Census: Fast, scalable and robust data aggregation in MANETs

V. Kulathumani*[1], A. Arora†[2], M. Sridharan‡[2], K. Parker§[2], and M. Nakagawa¶[1]

[1]Department of Computer Science, West Virginia University, Morgantown WV 26505
[2]The Samraksh Company, Dublin OH 43017

December 29, 2016

Abstract

This paper describes *Census*, a protocol for data aggregation and statistical counting in MANETs. *Census* operates by circulating a set of tokens in the network using *biased random walks* such that each node is visited by at least one token. The protocol is structure-free so as to avoid high messaging overhead for maintaining structure in the presence of node mobility. It biases the random walks of tokens so as to achieve fast cover time; the bias involves short albeit multi-hop gradients that guide the tokens towards hitherto unvisited nodes. *Census* thus achieves a cover time of $O(N)$ and message overhead of $O(Nlog(N))$ where N is the number of nodes. Notably, it enjoys scalability and robustness, which we demonstrate via simulations in networks ranging from 100 to 4000 nodes under different network densities and mobility models. We also observe a speedup by a factor of k when k different tokens are used ($1 \leq k \leq \sqrt{N}$).

Keywords: random walk, statistical aggregation, gossip, local gradients

1 Introduction

This paper presents *Census*, a fast, scalable and robust protocol for data aggregation in MANETs. *Census* is a structure-free protocol that relies on biased random walks to achieve aggregation. The protocol operates

*vinod.kulathumani@mail.wvu.edu
†anish.arora@samraksh.com
‡mukundan.sridharan@samraksh.com
§kenneth.parker@samraksh.com
¶manakagawa@mix.wvu.edu

The email point of contact is vinod.kulathumani@mail.wvu.edu. This work was supported in part by Defense Advanced Research Projects Agency (DARPA) contract FA8750-12-C-0278. The views, opinions, and/or findings contained in this paper are those of the authors and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

by circulating a set of one or more tokens using biased random walks, in such a way that every node in the network is visited by at least one token. We say that a node is *visited* by a token when the node gets exclusive access to the token; the visitation period can be used by the node to add node-specific information into the token, resulting in data aggregation. Note that the concept of visiting all nodes individually differs from that of token dissemination [10, 31] over the entire network where it suffices for every node to simply hear at least one token, as opposed to getting exclusive access to a token.

There are many applications for an all node visitation service, such as voting, computing aggregates (i.e., max, min, or average), statistical counting (i.e., estimating the fraction of nodes that satisfy a state predicate), and computing empirical distributions of data in a network [41]. Example application scenarios include computing average sensor measurements, counting battalions and ammunition in military networks, and computing aggregate traffic densities in vehicular networks.

In static networks with stable links, data aggregation can be realized by traversing fixed routing structures such as trees or network backbones [34, 33, 24, 25]. However, in mobile networks and networks with frequent link changes, topology driven structures are likely to be unstable and to incur a high communication overhead for maintenance. In a recent paper, we have analyzed why routing protocols for MANETs such as OLSR [26] are unable to scale beyond 100-150 nodes, in terms of a *scaling wall* [28]. Firstly, as network size increases, paths are more likely to fail — the path connectivity interval falls as $O(\sqrt{N})$, with increasing network size N. Secondly, even small changes in node speed significantly increase this path failure rate and instability. Finally, in order to be successfully used for routing, the broken paths need to be fixed much faster than the rate at which they break and this involves fast link estimation for discovering broken paths and then repairing those paths, both of which incur high message overhead.

Thus, in contrast to the static network case, *Census*, exploits the simplicity of random walks to achieve token coverage in MANETs. Random walks are attractive for MANETs because they are inherently stable in the presence of network dynamics, have no critical points of failure, avoid structure maintenance, and have very little state overhead [5]. In fact, *Census* is largely mobility agnostic — motion models and node speeds have hardly any impact on its performance, as a major component of the protocol is simply token passing. In fact, in sparse networks, we observe that high node speeds seem to benefit convergence time by enabling faster mixing of nodes.

In a pure random walk, a node that holds a token picks a random node in its neighborhood and transfers the token to that node. The first time that any token visits a node can be used by the node to add its state into the aggregate being computed. This process is repeated until all nodes have been visited. Unfortunately, the *cover time* for random walks (time to visit all nodes) is typically high because of wasted exploration when a token repeatedly encounters already visited nodes. In order to expedite the cover time, we explore in this paper the idea of partially guiding random walks towards unvisited nodes.

Self-repelling random walks: Let us first consider biasing based on local information only, by giving preference to nodes in the neighborhood that have been visited lesser number of times. So if there are one or more unvisited nodes in the direct one-hop neighborhood (i.e., within the communication range) of a token, the token is passed to one of the unvisited nodes chosen at random. If all the nodes in the neighborhood are visited, then the token is passed to the node that is visited least number of times (with ties broken randomly). Such random walks have been termed as *self-repelling* random walks [8]. The uniformity with which self-repelling random walks traverse a two dimensional lattice has been documented and studied in [21]. In this paper, for the first time to the best of our knowledge, we study the cover time of self-repelling random walks in mobile networks and show analytically that self-repelling random walks achieve a cover time of $O(N \log(N))$ in a MANET where the nodes themselves move according to a random walk mobility model. We moreover prove that by just using a local self-repelling strategy, a significant portion of the network can be covered without much wasted exploration at already visited nodes. However, when the fraction of already visited nodes in the network rises beyond a certain threshold, self-repelling random walks exhibit a slowdown. This is because when all the nodes within the communication range of a token holder are already visited, the scheme involves exploring the region of visited nodes until an unvisited node is found. While the order of convergence in relation to N remains $O(N \log(N))$, the slowdown creates a long tail in the convergence and increases cover time. To redress this shortcoming, we use a complementary method that further speeds up the cover time, which forms the basis of *Census*.

Multi-hop gradient bias a.k.a Census: To prevent the random walks from getting stuck in regions of visited nodes while there are still unvisited nodes to be explored, we set up short, temporary multi-hop gradients to pull the token towards unvisited nodes. We show analytically that this yields a cover time of $O(N)$. Thus, the order of convergence improves by a factor of $\log(N)$. In doing so, *Census* introduces a gradient message overhead of $O(N \log(N))$, to pull tokens towards unvisited nodes. Nevertheless, this overhead is compensated by a reduction in the required number of token transfers. In fact, our simulations show that the ratio of token transfers to that of node size, i.e., the exploration overhead of gradient biased random walk remains constant (irrespective of network size) and close to 1.

1.1 Summary of contributions

We introduce the idea of applying self-repelling and gradient biased random walks for addressing the data aggregation problem in MANETs, which to the best of our knowledge has not been explored before. We show analytically that self-repelling random walks have a cover time of $O(N \log N)$ in a mobile network modeled as a time-varying random geometric graph with almost uniform stationary distribution of nodes. We then show analytically that by using short temporary gradients, the cover time can be improved to $O(N)$ in terms of network size N. We call this protocol as *Census* and also show that the overall communication cost for *Census* is $O(N \log(N))$. We show an improvement in cover time by a factor of k when k tokens are used. We analytically compare the performance of *Census* with regular random walks, flooding, gossip, diffusion and structure based routing protocols as well. We corroborate all of our analytical results for *Census* using ns-3 based simulations of mobile networks ranging from 100 to 4000 nodes, under different network densities, node speeds and mobility models. We also empirically validate the performance of self-repelling and gradient biased random walks in mobility models that do not result in uniform distribution of nodes at all times and in sparse networks where there can be temporary disconnected partitions. Our simulation results demonstrate that gradient biased random walks are simple, yet effective tools for achieving scalable and robust data aggregation in MANETs.

1.2 Outline of the paper

In Section 2, we describe the system model. In Section 3, we describe the *Census* protocol. In Section 4, we present an analytical characterization of convergence time and message overhead for *Census*, and compare this against that of random walks with local bias and pure random walks (without bias). In Section 5, we evaluate the performance of Census and compare it against pure random walks, random walks with local bias, flooding, gossip, diffusion and structure-based aggregation protocols. In Section 6, we discuss implementation considerations for *Census* in a MANET such as handling message losses and termination detection. We discuss related work in Section 7 and make concluding remarks in Section 8.

2 System model

We consider a mobile network of N nodes modeled as a geometric Markovian evolving graph [11]. Each node has a communication range R. We assume that the N nodes are independently and uniformly deployed over a square region of sides $\sqrt{A}$ resulting in a network density $\rho = N/A$ of the deployed nodes. Consider the region to be divided into square cells of sides $R/\sqrt{2}$. Thus the diagonal of each such cell is the communication range R. Let $R^2 > 2c\log(N)/\rho$. It has been shown that there exists a constant $c > 1$ such that each such cell has $\theta(\log N)$ nodes whp, i.e., the degree of each node is $\theta(\log N)$ whp. Such graphs are referred to as geo-dense geometric graphs [5]. Denote $d = \theta(\log N)$ as the degree of connectivity.

For our analysis, we focus on a random direction mobility model (with reflection) [6, 35] for the nodes. This is a special case of the random walk mobility model [9] and sometimes referred to as random walk 2-d mobility model in network simulators [1]. In this mobility model, at each interval a node picks a random direction uniformly in the range $[0, 2\pi]$ and moves with a constant speed that is randomly chosen in the range $[v_l, v_h]$. At the end of each interval, a new direction and speed are calculated. If the node hits a boundary, the direction is reversed. Motion of the nodes is independent of each other. An important characteristic of this mobility model is that it preserves the uniformity of node distribution: given that at time $t = 0$ the position and orientation of nodes are independent and uniform, they remain uniformly distributed for all times $t > 0$ provided the nodes move independently of each other [35, 11]. Our analysis in this paper holds for all mobility models in which the stationary distribution of node locations is almost uniform and the average degree of connectivity is constant at all times whp. Several commonly used mobility models such as the 2-d grid random walk and random waypoint on spheres also satisfy this uniformity property [11]. We have also empirically evaluated *Census* in other mobility models such as 2-d random waypoint and Gauss-Markov where such uniformity assumptions may not hold [35]. We have also empirically evaluated *Census* in sparse networks where the network may be temporarily disconnected into multiple partitions.

One or more tokens are introduced at random locations that are uniformly distributed within the network. The objective of *Census* is to pass the tokens around the network such that every node in the network is visited by at least one token. Recall that we say that a node is visited by a token when it gets exclusive access to the token. The visitation period is used to aggregate the node's state information into the token. Note that when a node's information is aggregated into the token, the size of the information in the token remains constant. In other words, individual information pertaining to a node such as its id or state is not stored in the token. Instead, only the aggregate value is retained.

A standard definition for the cover time (C_u) of a random walk starting from a node u is the expected time to visit all nodes in the graph starting from that node. The Cover time of a random walk on a graph G is $\max_u C_u$ [3]. This standard definition has previously been refined for the context of k independent random walks as follows. Let $T(k, v_1, ..., v_k)$ be the time taken to visit all vertices using k independent walks starting at vertices $v_1, ..., v_k$. The k-particle cover time $C_k(G)$ is defined as $C_k(G) = \max_{v_1,...,v_k} E(T(k, v_1, ..., v_k))$.

In this paper, we have retained these standard definitions of cover times for both self-repelling random walks and gradient biased random walks.

However, we note that the cover time for random walks has been typically studied and computed for static (not mobile) graphs and that too without additional gradient support, in which case the number of token transfers (steps), *time*, and number of messages are all equivalent. In this paper, when using self-repelling and gradient biased random walks, each token transfer step consists of announcement, token request and token transfer messages. Thus, although proportional, the number of messages is different than the number of steps. For gradient biased random walks, there is also message overhead associated with the setup of temporary gradients. Hence, we separately characterize (both analytically and empirically) the message overhead for self-repelling and gradient biased random walks. Secondly, since we study random walks on mobile networks, notion of time is related to node speed. Moreover, when dealing with wireless networks, time also involves messaging delays. Therefore, during empirical evaluation we separately characterize both the number of token transfers and the actual convergence time.

3 Census protocol

3.1 Protocol overview

The *Census* protocol consists of two components: (i) token passing, and (ii) gradient setup. The token passing component is based on the notion of a self-repelling random walk. To avoid wasted explorations because of tokens being stuck in regions of already visited nodes, a gradient setup component is added to the protocol. In this subsection, we first provide a brief overview of these two components.

3.1.1 Token passing

A node that holds a token first announces that it has a token. Neighboring nodes that hear this request start a random timer to request this token. The random timer values for placing this request are staggered such that nodes which have not been visited or have been visited a fewer times are able to send the request earlier than those that have been visited more times. Also, a node suppresses its request if it has already heard another request from a node that has been visited equal or fewer number of times. These steps ensure that the number of requests generated per token transfer remain fairly constant, irrespective of network density. After a preset time, the token announcer processes the received requests and transfers the token to a node

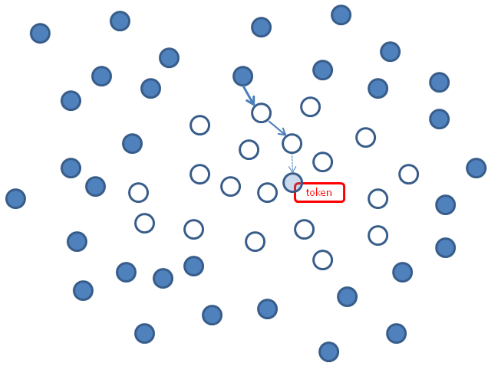


Figure 1: During token passing, a token may be surrounded by an island of visited nodes (white circles), i.e., all neighboring nodes have already been visited. Nodes that have not yet been visited (indicated by dark circles) periodically set up a gradient using the set of visited nodes to attract the token towards them.

which has been visited the fewest number of times. The details of the token passing component are described in Section 3.3.

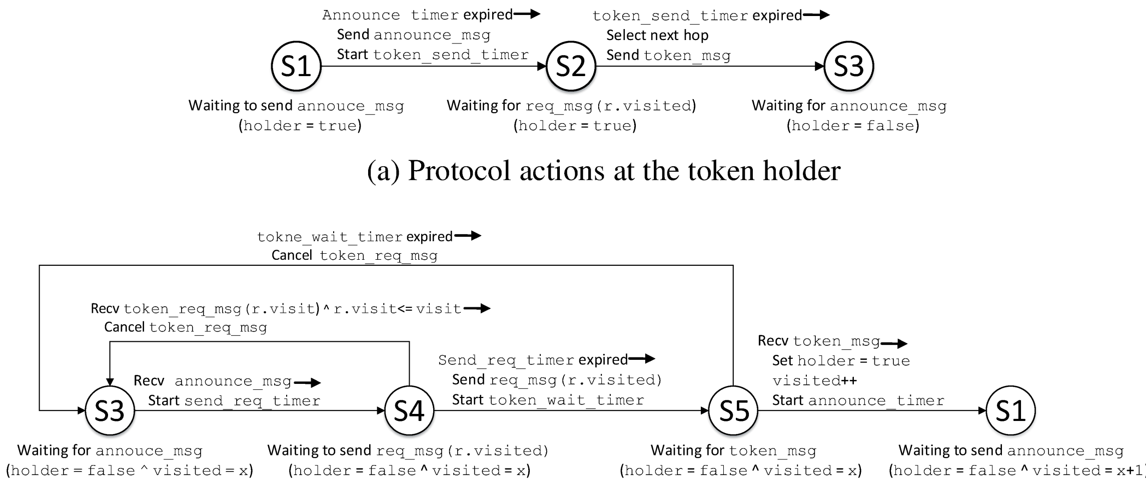
3.1.2 Gradient setup

As described before, self-repelling random walks may cause wasted explorations when all neighboring nodes have been visited. So, a gradient component is introduced. Nodes that have not yet been visited, set up short gradients that lead to the current token holder. When all neighboring nodes have been visited, these gradients are used as guides to transfer the token towards unvisited nodes (See Fig. 1). Since the network is mobile, the gradients need to be periodically refreshed. The details of the gradient setup component are described in Section 3.4.

3.2 Protocol state

To realize these components, each node stores three variables, *visited*, *holder* and *level*.

- The variable *visited* tracks the number of times that a node has been visited by any of the tokens; *visited* is initially 0 at all nodes. When a token first arrives at a node, *visited* is set to 1. Tokens are assumed to be initiated at a random set of nodes. All nodes in which a token is initiated are marked as visited by default and the token value is initialized to the data at the corresponding node.
- The variable *holder* is used to identify nodes that currently hold a token.



(a) Protocol actions at the token holder

(b) Protocol actions at non-token holder

Figure 2: State transition diagram for token passing

- When a gradient bias is used, each node also participates in a gradient setup process to attract tokens towards unvisited nodes. To do so, each node uses the state variable level where $0 \leq level \leq 1$. Nodes that are unvisited are at $level = 1$. Nodes that hold a token set *level* to 0 as soon as they receive a token. When a gradient is setup, a visited node may have value of *level* such that $0 < level < 1$, where a larger value indicates that the node is closer to an unvisited node along the gradient.

3.3 Token passing

3.3.1 Token passing with self-repelling random walks

For self-repelling random walks, a token holder announces that it has a token when the `announce_timer` expires. Nodes that hear this message and have not been visited, start a `send_req_timer` to send a request at a random slot within a chosen interval $[0, .., \frac{T_r}{2}]$. Nodes that hear this message and have already been visited, start the `send_req_timer` timer to send a request at a random slot within a chosen interval $[\frac{T_r}{2}, .., T_r]$. As a result, unvisited nodes get a chance to transmit before visited nodes. Also, if a node hears another request being sent with a value of *visited* that is lower than its own, it suppresses its own request. This staggered requesting procedure ensures that the number of requests being sent remain low and fairly constant irrespective of the network density.

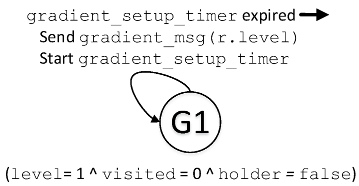
After a token holder announces that it has a token, it starts a `token_send_timer` with a value of

T_r. Until this timer expires, the token holder accepts requests for the token from its neighbors. When this timer expires, the token holder picks a random unvisited node if at least one unvisited node sends a request. Otherwise, the token holder picks the node that has been least visited. The token is transferred to the chosen node. The node that receives the token marks itself as visited if it was unvisited so far. If the token is used for data aggregation, an already visited node may not add its information again to a token. This concludes the procedure for token passing using self-repelling random walks. The token is continued to pass iteratively using this procedure. The protocol actions for the token passing component are illustrated in Fig. 2 using a state transition diagram.

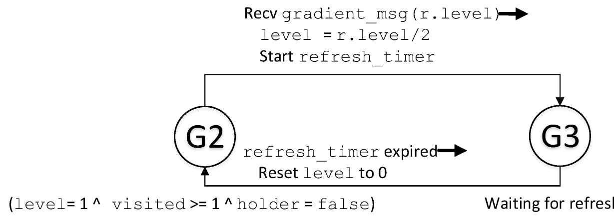
3.3.2 Token passing with gradient bias

For random walks with a gradient bias (i.e. *Census*), the procedure is similar. A token holder announces that it has a token. A `token_send_timer` with a value of T_r is then started at the token holder to accept requests for the token. Nodes that hear this message and have not been visited, start a `send_req_timer` to send a request at a random slot within a chosen interval $[0, .., \frac{T_r}{2}]$. All nodes with $(0 < level < 1)$ that hear the token announcement message send a request for the token at a random slot within a chosen interval $[\frac{T_r}{2}, .., T_r]$. Nodes with $(0 < level < 1)$ are nodes that have been visited and are now part of a gradient. If a node hears another request being sent with a *level* greater than itself, it suppresses its own request.

The token holder stores all requests received during time T_r. The token requests are sorted based on the level of the requestors and the token is sent to the node with the highest level. When multiple requestors exist with the same level, the token recipient is chosen randomly among that set. Thus if any unvisited node requests a token, the token will be sent to that node. If all nodes that have currently requested the token have been visited, the token is sent to the node with the highest value of level, which is expected to be the node that is closest to an unvisited node. As soon as a token reply has been sent, the token holding node resets *holder* to 0. Thus, the state transition sequence for token passing with gradient bias is similar to that of self-repelling random walks (Fig. 2) with the difference being the choice of the next hop. When gradient bias is used, the value of *level* is used to guide the tokens towards unvisited nodes. The following section describes how these short multi-hop gradients are setup.



(a) Protocol actions for gradient create



(b) Protocol actions at unvisited node to become part of gradient

Figure 3: State transition diagram for gradient setup

3.4 Gradients

3.4.1 Gradient setup

During *Census* operation a token can get stuck inside a region where all its neighbors have already been visited. To recover from such a scenario, a gradient is setup in the network to attract tokens towards unvisited nodes, i.e., nodes with $level = 1$ (See Figure 1). This is done as follows. Nodes with $level = 1$, for which none of their neighbors currently hold a token and have at least one neighboring node with $level = 0$, initiate a gradient setup by broadcasting a gradient message. Nodes with $level = 0$ that receive a gradient message from update their level to half of sender's level and rebroadcast the gradient message. Thus, gradient broadcasts propagate only till the region where nodes with non-zero level are present, filling up the gap between an unvisited node and other nodes with non-zero levels. The protocol actions for the gradient setup component are illustrated in Fig. 3 using a state transition diagram.

3.4.2 Gradient refresh

To account for node mobility, gradients have to be periodically refreshed. To do so, when a node updates its level from zero to some non-zero value < 1, it starts a timer proportionate to the new level and when the timer expires it resets its level back to 0. Thus nodes with higher values of level are refreshed slower than smaller values. This heuristic is based on two reasons. (1) Gradients should preferably not be refreshed before a token is able to climb up a gradient and reach an unvisited node. By refreshing at a rate proportional to the value of level, a token gets more time to reach closer to the source of the gradient. (2) Nodes that are far away from an unvisited node (closer to the bottom of the gradient) should prevent blocking of gradient setup from unvisited nodes that are nearby, for extended periods of time.

4 Census analysis

In this section, we quantify the expected bounds on cover time as well as message overhead for self-repelling random walks and random walks with gradient biasing. Our results show that self-repelling random walks have a cover time of $O(N \log(N))$, and *Census* with gradient bias further reduces the cover time to $O(N)$ in terms of network size using short, multi-hop gradient bias. In the remainder of this section, we analytically compare Census with traditional techniques for data aggregation such as structure based protocols, flooding, and gossip.

To begin with, let us prove a lemma that characterizes the expected geometric distance between unvisited tokens over time. This is useful because in the node visitation problem as long as there is even a single unvisited neighbor, a token will be transferred to that node. First, we state the following claim regarding the uniformity in the distribution of visited nodes during the progression of a self-repelling random walk.

Proposition 4.1. *The distribution of visited nodes (and unvisited nodes) remains uniform during the progression of a self-repelling random walk.*

Argument: Our claim is based on the analysis of uniformity in coverage of self-repelling random walks in [21]. In [21], the variance in the number of visits per node of *self-repelling* random walks is shown to be tightly bounded, resulting in a uniform distribution of visited nodes across the network. More precisely, let $n_i(t, x)$ be the number of times a node i has been visited, starting from a node x. The quantity studied in [21] is the variance $(1/N)(\sum_i (n_i(t, x) - \mu)^2)$, where $\mu = (1/N)(\sum_i n_i(t, x))$. It is seen that this variance is bounded by values less than 1 even in lattices of dimensions 2048×2048. We use this to infer that even after the walk started, the distribution of visited nodes (and by that token, unvisited nodes) remains uniform.

We have empirically characterized the remarkable uniformity with which self-repelling random walks visit nodes in a network in Section 5.1 (Fig. 5).

Lemma 4.2. *There exists at least one unvisited node within h-hops of a token holder in a self-repelling random walk with probability p as long as the fraction of unvisited nodes in the network ϕ_u satisfies $\phi_u > 1/(h^2 c)$.*

Proof. Recall that there are d nodes in each cell of size $R^2/2$ whp. There are $2A/R^2$ such cells. Given that a token is in any of these cells, there is one unvisited neighbor as long as each of the cells has at least one unvisited node. Let $n_u(t)$ denote the number of unvisited nodes at any time t. To find a lower bound on $n_u(t)$, we use the analogous coupon collector problem which studies the expected number of coupons to be drawn from B categories so that at least 1 coupon is drawn from each category [36, 17]. Using the coupon collector result [36, 17], if $n_u(t) > (2A/R^2)(\log(2A/R^2) + \gamma)$, then each cell has at least one unvisited node whp. In this case $\gamma = 0.5772$ is the Euler-Mascheroni constant.

First, we note the following:

$$\begin{aligned}
\log(N) - \log(2A/R^2) &= \log(NR^2/2A) \\
&\geq \log(2\rho Nc\log(N)/2\rho N) \\
&\geq \log(c\log(N)) \\
&> \log(\log(N)) \qquad \text{since } c > 1 \\
&> \gamma \qquad \text{since } N >> e^{e^{\gamma}} \text{ for the scale of networks we consider}
\end{aligned}$$

Therefore, having $(2A/R^2)(\log N)$ unvisited nodes instead of $(2A/R^2)(\log(2A/R^2)+\gamma)$ can only increase the chance of having an unvisited node in each cell. Hence, the required number of unvisited nodes in the network can be written as:

$$n_u(t) > \frac{2A\log(N)}{R^2} \tag{1}$$

Recall that $R^2 \geq 2c\log(N)/\rho$. Thus the required number of univisted nodes can be written as:

$$n_u(t) > \frac{2N}{2c} \tag{2}$$

Dividing by N, the fraction of unvisited nodes required so that each node has one unvisited neighbor can be written as:

$$\phi_u > \frac{1}{c} \tag{3}$$

To generalize this result to the availability of an unvisited neighbor within h hops of a token holder, we consider cells of sides $hR/\sqrt{2}$ which have a diagonal of length hR. Thus, the required fraction of unvisited nodes so that each node has one unvisited node within h communication hops can be written as follows, where $c > 1$.

$$\phi_u > \frac{1}{h^2 c} \tag{4}$$

This proves the Lemma. □

Significance: When $h = 1$ and $c = 2$, $1/(h^2c) = 0.5$. Thus, as long as less than 50% of the nodes are visited, self-repelling random walks are expected to find an unvisited node within one hop whp. With $h = 2$, we see that until about 88% coverage, an unvisited node can be expected within a 2 hop neighborhood of a token. This highlights the extent to which bounded self-repelling random walks can cover a significant portion of the network without much wasted exploration and without any supporting network structures. Note also that when c increases, we expect the degree d (i.e., the number of nodes in each cell) to increase and hence progressively larger portions of the network can be visited without much wasted exploration.

4.1 Self-repelling random walks

Theorem 4.3. *Both expected cover time and the expected number of token transfers for a self-repelling random walk in a connected, mobile network of N nodes with uniform stationary distribution of node locations and a single token are $O(N(\log(N)))$.*

Proof. From Lemma 4.2, we note that the expected number of unvisited neighbors remains greater than 1 as long as $\phi_u > 1/c$. Thus for a fraction $(1 - 1/c)$ of the nodes, the expected steps taken by a token is 1.

Once the fraction of visited nodes exceeds $(1 - 1/c)$, a slowdown is expected because the token might be randomly traversing an area of already visited nodes. But note that for a fraction $1/c - 1/4c$ nodes, there exists an unvisited node within 2 hops of a token holder and in general for a fraction $1/(h-1)^2c - 1/h^2c$ nodes, there exists an unvisited node within h hops of the token holder whp. Thus, for a fraction $1/(h-1)^2c - 1/h^2c$ nodes, an expected number of $O(h^2)$ steps is needed to find an unvisited node. Continuing upto a maximum of H hops, where $H = \sqrt{N}$, the total number of steps traversed by a token before visiting all nodes and the expected time for complete coverage is given by the following expression:

$$= O((N - \frac{N}{c}) + (\frac{N}{c} - \frac{N}{4c})4 + .. + (\frac{N}{(H-1)^2)c} - \frac{N}{H^2c})H^2)$$
$$= O(N - \frac{N}{c} + N(3 + 5/4 + .. + (2H+1)/(H)^2))$$
$$= O(N - N/c + N.\sum_{i=1}^{H}(2i+1)/i^2)$$
$$= O(N + N.\sum_{i=1}^{H}(2)/(i) + N.\sum_{i=1}^{H}(1)/(i^2))$$
$$= O(N(1 + \log(H))\ \{\texttt{Euler harmonic series approx.}\}$$
$$= O(N(1 + \log(N)))$$
$$= O(N(\log(N))) \quad \square$$

Now consider the region to be divided into k equi-sized areas with one token used to visit nodes in each area. Thus, each token is responsible for an area of N/k. Using this, we state the following corollary.

Corollary 4.4. *Both the expected cover time and the average number of transfers per token in random walks with local bias in a connected, mobile network of N nodes with uniform stationary distribution of nodes and k tokens are $O((N/k)\log(N))$.*

During performance evaluation, we relax the requirement that each of the k tokens is restricted to stay within a unique area. We initialize the k tokens uniformly at random across the network and validate empirically that the above bounds hold.

Using the above Corollary, we observe that with $\sqrt{N}$ tokens, the expected cover time is $O(\sqrt{N}\log(N))$. When $\log(N)$ tokens are used, the expected convergence time is $O(N)$.

4.2 Census with gradient bias

Theorem 4.5. *Both expected cover time and the expected number of token transfers in* Census *with gradient bias in a connected, mobile network of N nodes with uniform stationary distribution of nodes and a single token are $O(N)$.*

Proof. Similar to the analysis in Theorem 4.3, for a fraction $(1 - 1/c)$ of the nodes, the expected distance traveled by a token is 1. However, once the fraction of visited nodes exceeds $1/c$, the gradients will be used to pull the token towards unvisited nodes. Now note that for a fraction $1/(h-1)^2c - 1/h^2c$ of the nodes, there exists an unvisited node within h hops of the token holder whp. Thus, for a fraction $1/(h-1)^2c - 1/h^2c$ of the nodes, the expected distance traveled by a token is h.

Continuing up to a maximum distance of $H = \sqrt{N}$, the total average distance traversed by a token before visiting all nodes and the average time for complete coverage is given by the following expression:

$$
\begin{aligned}
&= O((N - \frac{N}{c}) + (\frac{N}{c} - \frac{N}{4c})2 + .. + (\frac{N}{(\sqrt{N}-1)^2c)} - \frac{N}{Nc})\sqrt{N}) \\
&= O(N + \frac{N}{c}(1 + 1/4 + 1/9 + ... + 1/N)) \\
&= O(N + \frac{N}{c}\sum_{i=1}^{\sqrt{N}}(1/i^2)) \\
&= O(N(1 + 1/c)) \\
&= O(N)
\end{aligned}
$$

□

In comparison to Theorem 4.3, we note a speed up by a factor of $\log(N)$.

Similar to Corollary 4.4, we note that with k tokens, the cover time for *Census* with gradient bias is $O(N/k)$. Thus, with $\sqrt{N}$ tokens, the expected cover time is $O(\sqrt{N})$. When $\log(N)$ tokens are used, the expected cover time is $O(N/\log(N))$. We also verify these results empirically.

Theorem 4.6. *The expected gradient message overhead in Census with gradient bias in a connected, mobile network of N nodes and 1 token is $O((N)\log(N))$.*

Proof. Following the lines of Theorem 4.3, we note that for a fraction $1/(h-1)^2c - 1/h^2c$ nodes, there exists an unvisited node within h hops of the token holder whp. Thus, for a fraction $1/(h-1)^2c - 1/h^2c$ nodes, a gradient of size $O(h^2)$ steps is needed to pull the token towards the unvisited node. Continuing upto a maximum of H hops, where $H = \sqrt{N}$, the total gradient setup cost follows from summing up the series as shown in the proof of Theorem 4.3. □

Thus, we note that when using gradient bias, there is an extra overhead to pull the tokens towards unvisited nodes, but this is compensated by reduction in the number of required token transfers and reduction in convergence time.

4.3 Comparison with other techniques for aggregation

4.3.1 Structured data aggregation

The problem of computing aggregates in ad-hoc networks is a well-studied one, especially for static sensor networks. Solutions include Directed Diffusion [25], Collection Tree Protocol [24], Sprinkler [34] and TAG [33]. However, in mobile networks and networks with frequent link changes, topology driven structures are likely to be unstable and incur a high communication overhead for maintenance. It has been observed that routing protocols for MANETs such as OLSR [26] are unable to scale beyond 100 - 150 nodes [16]. In a recent paper [30], we analyzed reasons for this *scaling wall* in MANETs. Firstly, as network size increases, paths are likely to fail more often - the median path connectivity interval in the network falls as $O(\sqrt{N})$, where N is the network size. Secondly, we observe that structure based approaches are not robust to motion models and node speeds because even small changes to node speed significantly increase the frequency of link changes and hence increase the path failure rate and instability. Finally, in order to be successfully used for routing, the broken paths need to be fixed much faster than the rate at which they break. This involves fast link estimation for discovering broken paths and then repairing those paths, both of which incur high message overhead. Not surprisingly, none of these protocols have been successfully adapted or evaluated for MANETs. On the other hand, we observe that *Census* is able to scale to several thousand of nodes in highly mobile networks. We also find that its performance is largely unaffected by motion model and node speeds (as shown later in Fig. 10 and Fig. 11).

4.3.2 Flooding and Gossip

A structure free approach such as flooding data from all nodes to every other node has a messaging cost of $O(N^2)$, and is not any faster than *Census*. Alternatively, for problems such as average consensus, one could use multiple rounds of local gossip where in each round a node averages the current state of all its neighbors and this procedure is repeated until convergence [22, 7]. However, this method requires several iterations and has also been shown to have a communication cost and completion time of $O(N^2)$ for convergence in grids or random geometric graphs, where connectivity is based on locality [40]. Census has a communication cost of $O(N \log(N))$, and it can be used for applications beyond just averaging.

A variant of flooding and gossip for general aggregation problems is a *diffusion*-like approach where the initiating node broadcasts an N-bucket register (one for each node in the network). Each time a node receives this register, it adds its own state into the register (if it is not already added) and rebroadcasts the register if it learned about any new node in this iteration. This process continues until all nodes have complete copies of the N-bucket register. Note that the size of each message in this technique is $O(N)$ and therefore the messaging cost is at least $\Omega(N^2)$. Moreover, this technique assumes that the ids of all the nodes in the network are known a priori. On the other hand, *Census* does not assume any knowledge about the nodes in the network and has a much lower communication cost.

5 Evaluation

In this section, we quantitatively evaluate the convergence characteristics and performance of *Census* under different network conditions. More specifically, in Section 5.1, we quantify the convergence characteristics of *Census* and compare with those of pure random walks and self-repelling random walks; our results characterize the improvement obtained by the gradient bias in *Census* by mitigating the long tail in cover time. And in Section 5.2, we evaluate the message overhead and cover time for *Census* under different network conditions such as communication range, number of tokens, mobility models, and speeds, and compare this with self-repelling random walks and regular random walks.

For simulations of *Census* using ns-3, we set up MANETs ranging from 125 to 4000 nodes with varying number of tokens that are initiated uniformly at random locations within the network. Nodes are deployed uniformly in the network. Initially, we choose the deployment area and communication range such that $R^2 = 4\log(N)/\rho$. Thus, the network is geo-dense with $c = 2$. We test such networks in our simulations with the following mobility models: 2-d random walk, random waypoint and Gauss-Markov. The average node speeds range from 3 to 15 m/s.

We then evaluate *Census* and self-repelling random walks when the network is sparse and the geo-density assumptions do not hold (Section 5.3. We choose $R^2 = 4/\rho$, i.e., the communication range does not grow as a function of $\log(N)$. In this case, the network may be disconnected at times. We show empirically that the analytical bounds on cover time and message overhead hold even in this sparse deployment scenario. We observe that the improvement offered by gradient bias is far more significant in this case.

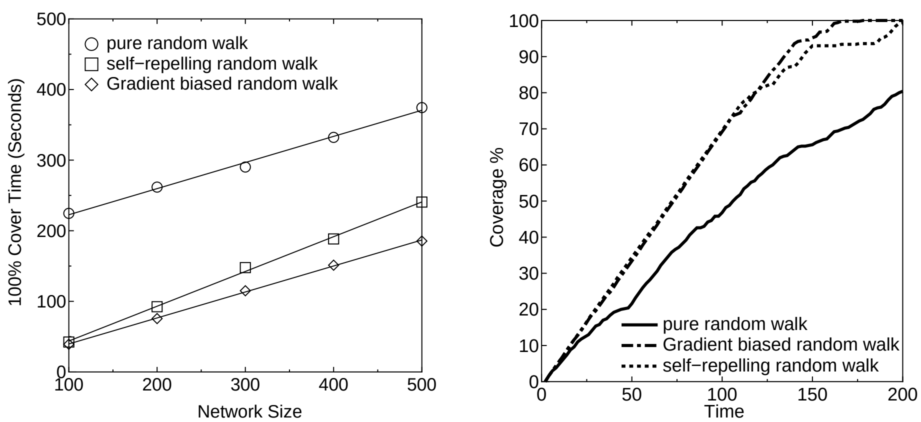
Finally, we compare self-repelling random walks with a special case of the same in which nodes only keep track of whether they are visited and do not keep track of the number of times they are visited. When all neighboring nodes have been visited, the token is transferred to a visited node chosen uniformly at random among the visited nodes. We observe that the performance of such locally biased random walks is quite similar to self-repelling random walks and the latter offers a constant factor of reduction in the exploration overhead. A detailed analysis of such locally biased random walks can be found in [29].

Table 1: Simulation parameters

Name	Values
Network size (N)	100, 200, 300, 400, 500, 1000, 2000, 4000
Mobility models	2D Random Walk, Random waypoint, Gauss-Markov
Deployment area (A)	100m X 100m
Network density (ρ)	N/A
Communication range (R)	$\sqrt{4 \log N/\rho}$
Average node speed	3 m/s, 7 m/s, 11 m/s, 15 m/s
Number of tokens	Ranging from 1 to $\sqrt{N}$

A summary of the various simulation parameters that have been used is listed in table 1.

5.1 Convergence characteristics

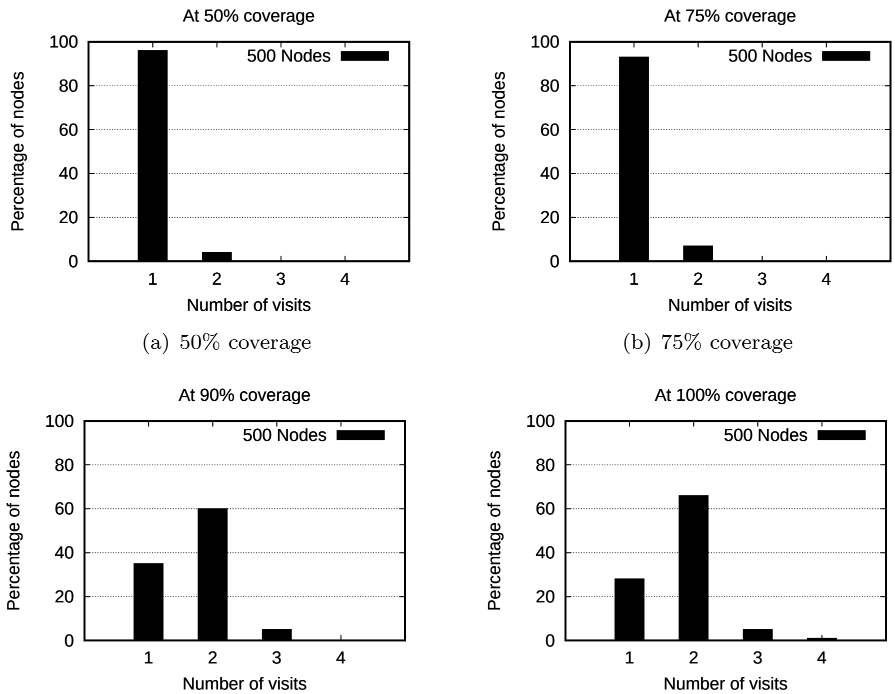


(a) Cover time vs network-size (single token) (b) Convergence pattern on a network of 500 nodes for a given trial with and without bias

Figure 4: Comparison of pure, local bias and gradient bias random walks

5.1.1 Comparison of pure, local bias and gradient bias random walks

Here, we compare the convergence characteristics of pure random walks with that of self-repelling random walks and random walks with gradient bias. We have used network sizes of $100 - 500$ and a single token in a random 2D-walk mobility model where the nodes move in a certain direction for a fixed distance and then choose a new random direction. The 100% cover time are shown in Fig. 4(a). The difference in convergence characteristics is illustrated in Fig. 4(b), where we show the convergence pattern in a single randomly chosen trial of 500 nodes for pure random walk, self-repelling and gradient biased Census. In this particular trial, we observe that until around the 80% mark, self-repelling proceeds on par with gradient bias, but then slows down slightly. This is because, until this point self-repelling enables a token to find an unvisited node directly and there are very few wasted explorations. A more pronounced slowdown for self-repelling random walk is noticed around 90%, whereas a pure random walk is slow throughout. After this point, there are wasted explorations in finding unvisited nodes when only self-repelling is used, whereas *Census* with gradient bias proceeds at a uniform rate throughout.



(a) 50% coverage

(b) 75% coverage

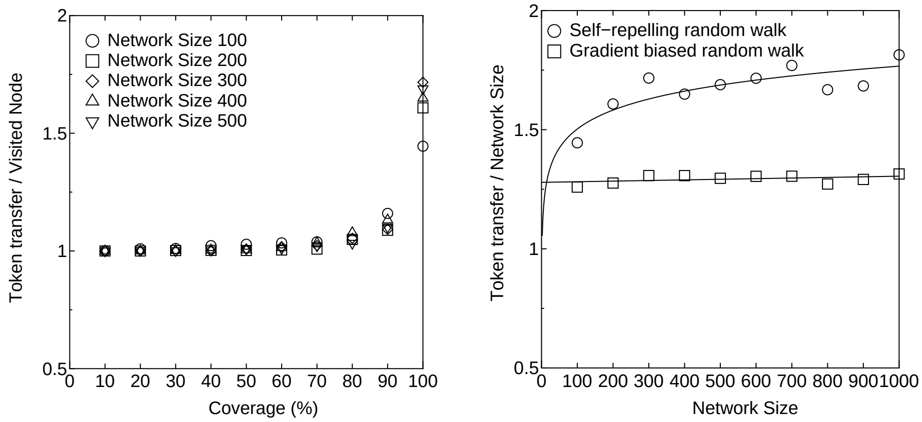
(c) 90% coverage

(d) 100% coverage

Figure 5: Number of times that each node is visited during various stages of coverage. $N = 500$. Results are averaged over multiple trials.

The remarkable uniformity with which self-repelling random walks visit nodes in a network is characterized in Fig. 5. In this figure, for the case of $N = 500$ nodes, we show the number of times nodes are visited by a token at different stages of coverage. We observe that even at 50% coverage, most of the nodes are visited only once. When all nodes are visited, the number of times each node is visited ranges only from 1 to 3. This matches the result in [21] and validates our claim about the uniformity in coverage with self-repelling random walks.

We analyze *exploration overhead* for self-repelling random walks in more detail in Fig. 6(a) which plots the ratio of token transfers to the number of visited nodes at different stages of coverage for self-repelling random walks (averaged over multiple trials). This ratio captures the wasted explorations, where a token is repeatedly passed to already visited nodes. We see that with self-repelling random walks, the ratio stays



(a) Token passing overhead as a function of coverage percentage (self-repelling random walk)

(b) Exploration overhead vs network size

Figure 6: Comparison of exploration overhead for self-repelling random walks and gradient biased random walks

low and close to 1 until about the $50-60\%$ mark and then starts rising. The observation matches our result in Lemma 4.2. But the token passing overhead for *Census* remains close to 1 throughout without any sharp rise at all network sizes as seen in Fig. 6(b). In Fig. 6(b), we compare the ratio of token transfers to the visited nodes at different network sizes with 100% coverage. We observe that for self-repelling random walks, this ratio grows as $\log(N)$, while for the gradient bias it is almost flat, matching our results in Theorem 4.3 and Theorem 4.5. Redundant token passes are very low with gradient bias. The exploration overhead factor indicates the number of token passing transactions required for coverage and is therefore a more accurate representation of convergence characteristics than the absolute cover time which is quite implementation specific and incorporates messaging latency in the wireless network. For instance, in our implementation each transaction (i.e., each iteration of token announcement, token requests and token passing) took on average $250ms$. But this number could be much smaller using methods such as [42] that use collaborative communication for estimating neighborhood sizes that satisfy given predicates.

5.2 Census properties under different network conditions

In this section, we evaluate the message overhead and cover time for *Census* (as well as self-repelling random walks) under different communication ranges, number of tokens, mobility models, and speeds.

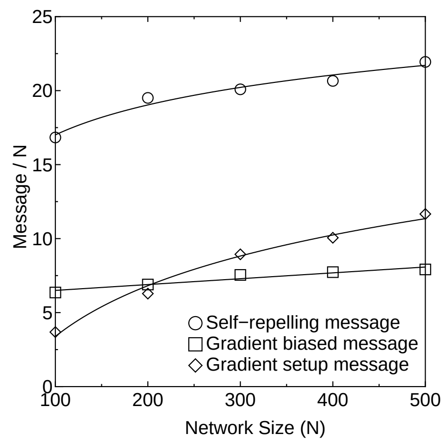
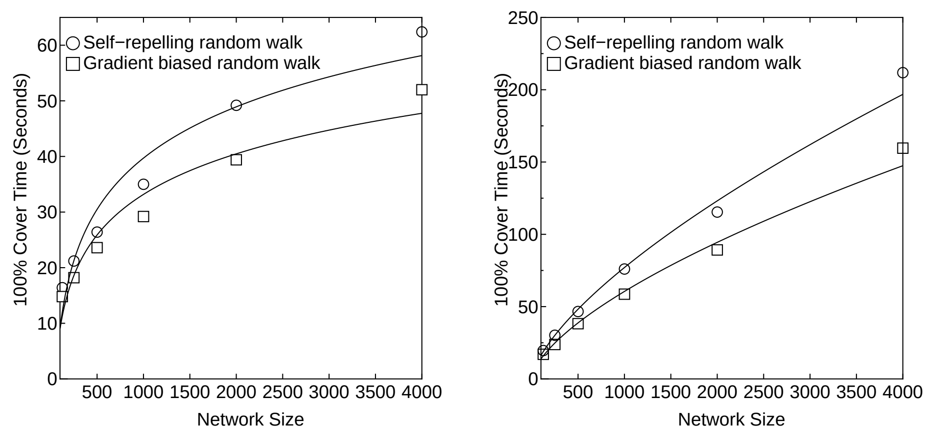


Figure 7: Analysis of message overhead for Census and self-repelling random walks



(a) Cover time vs network-size ($\sqrt{N}$ tokens)

(b) Cover time vs network-size ($\log_2 N$ tokens)

Figure 8: Impact of multiple tokens on the cover time

5.2.1 Message overhead

In Fig. 7, we compare the message overhead of *Census* and self-repelling random walks. To highlight the log factor overhead, we show messages normalized by network size on y-axis and network size on x-axis. The values for token messages with gradient bias stay constant indicating that the token messages grow linearly with network size. The curve for self-repelling token messages and the gradient setup messages grow as $\log(N)$, indicating a $N\log(N)$ messaging overhead. Note that the gradient setup cost is compensated by a significant reduction in the required token messages in *Census* as compared to the self-repelling scheme. These results show that *Census* achieves superior performance both in terms of cover time and reduction in message overhead over that of self-repelling random walks, despite the use of short multi-hop gradients.

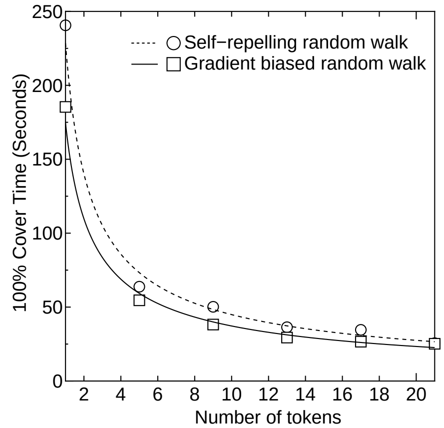
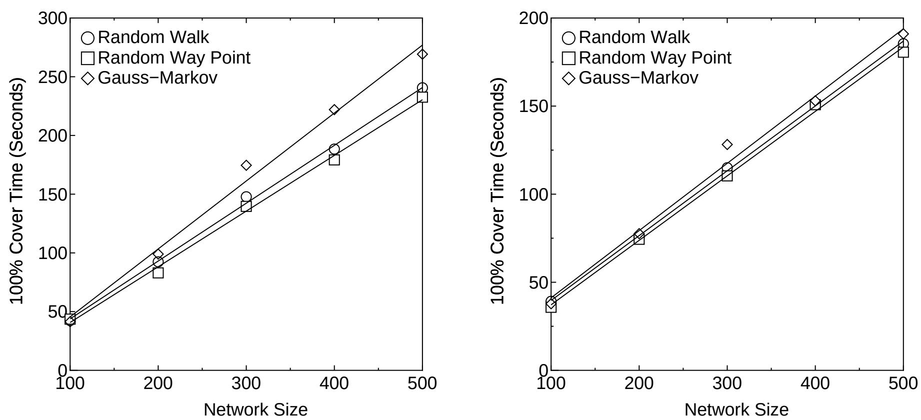


Figure 9: Cover time vs number of tokens ($N = 500$)

5.2.2 Multiple tokens

First, we quantify the impact of using $\sqrt{N}$ and $\log(N)$ tokens. Fig. 8(a) shows the cover time with $\sqrt{N}$ tokens in the network for Census and self-repelling random walks. The network sizes that we simulate are 125, 250, 500, 1000, 2000 and 4000. The corresponding number of tokens used in the network is 11, 15, 22, 31, 42 and 62 respectively. We observe that the coverage time grows only as $O(\sqrt{N})$, matching our analysis. In Fig. 8(b), we show the impact of using $\log_2(N)$ tokens. The network sizes that we simulate are 125, 250, 500, 1000, 2000 and 4000. The corresponding number of tokens used in the network is 7, 8, 9, 10, 11 and 12 respectively. Here, the trend is observed to be linear.



(a) Impact of mobility models (self-repelling) (b) Impact of mobility models (Census)

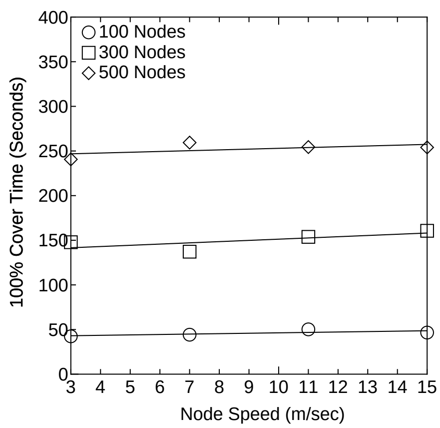
Figure 10: Cover time vs network size under different mobility models

Next, in Fig. 9, we analyze the cover time as a function of number of tokens. The network size is 500 and the number of tokens are varied from 1 to 22. We notice that the cover time decreases linearly with the number of tokens, matching our analysis.

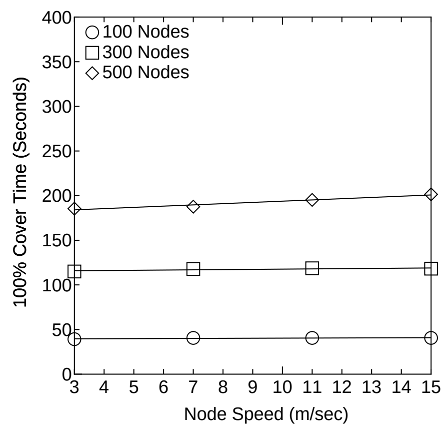
5.2.3 Impact of mobility model

We simulated *Census* under other mobility models, random waypoint and Gauss Markov. The node speeds remain in the range of $2-4$ m/s. For random way point, the pause time is set to 2 seconds between successive changes. In the Gauss Markov model, where motion characteristics are correlated with time, tuned with a parameter α, we set $\alpha = 0.75$. Velocity and direction are changed every 1 second in the Gauss Markov Model. For the random walk mobility model, it is known that the stationary distribution of the nodes is almost uniform. However, for the random waypoint model on a 2-d network, it is known that the uniformity distribution of nodes is not maintained (nodes tend to cluster towards the center). The uniformity properties of Gauss-Markov model are unknown. From Fig. 10, we observe that the cover time characteristics are similar for these different mobility models, even if the exact times show some variation, indicating robustness with respect to mobility model especially for *Census*.

5.2.4 Impact of node speed



(a) Impact of node speed (self-repelling)



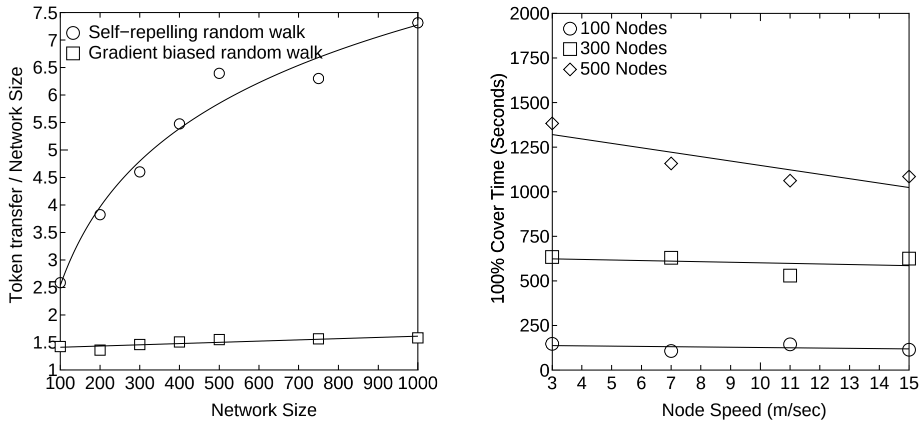
(b) Impact of node speed (gradient bias)

Figure 11: Impact of node speed

To highlight that *Census* is robust with respect to the rate of mobility induced link changes, we simulated *Census* under different node speeds. As observed in Fig. 11, even as the average node speed increases to $15m/s$, the cover time remains quite steady.

5.3 Census analysis in sparse networks

In this section, we evaluate self-repelling random walks and *Census* when the geo-dense property of network deployment does not hold. In other words, we choose a communication range that does not grow as a function



(a) Exploration overhead in sparse networks (below connectivity threshold)

(b) Impact of network speed in sparse networks for self-repelling random walks

Figure 12: Analysis in sparse networks

of log(N) and does not meet the connectivity threshold. Thus, the network may be partitioned at times and the node degree may not be uniform at all times. Specifically, we have chosen $R^2 = 4/\rho$ in the following results. In Fig. 12(a), we observe that the number of token transfers exhibits a similar trend to that of geo-dense networks for both self-repelling and gradient biased random walks. In comparison with Fig. 6(b), we observe that the improvement offered by *Census* with gradient biasing over that of self-repelling random walks is far higher in this case. This is because, the use of gradients allows a token to be quickly transferred to univisted nodes even when the network becomes connected momentarily. Self-repelling random walks spend more time in exploring for unvisited nodes and may not be able to utilize temporary moments when the network is connected.

In Fig. 12(b), we show the impact of higher network speed on self-repelling random walks in such sparse networks. We observe that at a network size of 500, the cover time actually starts improving with network speed. This is probably explained by the fact that at higher speeds, nodes which are temporarily disconnected from the portion that has a token, tend to converge with the connected component sooner and thus reduce the long tail in cover time.

5.4 Comparison with locally biased random walks

In this section, we compare self-repelling random walks with a special case of the same in which nodes only keep track of whether they are visited and do not keep track of the number of times they are visited. When all neighboring nodes have been visited, the token is transferred to a visited node chosen uniformly at random among the visited nodes. Thus, until there is at least one neighbor that has not been visited,

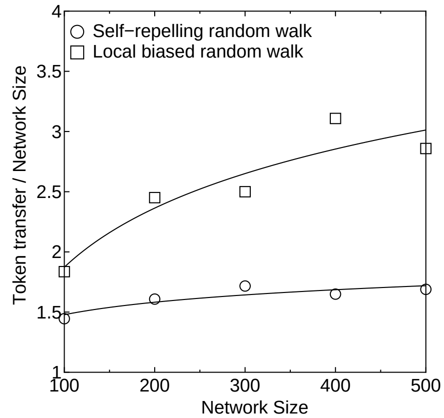


Figure 13: Exploration overhead (ratio of token transfer to network size) as a function of network size for locally biased random walks and self-repelling random walks

the action at each node is identical to that of self-repelling random walks. When all neighbors have been visited, locally biased random walks reduce to a cannonical random walk. We observe from Fig. 13 that the overhead of such locally biased random walks is quite similar to self-repelling random walks. However, self-repelling random walks offer a constant factor of reduction in the exploration overhead when compared to locally biased random walks. A detailed analysis of such locally biased random walks can be found in [29].

6 Discussion

In this section, we discuss some issues and optimization techniques that are not related to the core idea of using random walks for token coverage, but nevertheless are important in the context of implementing *Census* in a MANET.

6.1 Reliable token transfer

Reliable token transfer is critical for successful operation. If a token is released by a node, but the intended recipient did not receive the token reply message, the token is lost. At the same time, if the sending node relies on acknowledgements to release a token, it is possible that the acknowledgements are lost and duplicate tokens are created. For applications where duplicate counting is not permitted, this is a problem. This issue can be addressed in practice by using acknowledgments in conjunction with checkpoints. The procedure is described below.

As soon as a token reply has been sent, the sender releases the token (the node resets holder to zero). At the same time, it remains in a waiting state for acknowledgements from the recipient. If an acknowledgement is not received within a time T_a, the token send message is repeated up to a maximum of K re-tries. If the recipient receives the token multiple times, it simply repeats the acknowledgement message. However, if the token sender does not receive the acknowledgement even after K retries, it creates a *checkpoint* for the token: (a) the aggregate computed thus far is appended to the token along with the token id, (b) a fresh token id is created (unique ids can be created by simply assigning a node's id to the token during creation) and (c) the token aggregate is reset. It is possible that the token was actually successfully passed, but even in this case the checkpoint will not create duplicate counting. At the same time, the process ensures that data is not lost.

6.2 Termination detection

When using *Census*, termination can be deterministically detected. Note that when all nodes have been visited, the gradient setup will be terminated because the gradient setup is only initiated by nodes that have not been visited. As a result, a node that holds a token will continue to get only a level 0 reply for its token announcement. If a gradient is being setup, there would be at least one neighboring node with a value of level > 0. Therefore, when nodes holding the token get a level 0 reply from all its neighbors over an interval greater than the gradient refresh interval, the holder nodes can conclude that all nodes in the network have been visited.

In contrast, when using only self-repelling random walks, there is no deterministic way to detect termination. However, as the percentage of visited nodes increases, the ratio of token transfers to the visited nodes starts increasing. This ratio may be used to design an approximate threshold for termination detection. Moreover, the result in Fig. 6(a) shows the expected ratio of token transfers to the visited nodes at different levels of coverage for self-repelling random walks. In this figure, we see that until about $80-90\%$ coverage, there is very little variance in the token passing overhead ratio across different network sizes. Therefore, these values can be used to determine approximate thresholds for terminating a random walk trial at a desired level of coverage, irrespective of network size.

6.3 Token exfiltration

Once the initiated tokens have visited all nodes, it is necessary to ex-filtrate the tokens to a given location such as the operating base station or to one or more querying nodes in the network. Instead of using structures to route these aggregates towards querying nodes, a simple solution is to simply flood the aggregate tokens across the network in $O(D)$ time (where D is the network diameter) with an $O(Nk)$ message overhead where k is the number of tokens. This leads to a potential question: why not use flooding or diffusion based approaches all the way? Note that the cost of disseminating data from each node to all other nodes is $O(N^2)$ where N is the number of nodes in the network. By using a fixed number of k tokens to first compute the aggregates and then flooding the aggregates, the message overhead for flooding is only $O(Nk)$. Thus, our bounds on message overhead remain unaffected.

Note that other structure-free solutions are also possible for token ex-filtration. For instance, another potential solution is to transmit the k aggregated tokens using a long distance transmission link (such as cellular or satellite links) in hybrid MANETs where the *long links* are used for infrequent, high priority data.

7 Related work on random walks

Results on the cover time range for random walks in graphs have been shown to vary from $O(N \log(N))$ for complete graphs to $O(N^3)$ for lollipop graphs [32, 19, 20]. Typically, cover times are lower for dense, highly connected graphs and tend to increase as connectivity decreases [5]. A speed-up by a factor of k has been shown when k independent random walks are utilized in the graph [2, 23, 13].

Self-repelling random walks are those in which at each step, a random walk moves to a neighbor that has been visited least number of times (with ties broken using a random choice) [8]. The uniformity with which such random walks explore a graph has been analyzed in [21]. The analysis in [21] shows that the variance in the number of visits at each node is tightly bounded, resulting in a uniform distribution of visited nodes across the network. In this paper, we have exploited this observation to show that the cover time of self-repelling random walks in mobile networks is $O(N \log N)$. We further show that by using short temporary gradients to guide the self-repelling random walks, the cover time can be improved to $O(N)$. Cover times for biased and self-repelling random walks in time-varying graphs (that are relevant for mobile networks), have not been characterized before to the best of our knowledge.

For *static* mesh networks modeled as geometric graphs with uniform degree of connectivity, the expected cover time is known to be $O(N \log^2 N)$ [18]. It is also shown in [18] that if the communication radius is greater than a certain threshold, then the expected cover time is $O(N \log(N))$. In this paper, we have shown that the cover time can be improved to $O(N)$ with *Census* using gradient biasing. We also note that although the cover time for self-repelling random walks have the same upper bound as pure random walks on geometric graphs, the property of quickly covering a large fraction of nodes without much exploration is critical for the scalability and cover time bounds of the gradient biased random walk. In other words, applying the idea of gradients to regular random walks will not yield improvement in cover time.

We also note that in previous work there has been some *empirical* evidence of obtaining an $O(N \log(N))$ cover time for *static 2d grids* by exploiting some form of choice in random walks, where preference is given to less visited nodes at each step [5]. There is also empirical evidence from previous studies which show that locally biased random walks visit a significant fraction of the network without much wasted exploration. In [3], it is empirically observed that about 80% of the network can be covered in less than N steps, where N is the number of nodes in the network. These empirical observations further verify the claims made in our paper.

There has also been some work on unstructured techniques for data dissemination and informtaion sharing in mobile and dynamic networks (where the network topology is constantly evolving). In [11] and [12], the authors have modeled the network dynamics using a random walk mobility model and have obtained bounds on the flooding time, i.e., the number of steps to broadcast a message from a source to all nodes in the network. On the other hand, we study the time required to visit all nodes individually using random walk strategies in such mobile networks. In [39], the authors have studied the time required for a rumor to be shared among k agents performing independent random walks in a mobile network. Thus, they obtain bounds on the time required for multiple random walks to meet each other. In [4], the authors study cover times of random walks and lazy random walks on temporal graphs governed by adversarial strategies.

8 Conclusions

In this paper, we introduced the idea of using self-repelling random walks and gradient biased random walks for solving the data aggregation problem in MANETs, which to the best of our knowledge has not been explored before. Random walks are inherently structure-free and we showed that this yields a simple, yet robust strategy for data aggregation that is able to tolerate the dynamic topology of MANETs even as network size increases.

We observed that pure, unbiased random walks per se have high cover times, but a local, self-repelling strategy to bias the random walks significantly reduces the cover time. We showed analytically that self-repelling random walks have a cover time of $O(N \log N)$ in a mobile network. While self-repelling random walks are faster than pure random walks, we observed that they exhibit a long tail before all nodes are covered. This is because tokens tend to get stuck in regions with a dense concentration of previously visited nodes. To redress this shortcoming, we introduced a temporary multi-hop gradient bias to pull the tokens towards unvisited nodes. The resulting protocol (*Census*) has a cover time of only $O(N)$ and also significantly reduces the token passing overhead. We also showed that the overall communication cost for *Census* is $O(N \log(N))$. We further showed an improvement in cover time by a factor of k when k tokens are used.

We quantified the performance of *Census* under different network conditions. Our analysis shows that gradient bias outperforms unbiased as well as self-repelling random walks under all of these conditions. We showed that *Census* has very little state overhead, is naturally resilient to topology changes of MANETs, and is largely agnostic to mobility models and node speeds. In fact, in sparse networks, the convergence time for *Census* is seen to improve when average node speeds increase. We also showed how *Census* outperforms other techniques for node visitation such as flooding, gossip, and structure based routing protocols. Our analysis of *Census* is corroborated by simulations in ns-3 for networks ranging from 125-4000 nodes under different network densities, mobility models, and node speeds in MANETs, thus demonstrating the scalability and robustness of *Census*.

Importantly, *Census* is lightweight in terms of resource requirements and makes rather minimal assumptions of the underlying network. In particular, it does not assume knowledge of node addresses or locations, does not require a neighborhood discovery service or network topology information, and does not depend upon any particular routing or transport protocols such as TCP/IP. A key implication is that *Census* is readily instrumented for heterogeneous networks (and radio platforms).

We note that *Census* is not merely useful as an application layer protocol but can also serve as a network level routing service itself. In this sense, we regard *Census* as a candidate for a MANET network architecture that allows for Application Specific Network Patterns (ASNPs) so as to achieve scalability and robustness. These attributes have emerged as being important given recent high profile failures of MANETs, especially the JTRS program [27] which invested several billion dollars hoping to realize scalable, heterogeneous MANETs, that have motivated a clean-slate redesign of MANET solutions [15, 14]. A key learning from these failures is that network support for all to all communications does not scale well for mobile networks. The interested reader is referred to a companion document [38] on the ASNP architecture and its use of a minimal link layer abstraction to support multiple ASNPs, including *Census*, and their applications on heterogeneous radios as an alternative to an all-to-all communication standard.

Examples of applications that could utilize *Census* as a network level service are decentralized, network based control systems and distributed object tracking systems which frequently require access to consensus state estimates about different regions in the network [37]. Furthermore, the idea of successively visiting all nodes in the network using token passing itself can be utilized by other applications. For example, the protocol could be used to provide every node an access to a critical resource in a mutually exclusive manner, such as the use of a shared high-bandwidth link.

We are also working on refinement of *Census* for duplicate insensitive aggregation operations such as MAX and MIN in which the result remains intact even when the same information is counted multiple times. We expect further reduction in message complexity in such a scenario.

References

[1] The ns-3 network simulator. [Online - last accessed 2-June-2016].

[2] N. Alon, C. Avin, M. Koucky, G. Kozma, Z. Lotker, and M. R. Tuttle. Many random walks are faster than one. In *Proceedings of the Twentieth Annual Symposium on Parallelism in Algorithms and Architectures*, SPAA '08, pages 119–128, 2008.

[3] C. Avin and C. Brito. Efficient and robust query processing in dynamic environments using random walk techniques. In *International Symposium on Information Processing in Sensor Networks (IPSN)*, pages 277–286, 2004.

[4] C. Avin, M. Koucký, and Z. Lotker. How to explore a fast-changing world (cover time of a simple random walk on evolving graphs). In L. Aceto, I. Damgård, L. A. Goldberg, M. M. Halldórsson, A. Ingólfsdóttir, and I. Walukiewicz, editors, *Automata, Languages and Programming: 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part I*, pages 121–132. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[5] C. Avin and B. Krishnamachari. The power of choice in random walks: An empirical study. In *Proceedings of the 9th ACM International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems*, MSWiM '06, pages 219–228, 2006.

[6] J. Y. L. Boudec and M. Vojnovic. Perfect simulation and stationarity of a class of mobility models. In *IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 4, pages 2743–2754 vol. 4, 2005.

[7] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Trans. Info. Theory*, 52(6):2508–2530, 2006.

[8] C. Byrnes and A. J. Guttman. On self-repelling random walks. *Journal of Physics A: Mathemaical and General*, 17(17):3335–3342, 1984.

[9] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless Communication and Mobile Computing (WCMC): Special issue on mobile ad-hoc networking*, 2:483–502, 2002.

[10] Y. Chen, S. Shakkottai, and J. Andrews. On the role of mobility on multi-message gossip. *IEEE Transactions on Information Theory*, 56(12):3953–3970, 2013.

[11] A. E. F. Clementi, A. Monti, F. Pasquale, and R. Silvestri. Information spreading in stationary markovian evolving graphs. *IEEE Transactions on Parallel and Distributed Systems*, 22(9):1425–1432, 2011.

[12] A. E. F. Clementi, F. Pasquale, and R. Silvestri. Manets: High mobility can make up for low transmission power. In S. Albers, A. Marchetti-Spaccamela, Y. Matias, S. Nikoletseas, and W. Thomas, editors, *Automata, Languages and Programming: 36th Internatilonal Collogquium, ICALP 2009, Rhodes, greece, July 5-12, 2009, Proceedings, Part II*, pages 387–398. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[13] C. Cooper, A. Frieze, and T. Radik. Multiple random walks in random regular graphs. *SIAM Journal of Discrete Mathematics*, 23(4):1738–1761, 2009.

[14] DARPA. Clean slate ideas for manets. Online - last accessed 2-September-2015.

[15] DARPA. Fixed wireless at a distance. [Online - last accessed 2-September-2015].

[16] DARPA. Novel methods for information sharing in large scale mobile ad-hoc networks. Request for information: DARPA-SN-13-35, April 2013.

[17] A. V. Doumas and V. G. Papanicolaou. The Coupon Collector's Problem Revisited: Generalizing the Double Dixie Cup Problem of Newman and Shepp. *ArXiv e-prints*, Dec. 2014.

[18] G. Ercal and C. Avin. On the cover time of random geometric graphs. *Automata, Languages, and Programming*, 3580(1):677–689, 2005.

[19] U. Feige. A tight lower bound for the cover time of random walks on graphs. *Random structures and algorithms*, 6(4):433–438, 1995.

[20] U. Feige. A tight upper bound on the cover time for random walks on graphs. *Random structures and algorithms*, 6(1):51–54, 1995.

[21] H. Feund and P. Grassberger. How a random walk covers a finite lattice. *Physica*, A(192):465–470, 1993.

[22] R. Friedman, D. Gavidia, L. Rodrigues, A. Viana, and S. Voulgaris. Gossiping on manets: The beauty and the beast. *SIGOPS Oper. Syst. Rev.*, 41(5):67–74, Oct. 2007.

[23] A. Frieze and C. Cooper. The cover time of random regular graphs. *SIAM Journal of Discrete Mathematics*, 18(4):728–740, 2005.

[24] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis. Collection tree protocol. In *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, pages 1–14, 2009.

[25] C. Intanogonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva. Directed diffusion for wireless sensor networking. *IEEE Transactions on Networking*, 11(1):2–16, 2003.

[26] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot. Optimized link state routing protocol for ad hoc networks. In *IEEE International Multi-topic Conference (INMIC)*, pages 62–68, 2001.

[27] Joint Tactical Radio System. Jtrs — Wikipedia, the free encyclopedia, 2015. Online; accessed 2-September-2015.

[28] V. Kulathumani, A. Arora, M. Sridharan, K. Parker, and B. Lemon. On the repair time scaling wall for manets. *IEEE Communications Letters*, PP(99):1–4, 2016.

[29] V. Kulathumani, K. Parker, M. Sridharan, and A. Arora. Census: A protocol for visiting all nodes in manets using biased random walks. *CoRR*, abs/1409.7368, 2014.

[30] V. Kulathumani, M. Sridharan, A. Arora, B. Lemon, and K. Parker. On the repair time scaling wall for manets. *CoRR*, abs/1409.7370, 2014.

[31] P. Levis, N. Patel, S. Shenker, and D. Culler. Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks. In *USENIX/ACM Symposium on Networked Systems Design and Implementation (NSDI)*, pages 15–28, 2004.

[32] L. Lovascz. Random walks on graphs: A survey. Combinatorics, Paul Erdos 80, 1993.

[33] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: A tiny aggregation service for ad-hoc sensor networks. *SIGOPS Oper. Syst. Rev.*, 36(SI):131–146, Dec. 2002.

[34] V. Naik, A. Arora, P. Sinha, and H. Zhang. Sprinkler: A Reliable and Energy Efficient Data Dissemination Service for Extreme Scale Wireless Networks of Embedded Devices. *IEEE Transactions on Mobile Computing*, 6(7):777–789, 2007.

[35] P. Nain, D. Towsley, B. Liu, and Z. Liu. Properties of random direction models. In *IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 3, pages 1897–1907 vol. 3, 2005.

[36] D. Newman and L. Shepp. The double dixie cup problem. *American Math Monthly*, 67(1):58–61, 1960.

[37] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[38] K. Parker, M. Sridharan, A. Arora, V. Kulathumani, P. R. Kumar, J. Brown, R. Correa, H. Greenburg, D. Patru, C. Winters, and G. Thaker. Scaling mobile ad-hoc networks using application specific networking patterns. `https://samraksh.com/files/learning/Far\%20Out\%20Networking\%20-\%20Ad\%20Hoc\%20Mobility\%20Patterns.pdf`, July 2013. Public release: Technical book.

[39] A. Pettarin, A. Pietracaprina, G. Pucci, and E. Upfal. Tight bounds on information dissemination in sparse mobile networks. In *30th Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, PODC '11, pages 355–362, 2011.

[40] M. G. Rabbat. On spatial gossip algorithms for average consensus. In *2007 IEEE/SP 14th Workshop on Statistical Signal Processing*, pages 705–709, 2007.

[41] A. D. Sarwate and T. Javidi. Opinion dynamics and distributed learning of distributions. In *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*, pages 1151–1158, 2011.

[42] W. Zeng, A. Arora, and K. Srinivasan. Low power counting via collaborative wireless communications. In *Proceedings of the 12th International Conference on Information Processing in Sensor Networks*, IPSN '13, pages 43–54, 2013.