Estimating the Cardinality of a Mobile Peer-to-Peer Network

Shiping Chen, Yan Qiao, Shigang Chen, and Jianfeng Li

***Abstract*—Collecting information from mobile peer-to-peer (P2P) networks has important civilian and military applications. One problem is to determine the cardinality, i.e., the number of nodes, in a large mobile system. In a stationary wireless network, it can be trivially solved through a flooding-based query. However, the problem becomes much more challenging for mobile P2P networks whose topologies are constantly changing. In this paper, we present two novel statistical methods, called the circled random walk and the tokened random walk, to address this interesting problem. The circled random walk is simpler to implement and works well in networks of high mobility, whereas the tokened random walk works well with high or low mobility. These methods provide cardinality estimation by involving only a small subset of the nodes. They make tradeoff between overhead and estimation accuracy. The estimation error can be made arbitrarily small at the expense of larger overhead.**

***Index Terms*—Mobile P2P networks, random walk, cardinality estimation.**

I. Introduction

In mobile peer-to-peer networks, moving objects communicate using short-range wireless technologies such as IEEE 802.11 or Bluetooth [1], [2]. Nodes in these networks can become highly mobile. Constantly unstable network topologies may make it unsuitable to use some popular routing protocols such as DSDV [3], AODV [4] and DSR [5], which are designed for traditional mobile ad-hoc networks (whose topologies should be relatively stable enough for these protocols to work). In practice, peers may use smart phones, PDAs, or other advanced devices. If these devices are GPS-capable, geographical routing protocols [6], [7] can be used to route data over multiple hops.

Mobile P2P networks have many civilian and military applications. They are particularly useful when the communication infrastructure does not exist or cannot be accessed due to technical or economical reasons. For example, soldiers in a battlefield may carry small wireless devices and form a mobile P2P network among them. The future automobiles may be equipped with wireless devices that form vehicular networks to make our transportation systems more efficient. As vehicles pass by each other, data may be exchanged to inform road conditions ahead. Mobile networks not only serve as communication platforms but also provide other useful information. For example, the headquarters in a battlefield need to constantly monitor the remaining number of soldiers in each unit. The transportation department needs to know the information about traffic volume (i.e., number of moving cars[1]) in each district of a city. These applications rely on a basic function: estimating the cardinality of a mobile wireless network, i.e., the number of nodes in the network. Each node in a mobile P2P network only has knowledge about its immediate neighborhood. Therefore, cardinality estimation is a distributed process carried out among peer nodes.

This function can be easily implemented in a static wireless network where all nodes are stationary. A query node broadcasts a request message into the network. Each node transmits the message once when it receives the message for the first time. The node also remembers its predecessor, which is the one from which it receives the message for the first time. This simple broadcast protocol is not an efficient one, comparing with others [8]–[10]. But it quickly establishes a routing tree if each node treats its predecessor as the parent node. In this tree, each node learns the numbers of nodes in the subtrees rooted at its children, sum them up for the number of nodes in the subtree rooted at itself, and then reports that information to its parent. As this distributed computation is recursively carried out from the leaf nodes toward the root, the query node — which is located at the root of the tree — will learn the number of nodes in the whole network. However, any node failure will break the tree. To solve this problem, the synopsis diffusion approach [11] is proposed to collect information based on a more robust DAG (instead of tree) routing structure. Each node aggregates data from its upstream neighbors, integrates its own information, and broadcasts the aggregated information downstream. To ensure each node only transmits once, it must receive data from all upstream neighbors before its own transmission. This requires a static topology such as the type of sensor networks investigated in [11]. Other more efficient approaches have been invented to estimate the number of nodes in a stationary sensor network [12], [13] or estimate the number of tags in a large RFID system [14]–[18], assuming a centralized communication model where all tags directly communicate with a RFID reader.[2] Consequently, their methods cannot be applied to the multihop mobile network model in the context of this paper.

Manuscript received March 1, 2012; revised July 14, 2012.

S. Chen is with the Network Information Center & School of Optical-Electrical and Computer Eng., University of Shanghai for Science and Technology, Shanghai, 200093, China (e-mail: chensp@usst.edu.cn).

Y. Qiao and S. Chen are with the Department of Computer and Information Science and Engineering, University of Florida, Gainesville, FL, 32607, USA (e-mail: yqiao@cise.ufl.edu; sgchen@cise.ufl.edu).

J. Li is with the Business School, University of Shanghai for Science and Technology, Shanghai, 200093, China (e-mail: ljfwinner@163.com).

Digital Object Identifier 10.1109/JSAC.2013.SUP.0513032

[1] Parked cars do not contribute to traffic. They are mostly powered off and therefore do not participate in the network, either.

[2] Tags cannot communicate with each other to form a multihop wireless network.

In a mobile network, because nodes are moving, there is no stable routing structure for collecting information. Moreover, the number of nodes may change as new nodes join and existing nodes depart. This requires the operation of determining the number of nodes to be carried out frequently. On one hand, it is not efficient to rely on a flooding (or broadcasting) approach for such an operation due to high overhead. On the other hand, many applications may not require the exact count of the nodes, which varies over time anyway. An estimated number can already be very useful if the accuracy requirement is met. In the previous examples, the commander may not have to know the exact numbers of remaining soldiers in his units, and the transportation department may not have to know the exact number of vehicles that are present in a district. If an estimated number, say within $\pm 10\%$ of the exact number, meets their need, then we can adopt more efficient approaches that work well for mobile networks.

This paper proposes two statistical methods for estimating the number of nodes in a mobile P2P network. The first method is called the *circled random walk* (CRW). It estimates the number of nodes with an accuracy that is tunable. It makes tradeoff between the communication overhead and the estimation error. The error can be made arbitrarily small at the expense of increasing overhead. One problem of CRW is that it requires the nodes in the network to be randomly connected with one another through wireless links. This requirement may not be achievable in many realistic network settings. Our second method, called the *tokened random walk* (TRW), removes such a requirement. With the assistance of randomly distributed tokens, TRW provides a statistical estimation on the number of nodes in an arbitrarily connected wireless network. Our extensive simulations demonstrate that CRW works well in networks of high mobility, whereas TRW works well with high or low mobility.

The rest of the paper is organized as follows. Section II describes the network model and the problem to be solved. Section III presents our first method for estimating the number of nodes. Section IV presents our second method. Section V gives simulation results. Section VI draws the conclusion.

II. Network Model and Problem Statement

We consider a large mobile network system. Wireless links are established between nearby nodes that are within the communication range of each other. All links form a connected graph. Two nodes are neighbors of each other if there is a wireless link between them. Because the nodes move, the neighboring relationship changes over time.

The problem is to estimate the number n of nodes in a mobile P2P network, i.e., the cardinality of the network. Our goal is to develop statistical methods that estimate the value of n without flooding the network or requiring the participation of all nodes (for overhead reduction). Let $\hat{n}$ be the estimated number of nodes. By sampling only a subset of nodes, our methods ensure that the probability for n to fall in the range $[(1-\beta)\hat{n}, (1+\beta)\hat{n}]$ is at least α, where $\alpha(<1)$ and $\beta(<1)$ are the parameters of the accuracy requirement.

We assume that there exist end-to-end routing protocols [6], [7] that are able to route a message to a certain node.

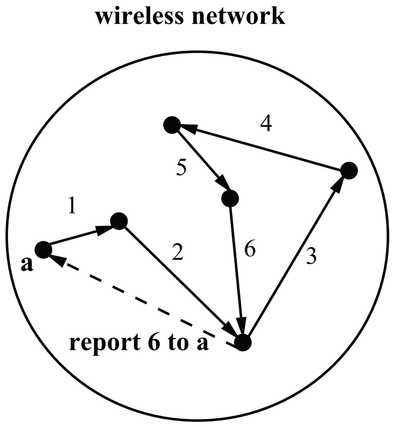


Fig. 1. Illustration of CRW.

In some scenarios, we *approximately* consider that the neighbors of a node are randomly selected from the network if the movement of a node allows it to have equal chance of being a neighbor of any other node. For example, when nodes move quickly and uniformly at random in a square area based on the waypoint model, they will all have chance to be each other's neighbors. In practice, rarely can this assumption be strictly met. Nevertheless, our simulation reveals that for highly dynamic networks where neighbors are frequently exchanged, even though this assumption is not accurate, our method based on this assumption still provides good cardinality estimation. We also observe that for static networks or networks of low mobility, the method based on the random neighbor assumption does not work well.

In this paper, we develop cardinality estimation methods for mobile networks. Our first method uses the random neighbor assumption. It is simpler and easy to implement. Removing the random neighbor assumption, our second method is able to work for network of high or low mobility. It is more robust but requires additional assistance in its execution. The preliminary results of this work were published in a conference paper [19].

III. Circled Random Walk (CRW)

In this section, we describe our first statistical method for estimating the number of nodes in a large mobile P2P network.

A. Description of the Method

A query node a sends out a number of probe messages. Each probe independently performs a *circled random walk* (CRW). The probe carries a globally unique identifier and a hop count. The identifier consists of node a's location and a sequence number. The hop count is initialized to be zero. When a node receives a probe, it records the probe's identifier and increases the hop count in the probe by one. If the node receives the probe for the first time, it forwards the probe to a randomly selected neighbor except for the one from which the probe was just received. If the node receives the probe for the second time, it discards the probe and sends the probe's hop count back to node a. The random walk of a probe terminates once its traversing path forms a circle. An illustration is given in Fig. 1.

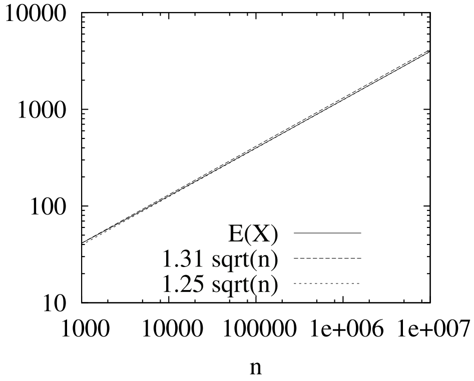


Fig. 2. $E(X)$ as a function of n.

Let X be the length of a circled random walk, which is the number of hops that the probe traverses before it reaches a node for the second time. X is a random number. In the next subsection, we establish the following mathematical formula that links $E(X)$ to n:

$$E(X) = f(n) = \sum_{i=3}^{n} i(\prod_{j=2}^{i-1}(1 - \frac{j-2}{n-2}))\frac{i-2}{n-2}, \qquad (1)$$
$$n = f^{-1}(E(X)).$$

Using the formula, we can estimate n after we measure $E(X)$ by performing a number of circled random walks and taking the average $\bar{X}$ of the received hop counts. Let $\hat{n}$ be the estimated value of n.

$$\hat{n} = f^{-1}(\bar{X}) \qquad (2)$$

The pseudo code of the algorithm we use to numerically compute $\hat{n}$ numerically from $\bar{X}$ based on (2) is given below. Note that $f(n)$ is a monotonically increasing function.

```
pick a small value $n_1$ and a large value $n_2$ such
that $f(n_1) < \bar{X}$ and $f(n_2) > \bar{X}$
while ($n_1 \neq n_2$) do
    let $n_3 = \lfloor (n_1 + n_2)/2 \rfloor$
    if $f(n_3) < \bar{X}$ then $n_1 = n_3$ else $n_2 = n_3$
end while
return $n_1$
```

B. Linking $E(X)$ to n

In this subsection, we derive Equation (1) and the variance of X. We also give a way for quick estimation of n. It is less accurate than what is computed by Equation (2) but is easier to calculate.

Let $q(i)$ be the probability of not visiting any node twice after a probe moves for i hops in its random walk. The node a that initializes the random walk is a visited node by default. It is obvious that $q(1) = q(2) = 100\%$. Consider the ith hop of the random walk, $\forall i > 2$. The previous $(i-1)$ hops visited i nodes, including node a. The next hop can be any node except for the current node and the previous node of the random walk. The probability for the ith hop to be an unvisited node is $1 - \frac{i-2}{n-2}$. Hence,

$$q(i) = q(i-1)(1 - \frac{i-2}{n-2}).$$

Here, we have used the random neighbor assumption in Section II, which is approximately true in highly-mobile networks. It does not hold well for low mobility as we will demonstrate through simulations. This assumption will be removed in the next section.

Recursively applying the above equation, we have

$$q(i) = q(1)\prod_{j=2}^{i}(1 - \frac{j-2}{n-2}) = \prod_{j=2}^{i}(1 - \frac{j-2}{n-2}).$$

X is a random variable with a distribution from 3 to n. We know that $q(i-1)$ is the probability of not visiting any node twice after the first $(i-1)$ hops, and $\frac{i-2}{n-2}$ is the conditional probability for the ith hop to reach a node that has been visited previously. Clearly, $q(i-1)\frac{i-2}{n-2}$ is the probability of visiting a node twice after i hops. Hence, the expected value of X is

$$E(X) = \sum_{i=3}^{n} iq(i-1)\frac{i-2}{n-2}$$
$$= \sum_{i=3}^{n} i(\prod_{j=2}^{i-1}(1 - \frac{j-2}{n-2}))\frac{i-2}{n-2}$$

The variance of X is

$$V(X) = \sum_{i=3}^{n}(i - E(X))^2(\prod_{j=2}^{i-1}(1 - \frac{j-2}{n-2}))\frac{i-2}{n-2}.$$

Fig. 2 shows $E(X)$ with respect to n in log scale. In the same plot, two additional curves, $1.31\sqrt{n}$ and $1.25\sqrt{n}$, are shown to closely overlap with $E(X)$. In fact, numerical computation shows that they are upper and lower bounds of $E(x)$ for a wide range of $n \in [10^3, 10^7]$. This indicates that $E(X) = O(\sqrt{n})$ in this range.

$$1.25\sqrt{n} < E(X) < 1.31\sqrt{n}$$
$$n < 0.64(E(X))^2 < 1.1n$$

Therefore, if the network size is in the range of $[10^3, 10^7]$, then $0.64\bar{X}^2$, which approximates $0.64(E(X))^2$, can be used as a quick but less accurate estimation of n. For accurate estimation, we use the estimator (2), which is based on (1). We analyze its accuracy below.

C. Determining the Number of Probes

The number m of probes initially sent from the query node controls the tradeoff between the communication overhead and the estimation accuracy. A wireless transmission is required when a node sends a probe to the next hop. We also know that $E(X) = O(\sqrt{n})$. Hence, the expected overhead of CRW in terms of the number of transmissions made by all nodes is $O(m\sqrt{n})$, which is linear in m.

Our goal is to determine the minimum number of probes that are needed to ensure that the probability for n to fall in the range $[(1-\beta)\hat{n}, (1+\beta)\hat{n}]$ is at least α, where α and β are the parameters of the accuracy requirement. In other words, the estimation has to achieve an α confidence interval that is bounded by $[(1-\beta)\hat{n}, (1+\beta)\hat{n}]$.

First, we prove that $\bar{X}$ is an unbiased estimate of $E(X)$, which is the mean hop count of a circled random walk. Let

X_i be the hop count of the circled random walk by the i^{th} probe. As $\bar{X}$ is the average hop counts of probes, we have,

$$\begin{aligned} \bar{X} &= \frac{1}{m}\sum_{i=1}^{m} X_i \\ E(\bar{X}) &= \frac{1}{m}E(\sum_{i=1}^{m} X_i) = \frac{1}{m}\sum_{i=1}^{m} E(X_i) \end{aligned} \tag{3}$$

Each X_i is an independent random sample of X. Therefore $E(X_i) = E(X)$. From (3), we have $E(\bar{X}) = E(X)$.

Next, we determine the value of m such that the probability for $E(X)$ to fall in the range $[(1-\beta')\bar{X}, (1+\beta')\bar{X}]$ is at least α, where β' is a constant whose value will be determined shortly. The α confidence interval of $E(X)$ is $\bar{X} \pm st^*/\sqrt{m}$, where s is the standard deviation calculated from the received hop counts and t^* is the upper $\frac{1+\alpha}{2}$ point of the t distribution with $(m-1)$ degree of freedom. The value of m is calculated as follows:

$$\begin{aligned} st^*/\sqrt{m} &= \beta'\bar{X} \\ m &= \frac{(st^*)^2}{(\beta'\bar{X})^2} \end{aligned} \tag{4}$$

Then, we determine an appropriate value for β'. We know that if m is chosen based on (4), then the following inequality is satisfied with probability α:

$$\begin{gathered} (1-\beta')\bar{X} \le E(X) \le (1+\beta')\bar{X} \\ f^{-1}((1-\beta')\bar{X}) \le f^{-1}(E(X)) \le f^{-1}((1+\beta')\bar{X}) \\ f^{-1}((1-\beta')\bar{X}) \le n \le f^{-1}((1+\beta')\bar{X}) \end{gathered}$$

Our estimation accuracy requirement is to satisfy $(1-\beta)\hat{n} \le n \le (1+\beta)\hat{n}$ with probability α. To meet this requirement, we select the maximum value of β' that meets the following conditions:

$$\begin{aligned} f^{-1}((1+\beta')\bar{X}) &\le (1+\beta)\hat{n} \\ f^{-1}((1-\beta')\bar{X}) &\ge (1-\beta)\hat{n}, \end{aligned} \tag{5}$$

where β is a constant specified in the accuracy requirement, $\bar{X}$ and $\hat{n}$ are results of the CRW execution, and we know how to solve f^{-1} based on the algorithm in Section III-A. Hence, β' can be computed numerically from the above inequalities.

Finally, based on (4) and (5), we design an iterative process for determining the number of probes that is required to meet a given accuracy requirement specified by α and β. The query node a begins with a certain number (e.g., 50) of probes. After it receives the hop counts of the probes and computes $\bar{X}$ and $\hat{n}$, node a determines β' from (5) and then the value of m from (4). If the total number m' of probes that have already been sent is equal to or greater than m, it returns the current value of $\hat{n}$ as the estimation of n. Otherwise, if m' is less than m, node a send $(m-m')$ more probes and use the returned hop counts to refine the value of m. This process continues until the total number of probes that have been sent is equal to or greater than m.

IV. Tokened Random Walk (TRW)

In this section, we describe our second statistical method for estimating the number of nodes in a large mobile P2P network.

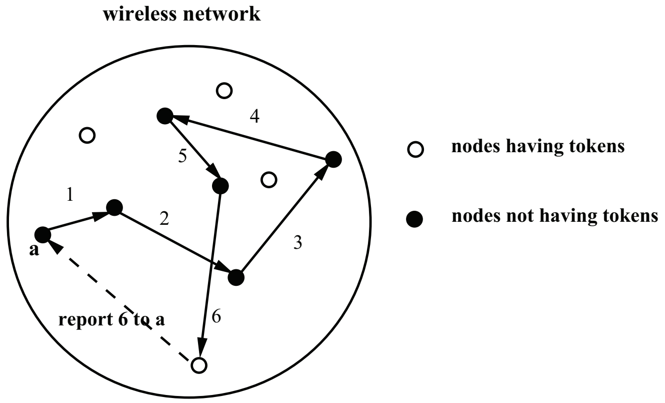


Fig. 3. Illustration of TRW.

A. Description of the Method

We randomly distribute a number T of tokens to nodes in the network. The problem of how to randomly distribute tokens will be discussed in Section IV-C. The nodes that hold tokens are called the *tokened nodes*.

A query node a sends out a number of probe messages. Each probe independently performs a tokened random walk (TRW) and counts the number of hops that it has traversed. When a node that does not hold a token receives a probe, it increases the hop count in the probe by one and forwards the probe to one of its neighbors. When a tokened node receives a probe, it discards the probe and sends the probe's hop count back to node a. An illustration of TRW is shown in Fig. 3.[3]

Let Y be the length of a tokened random walk, which is the number of hops that a probe traverses before it reaches a tokened node. In the next subsection, we establish a mathematical formula that links $E(Y)$ to n:

$$\begin{aligned} E(Y) = g(n) = &\sum_{i=1}^{n-T+1} i\frac{(n-i+1)!(n-T)!}{n!(n-i-T+1)!}\frac{T}{n-i+1}, \\ &n = g^{-1}(E(Y)). \end{aligned} \tag{6}$$

Using the formula, we can estimate n after we measure $E(Y)$ by performing a number of tokened random walks and taking the average $\bar{Y}$ of the received hop counts. The estimated number $\hat{n}$ of nodes is calculated as

$$\hat{n} = g^{-1}(\bar{Y}). \tag{7}$$

B. Linking $E(Y)$ to n

Let $p(i)$ be the probability of not reaching a tokened node after a probe has visited i nodes. This happens when and only when the set of T tokened nodes does not overlap with the set of i visited nodes. Recall that the tokened nodes are randomly chosen from the network. $\binom{T}{n}$ is the number of different ways for picking the set of tokened nodes. $\binom{T}{n-i}$ is the number of different ways for picking them from the nodes that are not

[3] In practice, we may allow the probes to make some random walks before starting to count the hops. Our simulation shows that it increases the randomness and returns better estimation results.

visited. Hence,

$$p(i) = \frac{\binom{T}{n-i}}{\binom{T}{n}} = \frac{(n-i)!(n-T)!}{n!(n-i-T)!}$$

Y is a random variable with a distribution from 1 to $n - T + 1$. We know that $p(i-1)$ is the probability of not reaching a tokened node after the first $(i-1)$ hops. The conditional probability for the ith visited node to have a token is $\frac{T}{n-2}$. Clearly, $p(i-1) \cdot \frac{T}{n-2}$ is the probability of reaching the first tokened node at the ith hop. The expected value of Y is

$$\begin{aligned} E(Y) &= \sum_{i=1}^{n-T+1} i \cdot p(i-1) \cdot \frac{T}{n-2} \\ &= \sum_{i=1}^{n-T+1} i \frac{(n-i+1)!(n-T)!}{n!(n-i-T+1)!} \frac{T}{n-2}. \end{aligned}$$

The variance of Y is

$$V(Y) = \sum_{i=1}^{n-T+1} (i - E(Y))^2 \frac{(n-i+1)!(n-T)!}{n!(n-i-T+1)!} \frac{T}{n-2}.$$

C. Random Token Distribution

The process of deriving (6) in the previous subsection does not require the assumption of a randomly-connected network. Instead, it requires that the tokens are randomly distributed. Essentially, we shift the random topology requirement of CRW to a random token distribution requirement, which can be implemented by the following distributed algorithm: Every token independently moves around in the network. When a node receives a token, it holds the token for a period that is inversely proportional to its current number of neighbors, i.e., the *node's degree*. After that period, it forwards the token to a neighbor selected uniformly at random. The transmission of a token may be piggybacked in the periodic hello exchange between the neighbors. In order to support communications in a mobile network, each wireless node has to periodically broadcast a hello packet (or called a beacon), which allows the node to learn its new neighbors. One bit in the hello packet can be used to encode a token transmission. '0' means there is no token carried in the hello packet, and '1' means there is.

We prove that, in the steady state, the rate at which a node receives tokens is proportional to the node's degree. Because the holding time after each receipt of a token is inversely proportional to the node's degree, the aggregate holding time at every node is about the same, which ensures the uniform random distribution of tokens.

Consider an arbitrary token. The movement of the token in the network is modeled as a discrete-time finite-state Markov chain. The current state is i if node i holds the token. The set of states is $\{1, ..., n\}$ for the n nodes. Let N_i be the set of neighbors of node i. Let $n_i = |N_i|$. The transition probability from state i to state j is

$$p_{ij} = \begin{cases} \frac{1}{n_i} & \text{if } j \in N_i \\ 0 & \text{if } j \notin N_i \end{cases}$$

The matrix of transition probabilities is $P = (p_{ij}, i, j \in [1, n])$. Let $\pi = (\pi_1, \pi_2, ..., \pi_n)$ be the stationary distribution of the Markov chain, which satisfies $\pi P = \pi$ and $\sum_{i=1}^{n} \pi_i = 1$, where π_i represents how often node i has (or receives) the token in the steady state of the stochastic token movement process. $\pi P = \pi$ can be rewritten as

$$\sum_{j \in N_i} \frac{\pi_j}{n_j} = \pi_i, \quad i \in [1, n].$$

It can be easily verified that the solution is

$$\pi_i = \frac{n_i}{\sum_{j=1}^{n} n_j}, \quad i \in [1, n].$$

Therefore, the rate at which a node receives a particular token is proportional to its degree. Because all tokens move independently, the rate at which a node receives tokens is also proportional to its degree. Our goal is for each node to have an equal chance of being a tokened node. To compensate the rate difference, when a node receives a token, the holding time should be inversely proportional to the degree. In particular, we may randomly pick a holding time from an exponential distribution whose mean is inversely proportional to the degree. If a node's degree changes during this period, we use the degree when it first receives the token.

In a mobile environment, a node may depart from the network or be powered down. To support the TRW protocol, we require that if a node has a token, it must transmit the token to a neighbor before it departs or is powered down. There may be rare cases where a node crashes to cause a permanent token loss. One solution is to release a new set of tokens periodically. Each release is identified by a sequence number s. The sequence number is initialized to be one and increased by one for every subsequent release. Each token must be associated with the sequence number that identifies which release it belongs to. When a node transmits a token to a neighbor piggybacked in a hello packet, instead of using one bit, the packet carries a sequence number. If the number is zero, it means no token; otherwise, it means a token of a certain release.

Each query is made with a sequence number, and each probe of the query carries that sequence number. When a node receives a probe, only if it has a token of the same sequence number, it will discard the probe and send the probe's hop count back to the query node.

Let D be the period between two consecutive token releases. Suppose D is chosen to be sufficiently large, such that tokens will be randomly distributed after they are released for a time period of D. Now, after tokens of sequence number s are released and before tokens of sequence number $s + 1$ are released, all queries can be made with sequence number $s - 1$. Moreover, all previous tokens (with sequence numbers $s - 2$ or smaller) are no longer useful. To remove outdated tokens, when a node receives a token with a new sequence number s, it knows that it should remove any token with sequence number $s - 2$ or less that it may receive in the future. Hence, as tokens with a new sequence number start to travel in the network, nodes begin to remove old, useless tokens.

D. Determining the Number of Probes

We now determine the minimum number of probes that are needed to ensure that the probability for n to fall in the range $[(1-\beta)\hat{n}, (1+\beta)\hat{n}]$ is at least α.

First, we prove that $\bar{Y}$ is an unbiased estimate of $E(Y)$, which is the mean hop count of a tokened random walk. Let Y_i be the hop count of the tokened random walk by the i^{th} probe. As $\bar{Y}$ is the average hop count of the probes, we have,

$$\begin{aligned} \bar{Y} &= \frac{1}{m}\sum_{i=1}^{m} Y_i \\ E(\bar{Y}) &= \frac{1}{m}E(\sum_{i=1}^{m} Y_i) = \frac{1}{m}\sum_{i=1}^{m} E(Y_i) \end{aligned} \tag{8}$$

Each Y_i is an independent random sample of Y. Therefore $E(Y_i) = E(Y)$. From (8), we have $E(\bar{Y}) = E(Y)$.

Next, based on the same process as Section III-C, it can be shown that if the number of probes is

$$m = \frac{(st^*)^2}{(\beta'\bar{Y})^2}, \tag{9}$$

then the probability for $E(Y)$ to fall in $[(1-\beta')\bar{Y}, (1+\beta')\bar{Y}]$ is at least α, where β' is a given value, s is the standard deviation calculated from the received hop counts, and t^* is the upper $\frac{1+\alpha}{2}$ point of the t distribution with $(m-1)$ degree of freedom. We have the following inequalities:

$$\begin{gathered} (1-\beta')\bar{Y} \leq E(Y) \leq (1+\beta')\bar{Y} \\ g^{-1}((1-\beta')\bar{Y}) \leq g^{-1}(E(Y)) \leq g^{-1}((1+\beta')\bar{Y}) \\ g^{-1}((1-\beta')\bar{Y}) \leq n \leq g^{-1}((1+\beta')\bar{Y}). \end{gathered}$$

The accuracy requirement is $(1-\beta)\hat{n} \leq n \leq (1+\beta)\hat{n}$ with probability α. To meet this requirement, we select the maximum value of β' that meets the following conditions:

$$\begin{aligned} g^{-1}((1+\beta')\bar{Y} &\leq (1+\beta)\hat{n} \\ g^{-1}((1-\beta')\bar{Y} &\geq (1-\beta)\hat{n}. \end{aligned} \tag{10}$$

Given an accuracy requirement specified by α and β, the query node a begins with a certain number of probes. After it receives the hop counts of the probes, node a determines β' from (10) and then the value of m from (6). If the total number m' of probes that have already been sent is equal to or greater than m, it returns the current value of $\hat{n}$ as the estimation of n. Otherwise, if m' is less than m, node a send $(m - m')$ more probes and use the returned hop counts to refine the value of m. This process continues until the total number of probes that have been sent is equal to or greater than m.

V. Simulation

In this section, we use simulations to evaluate the performance of our cardinality estimation methods in terms of accuracy and overhead. Our simulations are performed based on two models: (1) random waypoint model [5], [20]–[22], which has been extensively adopted in other research work on mobile systems; (2) street model, which captures an application scenario of estimating the number of moving vehicles in the streets of a city district.

A. Simulation Setup for Random Waypoint Model

Consider a square area of 1000×1000 units of length. The number n of wireless nodes ranges from 1,000 to 5,000. The transmission range of the nodes is 100 units of length. Each node is capable of forwarding messages to any other node in its transmission range. We use the random waypoint (RWP) model to simulate the mobility of nodes: Each node randomly selects a position within the area as destination and a velocity in the range of $[5, 30]$. Then it moves towards the destination at the selected speed. After it arrives at the destination, it moves again with a new destination and a new velocity.

We use CRW and TRW to estimate the number n of nodes in the area. Both methods send probe messages to the network. For CRW, when a node receives a probe message, it holds the message for 10 units of time before forwarding it. This holding period allows each node to have chance to change neighbors. TRW does not need the random-neighbor assumption, and therefore the holding period for probe messages is not necessary. For TRW, if a node receives a token, it keeps the token for 10 to 100 units of time before passing it to another node. The actual time that it holds the token is inversely proportional to the number of its neighbors. We set the number of tokens in the network to be 100. In order to increase randomness, we let probe messages to make some random walks before starting to count the hops. Assume each node knows its geographic location (possibly through GPS). A node also knows the geographic locations of its neighbors, which are piggybacked in the hello exchanges. We implement geographic routing [7] for the hop count result to be sent back to the query node whenever a probe message completes its random walk. The location of the query node is carried by the probe message.

B. Estimation Accuracy

We first present the estimation accuracy of the two methods. We vary the number n of wireless nodes from 1000 to 5000. We run the simulation under three different accuracy requirements: $\alpha = 95\%, \beta = 20\%$; $\alpha = 99\%, \beta = 20\%$; and $\alpha = 95\%, \beta = 10\%$.

Figs. 4 and 5 compare the actual number of nodes, n, and the estimated number, $\hat{n}$. Each point in the figures represents one simulation result; the x coordinate of the point is the actual number n of nodes, and the y coordinate is the estimated number $\hat{n}$. An estimation is more accurate if the point is closer to the equality line, $y = x$. In the figures, points surround the equality line, indicating good estimation performance; the tighter the accuracy requirement is, the closer the points are to the equality line. (Better accuracy comes with a price of higher overhead, which will be shown in the next subsection.)

In the leftmost plot, the requirement specified by α and β is that the estimated number $\hat{n}$ should have a 95% chance to fall within ± 20% of the actual number n. Our statistical measurement based on the data points in the figure shows that this requirement is met. The same is true for other plots in Figs. 4 and 5 with different combinations of α and β values.

C. Estimation Overhead

Next, we investigate the overhead of the two methods. For CRW, the overhead is caused by probe message transmissions.

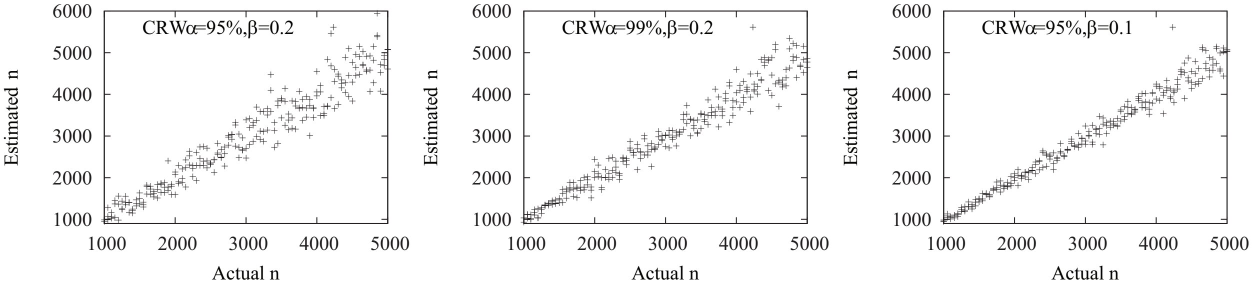


Fig. 4. CRW's estimation accuracy under the random waypoint model with nodal moving velocity in range of $[5, 30]$.

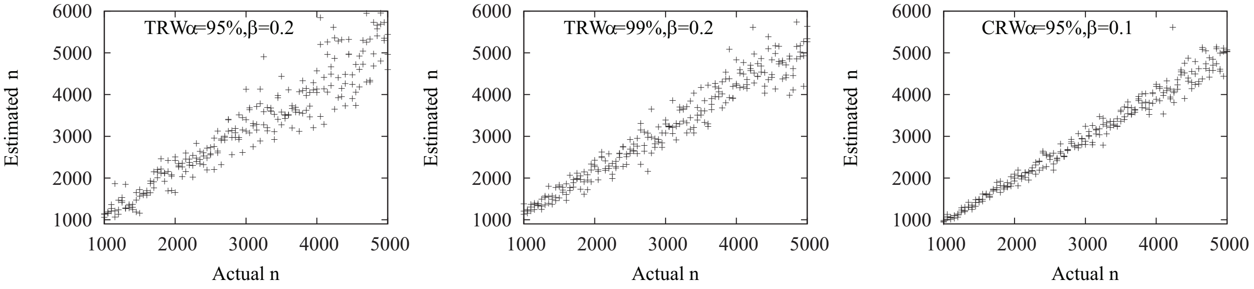


Fig. 5. TRW's estimation accuracy under the random waypoint model with nodal moving velocity in range of $[5, 30]$.

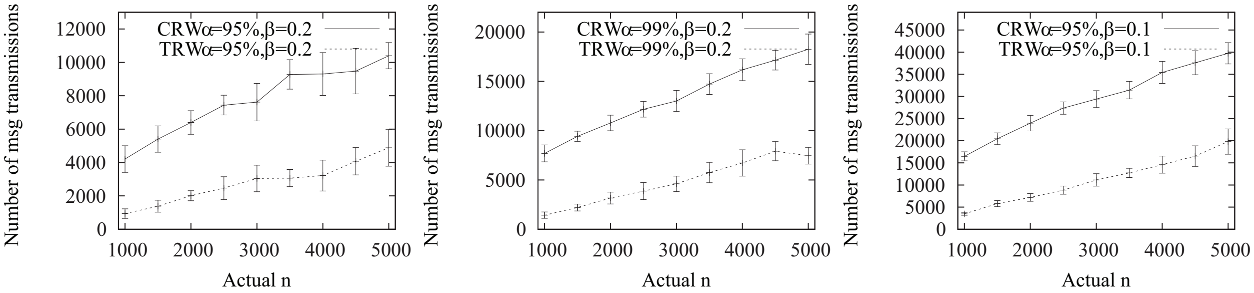


Fig. 6. Number of message transmissions for each estimation, under the random waypoint model with nodal moving velocity in range of $[5, 30]$.

For TRW, the overhead comes from both probe transmissions and token transmissions. A probe transmission is a separate packet with headers at MAC, network, and application (CRW/TRW) layers, as well as physical-layer overhead. A token transmission is piggybacked in the hello exchanges, which incurs far smaller overhead than a probe. Hence, we only count the probe message transmissions in overhead comparison.

Fig. 6 compares CRW and TRW in terms of the total number of message transmissions. For each data point, we make 100 simulation runs, average the overhead results, and present the mean overhead and its standard deviation. From the figure, we observe that TRW is much more efficient, only requiring 20% to 30% of the overhead incurred by CRW.

D. *Performance under Low Mobility*

As we have shown in the previous simulations, CRW produces good estimation when the nodes move relatively fast. Even though the random neighbor assumption is not accurate, it approximates well for the highly mobile scenarios. However, our next set of simulations show that CRW does not work well for networks of low mobility. On the contrary, TRW remains accurate for those networks.

The simulation setup is the same as described in Section V-A except that the velocity of each node is randomly selected from $[3, 10]$, instead of $[5, 30]$, when the node moves from one location to another. Fig. 7 shows that CRW consistently under-estimates the number of nodes in a network of low mobility. That is because nodes have relatively stable neighbors and as they forward probe messages, the messages tend to make short cycles within small neighborhood, which leads to the under-estimation.

Fig. 8 shows that TRW remains accurate for networks of low mobility because its design does not rely on the random neighbor assumption.

Fig. 9 compares the overhead of the two methods. Again, TRW outperforms CRW significantly.

E. *Simulation Setup for Street Model*

Suppose future cars are equipped with wireless devices that not only support user applications but also assist transportation management functions such as information gathering. One such function may be estimating the number of moving vehicles. When a vehicle is powered down, we can consider it to be in parking status (in most cases). When a vehicle is

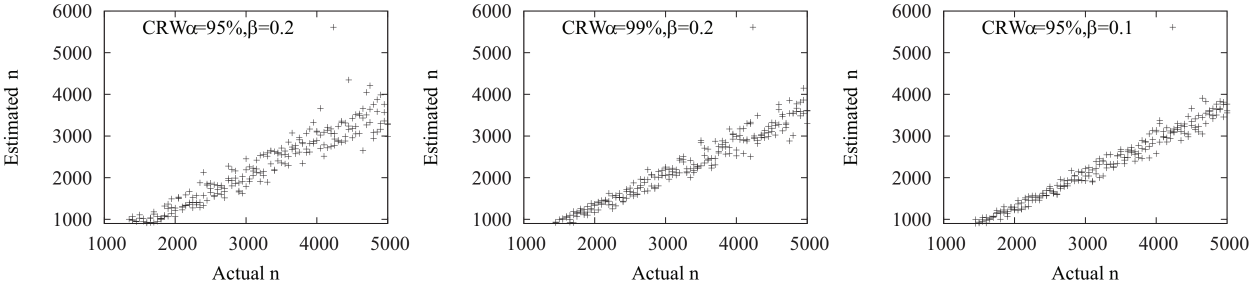


Fig. 7. CRW's estimation accuracy under the random waypoint model with nodal moving velocity in range of $[3, 10]$.

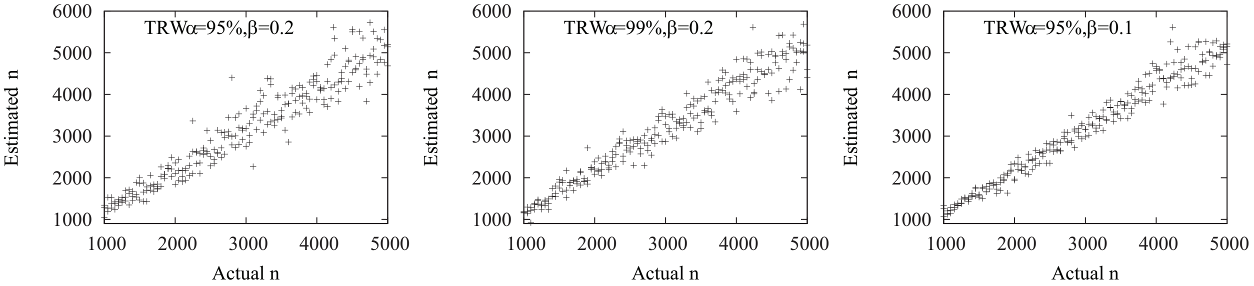


Fig. 8. TRW's estimation accuracy under the random waypoint model with nodal moving velocity in range of $[3, 10]$.

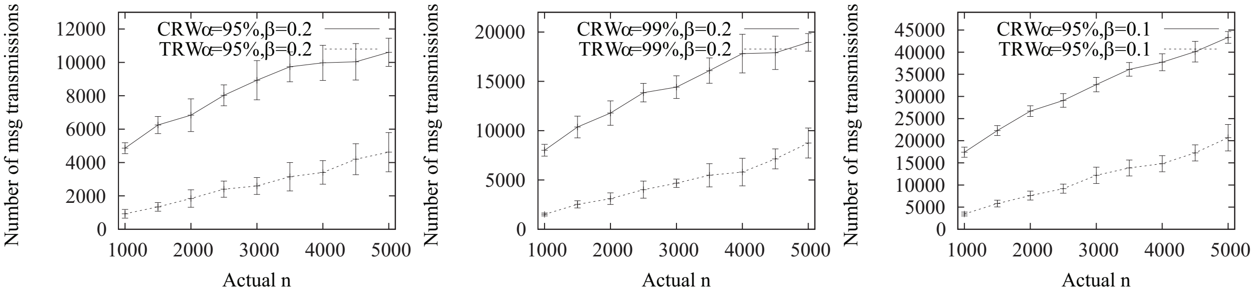


Fig. 9. Number of message transmissions for each estimation, under the random waypoint model with nodal moving velocity in range of $[3, 10]$.

powered up with its wireless device running, we can consider it be to in moving status (in most cases).

Consider a square area of 1000 meters by 1000 meters. Let the time unit be a second. Suppose the streets and avenues are arranged in a grid pattern and the distance between two adjacent streets (or avenues) is 100 meters. The number n of moving vehicles ranges from 1,000 to 5,000. The transmission range of a vehicle is 100 meters. Each vehicle is capable of forwarding messages to any other vehicle in its transmission range. Each moving vehicle selects an arbitrary intersection and a velocity in the range of $[20, 40]$ miles/hour, i.e., $[32, 64]$ kilometers/hour. Then it moves first along a street and then along an avenue or vice versa towards the destination at the selected speed. For simplicity, we do not implement variable speed and stops at signal lights. Other parameters are similar to those in Section V-A.

F. Accuracy and Overhead under Street Model

Fig. 10 presents the estimations made by CRW. The vertical axis is the estimated number of vehicles, and the horizontal axis is the true number of vehicles. In the leftmost plot, we let $\alpha = 95\%$ and $\beta = 0.2$, which require that 95% of estimated numbers should fall within ± 20% of the true numbers. In the middle plot, $\alpha = 99\%$ and $\beta = 0.2$. In the rightmost plot, $\alpha = 95\%$ and $\beta = 0.1$. Our statistical measurement based on the data points in the figure confirms that the accuracy requirement is indeed met. Fig. 11 presents the estimations made by TRW. Again, the accuracy requirement is met.

Fig. 12 compares CRW and TRW in terms of the total number of message transmissions. TRW outperforms CRW. Its communication overhead is only a fraction of CRW's overhead.

For a circled random walk, the time for a probe to travel each hop has two components: the holding period and the transmission time from one node to the next. When comparing the holding time (10 seconds in this simulation), the transmission time is negligible. Hence, our simulation only considers the holding time. The average completion time of CRW (including all its probes) is shown in Table I. Because of the long holding period, the time for CRW is significant. However, it is probably acceptable in practice to spend six to fifteen minutes for the task of measuring the number of vehicles in a city distinct. Because TRW does not need any holding period, its completion is much quicker. When $n = 1000$ and the number of tokens is 100, each tokened random walk has an average of just 10 hops, and only the

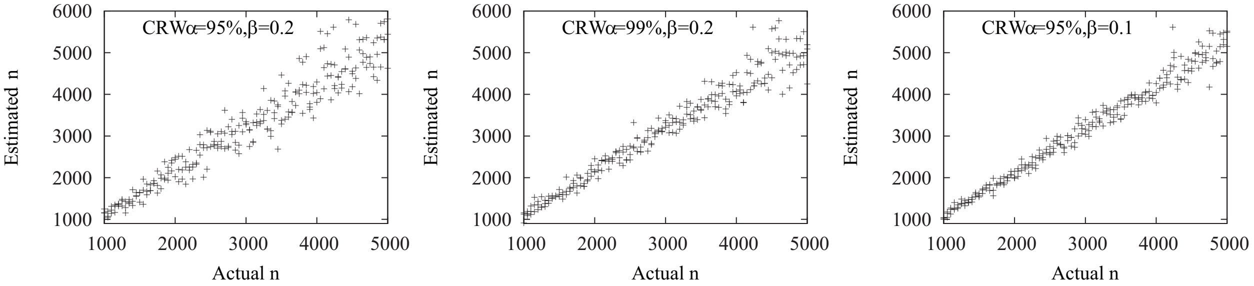


Fig. 10. CRW's estimation accuracy under the street model.

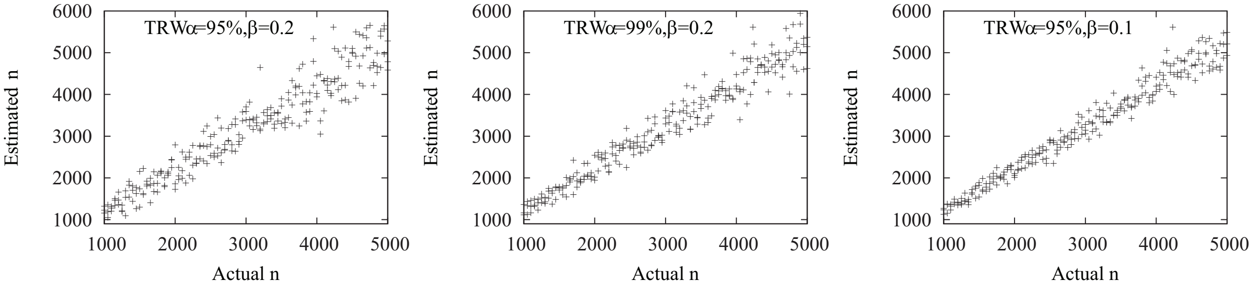


Fig. 11. TRW's estimation accuracy under the street model

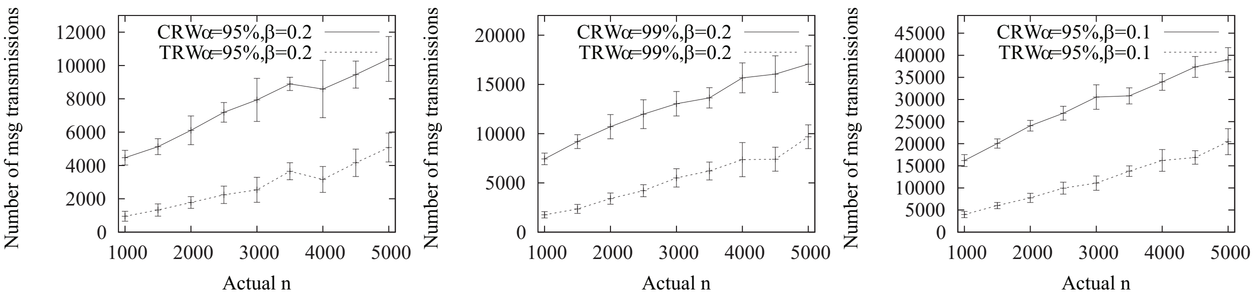


Fig. 12. Number of message transmissions for each estimation, under the street model.

transmission time is needed.

Overall, our simulation results are consistent across the random waypoint model and the street model. In summary, TRW is a better method in terms of communication overhead. It is also better in terms of estimation accuracy in low-mobility networks. CRW is able to produce results that meet the accuracy requirement in high-mobility networks. Its advantage is simplicity (without the assistance of tokens).

VI. Conclusion

This paper investigates the problem of how to estimate the number of nodes in a large mobile P2P network. We propose two statistical methods, called the circled random walk and the tokened random walk, respectively. The proposed methods are adjustable for communication overhead and estimation accuracy. Their estimation process involves only a subset of the nodes, and the estimation errors can be made arbitrarily small. While the circled random walk method requires a randomized neighboring relationship among the nodes and it works well in high-mobility networks, the tokened random walk is more practical for low-mobility networks because it removes such a requirement through a randomized token distribution process.

VII. Acknowledgements

This work was supported in part by the US National Science Foundation under grants CPS-0931969 and CNS-1115548, the National Natural Science Foundation of China under grant 61170277, the Innovation Program of the Shanghai Municipal Education Commission (Program Number 12zz137), and the first-class discipline construction project of Shanghai under grant S1201YLXK.

References

[1] B. Xu and O. Wolfson, "Data management in mobile peer-to-peer networks," in *Proc. Int. Workshop Databases, Inf. Syst. Peer-to-Peer Comput. (DBISP2P)*, 2004.

[2] O. Wolfson, B. Xu, H. Yin, and H. Cao, "Search-and-discover in mobile P2P network databases," in *Proc. IEEE ICDCS*, 2006.

[3] C. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance vector routing (DSDV) for mobile computers," in *Proc. ACM SIGCOMM*, Oct. 1994.

[4] C. E. Perkins and E. M. Royer, "Ad hoc on-demand distance vector routing," in *Proc. 2nd IEEE Workshop Mobile Comput. Syst. Applicat.*, Feb. 1999.

[5] D. Johnson and D. Maltz, "Dynamic source routing in ad Hoc wireless networks," *Mobile Comput.*, pp. 153–181, 1996.

[6] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, "Routing with guaranteed delivery in ad hoc wireless networks," in *Proc. 3rd Int. Workshop Discrete Algorithms Methods Mobile Comput. Commun. (DialM)*, Aug. 1999.

TABLE I
AVERAGE COMPLETION TIME OF CRW

n	1000	2000	3000	4000	5000
CRW Time	6 min 41 sec	9 min 18 sec	11 min 35 sec	13 min 32 sec	14 min 59 sec

[7] B. Karp and H. Kung, “GPSR: Greedy perimeter stateless routing for wireless networks,” in *Proc. ACM MobiCom*, Aug. 2000.

[8] W. Liang, “Constructing minimum-energy broadcast trees in wireless ad hoc networks,” in *Proc. ACM MobiHoc*, June 2002.

[9] S. C. H. Huang, P. J. Wan, X. Jia, H. Du, and W. Shang, “Minimum-latency broadcast scheduling in wireless ad hoc networks,” in *Proc. IEEE INFOCOM*, 2007.

[10] R. Mahiourian, F. Chen, R. Tiwari, M. T. Thai, H. Zhai, and Y. Fang, “An approximation algorithm for conflict-aware broadcast scheduling in wireless ad hoc networks,” in *Proc. ACM MobiHoc*, May 2008.

[11] S. Nath, P. B. Gibbons, S. Seshan, and Z. R. Anderson, “Synopsis diffusion for robust aggregation in sensor networks,” in *Proc. ACM SenSys*, 2004.

[12] A. Leshem and L. Tong, “Estimating sensor population via probabilistic sequential polling,” *IEEE Signal Process. Lett.*, vol. 12, no. 5, pp. 395–398, May 2005.

[13] C. Budianu, S. David, and L. Tong, “Estimation of the number of operating sensors in large-scale sensor networks with mobile access,” *IEEE Trans. Signal Process.*, vol. 54, no. 5, pp. 1703–1715, May 2006.

[14] Y. Zheng, M. Li, and C. Qian, “PET: Probabilistic estimating tree for large-scale RFID estimation,” in *Proc. IEEE ICDCS*, 2011.

[15] H. Han, B. Sheng, C. Tan, Q. Li, W. Mao, and S. Lu, “Counting RFID tags efficiently and anonymously,” in *Proc. IEEE INFOCOM*, 2010.

[16] M. Kodialam and T. Nandagopal, “Fast and reliable estimation schemes in RFID systems,” in *Proc. ACM MOBICOM*, 2006.

[17] M. Kodialam, T. Nandagopal, and W. Lau, “Anonymous tracking using RFID tags,” in *Proc. IEEE INFOCOM*, 2007.

[18] C. Qian, H. Ngan, and Y. Liu, “Cardinality estimation for large-scale RFID systems,” in *Proc. IEEE PerCom*, 2008.

[19] S. Chen, “Estimating the number of nodes in a mobile wireless network,” in *Proc. IEEE Globecom*, 2010.

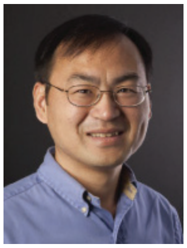
[20] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva, “Multi-hop wireless ad hoc network routing protocols,” in *Proc. ACM Mobicom*, 1998.

[21] W. Navidi and T. Camp, “Stationary distributions for the random waypoint mobility model,” *IEEE Trans. Mobile Comput.*, vol. 3, no. 1, pp. 99–108, 2004.

[22] E. Hyytia, P. Lassila, and J. Virtamo, “Spatial node distribution of the random waypoint mobility model with applications,” *IEEE Trans. Mobile Comput.*, vol. 5, no. 6, pp. 680–694, 2006.

Shiping Chen received the B.S. degree in electrical engineering from JiangXi University of China in 1984. He received the M.S. and Ph.D. degrees in computer science from the Institute of Computing Technology of the Chinese Academy of Sciences and Fudan University in 1990 and 2006, respectively. He joined the University of Shanghai for Science and Technology in 1990 and is currently a full professor in the School of Optical-Electrical and Computer Engineering. He is also the director of the network center of the university. His research interests include peer-to-peer networks, network communications, and database systems.

Yan Qiao received her B.S. degree in computer science and technology from Shanghai Jiao Tong University, China, in 2009. She is currently a Ph.D. student at the University of Florida (as of 2012) and her advisor is Dr. Shigang Chen. Her research interests include network measurement, algorithms, and RFID protocols.



Shigang Chen is an associate professor with the Department of Computer and Information Science and Engineering at the University of Florida. He received his B.S. degree in computer science from the University of Science and Technology of China in 1993. He received M.S. and Ph.D. degrees in computer science from the University of Illinois at Urbana-Champaign in 1996 and 1999, respectively. After graduation, he worked with Cisco Systems for 3 years before joining the University of Florida in 2002. He served on the technical advisory board for Protego Networks from 2002–2003. His research interests include computer networks, Internet security, wireless communications, and distributed computing. He has published more than 100 peer-reviewed papers, with about 5,000 citations. He received the IEEE Communications Society Best Tutorial Paper Award in 1999 and an NSF CAREER Award in 2007. He holds 11 US patents. He is an associate editor for the IEEE TRANSACTIONS ON NETWORKING, the *Elsevier Journal of Computer Networks*, and the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY. He has also served as guest editor for the *Journal of Advances in Multimedia* in 2007, for the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY in 2005, and for the *ACM/Baltzer Journal of Wireless Networks* (WINET) in 2005. He has served on the steering committee of IEEE IWQoS since 2010. He was a TPC co-chair for IWQoS 2009, a TPC co-chair for the Computer and Network Security Symposium of ACM IWCMC 2009, and an area co-chair for the Network and Data Communications Track of the 10th International Symposium on Pervasive Systems, Algorithms and Networks (I-SPAN) in 2009. He served as TPC co-chair for the Computer and Network Security Symposium of IEEE IWCCC 2006, vice TPC chair for IEEE MASS 2005, vice general chair for QShine 2005, and TPC co-chair for QShine 2004.

Jianfeng Li received the M.S. degree in management science and engineering from Haerbin University of Commerce in 2006. He is currently a Ph.D. student in the Business School at the University of Shanghai for Science and Technology. He is also a lecturer in the College of Economics and Management at China Jiliang University. His research interests include peer-to-peer networks, cloud computing, and supply chain management.