pond using capture-recapture method is now illustrated (refer to figure 1). In the first step, called "capture step", a random sample of fish is caught from the pond, as shown in subfigure 1(a). Let N_1 be the number of fish caught in this first sampling. These fish are marked and then set free into the pond. The fish are allowed to mix for some time (to reduce dependence between samplings). In the second step, called "recapture step", again a random

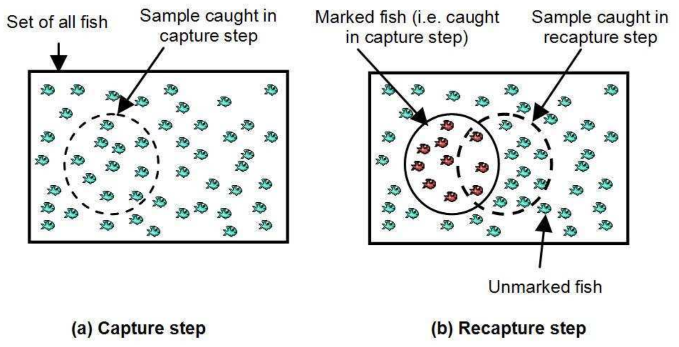

Set of all fish
Sample caught in capture step
Marked fish (i.e. caught in capture step)
Sample caught in recapture step
Unmarked fish
(a) Capture step
(b) Recapture step

Fig. 1. Capture-recapture method for estimating fish in a pond.

TABLE I
CONTINGENCY TABLE FOR TWO-SAMPLE CAPTURE-RECAPTURE METHOD

		Capture step 1	Capture step 0	
Recapture step	1	n_{11}	n_{01}	N_2
	0	n_{10}	n_{00}	
		N_1		N

sample of fish is caught from the pond and let N_2 be the number of fish captured in this step. This is shown in subfigure 1(b). Let n_{11} be number of marked fish caught in second step. Assuming that the fish population in pond remains constant during the estimation process as well as the two samplings are independent of each other, then the number of marked fish n_{11} captured in the second step (recapture step) will follow a hypergeometric distribution.

The fish captured in the two steps are cross-tabulated in a 2x2 contingency table, as shown in table I. Each cell represents the number of fish caught (or not caught) in each of the two steps. Thus, for a cell count n_{ij}, the first subscript $i = 1$ ($i = 0$) represents fish are caught (not caught) in capture step while the second subscript $j = 1$ ($j = 0$) represents fish are caught (not caught) in recapture step. Thus, n_{10} is the number of fish captured only in capture step and n_{01} is the number of fish captured only in recapture step. Also, n_{00} is the number of fish not captured in either of the two steps. The sum of all the cells in the table (N) is the total number of fish in the pond.

If the capture and recapture steps are independent of each other, then the probability of a fish being captured in both capture and recapture steps is the product of respective individual probabilities for the fish.

$$\begin{aligned} &Pr((capture = \ 1) \ AND \ (recapture = \ 1)) \\ &= Pr(capture = \ 1) \times Pr(recapture = \ 1) \\ &= \frac{N_1}{N} \times \frac{N_2}{N} \end{aligned} \tag{1}$$

But, from the table I, we have

$$Pr((capture = \ 1) \ AND \ (recapture = \ 1)) = \frac{n_{11}}{N} \tag{2}$$

Hence, using eq.1 and eq.2,

$$\begin{aligned} \frac{n_{11}}{N} &= \frac{N_1}{N}\frac{N_2}{N} \\ \therefore N &= \frac{N_1 \times N_2}{n_{11}} \end{aligned} \tag{3}$$

This equation gives the ML-estimate of total number of fish in the pond (similar to Goldberg and Wittes [6]). In case independence does not hold among cell frequencies in table I, then log-linear modeling techniques can be used to model the cell frequencies. Using the best-fit log-linear model for contingency table, an estimate for total number of fish in the pond is obtained (for more information on log-linear models, see Knoke and Burke [7]).

In capture-recapture method, the following assumptions are usually assumed to hold –

(i) The population of fish is closed i.e. there is not change in the population during estimation.
(ii) There are no loss of tags (individuals can be matched from capture to recapture).
(iii) For each step, each individual has equal chance of being in the sample.

2) Application of capture-recapture method to peer-to-peer network: The method of estimation of fish is applied to the peer-to-peer networks to estimate size of the network. The nodes in a peer-to-peer network are analogous to fish in pond and a sampling of population is analogous to a randomly detected set of nodes (or peers) in the network. Thus, for a two-sample strategy, two independent samplings are obtained of peer-to-peer network. Knowing which nodes have been captured in each of the two samples, an estimate of the total number of nodes in the network is obtained using eq.3.

In order to accommodate the case where there are no common nodes between the capture and recapture steps, we approximate the eq.3, similar to Petersen's approach [8], as below –

$$\therefore N = \frac{(N_1 + 1) \times (N_2 + 1)}{(n_{11} + 1)} \tag{4}$$

C. Sampling using Random Walks

For using capture-recapture method to estimate size of a peer-to-peer network, a method is required to sample nodes in the network. Such a method must give independent samplings of nodes while allowing for efficient implementation. In this paper, we use random walks for sampling of nodes in the network, as explained further.

A *random walk* (also known as "*drunkard's walk*") is formally defined as a sequence of successive steps on a graph with each step taken along a random direction. Such a walk does not have history, i.e. the next step taken depends only on the walker's current position. Thus, a random walk has 1^{st} order Markov property. The simplest random walk is a path constructed assuming the following conditions – (i) there is a starting point, (ii) the distance from one point in the path to the next is a constant, and (iii) the direction from one point in the path to the next is chosen at random, and no direction is more probable than another. Random walk on graphs is a well-researched field of study with a wide range of applications in areas like physics and economics. Lovász [9] provides a nice survey of random walks on graphs.

Gkantsidis et al, in their paper [1], showed that for a peer-to-peer network, (i) a random walk is an excellent candidate to simulate sampling of peer-to-peer networks, and (ii) the number of simulation steps required can be as low as number of samples in independent uniform sampling. Thus, they show that the nodes chosen in a random walk on peer-to-peer network can achieve same statistical properties as independent sampling of nodes from the network. This motivates us to use random walks for sampling nodes in a peer-to-peer network. If S is a node which wants to estimate the size of the network, we use two separate random walks from the same starting node S to sample nodes in the peer-to-peer network. Then, cross-tabulating the nodes sampled in two random walks as shown in Table I, we estimate the total number of nodes in the network.

IV. Random walk-based Capture-Recapture Approach (CR)

A. Algorithm

A node that needs to estimate size of the network is called *"source node"*. All remaining nodes in network are called *"non-source nodes"*. The source node acts as the start node for random walks in both capture and recapture steps. In capture step, the source node sends out a message (called *"gossip message"* or just *"gossip"*[1]) to a randomly selected subset of its neighbors. Each neighbor upon receiving the gossip forwards it to a randomly selected subset of its neighbors, provided that the *time-to-live* of gossip is greater than zero. The *time-to-live* for a gossip message is defined as the number of hops that a gossip message can traverse further into a network from the current node. If *time-to-live* of gossip becomes zero, the node builds a *"gossip-reply"* message which contains the list of nodes traversed by the gossip. It then sends that gossip-reply message to source node along the same path (backwards) along which the gossip traveled. The source node, upon receiving the gossip-reply message, retrieves the nodes along which the gossip traveled. The source node waits for a time (2**time-to-live*) before compiling the set of all nodes detected in capture step. This procedure is similarly repeated from same source node for the recapture step and the set of nodes detected is determined. An estimate of number of nodes in the network is then made using the nodes detected in these capture and recapture steps.

The algorithms 1 and 2 are implemented at each node of the network. For sampling, the two algorithms together enable sampling of the network. The algorithm 1 is used by a node which acts as source node while the algorithm 2 is used by non-source nodes. The CR algorithm has one input parameter – the *time-to-live* for gossip message. Also, it should be noted that since each non-source node may forward a gossip to more than one of its neighbors, a new random walk may start at a non-source node. Thus, in the CR algorithm, each sampling of the peer-to-peer network is basically a set of random number of random walks (not necessarily having same length) on the network. The sampling is also similar to randomized rumor spreading (see Karp et al [11]), where a gossip is introduced and randomly spread in a network and then the number of nodes, which know the gossip after a certain time interval, is determined.

[1]It is similar to a gossip spreading randomly in a network [10]. But, unlike the latter case, here the gossip is constrained in terms of how far (number of hops) it can spread.

Algorithm 1 CR algorithm for source node

Pseudo-code:

```
1.  Create a gossip message $m$, and add the source node to
    traversed path list in $m$.
2.  For each node in a randomly selected set of neighbors,
3.      Send gossip $m$ to neighbor.
4.  End For
5.  Wait for incoming messages.
6.  If (received message $m'$ is a gossip or gossip-reply
    message)
7.      Store nodes in traversed path in message $m'$.
8.  End If
```

Algorithm 2 CR algorithm for non-source node

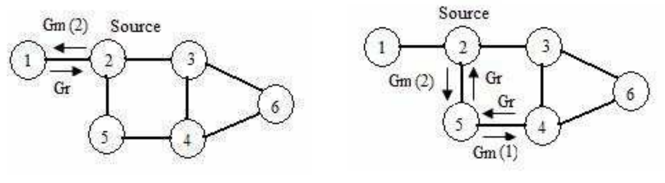
Pseudo-code:

```
1.  Wait for incoming messages.
2.  If (received message $m$, is a gossip message)
3.      Add node to the traversed path of the $m$.
4.      Decrement time-to-live of $m$.
5.      If (($m$ wasn't previously received and its time-to-live
        is Zero) or (number of neighbors is 1))
6.          Create a gossip-reply message containing $m$'s tra-
            versed path and send it back to source node.
7.      Else
8.          For each node in a randomly selected set of
            neighbors (except the sender)
9.              Send gossip $m$ to neighbor.
9.          End for
10.     End if
11. Else if (received message $m$ is a gossip-reply message)
12.     Forward the message $m$ to next node in the path to
        source node.
13. End if
```

B. Example

Network size estimation using CR algorithm is illustrated on the peer-to-peer network shown in Figure 2. For this network, consider a case where node 2 wants to estimate the size of the network. Suppose node 2 uses a gossip message with time-to-live of 2.

Capture Step: The capture step is shown in subfigure 2(a).



(a) Capture step (b) Recapture step

Fig. 2. Illustration of Random-walk based Capture-Recapture algorithm on a peer-to-peer network. (The number in parenthesis indicates the time-to-live of gossip message.)

Node 2 sends the gossip message (G_m) to randomly selected subset of its neighbors. Suppose that the node 2 decides to send G_m to node 1 (selected at random). Node 2 creates G_m with a time-to-live equal to 2 and adds itself to the traversed path in G_m. It then sends G_m to node 1. Node 1, on receiving G_m, decrements time-to-live for G_m and adds itself to the traversed path in G_m. Since node 1 has no new neighbors except node 2, it creates and sends back a gossip-reply message G_r to node 2. Node 2 thus compiles the list of nodes detected in capture step as -

List of nodes detected in capture step: {2, 1} (a)

Recapture Step: The recapture step is shown in subfigure 2(b). Assume, in this step, node 2 selects to forward the gossip message G_m to node 5. On receiving G_m, node 5 decrements time-to-live for G_m, adds itself to the traversed path in G_m and then forwards G_m to node 4 (selected at random). On receiving G_m, node 4 decrements the time-to-live and adds itself to the traversed path. Since time-to-live for G_m is zero, node 4 does not forward G_m. Instead it creates a gossip-reply message G_r and includes the traversed path from G_m in G_r. It then sends G_m back to the source node (node 2) along the path along which it received G_m. Node 2, on receiving G_r, determines the list of nodes visited by G_m as –

List of nodes detected in recapture step: {2, 5, 4} (b)

Using eq.4, N is thus calculated to be, $N = (3) * (2)/(1) = 6$ Note that the extra ones in eq.4 are implicitly added in the CR algorithm by including the source node in both lists (a) and (b). The results may vary depending on the nodes detected in each sampling. We explore this variation in our simulations in section V.

C. Time complexity and communication costs

The worst-case time complexity of each sampling (capture or recapture step) in the above approach is now analyzed. In our analysis, we assume that each hop requires same time. (In real-world situations, the time for each hop may not be constant. For such a case, one can instead use the maximum time for any hop.) Let t be the time-to-live (measured in number of hops) for a gossip message. Then, the gossip message, starting from a source node S, will travel a distance of at most t hops from S. Thus, a node E which builds the gossip-reply message is at a distance of at most t hops away from S. The gossip reply message will require at most t hops to return from node E to S. Thus, the total time required to gather the list of nodes detected in a step is $2 * t$ i.e., the time complexity of each sampling is $O(t)$.

Since the communication cost (network cost) is much more than the computation cost, complexity analysis of the later is not done. In the worst case where the time-to-live is greater than the diameter of network topology and the probabilities of a new random walk at non-source nodes are high, the CR algorithm will most likely degenerate into a broadcast algorithm. In such a case, a gossip message is sent to each node. Thus, the worst case communication cost (number of messages sent in network) of CR algorithm will be same as a broadcast algorithm.

V. SIMULATION

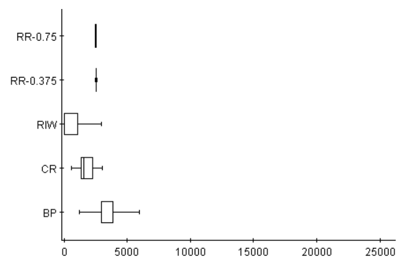
To evaluate the performance of size estimation algorithms, we simulate CR algorithm and three other size estimation algorithms on a peer-to-peer network. The simulation results are promising and experiments using real-world data will be carried out in future work.

A. Simulation methodology

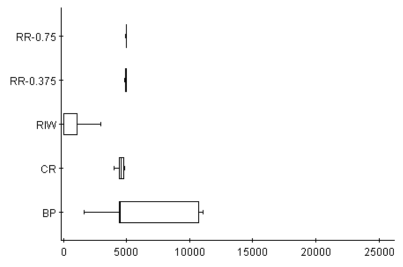
In order to evaluate the performance of CR algorithm, we simulate the algorithm on a peer-to-peer network using SimJava [12], a discrete event, process-oriented simulation package in Java. For each simulation, a network topology of the underlying network is generated. Here, each node represents a single node (peer) in the network. Networks of varying size, from 1000 nodes to 10,000 nodes are generated and CR algorithm is simulated on these networks. The CR algorithm has one parameter, viz., the *time-to-live* (measured in number of hops) for the gossip message. The performance of CR algorithm over a range of values for *time-to-live* is evaluated.

To evaluate other network size estimation techniques, we simulate following three algorithms proposed by Bawa et al. [3] – (i) Birthday Paradox (BP), (ii) Random Increasing Walk (RIW), and (iii) Randomized Report (RR). The performance of these three algorithms on same networks is evaluated. For CR algorithm, the number of trials (samplings) is constant i.e. two. The BP algorithm has one additional parameter ϵ, where (number of trials) $= 1/(\epsilon^2)$. Using analogy between CR and BP algorithms, we have (number of trials) $= 2 = 1/(\epsilon^2)$. Therefore, $\epsilon = 0.707$. There are no input parameters for RIW algorithm. For RR algorithm, we use the values 0.375 and 0.75 for p in order to evaluate the effect of change in probability (that a node will reply) on the performance of RR algorithm.

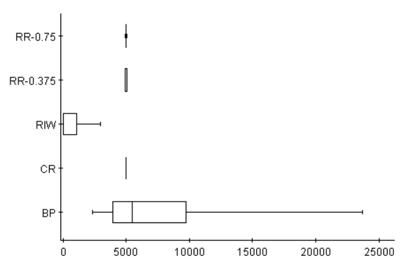
To measure the performance of various algorithms, an estimate of size of peer-to-peer network is determined, given actual size of the network. To remove any bias in results, each simulation is repeated several times and the results are presented as the averaged network size estimate and its standard deviation. The performance of various algorithms is evaluated based on two factors – (i) Range (which is analogous to time-to-live for CR algorithm), and (ii) Source node.



(a) range = 3

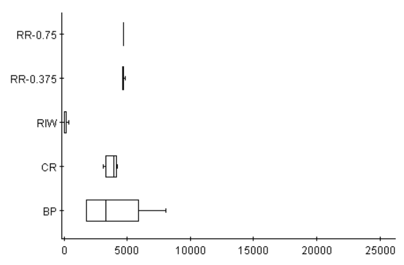


(b) range = 4

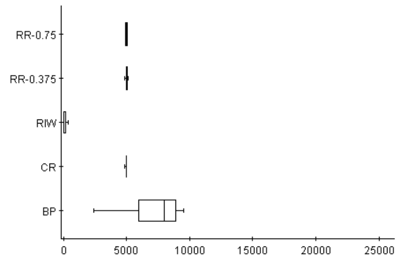


(c) range = 5

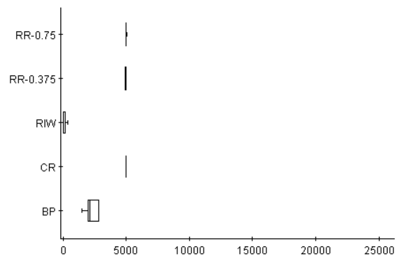
Fig. 3. Effect of range on network size estimate of a 5000-node Waxman network.



(a) range = 3



(b) range = 4



(c) range = 5

Fig. 4. Effect of range on network size estimate of a 5000-node Barabási-Albert network.

B. *Network topology*

Recent studies of real-world networks (e.g. Medina et al [13]) have shown a recurring discovery of a remarkable common property among all the natural networks – the inter-node connection distributions (i.e. the degree distributions) decay with a power-law tail. Networks having this remarkable property are called *scale-free* networks (see Barabási and Albert [14]). In such a network, most nodes have very few connections while a few nodes (hubs) have extremely large number of connections. Adamic et al [15] cite two examples of measurements – one of Gnutella network and the other of a simulated Freenet network – which show that this holds true even for peer-to-peer networks. In other words, peer-to-peer networks have a power-law degree distribution. Another research of degree distribution in a peer-to-peer network by Ripneau et al [16] showed that although Gnutella is not a power-law network, its configuration shows the benefits as well as drawbacks of a power-law structure. Hence, we evaluate Barabási-Albert model (scale-free model) in our simulations for peer-to-peer network.

Since this work can be applied to other distributed systems, we also consider the most commonly used random graph in such systems for simulations, namely, the Waxman model [17]. For a Waxman model, the topology generator discourages long links by making the probability of a link between two nodes proportional to the Euclidean distance between them. This is a fairly simple random graph used in most simulations. It should be noted that the Waxman model emphasizes connection locality. Newman [18] provides a nice survey of properties of complex networks.

The size estimation algorithms are evaluated using two network topologies, namely (i) Waxman router network, and (ii) Barabási-Albert model (scale-free model), generated using BRITE [19], a universal topology generator. The former network is an example of random graph while the later is an example of degree-based graph (see Tanguminarunkit et al. [20] for classification of network topology generators). We used following parameters for Barabási-Albert and Waxman models – $\alpha = 0.15$ and $\beta = 0.2$ with $m \in \{2, 5, 10, 15\}$. More details about the parameters in BRITE are explained by Medina et al [19]. In our simulations, we study the effect of two parameters of network topologies on the size estimated by each algorithm, namely – (i) Preferential connectivity and, (ii) Number of neighbors.

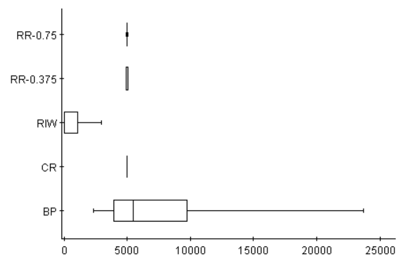
C. *Simulation results I: Effect of algorithm's parameters*

1) Performance of CR algorithm: Table II shows preliminary results of CR algorithm, each network having two neighbors ($m = 2$) per new node. The *time-to-live* for the gossip message is 10. The table shows that mean estimates obtained using CR algorithm do not deviate by a large amount for the different network sizes. In addition, it is observed that the standard deviation for the 5000-node Waxman router is 256.47 while for the 5000-node Barabási-Albert (BA) router network, it is 8.41. This shows that the accuracy of estimate of CR algorithm varies for diferent network topologies.

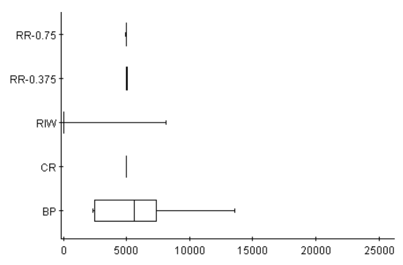
TABLE II

RESULTS OF SIMULATION ON TWO DIFFERENT NETWORK TOPOLOGIES. (EACH ROW IS AVERAGED OVER 10 DIFFERENT RUNS)

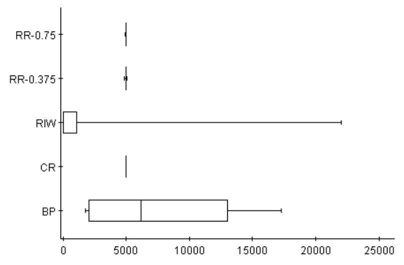
Type of network	No. of nodes	Mean estimate	Std. deviation
Waxmam router network	1000	995	4.86
	5000	4676	256.47
BA router network	1000	991	5.83
	5000	4962	8.41



(a) Source node is 0

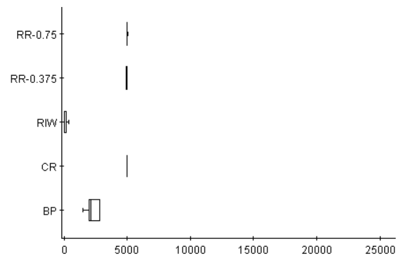


(b) Source node is 2499

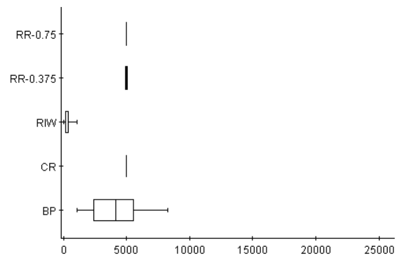


(c) Source node is 4999

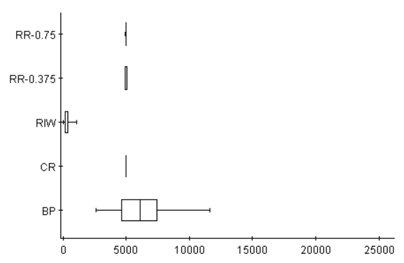
Fig. 5. Effect of source node on network size estimate of a 5000-node Waxman network.



(a) Source node is 0



(b) Source node is 2499



(c) Source node is 4999

Fig. 6. Effect of source node on network size estimate of a 5000-node Barabási-Albert network.

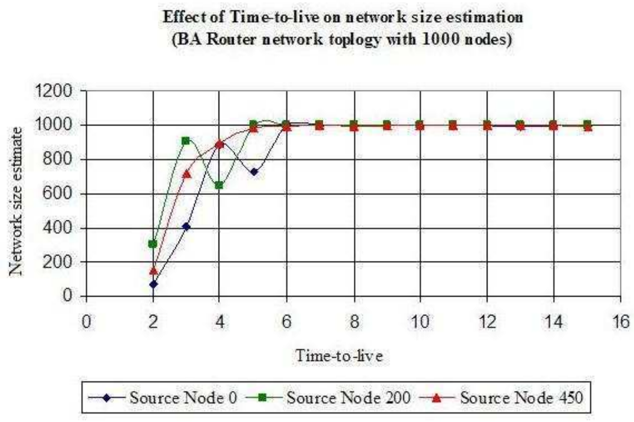
2) Effect of range: Figures 3 and 4 show the box-plots for the effect of the range (time-to-live) on the network size estimated using RR, RIW, CR and BP algorithms on a 5000-node Waxman network and a 5000-node Barabási-Albert network respectively. The box plots show the mean estimate and the standard deviation for five runs of each simulation. It is observed that the RIW algorithm performs poorly for both network topologies. The range has a noticeable effect on CR,RR and BP algorithms, with higher range values giving better predictions. Both cases of RR algorithms give a good estimate of the network size, with the one with higher probability p=0.75 giving lesser standard deviation. This is as expected since for higher probability, the RR algorithm tends to become a broadcast (complete walk) of the network.

3) Effect of source node: Similar to the previous section, figures 5 and 6 show the box-plots for the effect of source node on network size estimated using RR, RIW, CR and BP algorithms on a 5000-node Waxman network and a 5000-node Barabási-Albert network respectively. The reason we study this effect is because, in a preferentially-connected Waxman networks (or Barabási-Albert network), it is necessary to verify whether the algorithms give a good estimate irrespective of source node (i.e. location of node in the network). RIW algorithm does not perform well for either of the network models. Both CR and RR algorithms perform well in all the cases. BP algorithm shows a noticeable change in variance (standard deviation) for different source nodes. However, the mean estimate, which is useful in most practical cases, is approximately close to the actual network size for Waxman model.

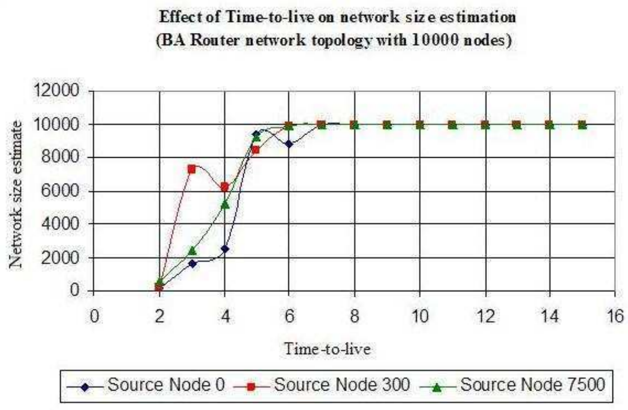
4) Convergence of estimate: The main motive of this part of simulations was to determine whether the estimated size converges to the actual network size with range. For these simulations, two BA router networks with 1,000 and 10,000 nodes respectively are used. Each simulation consists of estimating the network size using CR algorithm for each network over a range of values for *time-to-live*, until the estimate of the network converges to a certain value. Subfigures 3(a) and 3(b) show results for the two respective networks. It is observed from subfigure 3(a) that the estimates improve as *time-to-live* parameter increases. And for values of *time-to-live* greater than 6, the estimation converges to the actual network size. For subfigure 3(b), the estimate converges to actual network size for values of *time-to-live* greater than 7. One possible reason for these results is that small-world behavior is often incorporated in scale-free (power law) networks in the network topology generator. This results in the simulated scale-free network to have a small diameter. Thus, CR algorithm may be useful to determine approximately the diameter of a real-world peer-to-peer network and further study is required.

D. Simulation results II: Effect of network properties

1) Effect of number of neighbors: BRITE uses a parameter m, which is the number of neighbors per new node. This parameter specifies the linkage among nodes in the network and hence we study its effect on the estimate. The figures 8 and 9 show the box-plots for the effect of number of neighbors on network size estimated using RR, RIW, CR and BP algorithms on a 5000-node Waxman network and a 5000-node Barabási-Albert network respectively. As seen, CR algorithm shows slight variance for $m = 5$ but it shows very low variance or no variance for $m = 10$ or $m = 15$. The reason for this is that CR algorithm basically uses a random number of random walks for each sampling. When m is small, the number of such random walks is reduced and hence the number of nodes detected in a sampling is much low. However, as m increases, the number of

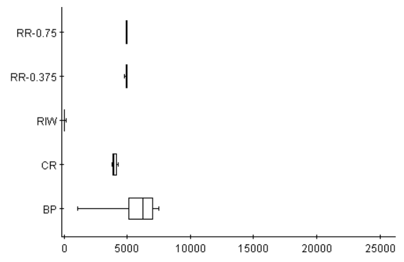


(a) 1,000-node Barabási-Albert network

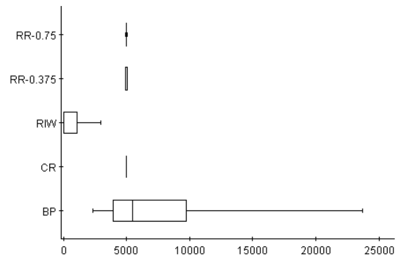


(b) 10,000-node Barabási-Albert network

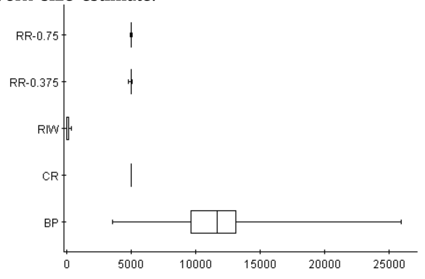
Fig. 7. Effect of gossip message's time-to-live on network size estimate.



(a) Number of neighbors = 5

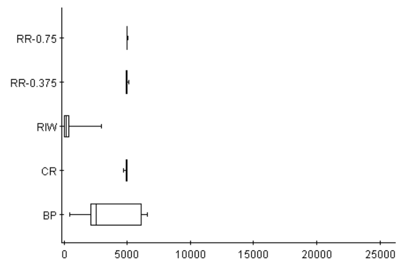


(b) Number of neighbors = 10

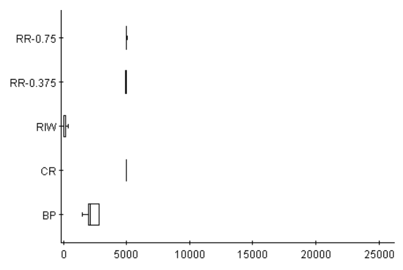


(c) Number of neighbors = 15

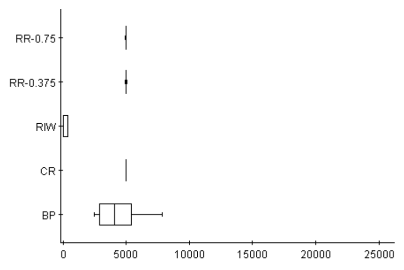
Fig. 8. Effect of number of neighbors on network size estimate of a 5000-node Waxman network.



(a) Number of neighbors = 5



(b) Number of neighbors = 10



(c) Number of neighbors = 15

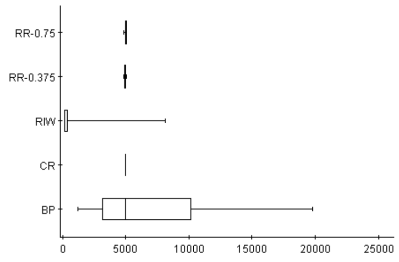
Fig. 9. Effect of number of neighbors on network size estimate of a 5000-node Barabási-Albert network.

random walks increases and hence it reduces the dependence between samplings. Thus, it gives a better estimate.

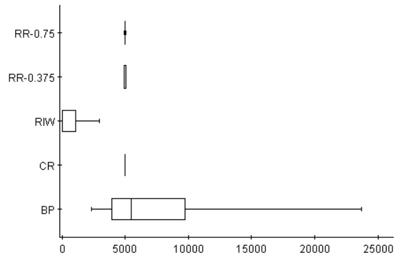
2) *Effect of preferential connectivity:* The figures 10 and 11 show the box-plots for the effect of preferential connectivity on the network size estimated using RR, RIW, CR and BP algorithms, for a 5000-node Waxman network and a 5000-node Barabási-Albert network respectively. In case of Waxman network, preferential connectivity is based on the Euclidean distance between the nodes, while in Barabási-Albert network, it is based on the degree distribution of the nodes in the network. The BP algorithm gives a better estimate for Waxman model than for Barabási-Albert model. The reason for this is that BP algorithm assumes that the underlying network is random, which holds in case of Waxman model but not for Barabási-Albert model. In Barabási-Albert model, preferential connectivity reduces the variance of estimate for BP algorithm but the mean still remains much farther away from the actual value. Further simulations have shown that preferential connectivity worsens the estimate of BP algorithm. RIW algorithm performs poorly irrespective of preferential connectivity. The RR and CR algorithms perform well in both cases because these algorithms are more dependent on the randomness (in sampling or in response) of nodes rather than the underlying link structure of network.

E. Simulation results III: Communication costs

It was previously noted that Capture Recapture and Randomized Report give better size estimates than the other two

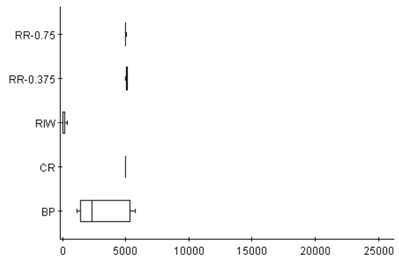


(a) Without preferential connectivity

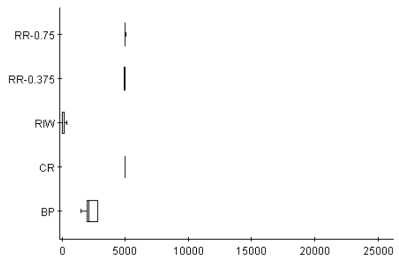


(b) With preferential connectivity

Fig. 10. Effect of preferential connectivity on network size estimate of a 5000-node Waxman network.

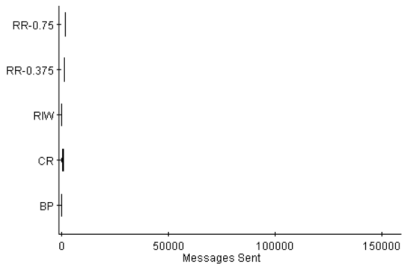


(a) Without preferential connectivity

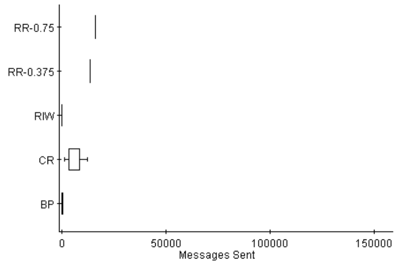


(b) With preferential connectivity

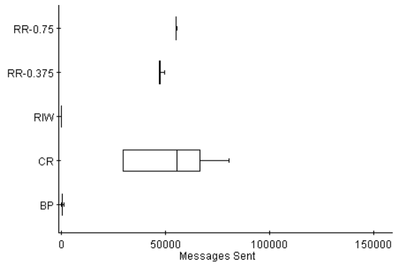
Fig. 11. Effect of preferential connectivity on network size estimate of a 5000-node Barabási-Albert network.



(a) Range = 3

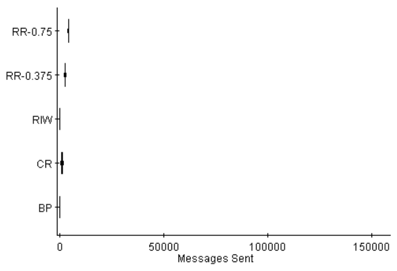


(b) Range = 4

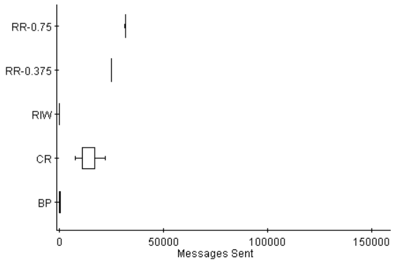


(c) Range = 5

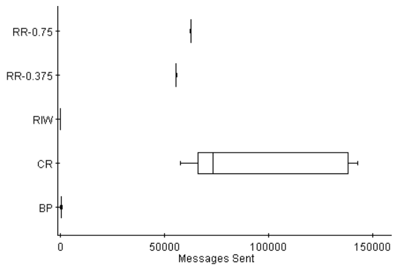
Fig. 12. Communication costs for a 5000-node Waxman network.



(a) Range = 3



(b) Range = 4



(c) Range = 5

Fig. 13. Communication costs for a 5000-node Barabási-Albert network.

algorithms. But these algorithms should also be evaluated in terms of the communication overheads associated with each algorithm (i.e. number of messages sent in the network). Such an evaluation is useful when applying these algorithms to a similar network (e.g. wireless sensor network), in which communication costs play a very important role. We had observed that the accuracy of size estimate using Capture Recapture and Randomized Report improves as range increases. However, the communication cost of these two algorithms also increases significantly as range increases (see figure 12 and figure 13). On the other hand, though Birthday Paradox gives lesser accurate size estimate than Capture Recapture and

Randomized Report, it has lesser average communication cost.

VI. Conclusions and Future Work

In this paper, we present a novel way to address an important question in peer-to-peer networks – how can a node in a peer-to-peer network estimate the size of network ? The main statistical techniques – capture-recapture method and random walk – used in this paper are explained and then a random-walk based capture-recapture method is explained for estimation of network size. Simulations using two network models (Waxman and Barabási-albert model) are shown for the proposed CR method as well as three other network size estimation techniques. Our paper also focuses on exploring the effects of algorithmic parameters as well as key properties of network models on performance (accuracy) of the four different network size estimation techniques. Randomized-Report approach performs as well as CR algorithm. Random Increasing Walk does not fare well in heavy tailed networks and may not be useful for real world networks which are known to have a heavy tailed distribution. We also observe that communication costs for BP and RIW are less than that of CR and RR. While the performance of RIW is observed to be poor in heavy tailed networks, BP has a comparable performance to CR and RR. Hence BP is a more viable option in applications which do not have stringent requirements for a low margin of error. The communication overheads of CR algorithm, though small, are present and it should be used where higher accuracy for network size estimate is required. It should be noted that though here we illustrate the proposed approach for peer-to-peer networks, it can be used to estimate size of distributed systems in general.

Our research has tested the different size estimation algorithms only on closed populations. Nodes are continuously joining and leaving a peer-to-peer network. This dynamic nature of such networks necessitates the study of size estimation methods using open populations techniques (e.g. Jolly-Seber model [21]). A comparative evaluation of the performance of different size estimation algorithms on open populations is also required. We observe that CR algorithm performs well in terms of accuracy. However, it has an overhead as it needs additional logic for random propagation. More specifically, it requires a random forwarding of gossip by a node to its neighbors. This is different from other algorithms which propagate the gossip message to all their neighbors (i.e. broadcasting approach). Thus efficient methods for implementation of randomness for a CR algorithm need to be explored. Another field of research is the study of application of these techniques to network topologies of Mobile Ad Hoc Network (MANET).

Acknowledgment

Sandeep Mane's and Jaideep Srivastava's work was supported in part by NSF Grant ISS-0308264 and by ARDA Grant F30602-03-C-0243.

References

[1] C. Gkantsidis, M. Mihail, and A. Saberi, "Random walks in peer-to-peer networks," in *Proceedings of IEEE INFOCOM*, 2004.

[2] K. Horowitz and D. Malkhi, "Estimating network size from local information," *Inf. Process. Lett.*, vol. 88, no. 5, pp. 237–243, 2003.

[3] M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani, "Estimating aggregates on a peer-to-peer network," Stanford University, Tech. Rep., 2003.

[4] D. Psaltoulis, D. Kostoulas, I. Gupta, K. Birman, and A. Demers. (2004) Practical algorithms for size estimation in large and dynamic groups. [Online]. Available: http://citeseer.ist.psu.edu/681255.html

[5] J. N. Darroch, "The multiple-recapture census: I. estimation of a closed population," *Biometrika*, vol. 45, no. 3/4, pp. 343–359, 1958.

[6] J. Goldberg and J. Wittes, "The estimation of false negatives in medical screening," *Biometrics*, vol. 34, no. 1, pp. 77–86, 1978.

[7] D. Knoke and P. Burke, *Log-Linear Models*. Newberry Park, California, USA: Sage publications Inc, 1980.

[8] C. G. J. Petersen, "The yearly immigration of young plaice into the limfjord from the german sea," *Report of the Danish Biological Station*, vol. 6, pp. 1–77, 1896.

[9] L. Lovász, "Random walks on graphs: A survey," *Combinatorics, Paul Erdös is eighty*, vol. 2, pp. 1–46, 1993.

[10] S. M. Hedetniemi, S. T. Hedetniemi, and A. Liestman, "A survey of gossiping and broadcasting in communication networks," *Networks*, vol. 18, pp. 129–134, 1988.

[11] R. M. Karp, C. Schindelhauer, S. Shenker, and B. Vöcking, "Randomized rumor spreading," in *IEEE Symposium on Foundations of Computer Science*, 2000, pp. 565–574.

[12] F. Howell and R. McNab, "SimJava: a discrete event simulation package for java with applications in computer systems modelling," in *First International Conference on Web-based Modelling and Simulation*, Jan 1998.

[13] A. Medina, I. Matta, and J. Byers, "On the origin of power laws in internet topologies," *ACM Computer Communication Review*, vol. 30, no. 2, pp. 18–28, 2000.

[14] A.-L.Barabási and R.Albert, "Emergence of scaling in random networks," *Science*, vol. 286, pp. 509–512, 1999.

[15] L. Adamic, R. Lukose, A. Puniyani, and B. Huberman, "Search in power-law networks," *Physical Review E*, vol. 64, 2001.

[16] M. Ripeanu, I. Foster, and A. Iamnitchi, "Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design," *IEEE Internet Computing Journal*, vol. 6, no. 1, 2002.

[17] B. M. Waxman, "Routing of multipoint connections," *IEEE Journal of Selected Areas in Communications*, pp. 1617–1622, 1988.

[18] M. E. J. Newman, "The structure and function of complex networks," *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[19] A. Medina, A. Lakhina, I. Matta, and J. Byers, "Brite: Universal topology generation from a user"s perspective," Boston University, Boston, MA, USA, Tech. Rep. BUCS-TR-2001-003, 2001.

[20] H. Tangmunarunkit, R. Govindan, S. Jamin, S. Shenker, and W. Willinger, "Network topology generators: degree-based vs. structural," in *SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, 2002, pp. 147–159.

[21] G. Seber, *The estimation of animal abundance and related parameters*. London, Great Britain: Griffin, 1982.