RaWMS - Random Walk based Lightweight Membership Service for Wireless Ad Hoc Networks

ZIV BAR-YOSSEF
Department of Electrical Engineering, Technion - Israel Institute of Technology and Google Haifa Engineering Center, Israel
zivby@ee.technion.ac.il
and
ROY FRIEDMAN and GABRIEL KLIOT
Computer Science Department, Technion - Israel Institute of Technology, Haifa 32000, Israel
{roy,gabik}@cs.technion.ac.il

This paper presents RaWMS, a novel lightweight random membership service for ad hoc networks. The service provides each node with a partial uniformly chosen view of network nodes. Such a membership service is useful, e.g., in data dissemination algorithms, lookup and discovery services, peer sampling services, and complete membership construction. The design of RaWMS is based on a novel reverse random walk (RW) sampling technique. The paper includes a formal analysis of both the reverse RW sampling technique and RaWMS and verifies it through a detailed simulation study. In addition, RaWMS is compared both analytically and by simulations with a number of other known methods such as flooding and gossip-based techniques.

Categories and Subject Descriptors: C.2.1 [**COMP.-COMMUNICATION NETWORKS**]: Network Architecture and Design—*Wireless communication*; C.2.4 [**COMP.-COMMUNICATION NETWORKS**]: Distributed Systems—*Distributed applications*

General Terms: Algorithms, Design

Additional Key Words and Phrases: Ad Hoc Networks, Membership service, Random Walk

1. INTRODUCTION

Context of this study. *Membership services* serve as essential building blocks in a variety of other services and applications in ad hoc networks. A membership service provides each node with a *view* regarding who are the other nodes in the network.

In traditional membership services [Chockler et al. 2001], the view of each process approximates the entire membership. Moreover, views must be consistent, and changes to views must be coordinated among all their members. This complete and strongly consis-

The first author is supported by the European Commission Marie Curie International Re-integration Grant.
This research is partially funded by the Israeli Science Foundation grant #44/03.
A shorter preliminary version of this paper appeared in the 7th ACM MobiHoc, May 2006.

tent approach works well in wired LANs. However, generally speaking, it is not suitable for large networks and mobile ad hoc networks. This is because maintaining such membership information consumes a lot of memory and requires large message and computational overheads for each membership change. In contrast, in mobile ad hoc networks, nodes often have limited memory capacities. The dynamic nature of the system implies frequent changes to the network membership. Additionally, wireless multi-hop networks are more sensitive to high message loads than wired LANs, and the energy consumption associated with sending and receiving many messages could quickly drain the batteries of mobile devices (making the usage of frequent flooding impractical). The mobility of nodes results in a continuous evolution of the physical structure of the network, causing frequent links and paths breakups, thereby discouraging the usage of multiple hop routing in such networks. These problems motivate the development of a membership service that avoids both flooding and multiple hop routing of messages.

Interestingly, many applications do not need complete membership information. Instead, they only require each member to hold a partial random view of the network membership. Examples of such applications are probabilistic reliable dissemination of data and events [Birman et al. 1999; Eugster et al. 2003; Kermarrec et al. 2003], peer sampling services [Jelasity et al. 2007], location services and uniform quorums [Haas and Liang 1999], random overlay constructions [Melamed and Keidar 2004], DHTs [Pucha et al. 2004], P2P anonymizers [Freedman and Morris 2002], etc. Therefore, it makes sense to offer an optimized membership service that indeed only provides nodes with partial random views. Such optimized services are the focus of this paper.

Contributions of this work. We start by introducing a novel reverse *Maximum Degree random walk* (RW) technique for peer sampling with an adaptation to ad hoc networks along with a formal analysis of this technique. Next, we present the *RAndom Walk based Membership Service* (RaWMS), which provides a random uniformly chosen partial membership view based on random walks. That is, every node in the network has an equal probability to appear in the view of every other node. In particular, the choice of the peers in the view of every node is independent of the locations of the peers in the network. [1]

In RaWMS, every Δ time units, every node starts a reverse Maximum Degree random walk, whose messages carry this node's identifier. Each RW traverses the network for a predefined number of steps and stops at some destination node. The length of the RW is such that the node in which the RW has stopped appears as if it was picked uniformly at random out of all network nodes. This way the source node advertises itself to the destination node, allowing the destination node to include the source node's identifer into its membership list. As we show in this paper, the result is that the membership list of the destination node includes a uniform random sample of nodes from the network.

Unlike many gossip-based algorithms, our service possesses five important properties. These include (i) proven uniform randomness of the constructed views, (ii) proven bounds on the load of an individual node (view size), (iii) enabling each node to set its view size independently of other nodes without any implications on the randomness of the views' content, (iv) a low chance of partition in the knowledge graph induced by the views, and (v) self healing from partitions when they do occur. Another important characteristic of

[1] The location-independence is importation for the target applications of such membership services, which depend on the fact that there is very little overlap in the views of any pair of neighboring nodes.

our algorithm is that it does not require multiple-hop routing. The analytically proven properties of our work are based on the assumption that the network graph is a connected static random geometric graph. We show through simulations that these properties indeed hold empirically for both static and mobile networks (yet, without a formal mathematical proof for the mobile case).

In the implementation of RaWMS, we seek to obtain a good tradeoff between the communication overhead incurred by our protocol vs. its memory consumption. To deal with this issue, our protocol allows every node to choose the target size of its view independently and without any correlation with other nodes. Moreover, a node can adjust its view size on-the-fly according to its currently available memory. In a small or medium size network, or if a node has plenty of memory, it may wish to maintain a large or even complete membership knowledge. On the other hand, in a sensor network or a large ad hoc network (with hundreds of nodes), nodes may wish to save memory and only maintain a partial membership view. If at some point in time a node with a small view requires knowledge of the entire membership, e.g., due to its application's demand, our service can reactively increase its view in a fast and efficient manner. This is done by consulting its neighboring nodes, which carries an additional small communication overhead.

We provide a detailed formal analysis of our implementation of RaWMS. Additionally, we extend the generic gossip-based peer sampling framework introduced in [Jelasity et al. 2007] to incorporate ad hoc networks. We utilize it to compare RaWMS with other membership construction techniques, such as lpbcast [Eugster et al. 2003], Shuffling [Gavidia et al. 2005; Voulgaris et al. 2005] and flooding. Finally, we study the performance of RaWMS by simulations, evaluating its properties and comparing it to the other known techniques mentioned above. These measurements largely confirm the insight from our theoretical analysis.

Paper's road-map:. Section 2 introduces the system model. In Section 3, we present the RW technique for peer sampling in ad hoc networks. Section 4 describes RaWMS and its formal analysis. Section 5 describes a generic framework used in a variety of gossiping algorithms for membership construction and compares a number of methods in this framework with RaWMS. Section 6 presents the simulation results for RaWMS vs. known gossip-based membership services. Section 7 discusses related work and we conclude with a discussion in Section 8.

2. SYSTEM MODEL

Consider a set of nodes spread across a geographical area and communicating by exchanging messages using a wireless medium. A node in the system is a device owning an omnidirectional antenna that enables wireless communication. Each node v may send messages that can be received by all other nodes within its transmission range r_v. A node u is a *neighbor* of another node v if u is located within the transmission range of v. The transmission disk of node v is a disk centered on v with radius r_v. The combination of the nodes and the transitive closure of their transmission disks forms a wireless ad hoc network.[2] The

[2]In practice, the transmission range does not behave exactly as a disk due to various physical phenomena. However, for the description of the protocol it does not matter, and on the other hand, a disk assumption greatly simplifies the formal model. At any event, our simulation results are carried on a simulator that simulates a real transmission range behavior including distortions, background noise, unidirectional links, etc.

network described above can also be modeled as a graph $G = (V, E)$ where V is the set of network nodes and E models the one-to-one neighboring relations.

The network connectivity graph $G = (V, E)$ of an ad hoc network is a special case of a d-dimensional *Unit Disk* graph, in which n nodes are embedded in the surface of a d-dimensional unit torus, and any two nodes within Euclidean distance r of each other are connected. When the nodes are placed uniformly at random on the surface the graph is known as a *Random Geometric Graph* (RGG) [Penrose 2003] and is denoted by $G^d(n, r)$. RGGs have been studied in the context of random walks, and thus we can utilize some of these results for our purposes. Specifically, the $G^2(n, r)$ graph is often used to model the network connectivity graph of 2-dimensional wireless ad hoc networks and sensor networks [Gupta and Kumar 1998]. See Appendix A for a formal description of the model.

We assume that nodes do not know their position and we do not use any geographic knowledge in our algorithms. Each node has a unique identifier that is used for sending messages to that node. The membership knowledge of a node, defined as the *view* of this node, is a list of identifiers of other nodes known to this node. In addition to the view structure, we assume that each node knows all of its direct neighbors, whose addresses are stored in the node's *neighbors list*. This list can be constructed, e.g., by a simple heartbeat mechanism that is present in any case in most routing algorithms for ad hoc networks. A node can communicate with its neighbors directly. Additionally, a node can communicate with other distant nodes whose address is present in its view by applying a routing algorithm.

New nodes may join and existing nodes may leave the network at any time, either gracefully or by suffering a crash failure. Nodes that crash or leave the network may rejoin it later (nodes that rejoin the network use their old identifiers).

Assumptions. For the theoretical analysis of random walk sampling in Section 3, we assume a static $G^2(n, r)$ connected network graph. The theoretical analysis of RaWMS in Section 4 allows nodes to leave and join the system, but still precludes mobility. However, the RaWMS algorithm itself is designed to operate in both static and mobile networks. In particular, the way RW is implemented in RaWMS can handle evolving neighborhoods, including recovering from disappearance of neighbors (either due to mobility, failure, or departure from the network). The correct behavior of RaWMS in mobile networks is shown by a simulation study in Section 6.

3. RANDOM WALK TECHNIQUES

Simple random walks. Let $G = (V, E)$ be an undirected graph, $n = |V|$. Let d_v denote the degree of a vertex $v \in V$. A *simple random walk* on G is a stochastic process in which a "token" is repeatedly forwarded from a node to a randomly chosen neighbor. Formally, the random walk is specified by an $n \times n$ probability transition matrix P, where $P_{v,u} = 1/d_v$, if $(v, u) \in E$, and $P_{v,u} = 0$ otherwise. For every time step $t \geq 0$, ϕ_t is a probability distribution over the vertex set V. It specifies, for each $v \in V$, the probability that the token is placed on vertex v at step t. The initial distribution ϕ_0 specifies the vertex at which the random walk is started. For every $t \geq 1$, $\phi_t = \phi_0 P^t$.

If the graph is connected and non-bipartite, then the sequence of distributions $\phi_0, \phi_1, \phi_2, \ldots$ is guaranteed to converge to a unique *limit distribution* π, which is *independent* of the initial distribution. π is also a *stationary* distribution of P, that is, $\pi P = \pi$.

A simple analysis (cf. [Lovász 1993]) shows that the stationary distribution of the simple random walk has a limit distribution that assigns probabilities to nodes proportionally to their degree: $\pi(v) = \frac{d_v}{2|E|}$, for every $v \in V$. Therefore, a stationary distribution of a simple random walk on a graph is uniform if and only if the graph is *regular*, i.e., all nodes have the same degree. Later in the section we will present the *Maximum Degree* random walk, whose stationary distribution is uniform even for non-regular graphs.

RW-based sampling. The following algorithm uses a random walk on G to sample nodes from the limit distribution π:

`sample` (p,T)
1) **start** a RW from p;
2) **run** the RW for T steps;
3) **return** the node in which the RW was stopped

If π happens to be the uniform distribution, then the algorithm generates uniform sample nodes. The idea of the algorithm is very simple: it starts the random walk at some start vertex p and runs it for T steps. The node reached after T steps is returned as a sample. If T is sufficiently large, then the distribution ϕ_T of the node returned is close to the limit distribution π.

Notice that this sampling technique does not require a priori knowledge of all network nodes and does not use multi-hop routing. A node must only be aware of its neighbors. This makes RW-based sampling attractive for ad hoc networks.

The main question to be addressed is how to set T to guarantee that ϕ_T is close to π. To this end, we define the *mixing time* of a random walk:

Definition 3.1. *For every node $v \in V$, let ϕ_0^v be the initial distribution concentrated on v. For every step $t \geq 0$, the total variation distance between $\phi_0^v P^t$ and π is defined as:*

$$\triangle_v(t) = \frac{1}{2} \sum_{u \in V} |\phi_0^v P^t(u) - \pi(u)|.$$

For every $\epsilon > 0$, the ϵ-mixing time *of the random walk is:*

$$T_{\text{mix}}(\epsilon) = \max_{v \in V} \min\{t \mid \triangle_v(t') \leq \epsilon, \forall t' \geq t\}.$$

Intuitively, the mixing time of a RW is the minimum number of steps t required to guarantee that, regardless of the start vertex of the random walk, the probability distribution reached after t steps is ϵ-close to the stationary distribution. Throughout this paper, when the parameter ϵ is omitted, we refer to mixing time with $\epsilon = \Theta(\frac{1}{n})$.

A popular method for bounding the mixing time of a random walk is via the *spectral gap* of its transition matrix. Let $\lambda_1, \lambda_2, \ldots, \lambda_n$ be the n eigenvalues of P ordered in decreasing absolute value. It can be shown that all these eigenvalues must be real and lie in the interval $[-1, 1]$, where the principal eigenvalue, $\lambda_1 = 1$. If G is connected and non-bipartite, then $|\lambda_2| < 1$. The difference $1 - |\lambda_2|$ is called the *spectral gap* of P and turns out to determine the mixing time of the random walk (cf. [Guruswami 2000]):

THEOREM 3.2. *The mixing time of a random walk with transition matrix P is upper bounded as follows:*

$$T_{\text{mix}}(\epsilon) \leq \frac{\ln \pi_{\min}^{-1} + \ln \epsilon^{-1}}{1 - |\lambda_2|},$$

where $\pi_{\min} = \min\{\pi(v) \mid v \in V\}$.

Note that when π is the uniform distribution then $\pi_{\min} = 1/n$.

Theorem 3.2 provides the means for setting the parameter T in the sampling algorithm. Given a bound on the spectral gap of P (which is typically derived by analyzing combinatorial properties of the graph G) and given the desired accuracy parameter ϵ, we can use the above formula to calculate T.

The Maximum Degree RW. As mentioned above, the simple RW on a graph converges to a uniform limit distribution if and only if the graph is regular. Ad hoc network graphs are typically non-regular, and thus we cannot use the simple RW directly to obtain uniform sampling of network nodes. Instead, we use a different RW, called the *Maximum Degree* (MD) random walk, which has been used before in various contexts [Bar-Yossef et al. 2000; Boyd et al. 2004; Boyd et al. 2005; Lovász 1993] to achieve uniform sampling.

Let $G = (V, E)$ be an undirected, connected, and non-bipartite graph, which is not necessarily regular. Suppose we have an upper bound D on $d_{\max}$, the maximum degree of G (we show how to obtain such a bound below). We use D to transform G into a regular graph G'. To this end, we add to each node v of G a *weighted self loop* (i.e., multiple edges from v to itself). The weight of the self loop of v is set to be $D - d_v$. The degrees of all nodes in the resulting graph G' are the same and equal D. The Maximum Degree random walk on G is the simple random walk on G'. The transition matrix of this random walk is then the following:

$$P_{v,u} = \begin{cases} 1/D, & \text{if } (v,u) \in E,\ v \neq u, \\ 0, & \text{if } (v,u) \notin E, \\ 1 - \sum_{u' \neq u} P_{u',u} & \text{if } v = u. \end{cases}$$

If G is connected, then G' is connected and non-bipartite, and hence (since G is undirected, connected and non-bipartite) the MD random walk has a limit distribution. Furthermore, since G' is regular, this distribution is uniform.

Many of the steps performed in a MD random walk are self loop steps. In many applications, including ours, self loop steps are "free": they can be executed in zero time and require no communication. Thus, it makes sense to define the *actual mixing time* of a random walk, denoted $T_{\text{actual\_mix}}$, which is the expected number of actual steps (i.e., non-self loop steps) needed for the random walk to approach its limit distribution.

As we shall see later, an overestimate of D may increase the mixing time of the MD random walk, but typically does not affect the actual mixing time. This is because an inflated D increases the mixing time at the same rate it increases the fraction of self loop steps, leaving the number of actual steps intact.

Another interesting aspect of MD random walks is that mobility does not affect the stationary distribution of the graph, as long as D is picked large enough to bound $d_{\max}$. As we discovered empirically (Section 6), it appears that (random) mobility even improves the mixing time.

Random walks on ad hoc networks. Wireless ad hoc and sensor networks are typically modelled as *Random Geometric Graphs* (RGG). We show that for an appropriate values of the radius r, a random geometric graph $G^2(n, r)$ is with high probability undirected and connected. Hence, the MD random walk on $G^2(n, r)$ is likely to converge to a uniform limit distribution.

Undirectedness. Recall that two nodes $u, v \in G^2(n,r)$ are connected by an edge if and only if the Euclidean distance between them is at most r. Since Euclidean distance is symmetric, $G^2(n,r)$ must be undirected. [3]

Connectivity. For RW based sampling to work, we must require the network graph to be connected. The connectivity of $G^2(n,r)$ was extensively studied in the context of the minimal transmission power necessary to ensure that with high probability a given ad hoc network graph is still connected as the number of nodes in the network grows to infinity. Gupta and Kumar [1998] have shown that if n nodes are placed on a unit disk and each node transmits at a power level that covers an area of $\pi r^2 = \frac{\log n + c(n)}{n}$, then the resulting network is asymptotically connected with probability one, if and only if $c(n) \to \infty$ as $n \to \infty$. In [Panchapakesan and Manjunath 2001], the authors obtain a similar result when nodes are distributed in the unit square $[0,1]^2$.

Throughout this paper we assume a radius $r = \sqrt{\frac{C \ln n}{n}}$ for the transmission range, where C is a constant. For $C > 1/\pi$, this is the minimal radius that satisfies the connectivity condition of Gupta and Kumar. Thus, we can assume w.h.p. that the ad hoc network graph is connected.

For technical reasons, we also assume the radius r is not too large ($r \leq 1/2$). If the radius is greater than $1/2$, then the resulting graph is a clique or close to a clique, and thus the random walk on this graph mixes very quickly.

Estimating the maximum degree bound. We now prove an upper bound on the maximum degree of the random graph $G^2(n,r)$. Note that the maximum degree is not being used by the MD RW or RaWMS in any way and does not influence its communication cost. The parameter D of the MD RW can be set arbitrarily high, in order to ensure that it bounds the actual maximum degree. Hence, the utility of the proposition below is mainly to get a feel for how the degrees of such graphs behave; this result is generic, and not tied down to RaWMS. We also use the analysis below in evaluating the actual mixing time in Theorem 3.4 (which indeed shows that the value of D does not affect the actual mixing time).

PROPOSITION 3.3. *Suppose $r \leq 1/2$. Fix any $0 < \alpha_d < 1$ and let*

$$\delta_d = \sqrt{\frac{3}{\pi r^2 (n-1)} \cdot \ln \frac{2n}{\alpha_d}}.$$

Let d_{avg}, $d_{\max}$, and $d_{\min}$ be, respectively, the average, maximum, and minimum degree of the random geometric graph $G^2(n,r)$. Then,

(1) $E(d_{\text{avg}}) = \pi r^2 (n-1)$

(2) with probability at least $1 - \alpha_d$, $d_{\min} \geq (1-\delta_d) \cdot \pi r^2 (n-1)$ and $d_{\max} \leq (1+\delta_d) \cdot \pi r^2 (n-1)$

PROOF. Fix any $i \in \{1, \ldots, n\}$. For each $j \neq i$, let X_j be a 0-1 random variable

[3] The symmetry assumed in the theoretical model of RGGs is not always valid in real ad hoc networks and the transmission range does not behave exactly as a disk due to various physical phenomena. In practice, it is possible that a node v receives messages sent from node u, but not vice versa. Yet, such phenomena are rare and on the other hand, those assumptions greatly simplify the formal model. At any event, our theoretical results were verified through an extensive simulation with real transmission range behavior including distortions, background noise, unidirectional links, etc.

indicating whether the j-th node of $G^2(n,r)$ is a neighbor of the i-th node of $G^2(n,r)$ or not. Since two nodes are neighbors if and only if they are at distance at most r from each other, then $E(X_j) = \Pr(X_j = 1) = \pi r^2$. (Here we use the fact $r \leq 1/2$. Otherwise, a disk of radius r centered at the i-th node "wraps around" itself, and thus contains multiple "copies" of the same points on the surface of the unit torus. In particular, this means that the probability to have the j-th node as a neighbor the i-th node is lower than πr^2.)

Let $Y_i = \sum_{j \neq i} X_j$ be the degree of the i-th node. By linearity of expectation, $E(Y_i) = \pi r^2(n-1)$. Note that $d_{\text{avg}} = \frac{1}{n}\sum_{i=1}^{n} Y_i$. Thus, using linearity of expectation again, $E(d_{\text{avg}}) = \pi r^2(n-1)$. By Chernoff bounds

$$\Pr(|Y_i - E(Y_i)| > \delta_d E(Y_i)) \leq 2 \cdot \exp(-\frac{\delta_d^2 E(Y_i)}{3}).$$

Substituting $E(Y_i) = \pi r^2(n-1)$ and the value of δ_d, we have:

$$\Pr(|Y_i - \pi r^2(n-1)| > \delta_d \cdot \pi r^2(n-1)) \leq \frac{\alpha_d}{n}.$$

Using the union bound, the probability that there is a node whose degree is less than $\pi r^2(n-1) \cdot (1-\delta_d)$ or more than $\pi r^2(n-1) \cdot (1+\delta_d)$ is at most α_d. □

As shown by the proposition, the average degree of every node in $G^2(n,r)$ is $(n-1)\pi r^2$. For example, for $C = 1$ and $\alpha_d = 0.1$, the average degree is around $\pi \ln n$ and the maximum degree is at most a factor $(1 + \sqrt{1 + 3/\ln n}) \sim 2$ away from the average degree with probability 0.9.

Mixing time. Next, we analyze the mixing time of the Maximum Degree random walk on $G^2(n,r)$. Avin and Ercal [2005] and Boyd et al. [2005] analyze the mixing time of the simple random walk on $G^2(n,r)$ and show it equals $\Theta(r^{-2}\log n)$. Boyd et al. [2005] mention in their paper that a similar analysis can show the same bound on the mixing time of the MD random walk. Yet, they do not give this analysis explicitly. Furthermore, the analysis provided in these papers is asymptotic, and does not include the exact constants.

We follow the footsteps of Boyd et al. and provide a rigorous analysis of the mixing time of the MD RW. We show that:

THEOREM 3.4. *Suppose $r \leq 1/2$ and $n \geq 10$. Let $G^2(n,r)$ be a random geometric graph chosen with n nodes and radius r. Let D be any value that upper bounds the maximum degree of $G^2(n,r)$. Let $T_{\text{mix}}(\epsilon)$ be the mixing time of the MD random walk on this graph, when applied with the value D. Let $T_{\text{actual\_mix}}(\epsilon)$ be the actual mixing time of this random walk (i.e., excluding self loop steps). For any $C > 49$, if $r = \sqrt{\frac{C \ln n}{n}}$, then with probability at least $2/3$ (over the choice of the graph),*

$$\begin{aligned}
T_{\text{mix}}(\epsilon) &\leq \frac{30}{(1-\frac{7}{\sqrt{C}})^2} \cdot \frac{D}{n} \cdot \frac{1}{r^4} \cdot (\ln n + \ln \epsilon^{-1}). \\
T_{\text{actual_mix}}(\epsilon) &\leq \frac{120}{(1-\frac{7}{\sqrt{C}})^2} \cdot \frac{1}{r^2} \cdot (\ln n + \ln \epsilon^{-1}). \\
T_{\text{actual_mix}}(\epsilon) &\leq \frac{d_{\max}}{D} \cdot T_{\text{mix}}(\epsilon).
\end{aligned}$$

The proof of Theorem 3.4 is rather involved, and is therefore deferred to Appendix E. The proof relies on Sinclair's canonical paths method [Sinclair 1992] for bounding the

spectral gap of a random walk. The construction and the analysis of these canonical paths are done via partitioning of the torus into a square grid, and defining "square paths" on this grid. Several additional remarks are in order.

(1) If $d_{\max} \approx \pi r^2 n$ (as guaranteed w.h.p. by Proposition 3.3) and if we choose D to be close to $d_{\max}$, then the mixing time of the MD random walk is $T_{\mathrm{mix}}(\epsilon) = O(r^{-2}(\ln n + \ln \epsilon^{-1}))$. For our choice of r, if C is a constant, then this mixing time is $T_{\mathrm{mix}}(\epsilon) = O(n(1 + \frac{\ln \epsilon^{-1}}{\ln n}))$. On the other hand, if D is a gross overestimate of $d_{\max}$, T_{mix} can get higher.

(2) As opposed to the standard mixing time, which can get large if D is an overestimate of $d_{\max}$, the actual mixing time is not affected by the difference between D and $d_{\max}$. That is, $T_{\mathrm{actual\_mix}}(\epsilon) = O(r^{-2}(\ln n + \ln \epsilon^{-1}))$ always, regardless of the value of D. For constant C, we have

$$T_{\mathrm{actual\_mix}}(\epsilon) = O(n(1 + \frac{\ln \epsilon^{-1}}{\ln n})).$$

(3) The theorem exhibits a tradeoff between the mixing time and the radius r: the larger is r, the smaller is the mixing time. This is to be expected, since a large transmission range improves the connectivity of the graph, which results in a faster mixing time. On the other hand, large transmission range increases the number of transmission collisions, reducing the quality of the wireless link.

(4) The minimum network size, for which the above theorem gives a non-trivial result is obtained by setting $C = 50$, in which case $n \geq 1,060$. For smaller networks, the lower bound on r implies $r > 1/2$, which means that the graph $G^2(n, r)$ is a clique. In cliques (with self loops), the random walk mixes in a single step.

(5) The theorem shows that the asymptotic behavior of the random walk is linear. The fact that the bounds provide non-trivial results only for sufficiently large networks and that Theorem 3.4 is applicable only for quite large radii ($C > 49$) are artifacts of the involved theoretical analysis and not of the algorithm itself. We believe that in practice the RW mixes quickly for much smaller transmission ranges and for small networks as well. This is supported by our experimental results, in which we have experienced with $C = 1$ and observed almost uniform quality of the RW sampling for $T_{\mathrm{mix}}(\epsilon) = n/2$.

3.1 Reverse RW-based uniform sampling in ad hoc networks

The naïve, direct, approach for applying the MD random walk for generating uniform samples in an ad hoc network is the following. Every node v starts the sampling algorithm described above using the MD random walk, passing its own id and the random walk's mixing time as parameters. The last node reached in the random walk notifies v of its id. This id represents a uniformly sampled node from the network. The notification can be done either by using the reverse path of the RW or by applying unicast routing. Both introduce significant additional communication overhead.

To solve this problem, we propose using a *reverse sampling technique*. That is, instead of informing the source node v about a sampled destination node u, the destination u is informed about the source v. We claim that this constitutes a random sample of source nodes. Using symmetry arguments, the destination node u can use the source v as if v was sampled by u directly. This way, there is no additional routing overhead for notifying the result of the RW to its initiating node. Since every node can initiate a number of RWs with its id simultaneously, we can use this technique to construct for each node a random

sample of s ($1 \leq s \leq n$) other nodes.

Below, we prove that reverse sampling indeed results in a uniform sample of nodes.

LEMMA 3.5. *Suppose every node v in a network chooses (via a random walk) a random node X_v. For every u, let Z_u be the set of nodes that selected u (the RWs started by them have stopped at u): $Z_u = \{v \mid X_v = u\}$. Then, given that the size of Z_u is k, Z_u is a random subset of the vertex set of size k.*

PROOF. The proof can be found in Appendix C. □

4. RANDOM WALK BASED MEMBERSHIP SERVICE

In RaWMS, a *View* at a node v is defined as a set of node descriptors, where each descriptor consists of <*NodeIdentifier*,*LastTime*>. *NodeIdentifier* is the unique identifier of a given node u and *LastTime* is the last time that v has "heard" from u. Every node v advertises itself every Δ time units by starting a reverse sampling process, as described in Section 3.1. In other words, each Δ time units, v starts a Maximum Degree random walk, whose messages carry v's identifier. Each of these RWs traverses the network for a number of steps that is equal to the mixing time and stops at some node u. If u already has a descriptor corresponding to v in its view, u refreshes the last time it heard from v and discards the RW. Otherwise, u stores the identifier of v in its view. We propose two methods for removal of nodes from the view: *size-based* and *time-based*.

In the size-based method, a node maintains a hard limit on its view size. Each node may choose the target size of its view independently and without any correlation with other nodes. In case that the view of a node u exceeds its limit upon storing a new identifier, u discards a descriptor with the oldest *LastTime* from its view.

In the time-based method, every node discards nodes' descriptors according to a predefined timeout. The descriptor of node v is removed from node's u view, if u has not heard from v for *Timeout* time units. Each node may choose the value of *Timeout* independently and without any correlation to other nodes. A node can probabilistically adjust its view size by setting the *Timeout* proportionally to the mixing time and Δ. Both methods automatically deal with purging descriptors of nodes that already left the network. The difference between the methods is the probabilistic versus deterministic guarantee of the view size.

The general structure of RaWMS is presented in Figure 1. The protocol consists of two threads: an active thread that initiates a new RW every Δ time units and a passive thread waiting for incoming messages. The `discardExpiredFromView`(*View, Timeout*) function discards all descriptors from the view that the node has not heard from in the last *Timeout* period; `discardOldestFromView`(*View*) discards the oldest descriptor from the view; `refreshInView`(*View,addr*) refreshes the *LastTime* attribute of a given descriptor in the view; `storeInView`(*View,addr*) stores a new descriptor corresponding to a given address and the current time in the view. `pickNextNode` picks either one of the neighbor nodes or a self-loop (of the current node) according to the RW transition matrix probabilities.

RaWMS can also support construction of different views for different groups. Nodes periodically advertise themselves to all groups they belong to (every RW advertises the source node to all groups simultaneously). When a RW stops, the destination node can filter the source node according to the groups it belongs to.

```
do forever
  wait(Δ time units);
  // start a new RW
  ttl ← MixingTime;
  handleRW(myAddress,ttl);
  if timeoutBasedMethod then
    discardExpiredFromView(View,Timeout)
  endif;
enddo

upon receive(RW_message<addr,ttl>) from u do
  // resend the RW to the next node
  ttl ← ttl-1;
  handleRW(addr,ttl) enddo

handleRW(addr,ttl)
  while ttl> 0 do
    next ← pickNextNode();
    if next != v then
      send (RW_message<addr,ttl>) to next;
      return
    else
      // self-loop step, only count ttl
      ttl ← ttl-1
    endif
  // the ttl count reached 0
  enddo
  publish(addr)

publish(addr)
  if addr ∈ View then
    refreshInView(View,addr)
  else
    storeInView(View,addr)
  endif
  if sizeBasedMethod and ViewIsFull then
    discardOldestFromView(View)
  endif
```

Fig. 1. RaWMS - code for node v

4.1 Formal performance analysis

For the purpose of analysis of RaWMS, we assume that all nodes start the algorithm simultaneously with initial empty views and all nodes have the same target view size, denoted $s(n)$. Notice that these assumptions are only required for the formal performance analysis of RaWMS. On the other hand, the correctness of the reverse sampling (and RaWMS) does rely on the fact that all nodes advertise themselves at the same average rate $1/\Delta$. Otherwise, a bias towards more frequently advertising nodes will be created.

We define the *convergence time* to be the number of protocol steps required until all views reach their target size. The period from the beginning of the protocol run until the convergence time has passed is the *convergence period*. In order to evaluate the performance of RaWMS, we study the time and the communication complexity of the protocol throughout the convergence period. Obviously, the target view size, that can be picked by each node independently from other network nodes by enforcing a view size limit or by using an aging timeout, has a direct impact on the memory consumption of the node, as well as on the time and the communication complexity of the convergence process. Clearly, the larger the target view size is, the more messages should be sent and the more time the view construction takes.

Intuitively, if each random walk started by some node v would have reached a different node, then in order to obtain a view of size $s(n)$, it would have been enough to start $s(n)$ RWs at each node during the convergence period. However, two random walks started at the same node v have a non-negligible probability of reaching the same node u. Thus, in order to obtain the target view size $s(n)$, each node should start a larger number of

RWs, which we denote by $r(n)$. Once we compute $r(n)$, we can immediately compute the communication and time complexity to reach convergence.

The average value of $r(n)$. In order to calculate $r(n)$, we refer to the famous *bins and balls* probabilistic problem: how many balls should be placed randomly into n bins in order to have at least one ball in s bins. In our case, we wish to calculate the number $r(n)$ of random trials (the "balls") that are required until $s(n)$ different destination nodes (the "bins") are picked. Each random trial corresponds to a single RW. (For simplicity of analysis, we assume below that each RW chooses a truly uniform node from the network, i.e., $\epsilon = 0$). We prove the following:

LEMMA 4.1. *Let $1 \leq s = s(n) \leq n$ and let $r = r(n)$ be the random variable specifying the number of balls needed to be randomly placed in n bins until s of the bins are non-empty. Then,*

$$E(r) = n(H_n - H_{n-s}) \leq \begin{cases} n \ln \frac{n}{n-s}, & s < n, \\ n \ln n + O(1), & s = n. \end{cases}$$

where $H_k = \sum_{i=1}^{k} \frac{1}{i}$ is the k-th harmonic number (and define $H_0 = 0$).

PROOF. The proof can be found in Appendix D. □

Note that using the inequality $1 + x < e^x$, which holds for all $x > 0$, we have:

$$n \ln \frac{n}{n-s} = n \ln(1 + \frac{s}{n-s}) < \frac{ns}{n-s}.$$

This gives a tight bound on $E(r)$ for $s \ll n$.

Note that nodes start new RW every Δ time units and do not have to be aware of $E(r(n))$ or make any use of it in RaWMS. $E(r(n))$ is used here only for the performance estimation of the algorithm.

However, $E(r(n))$ can be exploited by nodes working in the *time-based* method in order to adjust their average view size. A node that wishes to maintain an average view size of $s(n)$ can calculate the corresponding $E(r(n))$ independently based on its $s(n)$ and use the value $E(r(n)) \cdot \Delta$ as the *Timeout* for purging old descriptors out of its view. According to this strategy, no identifier stays in a node's view for more than $E(r(n)) \cdot \Delta$ time units on average without being refreshed by a new RW. Thus, an important property of RaWMS is that every view is refreshed to contain a completely new set of identifiers every $E(r(n)) \cdot \Delta$ time units on average.

Communication and time complexity for convergence. The communication complexity during the convergence period is determined by the number of random walks each node should start, i.e., the value $E(r(n))$ calculated above, multiplied by the length of each random walk. Thus, the total communication complexity during the convergence period is $n \cdot E(r(n)) \cdot T_{\text{actual\_mix}} = \Theta(n^2 \cdot E(r(n)))$. The time complexity is $E(r(n)) \cdot \Delta + T_{\text{actual\_mix}}$, i.e., the time to start $E(r(n))$ RWs and for the last RW to reach its destination.

For the special case of $s(n) = \sqrt{n}$, we get $E(r(n)) \approx \frac{ns}{n-s} \approx \sqrt{n}$. This means that for relatively small view sizes, there is a very little chance of getting collisions. The convergence time in this case is about $\sqrt{n} \cdot \Delta + T_{\text{actual\_mix}} = \Theta(\sqrt{n} \cdot \Delta + n)$ and the total communication complexity is $n \cdot \sqrt{n} \cdot T_{\text{actual\_mix}} = \Theta(n^2 \sqrt{n})$.

Bandwidth Consumption vs. Convergence Time. The main parameter that affects the bandwidth consumption of RaWMS is the frequency Δ in which nodes publish themselves. From the analysis above, the convergence time of RaWMS behaves linearly with Δ. Since in RaWMS all messages have the same size (every RW message includes an identifier of a RW originator), there is also a linear relationship between the bandwidth consumption of RaWMS and the number of messages. For example, doubling Δ would increase the bandwidth consumption by a factor of two, but will also halve the convergence time.

Join, leave, and maintenance. When a new node joins the network it starts the same algorithm as any other node, i.e., it starts advertising itself by initiating multiple RWs. After a convergence period, a new node will produce enough advertisements so that its identifier will be uniformly distributed across the network. Therefore, the time and communication complexities of a join process are $E(r(n)) \cdot \Delta + T_{\text{actual\_mix}}$ and $E(r(n)) \cdot T_{\text{actual\_mix}}$, respectively.

In order to speed up the uniform dissemination of new nodes in the network, a new node may initially advertise itself more frequently than $1/\Delta$. It can start the first $E(r(n))$ RWs at a fast rate, or even simultaneously. It is important, however, for the correctness of the reverse sampling, that after this initial phase, the joining node will return to advertising itself only once every Δ time units.

The algorithm purges the identifiers of failed or departed nodes automatically, without relying on any action on their side. In the time-based method, a failed node's identifier will be purged from the views of all other nodes precisely *Timeout* time units after its departure. In the size-based method, this will occur on average after $E(r(n)) \cdot \Delta$ time units.

The maintenance complexity of RaWMS is constant: all nodes keep advertising themselves at an average rate of $1/\Delta$ advertisements per time unit. The value of Δ can be tuned to tradeoff communication complexity with the time it takes to react to node leaves/failures and to purge their identifiers from all views.

Mobility. Nodes mobility is another important source of dynamic changes in the network graph of ad-hoc networks. This form of dynamism is not covered by our formal model and analysis. Very little is known in the literature about the behavior of random walks in mobile graphs. Moreover, dealing with mobility requires some knowledge about the mobility pattern.

Interestingly, our analysis for non-mobile networks can serve as a good approximation for the situation where nodes move slowly, or infrequently. Yet, at the other extreme, if nodes move fast and in a uniformly random fashion, then a partial uniform membership service can be trivially implemented by occasionally sampling the local neighborhood of each node. After a short duration, the physical network "mixes itself" well enough that the sample becomes uniform and random. However, in general, the speed of mobility cannot be trusted, and the mobility model is rarely uniformly random. Hence, even in mobile network, performing random walks is important for obtaining a good uniform sample of nodes. Formally analyzing the exact relationship between the mobility pattern and the required lengths of the random walks is left as an open research question. In this work we only study this issue by simulations.

Message loss. RaWMS uses a *salvation technique* to prevent dropping of RW messages. If a node v does not succeed to forward a RW message to the neighbor chosen in a given step (did not receive a MAC level acknowledgement), v makes a new attempt to send this

message to another random neighbor within the same step. This technique prevents a loss of RW messages in mobile networks, where nodes' mobility can lead to frequent breakages of neighborhood connections.

Notice that usage of such a salvation technique could potentially cause an undesired *forking* effect. That is, either a RW message was successfully received by the next node and propagated onward but the corresponding ack was dropped, or the next node failed after forwarding the RW message, but before acking it. In both cases, a RW messages would be resent to a different node, potentially creating a duplicate RW and leading to an additional message overhead and non uniform samples. We show that such forking in wireless ad hoc networks happens with a very low probability.

Forking probability. According to the IEEE 802.11 MAC protocol, when a source node does not receive an acknowledgement, it waits a backoff period and resends the message. Upon receiving a message for the second time, a destination node sends the ack again while discarding the duplicate message. The number of times the message is resent by the source node is defined by the protocol parameter, called *dot11ShortRetryLimit* (for short messages, up to 2347 bytes), whose default value is 7 [IEEE-802.11-Standard].

Therefore, in order for forking on a single link to occur, the first message should arrive, its ack should be lost and in six subsequent transmissions, either a message or an ack should be lost. Denote by p_{ack} the probability of an ack to be lost and by p_{msg} the probability of a subsequent (second, third and so on) transmission of a message to be lost. Therefore,

$$P(single\ forking) \leq p_{ack} * (p_{msg} + (1 - p_{msg})p_{ack})^6$$

The probability that no forking happens along the path of length n is:

$$P(no\ forking\ along\ the\ path) = (1 - P(single\ forking))^n$$

The value of p_{msg} is actually quite small, since it is a probability of a subsequent transmission to fail, after the first transmission succeeded (the nodes were neighbors during the first transmission and a subsequent transmission happens very soon after the first one). p_{ack} is small as well, given the 802.11 mechanism, which reserves the air link for an ack after a data message transmission. For example, for $p_{msg} = p_{ack} = 0.1$ and $n = 1000$, $P(no\ forking\ along\ the\ path) = 0.995$.

As for the second forking scenario, in which the next node fails or moves away after sending the RW message on, but before acking it, let us notice the following facts. The next node is a neighbor of the source node (since it received the first transmission) and the whole MAC transaction of resending the message and ack for 7 times occurs in a very short period of milliseconds. Thus, the probability that the next node will depart or move far after receiving the message and before acking it is very low as well.

We can therefore conclude that although theoretically possible, for the typical choice of parameters and network sizes we have considered, the probability of forking is very low. Our simulations (which inherently already include all these phenomena) validate that indeed forking almost never happens.

4.2 RaWMS usages and properties

Envisioned applications. Partial random membership can be very useful for construction of a variety of other services and applications in ad hoc networks (some of them are already mentioned in the introduction). Every node can pick its view size based on its memory

constraints and its envisioned applications' needs. For example, for the knowledge graph to be connected, the minimal view size should $\Theta(\log(n))$([Bollobas 2001]).

One of the applications we envision for RaWMS is a *data location service*. A data location service provides every node with the ability to share the data it posses with other network nodes, as well as to find and fetch for data stored in other nodes. In our implementation of a data location service, membership information is used in order to map data identifiers to nodes. Advertisements of new data items and lookups for existing data are based on this mapping. A good tradeoff between communication overhead and memory consumption for such a location service can be achieved using random views with an average size of $\Theta(\sqrt{n})$, as they help ensure intersections with high probability.

Another potential application that can benefit from our work is *P2P anonymization*. Consider, for example, a collection of Wi-Fi enabled cell-phones. Each cell phone can access the Internet directly using its cellular communication. However, this would leave explicit information identifying the surfer. The goal of an anonymizer is to utilize the ad-hoc network, created over the Wi-Fi capabilities of these cell phones, to anonymize such Internet accesses.

Systems like Crowds [Reiter and Rubin 1999] and Tarzan [Freedman and Morris 2002], to name a few, have been developed to provide user anonymity by reliance on P2P forwarding.[4] In such systems, the request first travels through a random path of nodes before being sent to the targeted web site by the last node in the path; the reply is sent back on the reverse path (each message carries a unique id and nodes remember a mapping between the ids of incoming messages and the node from which they came, so no intermediate node knows the entire path).

These ideas cannot be applied as is to ad-hoc networks, for example, since they assume that any node may communicate directly with any other node. In an ad-hoc network, multiple hop routing is expensive, and also might compromise the anonymity of the nodes. Hence, a more natural approach would be to perform a random walk, whose length is the mixing time of the network. Additionally, to avoid disclosure of the initial node, instead of adding a TTL to the message, it is possible to have each node decide with probability $P = 1/\textit{mixing\_time}$ to access the targeted web site, and with probability $1 - P$ to forward the walk. Our work is useful for this approach, since our analysis of the mixing time of random walks in ad-hoc networks, including the use of maximum degree random walks, can be applied here to compute the mixing time of the network.

Yet, the above usage of random walks may also be problematic, as the mixing time of the network is $O(n)$. We can improve on this by utilizing RaWMS directly. Specifically, the initiator of the request can include its RaWMS random view of size $\sqrt{n}$ (or a fraction of it, e.g., half the view) in the header of the message. In this approach, the random walk stops after reaching any of the nodes in the attached header, which will happen after $O(\sqrt{n})$ steps, on average. For this purpose, RaWMS is particularly attractive, since in RaWMS, nodes never disclose any part of their view to other nodes. Also, the views continuously evolve, making the task of identifying the initiator of a request extremely hard. Working out the exact details of this idea, analyzing the anonymity level, and bench-marking such a system is part of our future work.

Random Knowledge graph. In evaluation of RaWMS, we consider several properties

[4]The idea of a using a network of *mixers* to provide e-mail anonymity was first proposed by Chaum [1981].

of the generated random views that are important to the envisioned applications. The properties are best described using a graph-theoretic view [Jelasity et al. 2007] as follows. Define the *knowledge graph* as a directed graph, whose vertices are the network nodes, and that contains an edge from v to u if and only if u's identifier is in the view of v. If the views are truly uniform, then the graph induced by the views is actually a random graph. This framework allows us to study the *connectivity* of the knowledge graph and the *load* of an individual node (out-degree and in-degree).

In order to gain a better understanding of our generated knowledge graph, we adopt the model introduced by [Fenner and Frieze 1982] and later described in [Jelasity and van Steen 2002]. The random digraph $D_{k-in,l-out}$ is defined as follows: each vertex $v \in V$ chooses a set $in(v)$ of k random sources for edges directed into v and a set $out(v)$ of l random targets for edges directed out of v. Such a digraph is called a k-in, l-out digraph. The edges are chosen without replacement so the graph has $(k+l)|V|$ edges. When $l = 0$ we write D_{k-in}. Notice that D_{k-in} is a directed graph. The random knowledge graph generated by RaWMS is D_{k-in} (rather than a traditional Erdos and Renyi [1960] random graph, in which every edge is picked randomly, independently of other edges).

Uniformity of the views. A key feature of RaWMS, compared to other probabilistic methods like [Allavena et al. 2005; Jelasity et al. 2007], is that the distribution of node ids in the views is guaranteed to be ϵ-close to the uniform distribution (according to the definition of the total variation distance in Definition 3.1).

The sampling accuracy (the difference between the stationary distribution and the actual achieved distribution) is controlled by the RW length and is probabilistically guaranteed to differ by up to $\epsilon = \Theta(\frac{1}{n})$ from the uniform distribution, if the mixing time is set correctly. Setting of the mixing time relies on the assumption of a static random geometric graph. Even if the network graph is not a static random geometric graph, the stationary distribution of the RW remains uniform due to the regularization of the graph with self loops. However, in this case, the assumed mixing time could turn out to be insufficient. To explore deviations from this assumption we have explored different topologies, such as mobile networks, in the simulation study in Section 6. The results reported there show the uniformity of views generated by RaWMS both in static networks and under different mobility speeds.

Connectivity of the knowledge graph. The connectivity of a random graph depends on the graph model (see [Bollobas 2001] for a detailed description of various models). For example, in their classical paper Erdos and Renyi [1960] consider an undirected graph of n nodes, where an edge between each (unordered) pair of nodes is present with probability p_n, independent of other edges. They show that if $p_n = (\log(n) + c + o(1))/n$, then the probability that the graph is connected goes to $e^{-e^{-c}}$.

For $D_{k-in,l-out}$, strong connectivity with high probability is achieved if $k \geq 2$ and $l \geq 2$ [Fenner and Frieze 1982]. Dropping the orientation in such a graph results in an undirected graph $G_{(k+l)-out}$ which is naturally also connected with high probability. Therefore, the graph is connected only if a node is guaranteed to have at least 2 incoming and 2 outgoing edges.

For a directed D_{l-out} (when $k = 0$) constant out degree is not enough and one has to increase l logarithmically according to $l = c + O(\log n)$ to achieve the probability limit $e^{-e^{-c}}$ for the reachability of each vertex from a specified source as $n \to \infty$ [Jelasity and van Steen 2002; Kermarrec et al. 2003]. Although to the best of our knowledge an

equivalent result for D_{k-in} has never been published, due to symmetry considerations, we conjecture that the connectivity condition for D_{k-in} is the same as for D_{l-out}. That is, k has to be at least logarithmic to ensure strong connectivity.

Self healing from partitions. Another important property exhibited by RaWMS is a *self healing* from partitions. Since in RaWMS nodes are not restricted to communicate only with nodes in their views, even if partition in the knowledge graph does occur at some time, it will be fixed by itself after a short period of time.

View size. As we have already shown, the view size can be set independently by every node. Every node can pick its view size based on its memory constraints and applications' needs. For example, for our envisioned application of data location service the average view size should be $\Theta(\sqrt{n})$.

Distribution of the in-degrees and out-degrees in the knowledge graph. We first take a closer look at the in-degree of a given node (the number of nodes that have this node's identifier in their view) at the end of the convergence period when using the *time-based* method. We consider the period in which no identifier was already removed due to the *Timeout*. Fix some node v out of the n nodes. Let X_v be the random variable specifying the in-degree of v at the end of the convergence period. v advertises itself to $s(n)$ uniformly chosen nodes. Thus, each node has a probability of $s(n)/n$ to have v in its view. Since advertisements to different nodes are independent of each other, X_v has a binomial distribution with parameters n and $s(n)/n$. We conclude that the mean value of X_v (the mean in-degree) is $s(n)$. In order to investigate the possible deviation of X_v from its mean, we use Chernoff bounds (see Appendix B). We view X_v as the sum of n independent Bernoulli random variables $Y_1, \ldots, Y_n$, where Y_i is 1 if and only if the i-th network node advertises itself to v. By Chernoff bounds, for any $0 < \delta < 1$,

$$\Pr\left[|X_v - s(n)| > \delta s(n)\right] \; < \; 2 \cdot \exp(-s(n)\delta^2/3).$$

For example, for a value of $\delta = 0.5$, the probability for a given node to have an in-degree larger than $1.5 \cdot s(n)$ or smaller than $0.5 \cdot s(n)$ is less than $2/e^{s(n)/12}$.

By the union bound, the probability for any node to have an in-degree that differs from the average in-degree by a factor of δ is:

$$\Pr\left[\exists v : |X_v - s(n)| > \delta s(n)\right] \; < \; 2n \cdot \exp(-s(n)\delta^2/3).$$

Recall that the out-degree corresponds to the view size. Clearly, the average out-degree is $s(n)$, as expected by our construction. We can apply the same method as for the in-degree distribution and conclude that the probability that any out-degree will deviate from the mean view size is very low (the same as for the in-degrees).

For the *size-based* method the analysis of in and out degrees should take into account possible removals from the view (thus making the analysis more complicated). The maximum view size is exactly $s(n)$. The probability to have view smaller than $s(n)$ is exponentially small with $s(n)$ as well. It can be shown that the distribution of in-degrees is also highly concentrated around the mean value, with exponentially small deviation probability.

Conclusion. The view constructed by RaWMS in every node contains a random sample of nodes. Moreover, the probability for in and out degrees in the knowledge graph to deviate from their mean is very low (exponentially small with the average view size).

4.3 Reactive extension of the view

It is possible that a node will wish to extend its local view to a larger one upon its application's demand. Increasing the desired view size, $s(n)$, is a good long term solution, since it does not incur any additional communication and relies on existing advertisements. The drawback is that it may take a significant amount of time until the new target size is reached (increasing a view size by $s(n)$ will take $E(r(n)) \cdot \Delta$ time units). On the other hand, maintaining a large view size all the time may be wasteful in case such a large view is typically not required. Therefore, a method to extend the view on demand is required.

We propose two on demand RW-based methods for extending the views. The first method can be used for constructing a full membership view out of all partial memberships. A node v requesting to extend its view to a full membership, starts a RW including its current view and the estimated network size, n. Every node u that receives this message adds its view to the message while removing duplicates. If the combined view is smaller than n, u sends the combined view to one of its neighbors picked at random. Once a combined view reaches the target size n, it is sent back to v on the reverse path of the RW. Since in this method the RW path is remembered inside the message, we can further optimize the RW by preventing it from revisiting the same nodes more than once. Studying the potential performance gain of this optimization is left for future work. Let us note that in a mobile network, there is a chance that some link of the reverse path of the RW may not exist by the time it is used for sending the reply back. To overcome this problem, a unicast routing protocol should be used. Practically, this happens very rarely due to a short time proximity between the RW and the reply.

The first technique can also be used to extend a view to some bigger, yet still partial view (by sending as a target size the size of the requested extended view, $es(n)$). However, it can produce highly correlated views between nearby nodes. Therefore, we propose a different technique for an on demand extension of a view into a larger partial view. In this method, instead of collecting nodes from neighbors by one short RW, different partial memberships should be collected from different nodes, chosen uniformly at random from the network. A node v that requests to extend its view up to a size of $es(n)$, starts a number of MD random walks, each running for a number of steps equal to the mixing time. The node chosen this way returns its view on the reverse path of the RW. If the combined view at v is not enough, v initiates more RWs to sample more memberships. This technique is actually an extension of our regular sampling technique, when we sample the whole view of randomly chosen node at once.

4.4 Network size estimation

RaWMS assumes that the number of nodes in the network n is known. This is required in order to determine the length of the RW in the reverse sampling procedure (the mixing time). There are a number of methods for obtaining a loose upper bound on the network size, e.g., [Feige 1996; Servetto and Barrenechea 2002]. Once we have such a loose upper bound, we can use the birthday paradox principle to obtain a much tighter bound in the following manner. We have shown that according to the reverse sampling technique, every time RW stops at node u, it has the effect of having u pick uniformly at random a node identifier out of all n nodes. According to the famous "birthday paradox", it is well-known that after $m = \sqrt{2n}$ random trials such that each trial picks uniformly one of n distinct values, the probability to pick m distinct values is at most $\frac{1}{e}$ and it drops rapidly

as m increases ([Motwani and Raghavan 1995]). Therefore, every node can calculate the first time it receives the same advertisement again (denoted by m) and use this number to estimate n according to $n = \frac{m^2}{2}$. This process is repeated constantly and averaged across a number of measurements. In order to deal with accumulating errors, the loose upper bound should be re-used periodically and the tight bound estimation be restarted.

A recent work [Merrer et al. 2006], which was done concurrently and independently to ours, compares various algorithms for network size estimation of peer-to-peer networks. The authors report that the usage of "birthday paradox" in a manner very similar to ours renders the best tradeoff among all compared algorithms between the estimation accuracy and the associated overhead of bandwidth and computational resources.

5. GOSSIP-BASED MEMBERSHIP

5.1 The generic gossip framework

As discussed in Section 7, gossiping has been studied in the past as a way to implement partial view membership services. A generic framework for such gossip-based protocols in peer-to-peer networks has been presented in [Jelasity et al. 2007] and adapted to sensor networks in [Gavidia et al. 2005]. We have combined these two frameworks into a unified framework that is adapted to both static and mobile ad hoc networks.

The view in gossip-based membership algorithms is a set of s node descriptors, each descriptor consisting of <*NodeIdentifier*,*HopCount*,*NewFlag*>. In existing gossip-based membership protocols, the size of the view is usually assumed to be the same for all nodes, whether it is a constant or a function of n (the number of nodes in the network).

We assume that each node executes the same protocol whose skeleton is shown in Figure 2. As in RaWMS, the protocol consists of two threads: an active thread initiating communication with other nodes, and a passive thread waiting for incoming messages. The skeleton code is parameterized with three boolean parameters, namely *push*, *pull* and *NewFlag*, the desired fanout F, and three functions, namely `selectPeer`, `selectItemsToSend` and `selectItemsToKeep`.

Periodically, each node gossips with one of its neighbors to exchange the items in their views. A view is organized as a list of descriptors, ordered according to increasing hop counts. Entries with the same hop count are ordered in a random manner. We can thus meaningfully refer to the first or last k elements of a particular view. Notice that in the protocol's code, a call to `increaseHopCount`(*view*) increments the hop count of every element in a view.

The above skeleton enables us to evaluate within the same framework the important policies involved in gossip-based protocols along four dimensions: (i) peer selection, (ii) view propagation, (iii) keep selection, and (iv) send selection. By combining the possible values of each of these attributes, one can obtain many variations of gossip protocols, some of which have already been explored.

Peer selection. Periodically, each node v selects a peer in order to exchange membership information with it. This selection is implemented by the function `selectPeer`() that returns the address of a live node either in v's current view or in v's neighbors list. Below we list a few representative policies that have been mentioned in the literature.

```
do forever
  wait Δ time units;
  if push then
    // 0 is the initial hop count
    myDescriptor ← (myAddress,0,NewFlag);
    send_buff ← selectItemsToSend
          (view,myDescriptor,{})
  else
    // empty view to trigger response
    send_buff ← {}
  endif
  repeat
    v ← selectPeer();
    send (send_buff) to v
    if pull then
      receive (recv_buff) from u
      recv_buff ← increaseHopCount(recv_buff);
      view ← selectItemsToKeep(view,recv_buff)
    endif
  for F times // F is the fanout parameter
enddo
```

```
Upon receive(gossip_message,recv_buff) from u do
  recv_buff ← increaseHopCount(recv_buff);
  if pull then
    // 0 is the initial hop count
    myDescriptor ← (myAddress,0,NewFlag);
    send_buff ← selectItemsToSend
          (view,myDescriptor,recv_buff);
    send (send_buff) to v
  endif
  view ← selectItemsToKeep(view,recv_buff)
enddo
```

Fig. 2. A Generic Gossip Framework

`rand`	Uniform randomly select an available node from the view
`head`	Select the first node from the view (the one with the lowest hop count)
`tail`	Select the last node from the view (the one with the highest hop count)
`neighbor`	Randomly select an available node from the neighbors list
`broadcast`	Select all nodes in the neighbors list to send a broadcast message to them

View propagation. Once a peer has been chosen, there are several alternatives to exchanging information with that peer, as listed below.

`push`	The node sends its view to the selected peer
`pull`	The node requests the view from the selected peer
`pushpull`	Both the node and the selected peer exchange their respective views

Send selection. Once the peer and the way to contact it have been chosen, the sender must decide what information to send. The options that have been discussed in the literature are listed below:

`rand`	Randomly select up to X descriptors from the view
`head`	Select the first X descriptors from the view (the ones with the lowest hop count)
`tail`	Select the last X descriptors from the view (the ones with the highest hop count)
`new`	Pick all descriptors that have been received for the first time
`full`	Send all s descriptors of the view

Keep selection. Once the membership information has been exchanged between peers, the received descriptors should be integrated into the node's view. The integration procedure must adhere to the target size limit of s descriptors by choosing only the subset of all available descriptors. In the protocol above, this is done by the `merge`(*view*,*recv_buff*)

procedure, which merges the received view with the current one. In case a descriptor appears in both views, the merged view takes the version with the most up to date timestamp. The function `selectItemsToKeep`(*view*,*recv_buff*) selects a subset of at most s elements from merged views (ordered by increasing hop count) according to one of the policies listed below:

`rand`	Merge and randomly select s elements without replacement from the merged view
`head`	Merge and select the first s elements from the merged view
`tail`	Merge and select the last s elements from the merged view
`shuffle`	Merge and remove the elements that were sent in this data exchange to the other node until only s elements remain in the view

It is possible to obtain a large selection of gossip protocols by simply plugging any of the above policies in the skeleton protocol of Figure 2. Each combination is expressed by means of a 4-tuple (*peer selection, view propagation, send selection, keep selection*). In particular, various combinations of the above policies were investigated in [Jelasity et al. 2007]. One of the conclusion of [Jelasity et al. 2007] is that no gossiping algorithm succeeds in constructing views that form a truly random knowledge graph. Typically, the resulting knowledge graph induced by the view's edges has great resemblance to a small-world graph.

Speeding up the joining process with *NewFlag*. We can use the same technique here as in RaWMS in order to speedup the joining process of a new node. That is, a new node increases the rate of gossiping until it has managed, with high probability, to distribute its identifier to enough random nodes in the network. Note that in gossiping algorithms, other nodes must also gossip the identifiers of newly joined nodes more frequently than the standard gossip frequency. For that purpose, during a fixed period of *NewTimeout* time units, a new node v turns on the *NewFlag* flag of its descriptor each time v gossips its descriptor. When a node receives a gossip descriptor with the *NewFlag* flag turned on, it increases its own gossip rate for a duration of *NewTimeout* time units. As a result, when a new node joins the network, the gossip rate at all "infected" nodes is increased and the new identifier is gossiped faster. After *NewTimeout* time units elapse, the gossip rate returns to $1/\Delta$ in order to reduce the communication overhead.

5.2 Specific gossip methods

lpbcast. lpbcast [Eugster et al. 2003] corresponds to (`rand, push, full, rand`). In each round every node v sends its view to F (the fanout parameter) nodes, chosen randomly from v's view. The number of rounds is logarithmic in the network size. In order to establish a communication path between two nodes in an ad hoc network, some routing algorithm must be employed. Since destinations are chosen randomly among the network nodes, the number of network level messages required to send a single gossip message is equal to the average path length of the network.[5] The average path length in an ad hoc network is in the order of the diameter of the network divided by the transmission range. In our case, this amounts to $O(\sqrt{\frac{n}{\log n}})$. Also, in each gossip message, the entire view

[5]More precisely, v chooses F random nodes from its view. However, the view gradually converges to a random sample.

is sent. Therefore, the total communication overhead of lpbcast for a view of size s is $n \cdot \sqrt{n \log n} \cdot F \cdot s$.

The main drawback of lpbcast, which makes it unsuitable for ad hoc networks, is the extensive usage of unicast routing. Since each node sends messages to random network nodes, lpbcast uses $F \cdot \log(n)$ routes in the initial convergence stage and keeps utilizing more routes afterwards. Notice that the establishment of a unicast route if often obtained through flooding, which is costly in an ad hoc network. Since the potential number of source destination pairs is quadratic, lpbcast's traffic pattern virtually establishes all-to-all routing paths over time, which are created merely for lpbcast's usage and are not necessarily used by the application. Those routes break over time due to nodes mobility, which adds the cost of repairing them.

Another drawback of lpbcast is that according to [Jelasity et al. 2007], lpbcast fails to provide uniform views. In addition, the views at the same node but in different rounds are not truly independent, since nodes gossip at round $t + 1$ only with nodes they had in their view in round t. As a result, it was shown in [Jelasity et al. 2007] that lpbcast has a non-negligible chance of partitioning. When partitions do occur in lpbcast, or any other similar gossip algorithm, they cannot self-heal.

Shuffling. Shuffling [Gavidia et al. 2005; Voulgaris et al. 2005] corresponds to (`neighbor, pushpull, rand, shuffle`). Shuffling was first introduced in the context of sensor networks and originally used for information dissemination. Yet, shuffling can also be used for construction of random views, by disseminating nodes identifiers, as was done in [Voulgaris et al. 2005] for peer-to-peer networks. In shuffling, a node communicates only with its direct neighbors. Every round each node randomly picks B identifiers out of its view and shuffles them with its randomly chosen neighbor. The main idea of Shuffling is that unlike other gossiping algorithms, Shuffling avoids loss of data during items exchange. This is accomplished by having the peers agree on which data items will be kept by each of them after the exchange takes place. Any two nodes that engage in a shuffle essentially swap a number of items. In doing so, they "move" the data around in a seemingly random fashion.

We analyze the performance of shuffling by adapting some RW techniques to it. In the following analysis, let us assume that each node already possess a random view. We are interested in determining the number of rounds and the number of messages required for a new node joining the system to incorporate its identifier uniformly into the views of other nodes in the system.

Every round each node randomly picks B identifiers out of its view and shuffles them with its randomly chosen neighbor. Since the views are random, when two nodes shuffle, they pass to each other almost completely different sets of identifiers. Therefore, almost all ids that node v passes to node u will migrate to u and will be removed from v's view. In this process, ids already present in the network are not discarded, and almost never duplicated. This view exchange process has some resemblance with RWs; each identifier traverses the network from one node to its randomly chosen neighbor. However, there are a number of differences: 1) in shuffling, the "walks" of different identifiers are not independent since an exchange is performed on a batch of B identifiers, 2) an identifier may not be passed to a neighbor node every round, since only B identifiers out of the entire view are exchanged every round.

The first difference can be controlled by the size of the exchanged batch, B. Large

values of B indeed increase the dependence between disseminations of different identifiers. However, for small values of B, the effect of dependence is not significant, especially since in every round each node picks a different set of B ids from its view for an exchange. Indeed, shuffling is usually run with small, constant B.

As for the second difference, we can measure the *pause time*, i.e., the average amount of time that each identifier spends in the view before being shuffled. If the whole view is shuffled, the pause time is zero. If only B identifiers out of the entire s (the view size) are shuffled every time, the pause time is a geometric random variable with a mean of $\frac{s}{B}$. Therefore, the number of rounds until an arbitrary identifier reaches a random place (assuming no duplications and discarding and fanout 1) is $\frac{s}{B} \cdot T_{\text{actual\_mix}}$, where $T_{\text{actual\_mix}}$ is the actual mixing time of a RW of the underlying graph. Since we are interested in a situation when s random nodes have the identifier of the new node in their view, a new node must publish itself s times, once in each successive round. This yields a total of $s + \frac{s}{B} \cdot T_{\text{actual\_mix}}$ rounds until convergence.

Flooding. Flooding can be used to implement a membership service by having each node flood the network with its identifier. An efficient implementation of flooding requires memory which is linear in the number of nodes in the system. That is, in order to prevent a node from delivering (and retransmitting) the same message more than once, a node should remember the identifiers of the last few broadcast messages initiated by every other node. Since the implementation of flooding itself requires linear memory space, there is no point in limiting the view to include fewer than n identifiers.

5.3 Probabilistic starvation

One of the main usages of partial membership services is in gossip-based probabilistic multicast algorithms. Specifically, these algorithms attempt to deliver every message to almost every node with high probability. The percentage of nodes that receive a message is called the *reliability factor*. However, those algorithms usually make no attempt to provide reliability for a single node. When the views are not truly random, there is a possibility that while most nodes receive all messages, a small number of nodes do not receive messages at all or receive only a small fraction of all messages. In particular, if there are some nodes (e.g., low degree nodes) that are not uniformly distributed among other nodes' views, those nodes will be constantly denied messages and thus suffer from *probabilistic starvation*. On the other hand, views constructed by RaWMS are proven to be uniform and therefore any probabilistic multicast algorithm built a top of it will not suffer from such a phenomenon.

5.4 Comparison

Table I shows an asymptotic comparison of all the methods mentioned above based on the theoretical analysis. Table II shows an exact comparison with constants based on the simulation results (taken from Section 6 below) for view size $\sqrt{n}$. The tables compare the time and the communication complexity of the convergence period in static networks. The maintenance cost for each method is the communication cost during the convergence period divided by the convergence time. An interesting observation about this comparison is that the message complexity of lpbcast does not depend on the view size.[6] On the other

[6]The view size only affects the bit complexity of the protocol. However, in most networks, as long as messages are not too large, the number of messages dominate the performance limitations of the protocol.

	#rounds	time of a round	total time	msgs per round	total msgs sent	msg size	com. overhead	mem. overhead	additional overhead
RaWMS	$r(n)$	$T_{\text{actual\_mix}}$ = n	$n + r(n)$	n^2	$n^2 \cdot r(n)$	1	$n^2 \cdot r(n)$	view size s	
lpbcast	$\log n$	$\sqrt{\frac{n}{\log n}}$ Av. path length	$\sqrt{n \log n}$	$n \cdot \sqrt{\frac{n}{\log n}} \cdot F$	$n \cdot \sqrt{n}$ $F\sqrt{\log n}$	view size s	$n \cdot \sqrt{n \log n} \cdot F \cdot s$	memory for routing	Unicast routing
Shuffling	$\frac{s}{B} T_{\text{act\_mix}}$ $= \frac{s}{B} \cdot n$	1	$n\frac{s}{B}$	$2n$	$2n^2 \frac{s}{B}$	B	$2n^2 s$	view size s	
Flooding	1	$\sqrt{\frac{n}{\log n}}$ Av. path length	$\sqrt{\frac{n}{\log n}}$	n^2 broadcasts	n^2 broadcasts	1	n^2	linear memory for flooding	Memory for flood.

Table I. Comparing RaWMS with gossip-based membership and flooding - based on theoretical analysis

	#rounds	time of a round	total time	msgs per round	total msgs sent	msg size	com. overhead	mem. overhead	additional overhead
RaWMS	$\sqrt{n}$	$T_{\text{actual\_mix}}$ = $\frac{n}{4}$	$\frac{n}{4}$	$\frac{n^2}{4}$	$\frac{n^2 \cdot \sqrt{n}}{4}$	1	$\frac{n^2 \cdot \sqrt{n}}{4}$	view size $\sqrt{n}$	
lpbcast	$4 \log n$	$\sqrt{\frac{n}{\log n}}$ Av. path length	$4 \cdot \sqrt{n \log n}$	$3n \cdot \sqrt{\frac{n}{\log n}}$	$12n \cdot \sqrt{n} \cdot \sqrt{\log n}$	view size $\sqrt{n}$	$12n^2 \cdot \sqrt{n} \cdot \sqrt{\log n}$	memory for routing	Unicast routing
Shuffling	$\frac{s}{B} T_{\text{act\_mix}}$ $= \frac{n\sqrt{n}}{4B}$	1	$\frac{n\sqrt{n}}{4B}$	$2n$	$\frac{n^2\sqrt{n}}{2B}$	B	$\frac{n^2\sqrt{n}}{2}$	view size $\sqrt{n}$	
Flooding	1	$\sqrt{\frac{n}{\log n}}$ Av. path length	$\sqrt{\frac{n}{\log n}}$	n^2 broadcasts	n^2 broadcasts	1	n^2	linear memory for flooding	Memory for flood.

Table II. Comparing RaWMS with gossip-based membership and flooding - constants are based on simulation results for view size $\sqrt{n}$

hand, in RaWMS and Shuffling, the message overhead for the duration of the convergence period depends on the view size (since the length of the convergence period depends on the view size). Thus, had we taken a smaller view size, such as $\log n$, it would have placed RaWMS and Shuffling in a further advantage over lpbcast.

Note that when nodes are mobile, there is an additional cost due to routing. In particular, lpbcast is highly affected by mobility since it relies heavily on unicast routing. When nodes move, routes break and must then be reestablished or repaired. In contrast, neither RaWMS nor shuffling suffer due to mobility, since they do not use multi-hop routing. In fact, in these two approaches, nodes' mobility can actually facilitate faster and more random dissemination of membership information.

6. SIMULATIONS

6.1 Setup

The simulations were performed using the JiST/SWANS simulator [Barr et al.] from Cornell university. Nodes use two-ray ground radio reflection model as the radio propagation model, with IEEE 802.11 MAC protocol and 1Mb/sec throughput. The multi-hop routing protocol used by lpbcast is AODV (recall that RaWMS does not use routing at all). Mobility was modelled by the Random-Waypoint model [Johnson and Maltz 1996]. In this model, each node picks a random target location and moves there at a randomly chosen speed, chosen from a given range. The node then waits for a random amount of time and then chooses a new location, etc. We have used 3 ranges of the speed of movement: 0.5-2 m/s, 2-5 m/s and 5-10 m/s. The speed range of 0.5-2 m/s corresponds to a walking speed and unless stated differently was used as a default speed range in our simulations, while the speed ranges of 2-5 m/s and 5-10 m/s were investigated in Section 6.2.3. An average pause time is 30 seconds. All simulations were performed on networks of 10, 50, 100, 200, 400 and 800 nodes. We have used the default Java pseudo random number generator, initialized with the current system time in milliseconds as a seed.

The nodes were placed at uniformly random locations in a square universe.[7] The transmission range was fixed for all network sizes and all simulations at 200m. The size of the simulation area was scaled in order to comply with the analytical results of Gupta and Kumar [1998] regarding the critical transmission range. For a square area a^2 the radius of the transmission range is $r = a\sqrt{\frac{C \ln n}{n}}$, $r \in [0, a]$. The average number of nodes in the transmission range of any node (network density) was set to $d_{\text{avg}} = 3 \ln n$ ($d_{\text{avg}} = \frac{\pi r^2 n}{a^2} = \frac{\pi a^2 \frac{C \ln n}{n} n}{a^2} = \pi C \ln n = 3 \ln n$ for $C \approx 1$). That is - we kept a constant transmission range and scaled the simulation area a^2 for n nodes according to $a^2 = \frac{\pi 200^2 n}{3 \ln n}$. By proposition 3.3, for such d_{avg}, $d_{\max} \approx 2 d_{\text{avg}}$. Additional densities were studied in Section 6.2.2.

Each simulation lasted 1,000 seconds (of simulation time) and each data point was generated as an average of 10 simulation runs. Simulations started after a 60 seconds initialization period, which was enough to construct one hop neighborhood information. The neighbors discovery protocol was running throughout the entire simulation period in all scenarios. RaWMS was run with a *time-based* method; the node's descriptor timeout in the view was set so that the average view size will be $\sqrt{n}$. In each scenario of RaWMS, each

[7] We run our simulation on a flat topology rather than a torus. This places our scheme in a slight disadvantage, since the communication graph tends to be less uniform in a flat topology.

node started $E(r(n))$ RWs, calculated out of the expected view size of $\sqrt{n}$ as described in Section 4.1. These advertisements were spread over the whole simulation period.

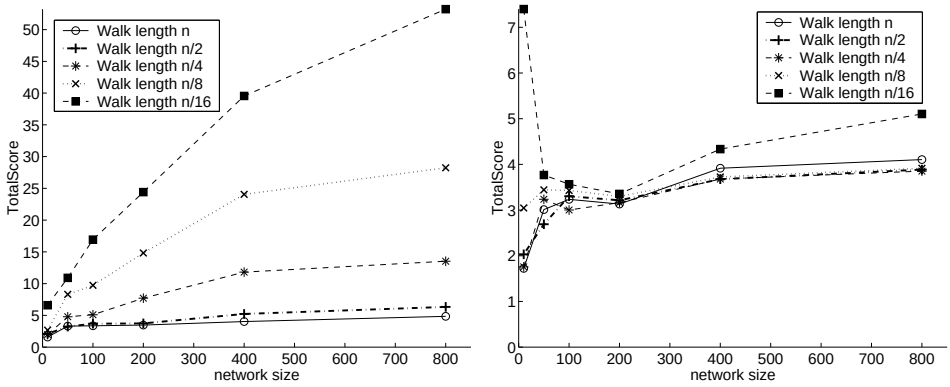
6.2 Properties of RaWMS

6.2.1 *Uniformness.* We have performed a number of tests to compare the views constructed by RaWMS with the ideal uniformly sampled views. These tests were picked to reflect the most important structural properties of the system: distribution of the path lengths from every node to all nodes in its view, dependence between views of neighbors in the ad hoc network, clustering coefficient of the knowledge graph, view size distribution, and connection between a node's degree (in the ad hoc network) and its view size. Since it is not possible to empirically prove the uniform randomness of the views, these statistical tests are used to strengthen our claim that the properties of the constructed views do not deviate from the properties of the theoretical uniform samples. The measures are explained and studied below.

Path Length Distribution. The first measure we used to evaluate RaWMS is the uniformness of the locations of nodes appearing in the views. That is, for each node v and corresponding view $\mathcal{V}$, we compare the ratio of nodes in $\mathcal{V}$ that are at a given distance from v to the ratio of such nodes in the real random network. If there is a strong match between these ratios for all views and distances, this gives a positive indication about the uniformness of the views created by RaWMS.

To this end, we used a χ^2 statistical test to compare the distribution of nodes in the view of every node at the end of the convergence period with the desired uniform distribution. Namely, we have partitioned all nodes into a number of bins according to their distance from the tested node. For every node v we have calculated the following score: $PathScore_v = \sum_{j=1}^{\#bins} \frac{(Actual_{v,j} - Expected_{v,j})^2}{Expected_{v,j}}$, $Actual_{v,j}$ being the actual number of nodes from distance j found in the view of node v and $Expected_{v,j}$ is the number of nodes from distance j that are expected to be found in the view of node v. The total network score $PathScore$ corresponds to the average of all $PathScore_v$s. Thus $PathScore$ is a statistical test for the difference between the distribution of path lengths obtained by simulations and the assumed uniform distribution (in a perfect uniform sample, each view should include a number of nodes at each distance that is proportional to the actual number of nodes at this distance in the network).

The results of the path length distribution test for static networks are depicted in Figure 3(a). The simulations were run with 5 different lengths of the random walk, corresponding to 5 different candidates for the mixing time, T_{mix}. Clearly, the longer the walk is, the closer is the distribution reached by the RW to the uniform stationary distribution, since a long walk has a "better" chance to reach a random node. We can see that for lengths of n and $n/2$ the $PathScore$ is relatively low and almost does not change as the number of nodes grows. This means that walks of $n/2$ steps are long enough to correspond to the mixing time of those networks. Shorter walks exhibit a dramatic degradation in the test's score. Those walks are shorter than T_{mix} and do not have enough steps to reach the uniform stationary distribution. The larger the network is, the worst are the results of these short RWs, since they do not get a chance to move far away from the originating node. As a result, every node ends up with relatively more nodes in its view that are geographically closer to it and with fewer nodes that are geographically far from it. This confirms the theoretical result that too short RW in static networks will not converge to a stationary



(a) Static network (b) Mobile network

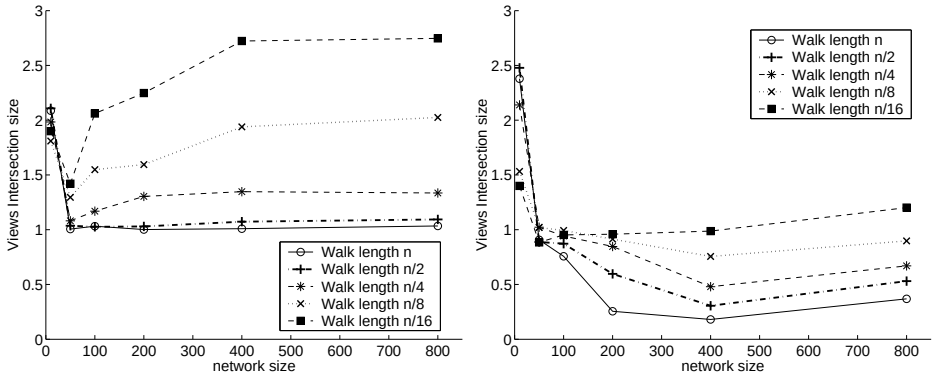
Fig. 3. RaWMS - the path length distribution test ($PathScore$ versus n)

distribution.

Figure 3(b) presents the results of our simulations with mobility. Interestingly, the random dissemination of membership information is actually improved by nodes movements, and even RWs of length $n/8$ get the same results as with length n. Nodes that used to be close to some node in the initial stage of the algorithm may end up in a completely different location in the network after some time, helping the "mixing" effect of the RW. Still, as can be seen from the graph, very short walks of length $n/16$ obtain worse results even with mobility. Also, notice that due to the salvation technique employed by RaWMS (if a node did not receive a MAC level ACK for a RW message, it sends this message to another random neighbor within the same step), RW messages are almost never dropped, even in mobile networks.

Intersection between views of neighboring nodes. In this test we have checked the amount of correlation between the views of (physically) neighboring nodes. For ideal uniformly chosen views there should not be any special correlation between the views of neighboring nodes. We have measured the average size of the intersection between the views of all pairs of neighboring nodes and compared it with a theoretically expected intersection. Since the average view size is $\sqrt{n}$, the expected intersection is $\sqrt{n}\frac{\sqrt{n}}{n} = 1$, for all network sizes. It can be seen from Figure 4(a) that indeed in static networks for long enough RWs (walks of length n and $n/2$) the average intersection size is very close to an expected one. However, walks shorter than the mixing time do not have enough steps to get far away from the originating node and tend to stop at its proximity instead of at a random node. As a result the neighbors of an originating node have a greater chance to have it in their views.

In mobile networks intersection between views of neighboring nodes is greatly reduced. Here, even short RWs can get a chance to escape the proximity of its originating node, due to mobile nodes carrying the RW message. Surprisingly, in mobile networks, the intersection is even smaller than expected. The reason for this is as follows. A fast moving mobile node v has a lower chance of getting a RW message, because if v passes next to a node u that has the RW message, v disappears from the transmission range of u before the neighbors discovery protocol at u detects v. The result is that long RWs tend to stop



(a) Static network (b) Mobile network

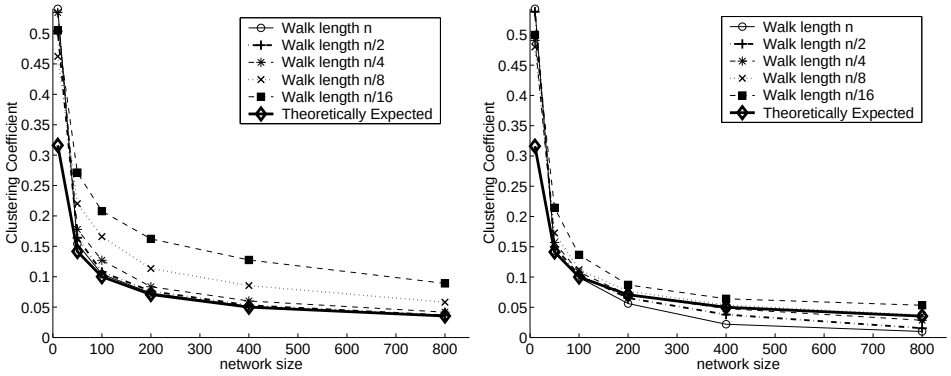
Fig. 4. Intersection between views of neighboring nodes

at static and slow nodes, that are surrounded by fast moving mobile nodes and usually are not neighbors of each other. Therefore, the intersection between views of static nodes with their fast moving neighbors is very small. This phenomenon becomes even worse in long RWs, since the longer the RW, the greater is the chance that it will be "stuck" at a static or slow moving node. The situation could be improved by a more aggressive and frequent neighborhood discovery protocol. These results also suggest that the length of RWs should be adjusted in reverse proportion to the observed mobility in the network.

Clustering Coefficient of the knowledge graph. A common measure for the uniformness of random graphs is their clustering coefficient [Watts and Strogatz 1998]. The clustering coefficient for a node v represents the probability that two neighbors of v will also be neighbors of each other. Hence, a graph with good uniformness will have a low clustering coefficient. Notice, however, that clustering coefficient alone, as being a statistical test, is not enough. For example, it only refers to the knowledge graph induced by the nodes, yet ignores the relationship between views of physical neighbors. The latter is covered by our measure above for the intersection between views of (physically) neighboring nodes.

The clustering coefficient of a node v is defined as the number of edges between the neighbors of v divided by the number of all possible edges between those neighbors. Intuitively, this coefficient indicates the extent to which the neighbors of v are also neighbors of each other. The clustering coefficient of the graph is the average of the clustering coefficients of the nodes, and always lies between 0 and 1. Figure 5 depicts the clustering coefficient of the knowledge graph induced by RaWMS compared to the theoretical clustering coefficient of a random graph (which is equal to the probability of existence of a link between any pair of nodes and equals $\frac{1}{View\_size} = \frac{1}{\sqrt{n}}$ in our case). It can be clearly seen that in static networks, the clustering coefficient for walks of length $n/4$ and longer closely follows the theoretically expected one (except for the small networks of 10 nodes). In dynamic networks, clustering coefficient behaves as expected even for shorter RWs.

View size distribution. Recall that the size of the view is a binomial distributed random variable with probability $\frac{s(n)}{n}$, mean value $s(n)$ and variance $s(n)(1 - \frac{s(n)}{n})$. We have compared those theoretically expected values with the actual mean and variance values of



(a) Static network (b) Mobile network

Fig. 5. Clustering coefficient

view sizes at the end of the convergence process.

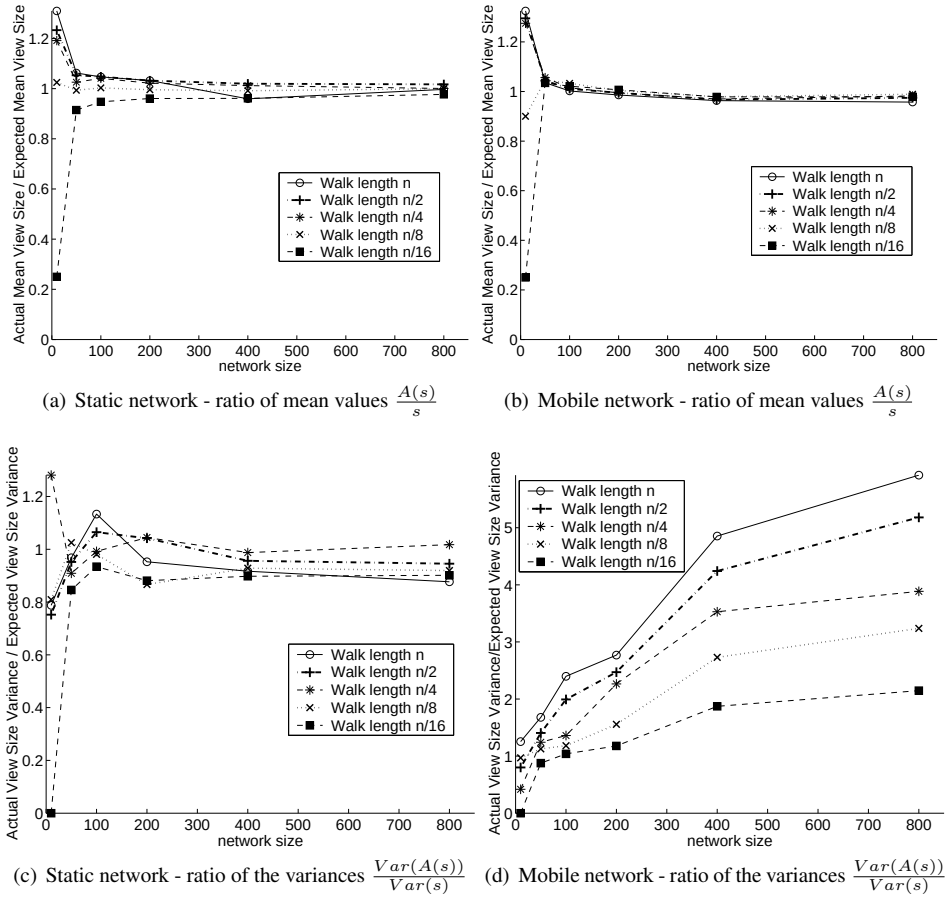
Figures 6 presents the graphs for $\frac{A(s)}{s}$ and $\frac{Var(A(s))}{Var(s)}$, with s representing the expected view size, $A(s)$ the actual mean view size, $Var(s)$ the expected variance, and $Var(A(s))$ the actual variance, calculated as $\frac{\sum_{i=1}^{n}(A(s)-view_i)^2}{n}$. For all network sizes and for all walk lengths, in both static and mobile networks, the average size of the view is almost equal (typically up to 90%) to the ideal, theoretically expected mean size. Only for small networks the mean view size is a bit larger than expected, due to the fact that for s of the order of n, $\frac{ns}{n-s}$ is not a tight bound of $E(r(n))$ (see Lemma 4.1). In these cases, nodes simply start too many RWs. The variance of the view sizes is also very close to the expected one in static networks, presenting another evidence to the fact that the view size is a binomial distributed random variable. The only exception is a small network of 10 nodes. For very short walks ($n/16$), the RWs did not get a chance to walk even a single step and the resulting view includes only the node itself. The variance is zero in such a case.

Notice that in mobile networks the variance is larger than in static networks. The variance is even larger for long RWs than for short RWs. This can be explained in the same way as with intersection between views of neighboring nodes. Long RWs tend to stop at static and slow nodes. As a result these nodes have a much larger view, at any given point in time, then fast moving nodes. The situation could be improved by a more aggressive and frequent neighborhood discovery protocol.

Correlation between node degree and view size. Additional tests were conducted to measure the correlation between nodes' degrees (the number of neighbors in the ad hoc network) and view sizes.

Figure 7 shows the distribution of view sizes accumulated into bins according to node degrees. The nodes were sorted by degree and then separated into 10 deciles, each containing 10% of the nodes. For each decile, the bar chart shows the ratio between the average view size of this decile and the average view size of the whole network. The results in Figures 7 and 8 were generated for walk length $n/2$. The same results were observed for other walk lengths both in static and mobile networks.

As stated before, the stationary distribution of a RW without self loops is degree-dependent,



(a) Static network - ratio of mean values $\frac{A(s)}{s}$

(b) Mobile network - ratio of mean values $\frac{A(s)}{s}$

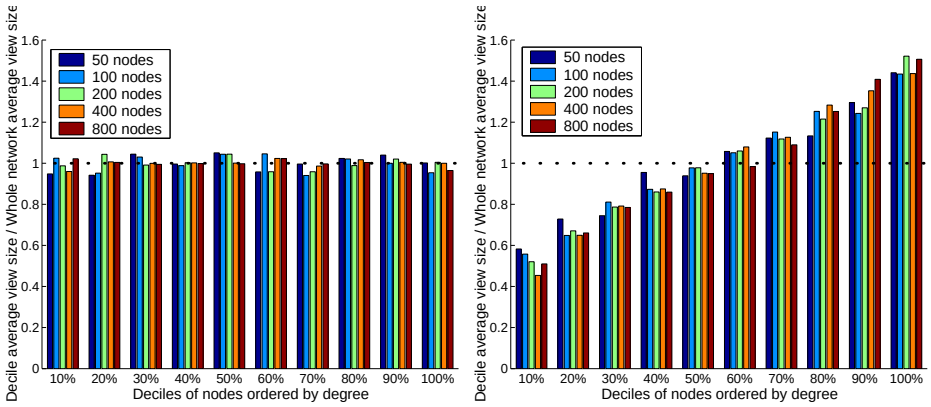
(c) Static network - ratio of the variances $\frac{Var(A(s))}{Var(s)}$

(d) Mobile network - ratio of the variances $\frac{Var(A(s))}{Var(s)}$

Fig. 6. View size distribution - the difference between actual and expected mean and variance values

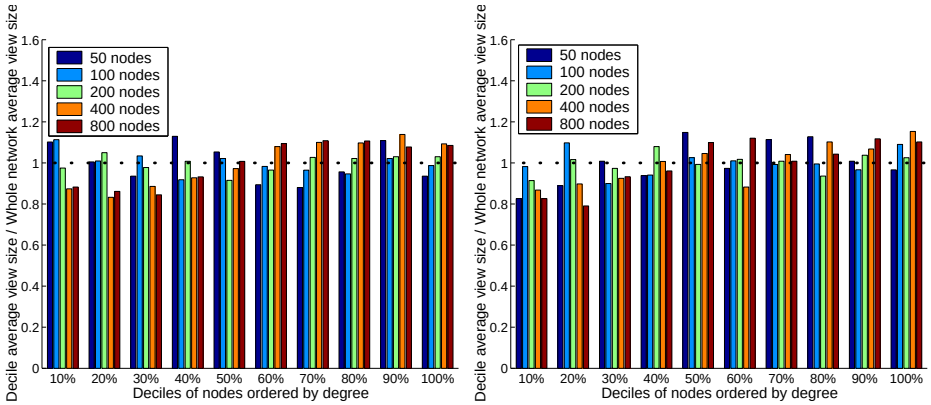
resulting in more RWs stopping at higher degree nodes. Indeed, it can be seen from Figure 7(b) that there is a significant bias towards high degree nodes - much more RWs stop at these nodes than at lower degree nodes, resulting in unbalanced view sizes. On the other hand, our Maximum Degree RW balances the node degree with self loops, generating a regular graph on which the RW has a uniform stationary distribution. Indeed, Figure 7(a) demonstrates that there is no bias towards high degree nodes and that the views have almost the same average size for all deciles.

The results for mobile networks are depicted in Figure 8. The results for RWs with self loops are essentially the same as in static networks: there is no bias towards high degree nodes. On the other hand, in a mobile network, RWs without self loops have very little bias as well. This is since in a mobile network the neighborhood of a node changes frequently. During the run every node has different degrees, and all nodes have approximately the same degree averaged over the whole simulation time. Therefore, we can note here that again mobility assists in introducing uniformity into the RW.



(a) RW with self loops - static network (b) RW without self loops - static network

Fig. 7. Correlation between node's degree and its view size. Static network, walk length $n/2$

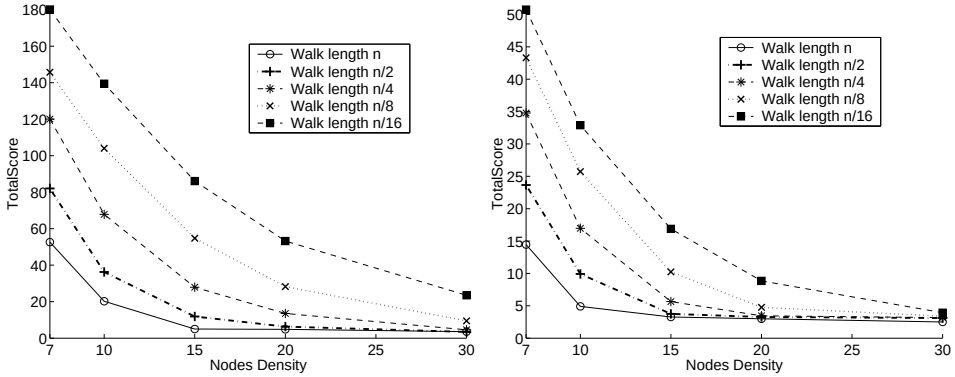


(a) RW with self loops - mobile network (b) RW without self loops - mobile network

Fig. 8. Correlation between node's degree and its view size. Mobile network, walk length $n/2$

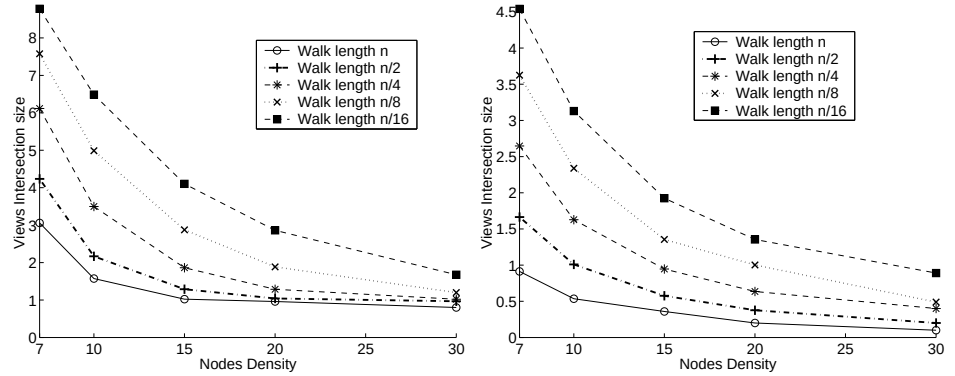
6.2.2 *Nodes density.* This section studies the performance of RaWMS in networks with varying nodes density. In these simulations we have changed the average number of neighbors to be 7, 10, 15, 20 and 30 (which corresponds to $d_{\text{avg}} = C \ln n$, for $C = 1, 1.5, 2.2, 3, 4.5$). We depict only the results for networks of 800 nodes. For other network sizes the qualitative effect of the results was the same, however as 800 nodes is our biggest network, it depicts the general trends of the results in the best way. Note that 7 neighbors is the smallest density which results in the connected network (even for 7 neighbors there are some individual nodes that may sometimes be disconnected, but their number is negligible).

We can see in Figure 9 the path length distribution test results as a function of network density. The denser the network is, the smaller is the $PathScore$, meaning that the views



(a) Static network (b) Mobile network

Fig. 9. Path length distribution test ($PathScore$) as a function of varying density



(a) Static network (b) Mobile network

Fig. 10. Intersection between views of neighboring nodes as a function of varying density

are more uniform. The same trend can be seen in Figure 10, depicting the intersection between views of neighboring nodes. The intersection is smaller for denser networks. We have not depicted the results of clustering coefficient and view size distribution (both average and variance) since the results were not affected by density and were very similar to the results in Figures 5 and Figures 6.

We can observe in both Figures 9 and 10 the effect of the average number of neighbors on RaWMS. The smaller the transmission radius is (resulting in smaller d_{avg}), the bigger the network diameter becomes and as a result a single RW has to walk more steps to reach the stationary distribution. Thus, a longer mixing time is needed to reach a uniform distribution of nodes in the views constructed by RaWMS. This complies with the general result of our analysis in Theorem 3.4. It also matches intuition, since the denser a network is, the closer it is to a clique, and hence its mixing time should be shorter. In mobile networks the same phenomenon can be seen. However, quantitatively, uniformity is achieved with much shorter mixing times.

6.2.3 *RaWMS in fast and medium speed mobile networks.* This section studies the performance of RaWMS in fast and medium moving mobile networks. In these simulations, we employed the Random Waypoint model with movement speeds ranging in 2-5 m/s, which corresponds to running, and with speeds ranging in 5-10 m/s, which corresponds to urban traffic or VANET networks. The average pause time was set to 30s.

According to the path length distribution test, we can see in Figures 11(a) and 11(b) that when the speed of movement is medium or fast, uniform path length distribution is achieved with relatively short RWs. Even a RW of a couple of steps (walk length of $n/64$) achieves a very good score. This is clearly due to the high rate of change in the network.

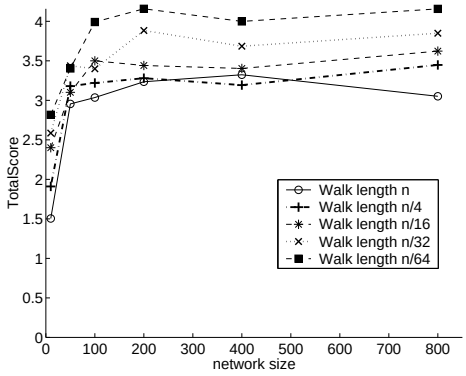
The average view size is very close to the theoretically expected view size and was therefore not depicted. The variance in the view sizes is depicted in Figures 11(c) and 11(d). We can see the same behavior as in Figure 6(d) for slow moving networks. The variance becomes larger for longer RWs than for shorter ones, due to the fact that moving nodes have a lower chance of getting any RW message. Recall that this is because the neighbors discovery protocol is not fast enough to notice their short presence. Hence, fast moving nodes have smaller views than slower (and static) nodes. However, this phenomenon is reduced in fast and medium moving networks compared to slow moving networks. In fast networks, the dynamics is so high and sporadic that almost all nodes are equally likely to be both fast and slow during the simulation time. Since those differences are averaged along the whole simulation period, the view size variance is reduced.

The intersection between the views of neighboring nodes is depicted in Figures 11(e) and 11(f) and resembles the same behavior as in Figure 4(b). The intersection is close to the optimal of one (of size 1) for short RWs and is even smaller for long RWs. This is again a result of the difference between fast and slow moving nodes. However, faster mobility does not assist in this case as with view size variance. This is due to the fact that we measure the intersection between views at the end of the simulation period, at which point the differences between fast and slow nodes are significant. Therefore, when it comes to view intersection, the effect of heterogeneity between fast and slow nodes is not averaged along the whole simulation period.

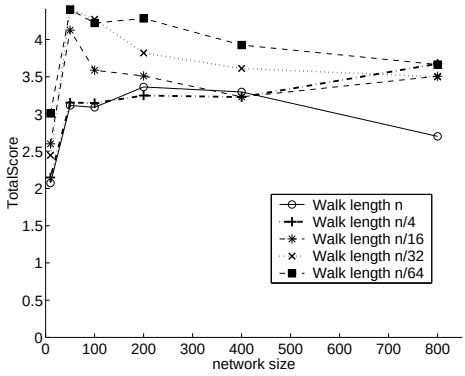
A general observation from all the simulations we have conducted for fast and medium speed networks is that generally speaking, mobility greatly assists in uniform membership dissemination. Yet, long RWs tend to have negative influence on the uniformity of some distribution properties (increased view size variance and decreased intersections). The conclusion is that the length of RWs should be set inversely proportional to the mobility level.

Another interesting phenomenon that was observed during simulations of fast moving networks with long RWs is as follows: since the neighborhood discovery protocol is not fast enough to detect frequent neighborhood changes, often an attempt is made to pass a RW to a neighbor that is no longer present in the sender's proximity. Consequently, the MAC protocol makes several attempts to send the message and gives up only after all attempts have failed (recall that due to the salvation technique of RaWMS, such a RW will not be lost and an attempt will be made to pass it to another neighbor). In the meanwhile, additional messages arrive and wait at the IP level queue to be passed to the MAC protocol, which is still busy with the previous message. This results in congestion and reduced bandwidth.

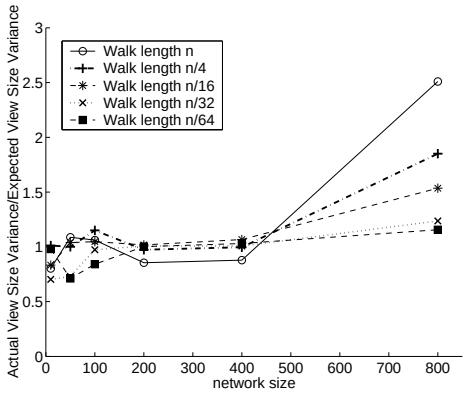
Consequently, we believe that in fast networks, a somewhat different approach for RW



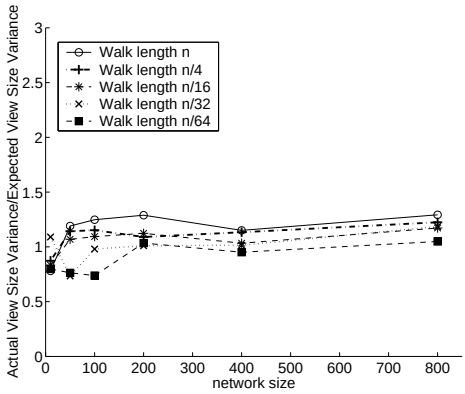
(a) Path length distribution - 2-5 m/s Mobile network



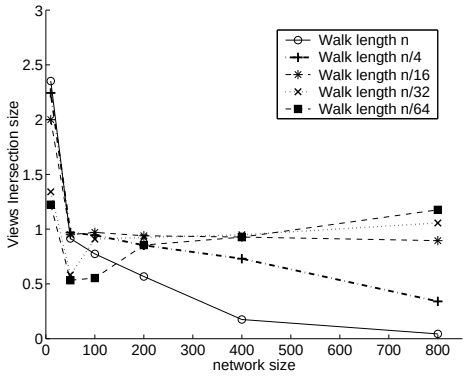
(b) Path length distribution - 5-10 m/s Mobile network



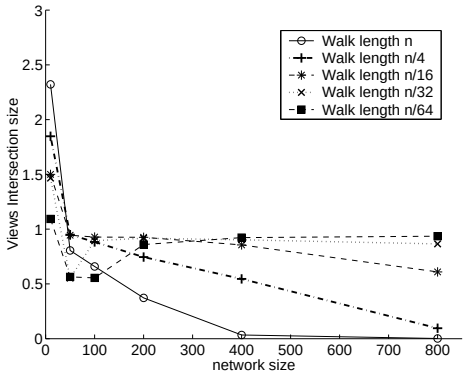
(c) View size distribution - ratio of the variances $\frac{Var(A(s))}{Var(s)}$ - 2-5 m/s Mobile network



(d) View size distribution - ratio of the variances $\frac{Var(A(s))}{Var(s)}$ - 5-10 m/s Mobile network

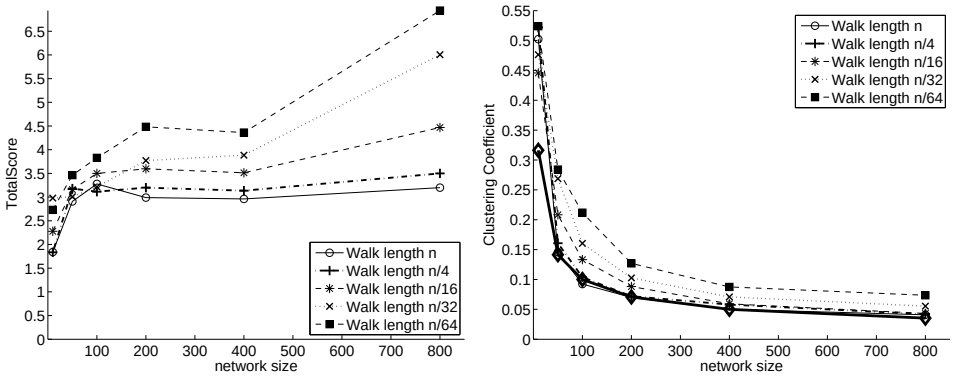


(e) Intersection between views of neighboring nodes - 2-5 m/s Mobile network



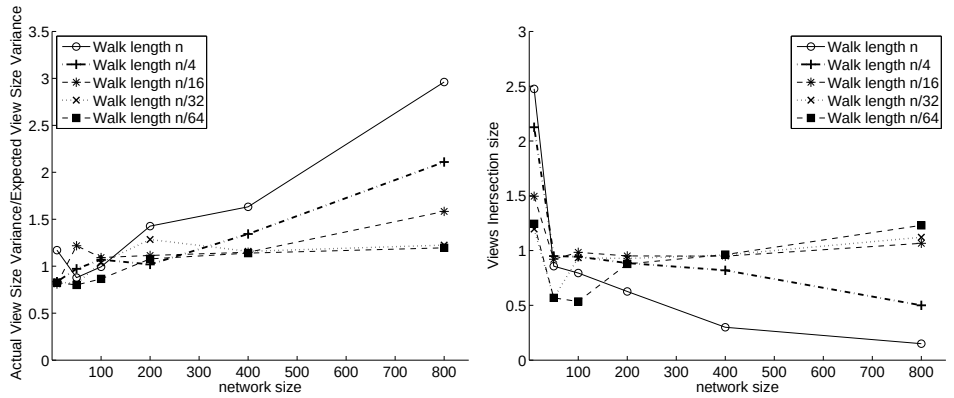
(f) Intersection between views of neighboring nodes - 5-10 m/s Mobile network

Fig. 11. Fast and medium speed mobile networks: Path length distribution, View size distribution and Intersection between views of neighboring nodes



(a) Path length distribution - Heterogeneous Mobile network

(b) Clustering coefficient - Heterogeneous Mobile network



(c) View size distribution - ratio of the variances $\frac{Var(A(s))}{Var(s)}$ - Heterogeneous Mobile network

(d) Intersection between views of neighboring nodes - Heterogeneous Mobile network

Fig. 12. Mobile networks with heterogeneous speeds: Path length distribution, Clustering coefficient, View size distribution, and Intersection between views of neighboring nodes

implementation is worth investigating. Instead of picking a next node based on neighborhood information that is likely to be obsolete, a node v that wishes to pass a RW message will broadcast this message to all its neighbors. Every node u that receives this message will choose with probability $\frac{1}{number\_of\_neighbors}$ to accept this RW message (and will ignore it otherwise). On average, a single node will accept this RW and pass it on to the next node. However, there is a non-negligible probability that no node will rebroadcast this message, i.e., the RW will be discarded, or more than one node will rebroadcast it, i.e., the RW will be duplicated. One possible solution is to allow RWs to be duplicated and discarded and compensate for that at the membership service level. Another option is to use a higher rebroadcasting probability, yet if a node u decided to accept the RW, then u will first ask for permission from the sender v. Node v will grant the RW to the first node asking for it. Further exploration of the above techniques in mobile networks is left for

future research.[8]

6.2.4 *RaWMS in mobile networks with heterogeneous speeds.* This section studies the performance of RaWMS in mobile networks with very heterogeneous speeds. In this simulations we employed the Random Waypoint model, when half of the nodes moved at the speed picked in the range of 0.5-2 m/s (slow nodes) and another half moved at the speed picked in the range of 5-10 m/s (fast nodes). The average pause time was set to 30s. We can see that the views are random according to the path length distribution test and clustering coefficient, for mixing times of $n/16$ and longer. However, both view size variance and the intersection between views of neighboring nodes show the same tendency that was seen in mobile networks before (Figures 4, 6 and 11). There is a growing difference between slow and fast nodes, which happens due to the fact that long RWs tend to stop at static and slow nodes rather than at fast nodes. As a result slow nodes have larger views and smaller intersection with the views of their fast moving neighbors. To deal with this a more aggressive and frequent neighborhood discovery protocol should be used.

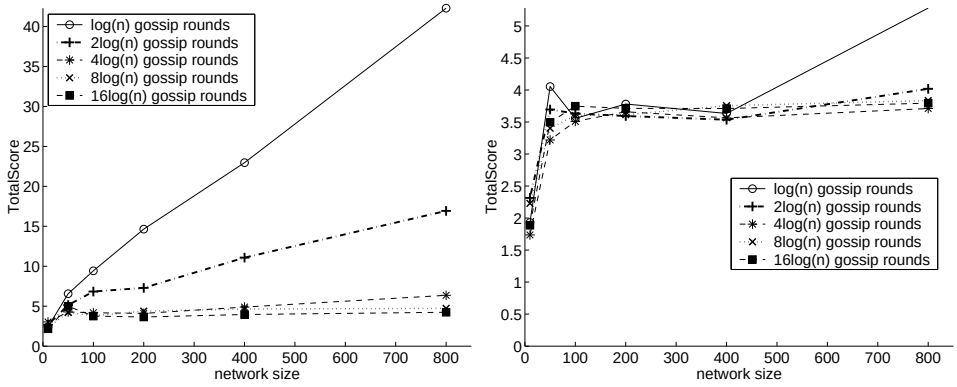
6.2.5 *Mixing time - the conclusion.* As we have shown in various tests, T_{mix} is well approximated by $n/2$ for static networks and by $n/8$ for dynamic networks. Moreover, for fast moving networks, T_{mix} can be set as low as $n/32$. However, measuring the level of mobility at runtime is an open challenge. We will therefore use an upper bound of $T_{\mathrm{mix}} = n/8$ for all mobile networks. According to Theorem 3.4, $T_{\mathrm{actual\_mix}} \leq T_{\mathrm{mix}} \frac{d_{\max}}{D}$. In our simulations, D was set large enough to bound $d_{\max}$ and the measured $T_{\mathrm{actual\_mix}}$ was about $T_{\mathrm{mix}}/2$. Therefore, $T_{\mathrm{actual\_mix}}$ is about $n/4$ for static networks and $n/16$ for dynamic networks.

6.3 Comparison with lpbcast

lpbcast. In our measurements, we have separated the routing communication overhead from the application communication overhead. This highlights why lpbcast is considered a very good protocol for peer-to-peer networks, but does not do so well in ad hoc networks. lpbcast was tested with a varying number of rounds: $\log n$, $2\log n$, $4\log n$, $8\log n$, $16\log n$. The fanout was set to 3 for all simulations and the view size limit was set to $\sqrt{n}$, to establish the same conditions as with RaWMS. As can be seen in Figure 13(a), in static networks, when the number of gossip rounds is $2\log(n)$ or less, the resulting view is not uniform according to the path length distribution test. As for a view size of lpbcast, since it was limited to $\sqrt{n}$ and since nodes gossip their entire view, in almost all cases the view was full. Here too, as can be seen in Figure 13(b), the uniformity of the views is dramatically improved when nodes are mobile.

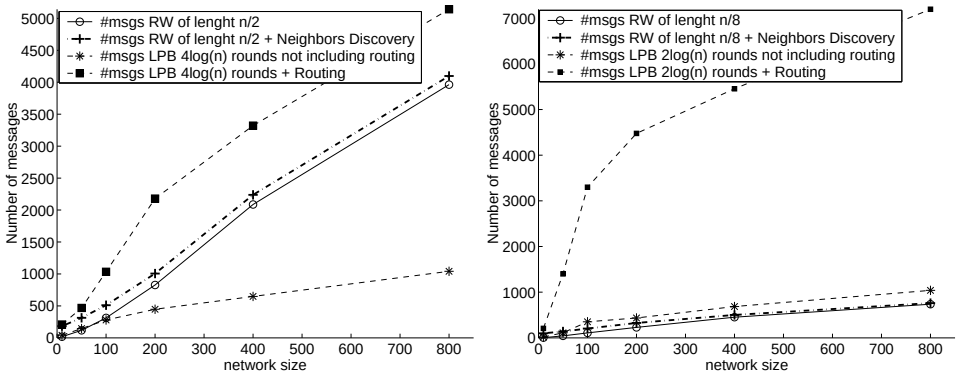
RaWMS versus lpbcast - communication overhead. Figure 14 depicts the number of messages sent by a single node during the entire simulation period, in both RaWMS and lpbcast. We have separated the number of application messages (messages directly generated by RaWMS and lpbcast) from the total number of network messages, which include the cost of routing and the neighbor discovery protocol messages. We have chosen to present RaWMS with a walk length of $n/2$ and lpbcast with $4\log n$ rounds, as these give optimal results, respectively. That is, these are the most efficient versions of both protocols, which still guarantee a fairly uniform distribution of views at the lowest possible cost.

[8]The ideas presented in this paragraph were proposed to us by Chen Avin.



(a) Static network (b) Mobile network

Fig. 13. lpbcast - path length distribution test ($PathScore$ versus n)



(a) Static network (b) Mobile network

Fig. 14. RaWMS versus lpbcast - comparing the number of messages

We can see that the results generally follow our theoretical discussion in Section 5.4. In RaWMS, each node starts roughly $\sqrt{n}+2$ RWs, each walk sending $T_{\text{actual\_mix}}$ messages. $T_{\text{actual\_mix}}$ is about $n/4$ as previously explained. Thus, every node sends a total number of $\frac{n\sqrt{n}}{4}$ messages. In lpbcast, every node starts $4\log n$ rounds with fanout 3 and each message traverses the network over an average path of $\sqrt{\frac{n}{\log n}}$. Therefore, each node sends $12\sqrt{n\log n}$ messages in total.

As is evident from Figure 14(a), lpbcast generates fewer application messages than RaWMS, as expected by our previous analysis. Yet, recall that in lpbcast each message contains the whole view, while in RaWMS messages carry only a single node identifier. Therefore, the total bit communication overhead of lpbcast is $12n\sqrt{\log n}$. In addition, lpbcast has a significant message overhead due to routing. When adding the cost of routing, RaWMS becomes considerably more efficient than lpbcast.

Figure 14(b) illustrates the communication costs of RaWMS with a walk length of $n/8$

and lpbcast with $2 \log n$ rounds in mobile scenarios. Again, those parameters guarantee a uniform distribution of views at the lowest possible cost. Here, the cost of RaWMS is significantly lower than lpbcast. This is due to a decreased walk length, yet without compromising the uniformness of the views. In this scenario, each node sends about $\frac{n\sqrt{n}}{16}$ messages. lpbcast sends approximately the same number of application messages as in the static case. However, with mobility, the cost of routing becomes considerable, which accounts for the dramatic affect on the overall performance of lpbcast in terms of network messages.

6.4 Comparison with Shuffling

Shuffling. We have measured the influence of a batch size, denoted by B, and the number of rounds on the uniformness of the views and the performance of Shuffling. Shuffling was tested with a varying number of rounds: $\frac{n\sqrt{n}}{B}$, $\frac{n\sqrt{n}}{2B}$, $\frac{n\sqrt{n}}{4B}$, $\frac{n\sqrt{n}}{8B}$, corresponding to different values for the actual mixing time and for different values of $B : 1, 2, 4, 8$. The view size limit was set to $\sqrt{n}$.

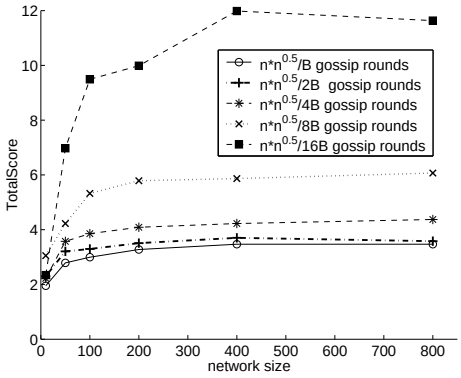
As can be seen from Figures 15, in static networks, when the number of gossip rounds is $\frac{n\sqrt{n}}{4B}$ or more, the resulting view is uniform according to the path length distribution test, for all values of B (due to the space considerations, we have chosen to present the results only for $B = 1$ and $B = 8$, but the same results were measured also for $B = 2$ and $B = 4$). According to our theoretical analysis, the number of rounds in Shuffling is $\frac{\sqrt{n}}{B} \cdot T_{\text{actual\_mix}}$. Indeed, as we have shown previously, $T_{\text{actual\_mix}}$ is well approximated by $n/4$. Thus, the number of $\frac{n\sqrt{n}}{4B}$ rounds, which results in a good uniformity according to the path length distribution test, confirms our theoretical analysis.

However, the value of a batch size has a direct impact on the size of the intersection between views of neighboring nodes. As can be seen from Figure 15(e) and Figure 15(f), which depict the average intersection size between views for $\frac{n\sqrt{n}}{4B}$ rounds as a function of B, the larger the value of B is, the bigger the intersection is. This can be explained by a certain amount of duplication that occurs when neighboring nodes shuffle their identifiers. We can therefore see the importance of conducting multiple statistical tests in order to compare the results with the ideal uniform sample: according to the path length distribution test, the views are uniform, but the intersection test clearly shows the opposite. The view size of Shuffling at the end of the convergence period was full in all cases.

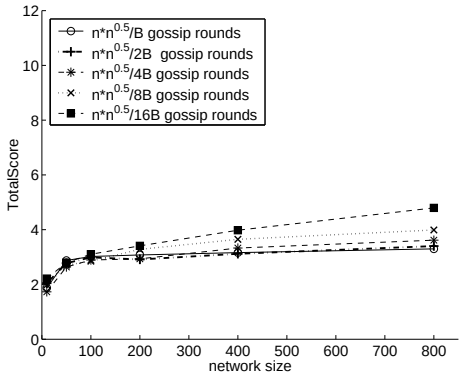
In mobile networks, good uniformity is reached after $\frac{n\sqrt{n}}{8B}$ rounds, irrespectively of the batch size as well. Increasing a batch size has the same effect on the views intersection size in mobile networks as in static networks, however to a lesser extent.

RaWMS versus Shuffling - communication overhead. Figure 16 depicts the number of application messages sent by a single node during the entire simulation period for different values of B. In Shuffling, as in RaWMS, the cost of routing and the neighbor discovery protocol messages is constant and is therefore not depicted.

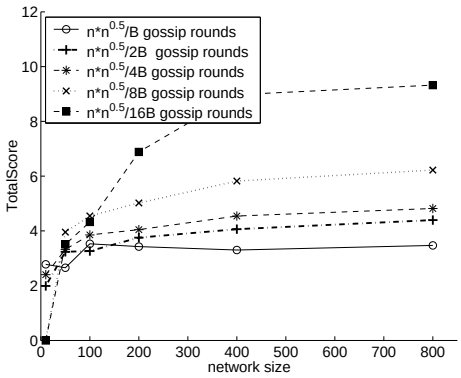
In static networks RaWMS is presented with a walk length of $n/2$ and Shuffling with $\frac{n\sqrt{n}}{4B}$ rounds, as these are the most efficient versions of both protocols, which still guarantee a fairly uniform distribution of views at the lowest possible cost. In RaWMS, each node sends a total number of $\frac{n\sqrt{n}}{4}$ messages (as explained in Section 6.3). In Shuffling, each node sends a total number of $2\frac{n\sqrt{n}}{4B}$ messages, since in each round every node starts one



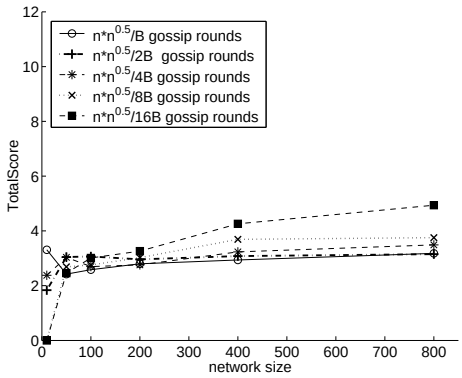
(a) Path length distribution, Static network - batch size 1



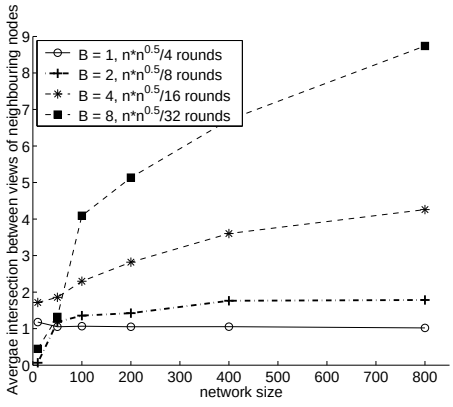
(b) Path length distribution, Mobile network - batch size 1



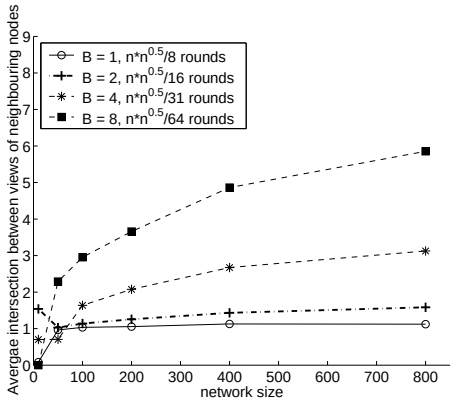
(c) Path length distribution, Static network - batch size 8



(d) Path length distribution, Mobile network - batch size 8

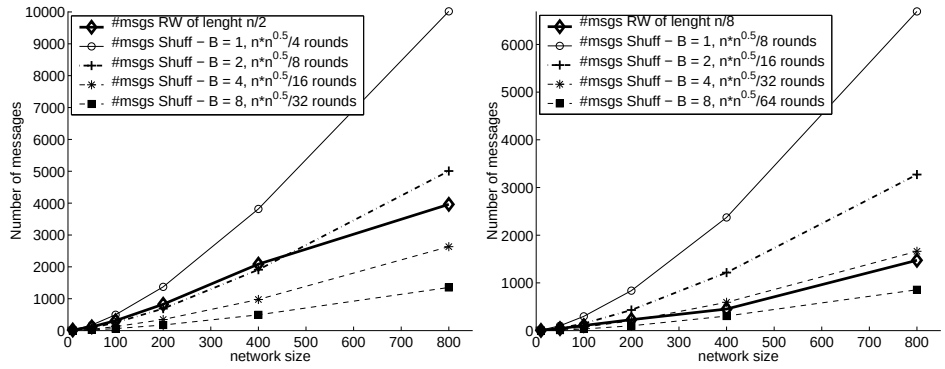


(e) Intersection between neighboring nodes, Static network



(f) Mobile network

Fig. 15. Shuffling - path length distribution test ($PathScore$ versus n) and intersection between views of neighboring nodes



(a) Static network (b) Mobile network

Fig. 16. RaWMS versus Shuffling - comparing the number of messages

shuffle with one of its neighbors, and therefore every shuffle results in an exchange of two messages of size B. For $B = 2$, the overhead of Shuffling matches that of RaWMS.

In mobile networks RaWMS is presented with a walk length of $n/8$ and Shuffling with $\frac{n\sqrt{n}}{8B}$ rounds. In RaWMS, each node sends a total number of $\frac{n\sqrt{n}}{16}$ messages, while in Shuffling, each node sends a total number of $2\frac{n\sqrt{n}}{8B}$ messages. For $B = 4$ the overhead of Shuffling matches that of RaWMS.

Figures 15(e) and 16 depict the tradeoff in the Shuffling algorithm: on the one hand the number of messages is reduced as the batch size B increases. On the one hand, larger B results in an increased correlation between neighboring nodes, damaging the uniformity of the views. In addition, increasing the batch size results in the increased message size. Therefore, the total bit communication complexity of Shuffling does not depend on B and equals $\frac{n\sqrt{n}}{2}$ in static network ($\frac{n\sqrt{n}}{4}$ in mobile network), while the bit communication complexity of RaWMS is $\frac{n\sqrt{n}}{4}$ in static network ($\frac{n\sqrt{n}}{16}$ in mobile network).

In summary, Shuffling introduces the possibility of batching a number of RW messages together, thus reducing the total number of messages (but not the total communication bandwidth, which remains constant w.r.t. B). Shuffling with small values of B behaves very similarly to RaWMS, but is much harder to analyze in a formal manner. Therefore, in systems in which the added confidence provided by a formally understood model is an issue, e.g., for legal reasons, RaWMS has an advantage over Shuffling. As for other practical purposes, increasing the size of B reduces the message complexity compared to RaWMS, but results in views that have worse uniformity properties. In other words, a network designer can choose between lower network complexity and better view uniformity: whenever message complexity is more important, Shuffling with a large value of B is a better choice. If view uniformity is more important, then both RaWMS and Shuffling with small values of B can be used. Another minor advantage of RaWMS is that its code is slightly simpler than Shuffling, and nodes are free to pick their view sizes independently.

7. RELATED WORK

Random walks. Comprehensive surveys of random walk techniques and their analysis appear in [Lovász 1993] and [Guruswami 2000]. The idea of using a "maximum-degree"

RW to reach a uniform limit distribution on the state space has been used before in a number of contexts [Bar-Yossef et al. 2000; Boyd et al. 2005].

Lv et al. [2002] propose to use simulated RWs for searching in unstructured peer-to-peer networks. They report that such a search is preferable to searching by flooding, due to RWs' adaptiveness to termination conditions and a fine-grain control of the search space. This work reported attractive empirical results, but does not provide any analytical evaluation of the RW properties.

Gkantsidis et al. [2004] explore the performance of RWs for searching and sampling in peer-to-peer networks and show that it is possible to simulate a uniform sample of elements from the network by performing a RW with an adequate length. We use a similar sampling technique, but on a completely different communication graph. Peer-to-peer networks graphs are usually assumed to be expanders. On the other hand, ad hoc network graphs are random geometric graphs [Penrose 2003], which are not expanders.

A recent work, which was done concurrently and independently to ours, proposed two RW based methods for peer counting and sampling in peer-to-peer networks: *Random Tour* and *Sample and Collide* [Massoulie et al. 2006]. The latter method is based on the "birthday paradox" in a manner very similar to ours (Section 4.4). There are a number of significant differences between our work and [Massoulie et al. 2006]. First, the methods in [Massoulie et al. 2006] use a continuous time RW to produce the samples, while RaWMS uses a Maximum Degree discrete RW with self-loops. Our work establishes the exact value of the mixing time for random geometric graphs, while [Massoulie et al. 2006] does not calculate the mixing time of the underlying graph and relies on the known expansion properties of random overlay graphs. Our work uses reverse sampling to reduce the communication overhead and constructs a membership service, while [Massoulie et al. 2006] uses its sampling methods only for counting. Additionally, [Massoulie et al. 2006] targets peer-to-peer networks, while RaWMS is meant for wireless ad hoc networks, which have different properties.

Various properties of RWs on random geometric graphs, including the mixing time and the partial cover time, have been investigated by [Boyd et al. 2005] and [Avin and Ercal 2005]. We rely on these results in our work.

Dolev et al. [2002] propose a randomized self-stabilizing group membership service for ad hoc networks. The group membership list is collected by a single random walk agent traversing the network. However, [Dolev et al. 2002] only constructs a full membership while RaWMS can be used to construct partial membership views. Moreover, they apply a single RW that covers the whole network and runs for a period of time that is equal to the cover time. We use multiple RWs simultaneously each running for a period that is equal to the mixing time. Thus, the time and communication complexities of the algorithm in [Dolev et al. 2002] are $O(n^3)$, while in RaWMS each RW runs for only $O(n)$. The communication complexity of RaWMS depends on the desired view size. For example, to construct a view of size $O(\sqrt{n})$ at every node, RaWMS sends a total of $O(n^2\sqrt{n})$ messages. A full membership can be constructed with RaWMS by an additional short RW that collects partial random views from different nodes. The total communication complexity in this case is $O(n^2\sqrt{n})$.

In [Servetto and Barrenechea 2002] RWs are used for routing in large-scale sensor networks. They assume a static network and only consider a grid topology. On the other hand, we also support mobility, and do not restrict the topology except for being connected.

Gossiping. Gossiping is another well-known scheme to establish a random sample. Recently, gossip-based dissemination of membership information was proposed in order to design scalable implementations of a peer sampling service. In particular, a general gossip-based peer sampling service was introduced in [Jelasity et al. 2007]. Examples of gossip-based lightweight membership services complying to this general framework are reported in [Allavena et al. 2005; Eugster et al. 2003; Ganesh et al. 2001; Jelasity and Babaoglu 2005; Voulgaris et al. 2005] and are discussed in more details in Section 5.

SCAMP [Ganesh et al. 2001] introduced a generic random membership service that is used for probabilistic reliable dissemination of data and events in peer-to-peer networks. The appealing property of SCAMP is that the partial view obtained by a node adapts automatically to the system's size, without any a priori knowledge of the total network size. However, [Ganesh et al. 2001] only proves that the mean value of the sum of all views of all nodes is $\Theta(n \log n)$ and that the actual sum of all view sizes is not far from the mean. No proof is provided about the view size of a single node, which may be far from the mean by orders of magnitude. In our work, we do bound the minimal and maximal view sizes of all nodes.

A gossip-based membership service for sensor and mobile ad hoc networks based on the Shuffling technique is described in [Gavidia et al. 2005] and discussed in section 5. The CYCLON [Voulgaris et al. 2005] protocol is an adaptation of Shuffling for membership construction in peer-to-peer networks. In [Voulgaris et al. 2005] the resemblance between the knowledge graph constructed by CYCLON and random graphs is shown by simulations. However, no formal analysis is presented. We believe that based on our observations in Section 5.2, a more through analysis of the CYCLON protocol is now possible.

RDG [Luo et al. 2003] is an adaptation of [Eugster et al. 2003] to ad hoc networks. It reduces the cost of routing compared to [Eugster et al. 2003] by utilizing routes created by other applications running in the same wireless node or by using proactive periodical flooding in order to establish those routes. Although RDG relies only on partial views for correct implementation of probabilistic multicast, in practice the views constructed by RDG are not necessarily partial and may even be almost full views. In addition, those views are not constructed by gossiping, but by the same flooding that establishes the routes. Gossiping is only used in RDG for data dissemination and for removal of nodes that left the network from the views. The usage of flooding results in a linear memory consumption, so there is no point in using it for constructing partial views.

Haas et al. [2002] have investigated various approaches for disseminating data using several gossip functions in ad hoc networks [Haas et al. 2002]. They investigate the impact of gossip on the message delivery ratio of broadcast messages. The *anonymous gossip* work has explored the use of gossip with direct neighbors in an ad hoc network to increase the reliability of broadcast and multicast protocols [Chandra et al. 2001]. Both these works, however, do not address membership maintenance.

Symphony [Manku et al. 2003] is a scalable and failure resilient protocol for maintaining distributed hash tables based on Kleinberg's Small World construction [Kleinberg 2000]. The distribution of nodes in the routing tables of Symphony (Symphony's membership) is comprised of local neighbors in the virtual ring structure, as well as of a constant number of long distance neighbors picked accordingly to a distribution which is inversely proportional to their distance on the ring. Such a choice of long distance neighbors guarantees, w.h.p., fast logarithmic DHT lookup.

8. DISCUSSION AND CONCLUSIONS

In this paper we have presented RaWMS, a random walk based lightweight membership service for ad hoc networks. We have presented a formal analysis of RaWMS, backed by simulations and have also compared RaWMS with gossip-based approaches for building such membership services. Overall, the results of the simulations confirm the formal analysis. They show that RWs present an attractive paradigm for implementing partial view based membership services in ad hoc networks. This is due to the fact that RWs do not require multi-hop routing and avoid flooding altogether. Moreover, when the network is mobile, RWs reach their target uniformity even faster than in static networks. In these cases, the mobility helps to disseminate messages to random places in the network.

We would like to highlight the fact that RaWMS is a fully decentralized algorithm. The only parameter that needs to be set for the correct behavior of RaWMS is the mixing time. No other assumptions or explicit prior knowledge about the structure of the network graph is being used in RaWMS. Specifically, no assumptions are made about the degree distribution of the graph – the stationary distribution of the RW remains uniform for any degree distribution due to the regularization of the graph with self loops. In particular, the maximal degree $d_{\max}$ need not be known, since an upper bound D on $d_{\max}$ can be chosen arbitrary large. Moreover, the network size can be estimated on the fly.

However, the calculation of the mixing time is based on an assumption that the network graph is a static connected unit disk graph. If the graph is not a connected unit disk graph, but is rather some sparser graph, e.g., a grid or a line, then the mixing time established in this work may be not sufficient to reach the uniform distribution. In such cases, RaWMS should be parameterized with a different mixing time. From a practical point of view, if the structure of the network graph cannot be estimated, the mixing time can be always overestimated without hurting the uniformity of the samples, but rather by increasing the communication complexity.

A surprising empirical result of our study is that in mobile networks short RWs obtain better results than long ones. In other words, the mixing time of mobile ad-hoc networks graphs is shorter than for static unit-disk graphs. The conclusion is that nodes should consider the degree of mobility in the network when determining the length of the RWs that they start. The faster nodes move, the shorter the RWs need to be. Recognizing the level of mobility can be implemented, e.g., by analyzing the frequency of neighborhood changes in each node's proximity. Studying this phenomenon in a formal manner and deriving a specific protocol from it is left for future work.

Power consumption and battery life are two important aspects of any mobile system in which nodes are battery operated. Power is governed by a fixed consumption per second plus a component that is directly affected by the number of messages sent and received. The cost of sending and receiving messages is almost the same in most WiFi wireless cards and even idle power consumption is of the same magnitude (325 mA for send, 215 mA for receive/idle and 25 mA for Power Save Mode [Ferro and Potorti 2005]). Hence, the energy requirements of RaWMS is directly proportional to the number of messages it generates, which we studied in Section 6.3. However, since RaWMS only uses unicast messages, it gives opportunity to 802.11 like networks to save energy by switching from idle to Power Save Mode [IEEE-802.11-Standard]. As for battery life, one can distinguish between two cases based on the level of heterogeneity (in terms of batteries and power specifications) of the network. If the network is homogeneous, then RWs are a very good mechanism, since

they inherently spread the load evenly between all nodes. On the other hand, if the network is very heterogeneous, then the above load balancing of RaWMS becomes a drawback, since in such cases we would rather utilize nodes that have more energy than others. It is actually possible to use our Maximum Degree RW method to bias the random walks in order to utilize the more powerful nodes, thereby extending the network's life. This can be done by adding more self-loops to powerful nodes and less self-loops to power-poor nodes. In such case the membership samples will not be uniformly random, but rather proportional to the power level of the nodes. Further investigating this direction is left for future work.

Our work leaves several additional open problems. These include, e.g., a more detailed investigation of the relation between random walks and gossip. In particular, combining random walks with occasional gossiping to far away nodes.

Finally, we believe that our analysis of RW's complexity for ad hoc networks can serve as a starting points for many additional RW-based algorithms in ad hoc networks.

ACKNOWLEDGMENT

We would like to thank Chen Avin and the anonymous referees for many insightful comments.

REFERENCES

ALLAVENA, A., DEMERS, A., AND HOPCROFT, J. E. 2005. Correctness of a Gossip based Membership Protocol. In *Proceedings of the 24th annual ACM symposium on Principles of Distributed Computing (PODC)*. 292–301.

AVIN, C. AND ERCAL, G. 2005. Bounds on the Mixing Time and Partial Cover of Ad-Hoc and Sensor Networks. In *Proceedings of the 2nd European Workshop on Wireless Sensor Networks (EWSN)*.

BAR-YOSSEF, Z., BERG, A., CHIEN, S., FAKCHAROENPHOL, J., AND WEITZ, D. 2000. Approximating Aggregate Queries about Web Pages via Random Walks. In *Proc. of the 26th International Conference on Very Large Data Bases (VLDB)*. 535–544.

BARR, R., HAAS, Z. J., AND VAN RENESSE, R. JiST/SWANS Java in Simulation Time / Scalable Wireless Ad Hoc Network Simulator. Available at `http://jist.ece.cornell.edu/`, Cornell University.

BIRMAN, K. P., HAYDEN, M., OZKASAP, O., XIAO, Z., BUDIU, M., AND MINSKY, Y. 1999. Bimodal Multicast. *ACM Transactions on Computer Systems 17,* 2, 41–88.

BOLLOBAS, B. 2001. *Random Graphs*, 2nd ed. Cambridge University Press.

BOYD, S., DIACONIS, P., AND XIAO, L. 2004. Fastest Mixing Markov Chain on a Graph. *SIAM Review 46,* 4, 667–689.

BOYD, S., GHOSH, A., PRABHAKAR, B., AND SHAH, D. 2005. Mixing Times for Random Walks on Geometric Random Graphs. In *Proceedings of the 2nd SIAM Workshop on Analytic Algorithmics and Combinatorics (ANALCO)*.

CHANDRA, R., RAMASUBRAMANIAN, V., AND BIRMAN, K. 2001. Anonymous Gossip: Improving Multicast Reliability in Mobile Ad-Hoc Networks. In *Proceedings of the 21st International Conference on Distributed Computing Systems (ICDCS)*. 275.

CHAUM, D. L. 1981. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM 24,* 2, 84–90.

CHERNOFF, H. 1952. A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the Sum of Observations. *American Mathematical Society 23*, 493–507.

CHOCKLER, G., KEIDAR, I., AND VITENBERG, R. 2001. Group Communication Specifications: a Comprehensive Study. *ACM Computing Surveys 33,* 4, 427–469.

DIACONIS, P. AND STROOCK, D. 1991. Geometric Bounds for Eigenvalues of Markov Chains. *Annals of Applied Probability 1*, 36–61.

DOLEV, S., SCHILLER, E., AND WELCH, J. 2002. Random Walk for Self-Stabilizing Group Communication in Ad Hoc Networks. In *Proceedings of the 21st annual symposium on Principles of Distributed Computing (PODC)*. 259–259.

ERDOS, P. AND RENYI, A. 1960. On the Evolution of Random Graphs. *PubI. Math. Inst. Hungar. Acad. Sci. 5*, 17–61.

EUGSTER, P. T., GUERRAOUI, R., HANDURUKANDE, S. B., KOUZNETSOV, P., AND KERMARREC, A.-M. 2003. Lightweight Probabilistic Broadcast. *ACM Transactions on Computer Systems (TOCS) 21,* 4, 341–374.

FEIGE, U. 1996. A fast randomized LOGSPACE algorithm for graph connectivity. *Theoretical Computer Science 169,* 2, 147–160.

FENNER, T. I. AND FRIEZE, A. M. 1982. On the Connectivity of Random m-orientable Graphs and Digraphs. *Combinatorica 2*, 347–359.

FERRO, E. AND POTORTI, F. 2005. Bluetooth and Wi-Fi Wireless Protocols: A Survey and a Comparison. *IEEE Wireless Communications 12*, 12 – 26.

FREEDMAN, M. J. AND MORRIS, R. 2002. Tarzan: a Peer-to-Peer Anonymizing Network Layer. In *Proceedings of the 9th ACM Cconference on Computer and Communications Security (CCS)*. 193–206.

GANESH, A. J., KERMARREC, A.-M., AND MASSOULIE, L. 2001. SCAMP: Peer-to-Peer Lightweight Membership Service for Large-Scale Group Communication. In *Networked Group Communication*. 44–55.

GAVIDIA, D., VOULGARIS, S., AND VAN STEEN, M. 2005. Epidemic-style Monitoring in Large-Scale Sensor Networks. Tech. Rep. IR-CS-012, Vrije Universiteit, Amsterdam, Netherlands. March.

GKANTSIDIS, C., MIHAIL, M., AND SABERI, A. 2004. Random Walks in Peer-to-Peer Networks. In *Proceedings of the 23rd Conference of the IEEE Communications Society (INFOCOM)*. 259–259.

GUPTA, P. AND KUMAR, P. 1998. Critical Power for Asymptotic Connectivity in Wireless Networks. In *Stochastic Analysis, Control, Optimization and Applications*. 547–566.

GURUSWAMI, V. 2000. Rapidly Mixing Markov Chains: a Comparison of Techniques. Available at `http://www.cs.washington.edu/homes/venkat/pubs/pubs.html`.

HAAS, Z., HALPERN, J., AND LI, L. 2002. Gossip-Based Ad Hoc Routing. In *Proceedings of the 21st Conference of the IEEE Communications Society (INFOCOM)*. 1707–1716.

HAAS, Z. AND LIANG, B. 1999. Ad Hoc Mobility Management with Randomized Database Groups. In *Proceedings of IEEE International Conference on Communications (ICC)*. Vol. 3. 1756 – 1762.

HORN, R. AND JOHNSON, C. 1985. *Matrix Analysis*. Cambridge University Press.

IEEE-802.11-STANDARD. Wireless LAN Media Access Control (MAC) and Physical Layer (PHY) Specifications. Downloadable at `http://standards.ieee.org/getieee802/`.

JELASITY, M. AND BABAOGLU, O. 2005. T-Man: Gossip-Based Overlay Topology Management. In *Proceedings of the 3rd International Workshop on Engineering Self-Organising Systems (ESOA)*.

JELASITY, M. AND VAN STEEN, M. 2002. Large-Scale Newscast Computing on the Internet. Tech. Rep. IR-503, Vrije Universiteit, Amsterdam, Netherlands. October.

JELASITY, M., VOULGARIS, S., GUERRAOUI, R., KERMARREC, A.-M., AND VAN STEEN, M. 2007. Gossip-based peer sampling. *ACM Trans. Comput. Syst. 25,* 3, 8.

JOHNSON, D. AND MALTZ, D. 1996. Dynamic Source Routing in Ad Hoc Wireless Networks. *Mobile Computing 353*.

KERMARREC, A.-M., MASSOULIE, L., AND GANESH, A. J. 2003. Probabilistic Reliable Dissemination in Large-Scale Systems. *IEEE Transactions on Parallel and Distributed Systems 14,* 3 (March), 248–258.

KLEINBERG, J. 2000. The Small-World Phenomenon: An Algorithmic Perspective. In *Proceedings of the 32nd ACM Symposium on Theory of Computing (STOC)*. 163–170.

LOVÁSZ, L. 1993. Random Walks on Graphs: A Survey. *Combinatorics 2*, 1–46.

LUO, J., EUGSTER, P., AND HUBAUX, J.-P. 2003. Route Driven Gossip: Probabilistic Reliable Multicast in Ad Hoc Networks. In *Proceedings of the 23rd Conference of the IEEE Communications Society (INFOCOM)*.

LV, C., CAO, P., COHEN, E., LI, K., AND SHENKER, S. 2002. Search and Replication in Unstructured Peer-to-Peer Networks. In *Proceedings of the 16th International Conference on Supercomputing (ICS)*. 84–95.

MANKU, G., BAWA, M., AND RAGHAVAN, P. 2003. Symphony: Distributed Hashing in a Small World. In *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems (USITS)*.

MASSOULIE, L., MERRER, E. L., KERMARREC, A.-M., AND GANESH, A. J. 2006. Peer Counting and Sampling in Overlay Networks: Random Walk Methods. In *Proceedings of the 25th ACM symposium on Principles of Distributed Computing (PODC)*. 123–132.

MELAMED, R. AND KEIDAR, I. 2004. Araneola: A Scalable Reliable Multicast System for Dynamic Environments. In *3rd IEEE International Symposium on Network Computing and Applications, (IEEE NCA)*. 5–14.

MERRER, E. L., KERMARREC, A.-M., AND MASSOULIE, L. 2006. Peer to Peer Size Estimation in Large and Dynamic Networks: A Comparative Study. In *Proceedings of the 15th IEEE International Symposium on High Performance Distributed Computing (HPDC)*. 7–17.

MOTWANI, R. AND RAGHAVAN, P. 1995. *Randomized Algorithms*. Cambridge University Press.

PANCHAPAKESAN, P. AND MANJUNATH, D. 2001. On the Transmission Range in Dense Ad Hoc Radio Networks. In *Proceedings of IEEE Signal Processing Communication (SPCOM)*.

PENROSE, M. D. 2003. *Random Geometric Graphs*. Oxford University Press.

PUCHA, H., DAS, S., AND HU, Y. C. 2004. Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks. In *Proceedings of the 6th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*. 163–173.

REITER, M. K. AND RUBIN, A. D. 1999. Anonymous Web Transactions with Crowds. *Communications of the ACM 42,* 2, 32–48.

SERVETTO, S. AND BARRENECHEA, G. 2002. Constrained Random Walks on Random Graphs: Routing Algorithms for Large Scale Wireless Sensor Networks. In *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*.

SIEGMUND, D. 1985. *Sequential Analysis - Tests and Confidence Intervals*. Springer-Verlag.

SINCLAIR, A. 1992. Improved Bounds for Mixing Rates of Markov Chains and Multicommodity Flow. *Combinatorics, Probability & Computing 1*, 351–370.

SONDOW, J. AND WEISSTEIN, E. W. Harmonic Number. From MathWorld–A Wolfram Web Resource. `http://mathworld.wolfram.com/HarmonicNumber.html`.

VOULGARIS, S., GAVIDIA, D., AND VAN STEEN, M. 2005. CYCLON: Inexpensive Membership Management for Unstructured P2P Overlays. *Journal of Network and Systems Management 13,* 2 (July), 197–217.

WATTS, D. J. AND STROGATZ, S. H. 1998. Collective Dynamics of Small-World Networks. *Nature 393,* 4 (June), 440–442.

APPENDIX

A. RANDOM GEOMETRIC GRAPHS

We provide below a formal definition of the Random Geometric Graph $G^2(n, r)$. To this end, we need to introduce some basic facts about the geometry on the surface of a torus.

Geometry on the surface of a torus. A 2-dimensional *unit torus* is the set of points in the unit square $[0, 1] \times [0, 1]$ endowed with a special measure of distance, called the *geodesic distance*. It is convenient to visualize a torus as taking the flat unit square, and then "gluing" together the two vertical edges and the two horizontal edges. What we get is a surface of 3-dimensional object, whose shape resembles a holed donut. The important point to notice is that because of the gluing points near the left vertical edge are close to points near the right vertical edge, and similarly points near the top horizontal edge are close to points near the bottom horizontal edge.

Every point u on the surface of a torus has two coordinates: $u_x \in [0, 1]$ and $u_y \in [0, 1]$. Every two points u, v on a torus have two straight lines connecting them (going in opposite directions). The *geodesic distance* between u and v is the length of the shorter of these two lines. To formally define the geodesic distance we introduce the following notion of "circle distance" between real numbers.

Every real number $a \in \mathbb{R}$ can be embedded into a circle whose circumference is 1 as

follows:

$$a \bmod 1 = \begin{cases} a - \lfloor a \rfloor & \text{if } a \geq 0 \\ 1 - (|a| \bmod 1) & \text{if } a < 0 \end{cases}$$

For two numbers $a, b \in [0, 1]$, we define the *circle distance* between a and b as:

$$\text{cd}(a, b) = \min\{(a - b) \bmod 1, (b - a) \bmod 1\}.$$

For example, if $a = 7/8$ and $b = 1/8$, then $(a - b) \bmod 1 = 6/8$, while $(b - a) \bmod 1 = 2/8$, and hence $\text{cd}(a, b) = 2/8$. Note that the circle distance between any two numbers is always at most $1/2$.

Given two points u, v on the surface of a unit torus, we define the *geodesic distance* between u and v as follows:

$$\text{gd}(u, v) = \sqrt{\text{cd}(u_x, v_x)^2 + \text{cd}(u_y, v_y)^2}.$$

Random geometric graphs. Let n be a positive integer and let $r \geq 0$ be a real number. The random geometric graph $G^2(n, r)$ is generated as follows. The graph has n vertices associated with n uniformly chosen points on the surface of a 2-dimensional torus. Two vertices u, v are connected by an edge if and only if $\text{gd}(u, v) \leq r$.

B. CHERNOFF BOUNDS

We state below the exact version of the Chernoff bounds [Chernoff 1952] we use in this paper. A proof can be found, e.g., in [Motwani and Raghavan 1995].

THEOREM B.1 CHERNOFF BOUNDS. *Let $X_1, X_2, \ldots, X_n$ be independent and identically distributed Bernoulli random variables with probability of success p. (That is, for all $1 \leq i \leq n$, X_i is a 0-1 random variable and $\Pr[X_i = 1] = p$.) Let $X = \sum_{i=1}^{n} X_i$ and let $\mu = E(X) = np$. Then, for any $0 < \delta < 1$,*

$$\Pr(X > (1 + \delta)\mu) < e^{-\mu\delta^2/3} \quad \textbf{\textit{[Upper tail]}},$$

and

$$\Pr(X < (1 - \delta)\mu) < e^{-\mu\delta^2/2} \quad \textbf{\textit{[Lower tail]}}.$$

Note that by combining the upper and lower tail bounds we obtain:

$$\Pr(|X - \mu| > \delta\mu) < 2e^{-\mu\delta^2/3}.$$

C. REVERSE RW-BASED UNIFORM SAMPLING - PROOF OF LEMMA 3.5

Lemma 3.5 (restated) *Suppose every node v in a network chooses (via a random walk) a random node X_v. For every u, let Z_u be the set of nodes that selected u (the RWs started by them have stopped at u): $Z_u = \{v \mid X_v = u\}$. Then, given that the size of Z_u is k, Z_u is a random subset of the vertex set of size k.*

PROOF. For simplicity of analysis, we assume in the proof that each RW produces a truly-uniform node and not a nearly-uniform node. The extension to deal with nearly-uniform samples is rather straightforward.

To prove the lemma, we need to show that $\forall u \in V$, $\forall 1 \leq k \leq n$, and for set S of k distinct nodes, $\Pr(Z_u = S \mid |Z_u| = k) = 1/\binom{n}{k}$.

Fix any u, any $k \in \{1, \ldots, n\}$, and any set $S = \{v_1, \ldots, v_k\}$ of k nodes. By Bayes rule,

$$\begin{aligned} &\Pr(Z_u = S \mid |Z_u| = k) = \\ &= \Pr(|Z_u| = k \mid Z_u = S) \cdot \frac{\Pr(Z_u = S)}{\Pr(|Z_u| = k)} \end{aligned} \tag{1}$$

We next analyze each of the three terms on the RHS of Equation 1. For the first term, we have $\Pr(|Z_u| = k \mid Z_u = S) = 1$. Regarding $\Pr(Z_u = S)$, note that $Z_u = S = \{v_1, \ldots, v_k\}$ iff $X_{v_1} = u, X_{v_2} = u, \ldots, X_{v_k} = u$, and for every $v \notin S$, $X_v \neq u$. The events $\{X_v = u\}_{v \in V}$ are independent of each other (because the random walks are independent). Furthermore, for every v, $\Pr(X_v = u) = \frac{1}{n}$. Therefore, $\Pr(Z_u = S) = (\frac{1}{n})^k \cdot (1 - \frac{1}{n})^{n-k}$.

Regarding $\Pr(|Z_u| = k)$, $|Z_u|$ has a binomial distribution with n trials and a probability of success $\frac{1}{n}$. Therefore, $\Pr(|Z_u| = k) = \binom{n}{k} \cdot (\frac{1}{n})^k \cdot (1 - \frac{1}{n})^{n-k}$. Substituting the three terms into Equation 1, we have the desired result. □

D. PROOF OF LEMMA 4.1

Lemma 4.1 (restated) *Let $1 \leq s = s(n) \leq n$ and let $r = r(n)$ be the random variable specifying the number of balls needed to be randomly placed in n bins until s of the bins are non-empty. Then,*

$$E(r) = n(H_n - H_{n-s}) \leq \begin{cases} n \ln \frac{n}{n-s}, & s < n, \\ n \ln n + O(1), & s = n. \end{cases}$$

where $H_k = \sum_{i=1}^{k} \frac{1}{i}$ is the k-th harmonic number (and define $H_0 = 0$).

PROOF. We view the balls as being placed in the bins sequentially, one by one. The first ball is inserted into an empty bin. The second ball is placed into an empty bin with probability $\frac{n-1}{n}$ and into a non empty bin with probability $\frac{1}{n}$. Using the independence assumption, the expected number of balls required to have a second non empty bin is a geometric random variable with parameter $p = \frac{n-1}{n}$ and mean $\frac{1}{p} = \frac{n}{n-1}$. The additional number of balls required to get the third non empty bin is a geometric random variable with parameter $p = \frac{n-2}{n}$ and mean $\frac{1}{p} = \frac{n}{n-2}$. This process goes on until s bins have at least one ball. r is the number of balls used in this process and is therefore a sum of geometric random variables. By linearity of expectation, we have:

$$\begin{aligned} E(r) &= 1 + \frac{n}{n-1} + \frac{n}{n-2} + \cdots + \frac{n}{n-s+1} \\ &= n(\frac{1}{n} + \frac{1}{n-1} + \frac{1}{n-2} + \cdots + \frac{1}{n-s+1}) \\ &= n(H_n - H_{n-s}). \end{aligned}$$

In order to bound the difference $H_n - H_{n-s}$, we use the following well-known bounds on the harmonic number (see, e.g., [Sondow and Weisstein]):

$$\ln n + \gamma + \frac{1}{2(n+1)} \;\leq\; H_n \;\leq\; \ln n + \gamma + \frac{1}{2n},$$

where γ is a constant. The case $s = n$ immediately follows from the above bound on H_n. For $s < n$,

$$\begin{aligned} n(H_n - H_{n-s}) &\leq n\left(\ln n + \frac{1}{2n} - \ln(n-s) - \frac{1}{2(n-s+1)}\right) \\ &\leq n(\ln n - \ln(n-s)) = n \ln \frac{n}{n-s}. \end{aligned}$$

□

E. MIXING TIME BOUND FOR THE MD RANDOM WALK

In this section we prove the upper bound on the actual mixing time of the Maximum Degree random walk on a random geometric graph $G^2(n, r)$ (Theorem 3.4). Our proof is based on Sinclair's bound [Sinclair 1992] on the spectral gap of a random walk.

E.1 Sinclair's bound

In this section we overview Sinclair's bound [Sinclair 1992] on the spectral gap of a Markov chain. Sinclair's result holds for general reversible Markov chains. Yet, in order to avoid cumbersome notation, we restrict to random walks on regular graphs, which is the case of interest to us.

Let $G = (V, E)$ be a connected non-bipartite D-regular graph on n nodes. G possibly has weighted self loops but does not have parallel edges. The probability transition matrix P corresponding to a random walk on G is an $n \times n$ stochastic matrix defined as follows. For every $u \neq v$, $P_{uv} = \frac{1}{D}$ if u and v are connected by and edge and $P_{uv} = 0$, if they are not. The diagonal entries are $P_{uu} = 1 - \sum_{u' \neq u} P_{uu'}$.

Since P is a symmetric matrix, the stationary distribution of this random walk is the uniform distribution. The principal eigenvalue of P is 1. Let $\lambda_{\max}$ denote its second largest eigenvalue in absolute value. Sinclair showed a bound on the spectral gap $1 - \lambda_{\max}$ using the notion of *canonical paths*.

A family of canonical paths is a collection of paths $\gamma = \{\gamma_{uv}\}_{u \neq v \in V}$, one for each pair of distinct node u, v in G. Sinclair defines two parameters of such a family: the *maximum path length* and the *maximum edge load*. The *maximum path length* of a family of canonical paths γ is defined as:

$$\ell(\gamma) = \max_{\gamma_{uv} \in \gamma} |\gamma_{uv}|.$$

For an edge $e \in E$, we denote by $\rho(\gamma, e)$ the number of paths in γ that pass through e:

$$\rho(\gamma, e) = |\{\gamma_{uv} \in \gamma \mid \gamma_{uv} \ni e\}|.$$

The *maximum edge load* of γ, denoted $\rho(\gamma)$, is defined as:

$$\rho(\gamma) = \max_{e \in E} \rho(\gamma, e).$$

Sinclair's bound is then the following:

THEOREM E.1 SINCLAIR. *For any D-regular graph G and for any family of canonical paths γ on G,*

$$1 - \lambda_{\max} \geq \frac{n}{D \cdot \ell(\gamma) \cdot \rho(\gamma)}.$$

To derive bounds on the mixing time of the Maximum Degree random walk on $G^2(n,r)$, we need to resort to a stronger version of Sinclair's bound. (Our extension of Sinclair's bound is identical to the extension done by Boyd et al. [2005] to the bound of Diaconis and Stroock [1991].) Let Γ be the collection of all possible families of canonical paths. Let p be a probability distribution over Γ. Let $\text{SUPP}(p)$ be the set of canonical path families that have non-zero probability under p. The *maximum path length* of p is defined as:

$$\ell(p) = \max_{\gamma \in \text{SUPP}(p)} \ell(\gamma).$$

The *maximum expected load* of p is defined as:

$$\rho(p) = \max_{e \in E} \sum_{\gamma \in \Gamma} p(\gamma) \cdot \rho(\gamma, e).$$

We prove the following:

THEOREM E.2. *For any D-regular graph G and for any distribution p over Γ,*

$$1 - \lambda_{\max} \geq \frac{n}{D \cdot \ell(p) \cdot \rho(p)}.$$

PROOF. We use the variational characterization of the second eigenvalue (cf. [Horn and Johnson 1985]):

$$1 - \lambda_{\max} = \frac{n}{D} \cdot \inf_{\psi} \frac{\sum_{e \in E} (\psi(e^+) - \psi(e^-))^2}{\sum_{u,v \in V} (\psi(u) - \psi(v))^2}, \tag{2}$$

where the infimum is over all non-constant functions $\psi : V \to \mathbb{R}$, and e^+ and e^- denote the two vertices comprising an edge e. Consider any term (u,v) in the denominator. Using any $\gamma \in \Gamma$, we can rewrite this term as the following telescopic sum:

$$(\psi(u) - \psi(v))^2 = \left(\sum_{e \in \gamma_{uv}} (\psi(e^+) - \psi(e^-)) \right)^2.$$

Since this holds for all γ, then we can also write:

$$(\psi(u) - \psi(v))^2 = \left(\sum_{\gamma \in \Gamma} p(\gamma) \cdot \sum_{e \in \gamma_{uv}} (\psi(e^+) - \psi(e^-)) \right)^2.$$

Applying the Cauchy-Schwarz inequality, we have:

$$(\psi(u) - \psi(v))^2 \leq \left(\sum_{\gamma \in \Gamma} \sum_{e \in \gamma_{uv}} p(\gamma) \right) \cdot \left(\sum_{\gamma \in \Gamma} \sum_{e \in \gamma_{uv}} p(\gamma) \cdot (\psi(e^+) - \psi(e^-))^2 \right).$$

We first bound the left factor:

$$\sum_{\gamma \in \Gamma} \sum_{e \in \gamma_{uv}} p(\gamma) = \sum_{\gamma \in \text{SUPP}(p)} p(\gamma) \cdot |\gamma_{uv}| \leq \ell(p).$$

Substituting the above back in the denominator of the expression appearing in Equation 2,

we have:

$$\begin{aligned}\sum_{u,v\in V}(\psi(u)-\psi(v))^2 &\le \ell(p)\cdot\sum_{u,v\in V}\sum_{\gamma\in\Gamma}\sum_{e\in\gamma_{uv}}p(\gamma)\cdot(\psi(e^+)-\psi(e^-))^2\\ &= \ell(p)\cdot\sum_{e\in E}(\psi(e^+)-\psi(e^-))^2\cdot\sum_{\gamma\in\Gamma}p(\gamma)\cdot\sum_{\gamma_{uv}\ni e}1\\ &= \ell(p)\cdot\sum_{e\in E}(\psi(e^+)-\psi(e^-))^2\cdot\sum_{\gamma\in\Gamma}p(\gamma)\cdot\rho(\gamma,e)\\ &\le \ell(p)\cdot\rho(p)\cdot\sum_{e\in E}(\psi(e^+)-\psi(e^-))^2.\end{aligned}$$

Substitution in Equation 2 completes the proof. □

E.2 Mixing time bound for $G^2(n,r)$

In this section we show that the Maximum Degree random walk on the random geometric graph $G^2(n,r)$ has a mixing time of about n with high probability. The proof basically repeats the argument made by [Boyd et al. 2005]. We need to repeat the analysis, in order to figure out the best constants. Moreover, we provide details that are missing in the current version of [Boyd et al. 2005].

Theorem 3.4 (restated) *Suppose $r \le 1/2$ and $n \ge 10$. Let $G^2(n,r)$ be a random geometric graph chosen with n nodes and radius r. Let D be any value that upper bounds the maximum degree of $G^2(n,r)$. Let $T_{\mathrm{mix}}(\epsilon)$ be the mixing time of the MD random walk on this graph, when applied with the value D. Let $T_{\mathrm{actual\_mix}}(\epsilon)$ be the actual mixing time of this random walk (i.e., excluding self loop steps). For any $C > 49$, if $r = \sqrt{\frac{C\ln n}{n}}$, then with probability at least $2/3$ (over the choice of the graph),*

$$\begin{aligned}T_{\mathrm{mix}}(\epsilon) &\le \frac{30}{(1-\frac{7}{\sqrt{C}})^2}\cdot\frac{D}{n}\cdot\frac{1}{r^4}\cdot(\ln n+\ln\epsilon^{-1}).\\ T_{\mathrm{actual\_mix}}(\epsilon) &\le \frac{120}{(1-\frac{7}{\sqrt{C}})^2}\cdot\frac{1}{r^2}\cdot(\ln n+\ln\epsilon^{-1}).\end{aligned}$$

We prove the theorem by bounding the spectral gap of the Maximum Degree random walk using Theorem E.2. In order to define a distribution over canonical paths on $G^2(n,r)$, we introduce the notion of a *square grid* on the unit torus.

Square grid. Let $t = \lceil\frac{\sqrt{8}}{r}\rceil$. We divide the unit square $[0,1]^2$ into t^2 squares, each one of side length $1/t$. Each square is surrounded by eight neighboring squares. Consider any two nodes $u, v \in G^2(n,r)$ belonging to neighboring squares. Since the square side length is at most $r/\sqrt{8}$, then $\mathrm{cd}(u_x, v_x) \le 2r/\sqrt{8} = r/\sqrt{2}$ and similarly $\mathrm{cd}(u_y, v_y) \le 2r/\sqrt{8} = r/\sqrt{2}$. It follows that $\mathrm{gd}(u,v) \le r$, implying u and v are neighbors in the graph $G^2(n,r)$. We next prove that with high probability each square in the grid contains about n/t^2 nodes:

PROPOSITION E.3. *Fix any $0 < \alpha_s < 1$. Let*

$$\delta_s = \sqrt{\frac{3t^2}{n}\cdot\ln\frac{2t^2}{\alpha_s}}.$$

With probability at least $1-\alpha_s$ (over the choice of the random graph), every square in the square grid contains between $(n/t^2)\cdot(1-\delta_s)$ and $(n/t^2)\cdot(1+\delta_s)$ nodes of $G^2(n,r)$.

PROOF. Fix any square C in the square grid. For each $i=1,\ldots,n$, let X_i be the 0-1 random variable indicating whether the i-th node of $G^2(n,r)$ lands in C or not. Clearly, $E(X_i)=\Pr(X_i=1)=1/t^2$.

Let $X=\sum_{i=1}^{n}X_i$ be the total number of nodes of $G^2(n,r)$ that fall into C. By linearity of expectation, $E(X)=n/t^2$. By Chernoff bounds,

$$\Pr(|X-E(X)|>\delta_s E(X))\leq 2\cdot\exp(-\frac{\delta_s^2 E(X)}{3}).$$

Then,

$$\Pr(|X-\frac{n}{t^2}|>\delta_s\cdot\frac{n}{t^2})\leq 2\cdot\exp(-\frac{\delta_s^2 n}{3t^2})=\frac{\alpha_s}{t^2}.$$

The total number of squares is t^2. Hence, by the union bound, the probability there is a square that contains less than $\frac{n}{t^2}\cdot(1-\delta_s)$ nodes or more than $\frac{n}{t^2}\cdot(1+\delta_s)$ nodes is at most α_s. □

Square paths. Fix any realization G of the random graph $G^2(n,r)$ that has at least one node in each of the squares of the square grid (by Proposition E.3 the vast majority of the realizations of $G^2(n,r)$ have this property). Let $u\neq v$ be any two distinct nodes in this graph. We next define a family of paths between u and v, which we call *square paths*. Let C_u be the square to which u belongs and let C_v be the square to which v belongs (possibly, $C_u=C_v$). Let L_{uv} be the shortest straight line connecting u and v. Let $C_1,\ldots,C_k$ be the sequence of squares through which L_{uv} passes. Clearly, $C_1=C_u$ and $C_k=C_v$. A *square path* is a sequence $u_1,\ldots,u_k$ of k nodes that satisfies the following:

(1) $u_1=u$.
(2) $u_k=v$.
(3) For every $i=1,\ldots,k$, u_i belongs to C_i.

Note that there can be many square paths connecting u and v. We next show an upper bound on the length of square paths:

PROPOSITION E.4. *Let G be any realization of the random graph $G^2(n,r)$ that has at least one node in each square. Let $u\neq v\in G$ be any two distinct nodes. Then, every square path between u and v is of length at most $t+2$.*

PROOF. Let L_{uv} be the shortest straight line connecting $u=(u_x,u_y)$ and $v=(v_x,v_y)$. Let $C_1,\ldots,C_k$ be the squares through which L_{uv} passes and let $(x_1,y_1),\ldots,(x_k,y_k)$ be the bottom-left corners of these squares, respectively. For every $i=1,\ldots,k-1$, the squares C_i and C_{i+1} are neighboring squares, meaning that either $\mathrm{cd}(x_i,x_{i+1})=1/t$ and/or $\mathrm{cd}(y_i,y_{i+1})=1/t$. Since $\mathrm{cd}(u_x,v_x)\leq 1/2$, then the number of $i\in\{2,\ldots,k-1\}$ for which $\mathrm{cd}(x_i,x_{i+1})=1/t$ is at most $\lfloor\frac{1}{2}/\frac{1}{t}\rfloor\leq\frac{t}{2}$. Similarly, the number of $i\in\{2,\ldots,k-1\}$ for which $\mathrm{cd}(y_i,y_{i+1})=1/t$ is at most $\frac{t}{2}$. We conclude that k can be at most $t+2$. □

Canonical path distribution. Fix any realization G of the random graph $G^2(n,r)$ that has at least one node in each square. Let Γ_G be the set of all families of canonical paths $\gamma=\{\gamma_{uv}\}_{u,v\in G}$ on G. We now define a probability distribution p_G on Γ_G. The support of

p_G will consist only of families of square paths. We pick such a family γ as follows. The $\binom{n}{2}$ paths in γ are selected independently. For each $u \neq v \in G$, a canonical path between u and v is chosen uniformly at random among all the square paths between u and v. By Proposition E.4, we have an immediate bound on the maximum path length of p_G:

$$\ell(p_G) \leq t + 2.$$

Before we prove the upper bound on the maximum expected edge load of p_G, we show the next upper bound on the number of paths that pass through each square:

LEMMA E.5. *Let*

$$\delta_\ell = \frac{t}{\sqrt{\binom{n}{2} \cdot \alpha_\ell}}.$$

With probability at least $1 - \alpha_\ell$ *(over the choice of the random graph), the number of paths passing through each square of the square grid is at most* $\binom{n}{2} \cdot (\frac{1}{t} + \frac{2}{t^2}) \cdot (1 + \delta_\ell)$.

PROOF. Fix any square C. Let $U_1, \ldots, U_n$ be the n random points chosen on the surface of the unit torus. For each $i \neq j$, let $L_{U_iU_j}$ be the shortest straight line connecting U_i and U_j. We define X_{ij}^C to be the 0-1 random variable indicating whether C intersects $L_{U_iU_j}$ or not. Let $X^C = \sum_{1 \leq i < j \leq n} X_{ij}^C$ be the number of paths passing through C. Our goal is to show that X^C is small with high probability. To this end, we first bound the expectation of X^C and then use Chebyshev's inequality to prove that with high probability X^C does not exceed its expectation by much.

By linearity of expectation, $E(X^C) = \sum_{1 \leq i < j \leq n} E(X_{ij}^C)$. It thus suffices to bound the expectation of X_{ij}^C:

CLAIM E.6. *For all* C, i, j, $\frac{1}{t^2} \leq E(X_{ij}^C) \leq \frac{1}{t} + \frac{2}{t^2}$.

PROOF. For every square C and every two points u, v on the torus, define:

$$T(C, u, v) = \begin{cases} 1 & \text{if } C \text{ intersects the line } L_{uv} \\ 0 & \text{otherwise} \end{cases}$$

Let U and V be uniformly chosen points on the torus surface. Clearly, $E(X_{ij}^C) = E(T(C, U, V))$. $E(T(C, U, V))$ is the probability that the line L_{UV} intersects C. We next show that this probability is the same for all C:

CLAIM E.7. *Let* U *and* V *be uniformly chosen points on the surface of the torus and let* L_{UV} *be the shortest straight line connecting* U *and* V*. Then all squares in the square grid are equally likely to intersect* L_{UV}*.*

PROOF. The proof is based on the symmetry of the torus. For each square C, let S_C be the set of pairs of points whose shortest connecting line passes through C:

$$S_C = \{(u, v) \mid L_{uv} \cap C \neq \emptyset\}.$$

Fix any two squares C, C'. We would like to show that $|S_C| = |S_{C'}|$. That would imply that all squares are equally likely to intersect the line L_{UV} connecting two random points U, V on the torus surface. To this end, we define a 1-1 function f from the torus surface to itself and prove that f induces a 1-1 mapping from S_C onto $S_{C'}$.

Let (x, y) and (x', y') be the leftmost bottom corners of C and C', respectively. We define the function f as follows. For every point $w = (w_x, w_y)$:

$$f(w_x, w_y) = ((w_x + \mathrm{cd}(x, x')) \bmod 1, (w_y + \mathrm{cd}(y, y')) \bmod 1).$$

To show f is 1-1 we present an inverse mapping:

$$g(z_x, z_y) = ((z_x - \mathrm{cd}(x, x')) \bmod 1, (z_y - \mathrm{cd}(y, y')) \bmod 1).$$

Indeed, let $w = (w_x, w_y)$ be any point on the torus surface. The x-coordinate of $g(f(w))$ is:

$$\begin{aligned}
&((w_x + \mathrm{cd}(x, x')) \bmod 1 - \mathrm{cd}(x, x')) \bmod 1 \\
&= (w_x \bmod 1 + \mathrm{cd}(x, x') \bmod 1 - \mathrm{cd}(x, x') \bmod 1) \bmod 1 \\
&= (w_x \bmod 1) \bmod 1 = w_x.
\end{aligned}$$

Similarly, the y-coordinate of $g(f(w))$ is w_y and hence $g(f(w)) = w$.

We observe that f maps lines to lines and squares to squares. Furthermore, $f(C) = C'$. Let F be the following mapping from S_C: $F(u, v) = (f(u), f(u))$. Next, we prove that F is a 1-1 mapping from S_C onto $S_{C'}$. Let (u, v) be any pair in S_C. This means that C intersects the line L_{uv}. Let w be a point in $C \cap L_{uv}$. Since f maps lines to lines, then $f(w)$ must lie also on the line $L_{f(u)f(v)}$ connecting $f(u)$ and $f(v)$. On the other hand, since $w \in C$ and $f(C) = f(C')$, then $f(w) \in C'$. We conclude that $L_{f(u)f(v)}$ intersects C', and thus $(f(u), f(v)) \in S_{C'}$. F is then a mapping from S_C to $S_{C'}$. It is 1-1 due to the fact f is 1-1. A similar argument can show that $G(u', v') = (g(u), g(v))$ (where $g = f^{-1}$) is the inverse mapping of F. Hence, F is a 1-1 mapping from $S_{C'}$ onto S_C implying S_C and $S_{C'}$ are of equal size. □

Going back to the proof of Claim E.6, since $E(T(C, U, V))$ is independent of C, we can write it as:

$$E(T(C, U, V)) = \frac{1}{t^2} \sum_{C'} E(T(C', U, V)),$$

where the summation is over all squares C' in the grid and t^2 is the number of such squares. By linearity of expectation,

$$\frac{1}{t^2} \sum_{C'} E(T(C', U, V)) = \frac{1}{t^2} E(\sum_{C'} T(C', U, V)).$$

Since for every u, v, the number of squares that intersect L_{uv} is at least 1 and at most $t + 2$ (Proposition E.4), then

$$1 \leq \sum_{C'} T(C', U, V) \leq t + 2.$$

We conclude that:

$$E(X_{ij}^C) = E(T(C, U, V)) \leq \frac{1}{t^2} \cdot (t + 2) = \frac{1}{t} + \frac{2}{t^2}$$

and

$$E(X_{ij}^C) = E(T(C, U, V)) \geq \frac{1}{t^2}.$$

We conclude from the above claim that

$$E(X^C) \leq \binom{n}{2} \cdot (\frac{1}{t} + \frac{2}{t^2}).$$

Next, we prove that the sequence of random variables $\{X_{ij}^C\}_{1 \leq i < j \leq n}$ is pairwise independent:

CLAIM E.8. *For every $(i,j) \neq (i',j')$, X_{ij}^C and $X_{i'j'}^C$ are independent.*

PROOF. If $\{i,j\} \cap \{i',j'\} = \emptyset$, then the independence of X_{ij}^C and $X_{i'j'}^C$ follows from the independence of the two pairs (U_i, U_j) and $(U_{i'}, U_{j'})$. So suppose, e.g., that $i = i'$ but $j \neq j'$. The independence will follow from the following stronger claim: for every two points u, v on the torus surface,

$$\Pr(X_{ij}^C = 1 \mid U_i = u, U_{j'} = v) = \Pr(X_{ij}^C = 1).$$

Since X_{ij}^C and U_i are independent of $U_{j'}$, it suffices to show that for all u,

$$\Pr(X_{ij}^C = 1 \mid U_i = u) = \Pr(X_{ij}^C = 1).$$

That is, even if we know that U_i was chosen to be u, this does not change the probability of the line $L_{U_iU_j}$ to pass through C. This statement follows by a symmetry argument, similar to the one done in the proof of Claim E.7: for every fixed point u, all squares are equally likely to intersect the line L_{uV} (where V is chosen at random). □

We now finally return to the proof of Lemma E.5. Since $E(X^C) \leq \binom{n}{2} \cdot (\frac{1}{t} + \frac{2}{t^2})$, then

$$\begin{aligned} \Pr(X^C > \binom{n}{2} \cdot (\frac{1}{t} + \frac{2}{t^2})(1+\delta_\ell)) &\leq \Pr(X^C > E(X^C) \cdot (1+\delta_\ell)) \\ &\leq \Pr(|X^C - E(X^C)| > \delta_\ell \cdot E(X^C)). \end{aligned}$$

By Chebyshev's inequality,

$$\Pr(|X^C - E(X^C)| > \delta_\ell \cdot E(X^C)) \leq \frac{\text{var}(X^C)}{\delta_\ell^2 \cdot E^2(X^C)}.$$

Recall that $X^C = \sum_{i,j} X_{ij}^C$. The random variables $\{X_{ij}^C\}_{i,j}$ are identically distributed and pairwise independent. Let $p = \Pr(X_{ij}^C = 1)$. Then, $\text{var}(X^C) = \binom{n}{2} \cdot p(1-p) \leq \binom{n}{2} \cdot p$. On the other hand, $E(X^C) = \binom{n}{2} \cdot p$. Therefore,

$$\frac{\text{var}(X^C)}{\delta_\ell^2 \cdot E^2(X^C)} \leq \frac{1}{\delta_\ell^2 \binom{n}{2} p}.$$

By Claim E.6, $p \geq 1/t^2$. Also recall that $\delta_\ell = t/\sqrt{\binom{n}{2} \cdot \alpha_\ell}$. We conclude that

$$\Pr(X^C > \binom{n}{2} \cdot (\frac{1}{t} + \frac{2}{t^2}) \cdot (1+\delta_\ell)) \leq \frac{\alpha_\ell}{t^2}.$$

Using the union bound and based on the fact there are t^2 squares, the probability there is at least one square C for which $X^C > \binom{n}{2} \cdot (\frac{1}{t} + \frac{2}{t^2}) \cdot (1+\delta_\ell)$ is at most α_ℓ.

We are now ready to prove the upper bound on the maximum expected edge load of p_G:

LEMMA E.9. *Fix any $0 < \alpha_\ell, \alpha_s < 1$. Let G be a random realization of the random graph $G^2(n,r)$ and let p_G be the canonical path distribution defined above. Then, with probability at least $1 - \alpha_\ell - \alpha_s$ (over the choice of G),*

$$\rho(p_G) \leq \frac{t^3}{16} \cdot (1 + \frac{2}{t}) \cdot \frac{1+\delta_\ell}{(1-\delta_s)^2}.$$

PROOF. Fix any square C. By Lemma E.5, with probability at least $1 - \alpha_\ell$, the number of paths that pass through C is at most

$$\binom{n}{2} \cdot (\frac{1}{t} + \frac{2}{t^2}) \cdot (1 + \delta_\ell).$$

The canonical path distribution disseminates the paths that pass through C evenly among the nodes in C. By Proposition E.3, with probability at least $1 - \alpha_s$, the number of nodes in C is at least

$$\frac{n}{t^2} \cdot (1 - \delta_s).$$

Fix any node $u \in C$. We conclude that with probability at least $1 - \alpha_\ell - \alpha_s$, the expected number of paths that pass through u is at most

$$\frac{\binom{n}{2} \cdot (\frac{1}{t} + \frac{2}{t^2}) \cdot (1 + \delta_\ell)}{\frac{n}{t^2} \cdot (1 - \delta_s)} \leq \frac{n}{2} \cdot t \cdot (1 + \frac{2}{t}) \cdot \frac{1+\delta_\ell}{1-\delta_s}.$$

A symmetry argument similar to the one shown in the proof of Claim E.6 can show that in expectation exactly 1/8 of the paths that pass through square C go to each one of its neighboring squares. Fix a neighboring square C'. Recall that any node in C is connected to any node in these neighboring squares. Hence, 1/8 of the paths that pass through u are expected to use the edges that connect u with nodes in C'. Since the canonical path distribution picks a random node from each square independently, then all the edges that connect u and C' are expected to carry the same load. This load then equals the number of paths that pass through u divided by 8 and divided again by the number of nodes in C'. We already know (Proposition E.3) that the number of nodes in C' is at least $\frac{n}{t^2} \cdot (1 - \delta_s)$, hence the expected load on edges connecting u with C' is at most:

$$\frac{\frac{n}{2} \cdot t \cdot (1 + \frac{2}{t}) \cdot \frac{1+\delta_\ell}{1-\delta_s}}{8 \cdot \frac{n}{t^2} \cdot (1 - \delta_s)} = \frac{t^3}{16} \cdot (1 + \frac{2}{t}) \cdot \frac{1+\delta_\ell}{(1-\delta_s)^2}.$$

Since the choice of C, C' and u was arbitrary this is also the maximum expected load on edges of the graph G. □

We are now ready to prove Theorem 3.4:

PROOF PROOF OF THEOREM 3.4. Suppose D is the upper bound on $d_{\max}$ used in the random walk. We start by analyzing the standard mixing time of the MD random walk, including the self loops. Let P be the probability transition matrix of the random walk. By Theorem 3.2,

$$T_{\text{mix}}(\epsilon) \leq \frac{\ln n + \ln(1/\epsilon)}{1 - \lambda_{\max}(P)}.$$

By the strong version of Sinclair's bound (Theorem E.1),

$$\frac{1}{1 - \lambda_{\max}(P)} \leq \frac{D}{n} \cdot \ell(p_G) \cdot \rho(p_G).$$

Hence,

$$T_{\mathrm{mix}}(\epsilon) \;\leq\; \frac{D}{n} \cdot \ell(p_G) \cdot \rho(p_G) \cdot (\ln n + \ln(1/\epsilon)).$$

We set $\alpha_d = \alpha_s = \alpha_\ell = 1/9$. Then, with probability at least $2/3$, the chosen random graph G satisfies the three following conditions:

(1) By Proposition 3.3, its maximum degree, $d_{\max}$, is at most $\pi r^2(n-1) \cdot (1+\delta_d)$.

(2) By Proposition E.4, $\ell(p_G) \leq t+2$.

(3) By Lemma E.9, $\rho(p_G) \leq \frac{t^3}{16} \cdot (1+\frac{2}{t}) \cdot \frac{1+\delta_\ell}{(1-\delta_s)^2}$.

Therefore,

$$\begin{aligned} T_{\mathrm{mix}}(\epsilon) \;&\leq\; \frac{D}{n} \cdot (t+2) \cdot \frac{t^3}{16} \cdot (1+\frac{2}{t}) \cdot \frac{1+\delta_\ell}{(1-\delta_s)^2} \cdot (\ln n + \ln(1/\epsilon)) \\ &=\; \frac{D}{n} \cdot \frac{(t+2)^2 \cdot t^2}{16} \cdot \frac{1+\delta_\ell}{(1-\delta_s)^2} \cdot (\ln n + \ln(1/\epsilon)). \end{aligned}$$

Now, recall that $t = \lceil \sqrt{8}/r \rceil \leq \sqrt{8}/r + 1$. Therefore, $t+2 \leq \sqrt{8}/r + 3$. r was chosen so that $r \leq 1/2$, hence $\sqrt{8}/r + 3 \leq (\sqrt{8}+3/2)/r < 5/r$. Similarly, $t^2 \leq (\frac{\sqrt{8}}{r}+1)^2 = \frac{8}{r^2} + 1 + \frac{2\sqrt{8}}{r} \leq \frac{12}{r^2}$. Therefore, $(t+2)^2 \cdot t^2/16 < 19/r^4$. We conclude that:

$$T_{\mathrm{mix}}(\epsilon) \;\leq\; 19 \cdot \frac{D}{n} \cdot \frac{1}{r^4} \cdot \frac{1+\delta_\ell}{(1-\delta_s)^2} \cdot (\ln n + \ln(1/\epsilon)).$$

Recall that:

$$\delta_\ell = \frac{t}{\sqrt{\binom{n}{2} \cdot \alpha_\ell}}$$

and

$$\delta_s = \sqrt{\frac{3t^2}{n} \cdot \ln \frac{2t^2}{\alpha_s}}.$$

Since $r \leq 1/2$, then $t \leq \sqrt{8}/r + 1 \leq 4/r$. Also, $\alpha_\ell = 1/9$. Hence,

$$\delta_\ell \leq \frac{12\sqrt{2}}{r\sqrt{n(n-1)}}.$$

Recall that $r = \sqrt{C \ln n/n}$ for $C > 49$ and that $n \geq 10$. Therefore, $\delta_\ell < 0.55$.

As for δ_s, $t^2 \leq 12/r^2$ and $\alpha_s = 1/9$. Hence,

$$\delta_s \leq \sqrt{\frac{36}{r^2 n} \ln \frac{216}{r^2}}.$$

Rewriting r as $\sqrt{C \ln n/n}$, we have:

$$\delta_s \leq \sqrt{\frac{36}{C \ln n} \ln \frac{216n}{C \ln n}} = \sqrt{\frac{36}{C} \cdot \left(1 + \frac{\ln(\frac{216}{C \ln n})}{\ln n}\right)}.$$

Since $C > 49$ and $n \geq 10$, then $\delta_s < \sqrt{47/C} < 7/\sqrt{C}$. By incorporating the bounds on δ_ℓ and δ_s, we obtain the desired bound on the mixing time:

$$T_{\mathrm{mix}}(\epsilon) \;\leq\; \frac{30}{(1-\frac{7}{\sqrt{C}})^2} \cdot \frac{D}{n} \cdot \frac{1}{r^4} \cdot (\ln n + \ln(1/\epsilon)).$$

We now turn to the calculation of the actual mixing time. Consider a run of the MD random walk, and let $U_1, U_2, \ldots$ be the distinct nodes visited during the random walk. (Note that $U_1, U_2, \ldots$ are random variables.) For each $i = 1, 2, \ldots$, let X_i denote the number of steps the random walk spends at U_i. That is, X_i is 1 plus the number of steps the random walk spends at the self loop of U_i until moving to U_{i+1}. For every infinite sequence of nodes $v_1, v_2, \ldots$ the random variables $X_1, X_2, \ldots$ are independent given that $U_1 = v_1, U_2 = v_2, \ldots$ (that is, the number of self loop steps spent at U_i depends only on U_i and not on the other nodes visited during the random walk).

Consider any step i. Given that $U_i = v_i$, X_i is a geometric random variable with probability of success d_{v_i}/D, where d_{v_i} is the degree of v_i, excluding the weighted self loop. Hence, $E(X_i | U_1 = v_1, U_2 = v_2, \ldots, U_i = v_i, \ldots) = E(X_i | U_i = v_i) = D/d_{v_i}$. Let $d_{\max} = \max_v d_v$. Then, $E(X_i | U_1 = v_1, U_2 = v_2, \ldots) \geq D/d_{\max}$.

Let $m = T_{\mathrm{mix}}(\epsilon)$ be the mixing time of the random walk. The random walk runs for m steps, including self loop steps, until it is stopped. Let T denote the number of non-self loop steps made by the random walk. Note that T is a random variable and $E(T)$ is the actual mixing time $T_{\mathrm{actual\_mix}}(\epsilon)$ we wish to calculate. Furthermore, $\sum_{i=1}^{T} X_i = m$. Since for every sequence of nodes $v_1, v_2, \ldots$, the random variables $X_1, X_2, \ldots$ are independent given that $U_1 = v_1, U_2 = v_2, \ldots$, the conditions of Wald's identity (cf. [Siegmund 1985]) are met, implying that:

$$E(\sum_{i=1}^{T} X_i \mid U_1 = v_1, U_2 = v_2, \ldots) \geq E(T \mid U_1 = v_1, U_2 = v_2, \ldots) \cdot \frac{D}{d_{\max}}.$$

Since $\sum_{i=1}^{T} X_i = m$ always, we have:

$$E(T \mid U_1 = v_1, U_2 = v_2, \ldots) \leq m \cdot \frac{d_{\max}}{D}.$$

This holds for every sequence $v_1, v_2, \ldots$. Thus,

$$E(T) \leq m \cdot \frac{d_{\max}}{D}.$$

Hence,

$$\begin{aligned} T_{\mathrm{actual\_mix}}(\epsilon) \;&=\; E(T) \;\leq\; m \cdot \frac{d_{\max}}{D} \;=\; T_{\mathrm{mix}}(\epsilon) \cdot \frac{d_{\max}}{D} \\ &\leq\; \frac{30}{(1-\frac{7}{\sqrt{C}})^2} \cdot \frac{D}{n} \cdot \frac{1}{r^4} \cdot (\ln n + \ln(1/\epsilon)) \cdot \frac{d_{\max}}{D} \\ &=\; \frac{30}{(1-\frac{7}{\sqrt{C}})^2} \cdot \frac{d_{\max}}{n} \cdot \frac{1}{r^4} \cdot (\ln n + \ln(1/\epsilon)). \end{aligned}$$

Recall that we assumed the chosen random graph satisfies

$$d_{\max} \leq \pi r^2 (n-1) \cdot (1 + \delta_d),$$

where

$$\delta_d = \sqrt{\frac{3}{\pi r^2(n-1)} \cdot \ln \frac{2n}{\alpha_d}}.$$

Writing r as $\sqrt{C \ln n/n}$ and recalling that $\alpha_d = 1/9$, we have:

$$\delta_d = \sqrt{\frac{3n}{\pi C \ln n(n-1)} \cdot \ln(18n)}.$$

Since $C > 49$ and $n \geq 10$, we have: $\delta_d < 0.25$. Therefore, $d_{\max} \leq 1.25\pi r^2(n-1) < 4r^2 n$. Substituting in the bound for $T_{\text{actual\_mix}}(\epsilon)$, we have:

$$T_{\text{actual\_mix}}(\epsilon) \;\leq\; \frac{120}{(1-\frac{7}{\sqrt{C}})^2} \cdot \frac{1}{r^2} \cdot (\ln n + \ln(1/\epsilon)).$$

□