A New Random Walk for Efficient Data Collection in Sensor Networks

C.M. Angelopoulos  S. Nikoletseas  D. Patroumpa  C. Raptopoulos
{aggeloko, patroump, raptopox}@ceid.upatras.gr, nikole@cti.gr
University of Patras and
Research Academic Computer Technology Institute (RACTI),
Greece

ABSTRACT
Motivated by the problem of efficiently collecting data from wireless sensor networks via a mobile sink, we present an accelerated random walk on Random Geometric Graphs. Random walks in wireless sensor networks can serve as fully local, very simple strategies for sink motion that significantly reduce energy dissipation but introduce higher latency in the data collection process. While in most cases random walks are studied on graphs like $G_{n,p}$ and Grid, we define and experimentally evaluate our newly proposed random walk on the Random Geometric Graphs model, that more accurately abstracts spatial proximity in a wireless sensor network. We call this new random walk the $\gamma$-stretched random walk, and compare it to two known random walks; its basic idea is to favour visiting distant neighbours of the current node towards reducing node overlap. We also define a new performance metric called Proximity Cover Time which, along with other metrics such as visit overlap statistics and proximity variation, we use to evaluate the performance properties and features of the various walks.

Categories and Subject Descriptors
C.2.1 [Network Architecture and Design]: Distributed networks, Network topology, Wireless communication

Keywords
Wireless Sensor Networks, Random Walks, Data Collection, Sink Mobility

1. INTRODUCTION
Wireless Sensor Networks are envisioned as large ad-hoc collections of very small autonomous devices, that can sense environmental conditions in their immediate surroundings while having limited processing, communication capabilities and energy reserves. The collected sensory data is usually disseminated to a static control center (called data sink) in the network, using node to node multi-hop data propagation. Such settings have increased implementation complexity and sensor devices consume significant amounts of energy. This is due to the fact that multi-hop data propagation protocols in sensor networks, where the sink is static, leads the sensor devices to consume significant amounts of energy in operations other than sensing (e.g. inter-node communication for synchronisation purposes, exploratory messages, etc). Furthermore, in the area around the control center, nodes need to heavily relay the data from the entire network, thus a bottleneck of increased energy consumption emerges and failures due to strained energy resources of these nodes leads to an early disconnected and dysfunctional network (see e.g. [6]).

Sink mobility can be used as a simple and energy efficient alternative for data collection. The sink is lifting the burden of inter-node coordination by traversing the network itself. A mobile sink may also substitute connectivity as it is capable of bypassing obstacles and reaching disconnected components of the network. Sensor motes maintain a rather passive role, in terms of data propagation, by simply waiting to contact the sink. When contact is established they deliver data via cheap (energy wise) one-hop transmissions.

Apparently, in such data collection schemes, new critical issues emerge regarding the pattern of movement to be adopted by the sink. The network has to be traversed in a timely and efficient manner. Also, the mobility pattern of the sink should guarantee that the entire network area, or at least the vast part of it, will eventually be visited.

In the light of the above discussion, random walks in wireless sensor networks can serve as fully local, very simple, distributed strategies for sink motion that significantly reduce energy dissipation and also probabilistically guarantee that eventually the entire network area will be covered. In this study, we choose to model wireless sensor networks via Random Geometric Graphs (RGG). We take this option in contrast to other graph models (e.g. $G_{n,p}$ and Grid), that are well established in random walk studies, as RGG better capture certain characteristics of WSN’s such as link existence dependencies of neighbouring nodes due to geometric
proximity.

Our goal is mainly to accelerate the random walk coverage of the network via reducing the node overlaps. Also, to improve other (network related) features of the walk, such as how fast it gets close to the network nodes.

2. RELATED WORK AND COMPARISON
Random Walks have been extensively studied in the past decades in the context of several disciplines. However, despite their numerous applications in ad hoc and overlay networks, not much research has been done on how they can be particularly applied in sensor networks with mobile entities, in a way that addresses the peculiarities of such networks (such as severe computing and communication constraints, the small memory and constrained battery, as well as the time-criticality of important applications).

A comparison of different random walk strategies for ad hoc networks is performed in [4]. The authors investigate the effectiveness of each strategy in terms of the expected hop count and the occurrence of deadlocks. In our work, we propose a new random walk strategy and compare two variations of it with two known walks. We study the performance of the strategies in terms of cover time, approximate cover time, proximity variation and a new metric that we define here, proximity cover time.

In [7] the problem of data gathering in a large-scale wireless sensor network with static nodes and a mobile patrol node is formulated as a classical random walk on a random geometric graph. The authors derive analytical bounds for the performance of the random walk in terms of node coverage. In order to improve this performance they propose an algorithm to constrain the random walk using the available side information, such as the awareness of previously visited sites. We also use some information (just the information of geometric distance) in order to speed up the network traversal and our new walks are different.

In [1] the authors try to achieve satisfactory energy-latency trade-offs via adaptive random walks, depending on network parameters such as density and/or history of past visits in each network region. Towards this direction they propose three new random walks which they evaluate on a grid overlaying the wireless sensor network. Here, in order to get more insight of the random walks’ behavior, we use a more detailed network model to abstract WSNs, the so-called random geometric graphs. Furthermore, our new random walk only uses local information concerning the geometric distance of neighbouring nodes.

In [11] authors investigate the Random Waypoint Model, which is widely used in the simulation studies of mobile ad hoc networks and show that it fails to provide a steady state in that the average nodal speed consistently decreases over time; therefore it should not be directly used for simulation. They also propose a simple fix of the problem and discuss a few alternatives. In our study, the moving element of the network (the sink) is visiting the nodes in a sequential way. Therefore we abstract its motion by considering that it is moving on the edges of a Random Geometric Graph, that models a wireless sensor network.

Nice research has also been conducted for deterministic sink mobility, such as in [8, 9] where authors address the problem of maximizing the network lifetime. In [8] authors suggest that the base station can be mobile and conclude that the best mobility strategy consists in following the periphery of the network (assuming that the sensors are deployed within a circle). In [9] authors investigate the approach that makes use of a mobile sink for balancing the traffic load and in turn improving network lifetime. They engineer a routing protocol that effectively supports sink mobility and through intensive simulations they prove the feasibility of their mobile sink approach.

3. THE NEW RANDOM WALK AND PERFORMANCE METRICS

3.1 The network model
Sensor networks comprise of a vast number of ultra-small homogeneous sensor devices (which we also refer to as sensors), whose purpose is to monitor local environmental conditions. Each sensor is a fully-autonomous computing and communication device, characterized mainly by its available power supply (battery), its transmission range \( r \), the energy cost of data transmission and the (limited) processing and memory capabilities. Sensors (in our model here) do not move. The positions of sensors within the network area are random and follow a uniform distribution. We focus on data collection methods, so we assume that initially all sensors have some data to deliver to the sink. For clarity, we also assume that no data is generated during the network traversal. That is if a node is called “visited”, then it has no data to send to the sink.

There is a special node within the network region, which we call the sink \( S \), that represents a control center where data should be collected. Here, we assume that the sink is mobile. The sink is not resource constrained i.e. it is assumed to be powerful in terms of computing, memory and energy supplies.

We consider that the random uniform placement of the sensors inside the network area is abstracted by a Random Geometric Graph. Random Geometric Graphs are formed by \( n \) vertices that are placed uniformly at random in the \([0,1]^2\) square. An edge \((u, v)\) exists iff the Euclidean distance of vertices \( u \) and \( v \) is at most \( r \), where \( r \) corresponds to the wireless communication radius \( r \) of the sensors. This holds assuming a disc radio model; two sensors can communicate with each other iff each one lies inside the communication range of the other. Random Geometric Graphs also have an important nice property: unlike other random graphs, like \( G_{n,p} \), edges are not statistically independent of each other. That is, the existence of an edge \((u, v)\) is not independent of the existence of edges \((u, w)\) and \((w, v)\). This property makes RGG a quite realistic model for wireless sensor networks that captures to a great extent the communication structure of real WSNs (at least their spatial aspects).

More strictly, consider an area \( A \subset \mathbb{R}^2 \) in two dimensional space. An instance of the random geometric graphs model \( G(X_n; r) \) is constructed as follows: select \( n \) points \( X_n \) uniformly at random in \( A \). The set \( V = X_n \) is the set of vertices of the graph and we connect two vertices if their euclidean distance is at most \( r \).
distance is at most \( r \). For any vertex \( v \in V \) we will denote by \( N(v) \) the set of neighbours of \( v \) and by \( \deg(v) = |N(v)| \) its degree. Furthermore, we will denote by \( \|u-v\| \) the euclidean distance between the points corresponding to vertices \( v, u \).

In [5, 10] it is shown that the connectivity threshold for \( G(X_n; r) \) is \( r_c = \sqrt{\frac{\ln n}{n}} \). In this paper we will consider random instances of \( G(X_n; r) \) of varying density, by selecting \( r = \sqrt{\frac{\ln n}{n}} \), for different values of \( c > 1 \), which guarantees that the produced random instance is connected with high probability.

Besides the information about the set of neighbours of each vertex \( v \in V \), an instance \( G(X_n; r) \) of the random geometric graphs model also contains extra information about the exact euclidean distance between \( v \) and any of its neighbours. We used this information in order to define a new random walk, namely the \( \gamma \)-stretched random walk on \( G(X_n; r) \), which is described below. This random walk aims to accelerate the data collection process while keeping the memory requirements restricted. The basic idea is that “more distant the data collection process while keeping the memory which is described below. This random walk aims to accelerate the data collection process while keeping the memory requirements restricted. The basic idea is that “more distant” neighbours of the current vertex (state of the walk) are favoured in a probabilistic manner. Therefore, our new walk is in fact a biased transitions random walk (see e.g. [2] for a nice discussion of several types of random walks based on the assumptions they made).

### 3.2 The \( \gamma \)-Stretched Random Walk

Let \( G(X_n; r) \) be a random instance of the random geometric graphs model with vertex set \( V = X_n \), where \( |V| = n \). Consider a particle moving on the vertices of the graph \( G(X_n; r) \). Given that it occupies a specific vertex \( v \in V \) at time \( t \geq 0 \), it decides where to move at time \( t+1 \) by choosing a vertex \( u \in V \) with probability

\[
p_{u,v} = \begin{cases} \frac{|u-v|^\gamma}{\sum_{w \in N(v)} |w-v|^\gamma} & \text{if } u \in N(v) \\ 0 & \text{otherwise.} \end{cases}
\]

(1)

The Markov chain describing the above process will be called \( \gamma \)-stretched random walk on \( G(X_n; r) \) and will be denoted by \( W_{G(X_n; r)}^\gamma \). Furthermore, we will denote by \( W_{G(X_n; r)}^\gamma(t) \) the state of the walk that begins at \( v \) at time \( t \). More formally, the state space of the \( \gamma \)-stretched random walk is the set of vertices \( V = X_n \) of the graph and its transition probability matrix is given by \( P = [p_{u,v}]_{u,v \in V} \).

This new random walk has minimum memory requirements, since every step is decided only by using information of the current state. Notice also that the larger \( \gamma \) is, the more distant neighbours are favoured over neighbours that are close by. In the simulation, we will often set \( \gamma = \frac{2}{c} \), so that the bias to visit distant neighbours is stronger as the density of \( G(X_n; r) \) increases. The special case \( \gamma = 1 \) will be referred to simply as stretched random walk.

### 3.3 Known Random Walks

The \( \gamma \)-stretched random walk will be compared to the following two known random walks which we describe below.

#### 3.3.1 Blind Random Walk

This is the usual random walk model. The blind random walk on a random instance \( G(X_n; r) \) of the random geometric graphs model is the simplest of all possible sink mobility patterns, since the next move of the sink is stochastically independent to the previous ones. Furthermore, given that the current vertex is \( v \in V \), the probability of moving to any neighbouring vertex \( u \in N(v) \) is \( p_{v,u} = \frac{\deg(u)}{|N(v)|} \).

This method is very robust, since it probabilistically guarantees that eventually all network regions and nodes will be visited and thus all data will be collected given that the network is connected. However, in some network structures it may become inefficient, mostly with respect to high latency, since the sink uses no memory of the past movements in order to select the next one and thus overlaps (i.e. visits to already visited vertices) occur.

#### 3.3.2 Random Walk with Memory

The performance of the blind random walk can be improved using some memory of past visits. In random walk with memory \( K \), the sink maintains a first-in-first-out (FIFO) list \( M \) which contains the last \( K \) nodes visited during the random walk, i.e. \( M = \{ c_1, c_2, ..., c_K \} \). The next hop is chosen uniformly at random among the neighbours of the node that are not in the memory list \( M \). The use of memory eliminates the possibility of short loops in random walks. Setting \( K = 0 \), we simply get the blind random walk. In this study the random walk with memory 1 will be used for comparison purposes. Note that this walk has stronger memory requirements and overhead than both the blind and the \( \gamma \)-stretched random walk.

### 3.4 Metrics

In this section we define and discuss the metrics that will be used in the evaluation of our new random walk, when compared to the blind random walk and the random walk with memory 1. In the following, let \( W \) denote any random walk model defined on a random instance \( G(X_n; r) \) of the random geometric graphs model.

#### 3.4.1 Cover time and approximate cover time

Consider the random walk \( W_n \) that begins on vertex \( v \) of a random instance of the random geometric graphs model \( G(X_n; r) \). Define \( T_n \) as \( \inf\{t : \forall u \in V, \exists t' \leq t : W_n(t') = u \} \) to be the time needed until \( W_n \) has visited all the vertices in the graph. The cover time \( C \) of \( G(X_n; r) \) is defined as \( C = \max_{v \in V} \mathbb{E}[T_v] \), where \( \mathbb{E} \) denotes the expected value of the random variable \( T_v \). It was shown in [3] that when \( r = \sqrt{\frac{\ln n}{n}} \), for \( c > 1 \), the cover time of \( G(X_n; r) \) for the blind random walk is asymptotically \( c \ln \left( \frac{r}{\sqrt{c-1}} \right)n \ln n \).

For \( \epsilon \in (0, 1) \) define also the \( \epsilon \)-approximate cover time as the mean number of steps that \( W \) needs in order to visit a fraction \( 1 - \epsilon \) of the vertices. More formally, define \( T^{(\epsilon)}_n \) to be the time needed until \( W_n \) has visited \( (1 - \epsilon)n \) vertices. The \( \epsilon \)-approximate cover time \( C^{(\epsilon)} \) of \( G(X_n; r) \) is defined as \( C^{(\epsilon)} = \max_{v \in V} \mathbb{E}[T^{(\epsilon)}_v] \).

These cover times are related to latency, as they capture the time the sink needs to collect the sensory data from the
entire network. The approximate cover time metric is of great interest as the majority of overlaps occur while the sink tries to locate the last few unvisited sub-regions that are scattered in the network area; however, in most sensor network applications it is sufficient to collect a vast percentage of the total sensory data, so this metric is relevant and informative.

### 3.4.2 A new metric: Proximity Cover Time

In real-life sensor networks, the sink is capable of collecting data not only from the currently visited node, but also from all nodes that are inside the communication range $r$ (via a single-hop data transmission). Therefore, there is strong motivation from real-life WSN to consider as visited not only the vertex of the current state of the walk, but also all neighbouring vertices. Following is the formal definition of the corresponding metric, Proximity Cover Time.

Denote by $D(v, \rho)$ the disc of radius $\rho > 0$, centered at $v$. Consider the random walk $\mathcal{W}_v$ that begins on vertex $v$ of a random instance of the random geometric graphs model $G(X; r)$. We will say that $\mathcal{W}_v$ is within distance $\rho$ to vertex $u$ at time $t$ if $W_v(t) \in D(u, \rho)$, i.e. the random walk started at $v$ occupies a vertex that lies in $D(v, \rho)$ at time $t$.

Let $T_v(\rho) \overset{def}{=} \inf\{\forall u \in V; \exists t' : W_v(t') \in D(u, \rho)\}$ be the time needed until $\mathcal{W}_v$ has come within distance $\rho$ to all vertices of the graph. We define the $\rho$-proximity cover time of $G(X_n; r)$ as $C(\rho) = \max_{v \in V} E[T_v(\rho)]$.

A similar but different metric is presented in [2] where each time the random walk visits an already visited vertex, then u.a.r. picks an unvisited neighbour (if one exists), marks it as visited and then continues the random walk from its current vertex (i.e. it does not make a transition to the marked neighbour).

### 3.4.3 Proximity Variation

For a vertex $v \in V$ and time $t_0 \geq 0$ let

$$
dist(v, t_0) = \min_{0 \leq t \leq t_0} \|W(t) - v\|
$$

be the function that returns the minimum distance between the random walk $W$ up to time $t_0$ and node $v$. Then the proximity variation $PV(t_0)$ at time $t_0$, is given by

$$
PV(t_0) = \frac{\sum_{v \in V} dist(v, t_0)}{n}
$$

where $n$ is the total number of nodes.

The rationale behind this metric is the following: In a real network, if the PV metric converges to zero quickly with $t$, this means that the sink gets close to all sensors quite soon and data collection progresses fast; this is especially relevant in case when the role of sensors is not completely passive but includes some limited multi-hop propagation of data to accelerate data propagation at a reasonable energy cost. On the contrary, when the PV converges slowly to zero, it means that the network traversal is performed in a way that some areas may stay unvisited for long time. This metric was first introduced in [1].

### 3.4.4 Visit overlap statistics

Other metrics that will be used in order to further characterize each walk’s evolution include the number of visits to a specific vertex $v$, as well as the distribution of the number of visits for every vertex in the graph. These overlap statistics will provide an insight on how exactly each walk traverses the graph, how different components of the graph are visited and in what rate.

![Figure 1: Cover Time](image)

#### 4. PERFORMANCE EVALUATION

##### 4.1 Simulation Setup

We conducted our experimental evaluation using Matlab R2009a as our simulation environment. We evaluate the walks on several instances of the Random Geometric Graphs model in order to address its random nature. More particularly, we construct Random Geometric Graphs consisting of $n = 2000$ nodes placed uniformly at random in a $100 \times 100$ square area. For such RGG graphs, the connectivity threshold is computed to be of radius $r = 100\sqrt{\frac{\ln n}{2000}} = 3.47$. For our performance evaluation we use three characteristic values for $r$: $r_1 = 5$, $r_2 = 7$, $r_3 = 10$, corresponding to sparse, average and denser graphs respectively. For every value of $r$, we construct 50 RGG instances on which we evaluate all walks for 100 iterations per instance. We consider these randomly constructed RGG instances to be representative of the RGG space.

We experimentally evaluate four random walks. The well known Blind Random Walk; two versions of our newly proposed walk: the Stretched Walk where $\gamma = 1$ and the $\gamma$-Stretched Walk, where $\gamma = \frac{1}{2}$, in order to investigate the impact of $\gamma$; the Random Walk with Memory 1, that somehow represents an upper bound in terms of memory usage and therefore is considered to be more powerful than the rest of the walks.

##### 4.2 Simulation Findings

Figure 1 depicts the mean number of hops each walk needs in order to visit all the nodes of the network for three representative values of the communication range $r$. For all the random walks we observe that as the density of the graph increases, the total cover time is reduced even by nearly 50%.
This is due to the fact that in a dense network the walk has more edges to traverse on, and therefore there is higher probability to visit any given vertex. In other words, a random walk reaches its stationary distribution much faster. We also observe that for all graph densities, the Blind Random Walk has the biggest cover time, while the impact of the fixed-sized memory of the Walk with Memory declines as the density of the graph increases. In fact, in sparse graphs, where the available options are limited, avoiding 1-hop overlaps by not visiting the last position significantly reduces the cover time. In our Stretched Walks, on the contrary, the impact of the $\gamma$ factor is proportional to the number of neighbouring nodes as the graph density increases. This is why for small values of radius $r$ the performance of the Stretched Walks is very similar. However, in more dense graphs, biasing the walk towards the more distant neighbours seems to have a well noticeable effect on the speed the random walk traverses the graph.

Figures 2 and 3 depict the rate at which each walk discovers unvisited nodes inside the network area for range $r = 5$ and $r = 10$ respectively. For sparse networks ($r = 5$) we note that the Stretched Walk outperforms even Random Walk with Memory. The Random Walk with Memory, while avoiding its previous position, may move towards a neighbouring node that geometrically is very approximate; thus allowing a high probability for early overlaps. On the other hand, the Stretched Walk, by favouring the more distant neighbours, quickly changes the subregions of the networks it traverses. This way it avoids early overlaps, thus discovering unvisited nodes much faster. However, in sparse networks, if the bias factor towards distant neighbours is very crude it may have an opposite effect. As seen in the performance of the $\gamma$-Stretched Walk, a crude bias factor may narrow the available options for the next step down to the point the walk is forced to make long cycles during its traversal, thus visiting already visited nodes.

In dense networks ($r = 10$) the rate at which unvisited nodes are discovered is accelerated for all walks. However, $\gamma$-Stretched Walk performs slightly better as the bias factor strongly favours distant nodes over geometrically approximate ones. We note that for all walks while they all discover relatively quick the 95% of the total number of vertices in the graph, they spend nearly half of the cover time in order to visit the last 5% of the vertices, thus creating a long convergence tail at the end of the traversal.
As discussed in subsection 3.4.2, there is strong motivation from real-life wireless sensor networks to consider as visited not only the vertex of the current state of the walk, but also all neighbouring vertices in the RGG network. Figure 4 depicts the performance evaluation of all walks in RGG for the Proximity Cover Time metric. Intuitively, the dominant factor for this metric is the dislocation attribute of the motion (i.e. the mean geometric distance the sink covers per hop). In sparse networks the use of memory by the Random Walk with Memory slightly improves the Proximity Cover Time of the Blind Random Walk. The Stretched Walk with $\gamma=1$ shows the best performance from all other walks. The $\gamma$-Stretched Walk is outperformed due to the crude way it is biased towards the few distant neighbours. However, for higher network densities this factor aids the walk to outperform even the Walk with Memory. In a network consisting of 2000 nodes, in less than 2000 hops the sink has been inside the communication radius of every node in the network.

Figure 5 depicts the overlaps distribution over the nodes of the graph. That is what percentage of the total number of the nodes has been visited $t$ times by the sink during the entire network traversal process. In sparse graphs the Random Walk with Memory outperforms the two Stretched Walks, that is more nodes of the network have been visited fewer times (i.e. the curve corresponding to Walk with Memory is above and to the left than the rest). At first glance this contradicts the fact that the Stretched Walk is discovering unvisited nodes in sparse networks at a higher rate than the Walk with Memory (Fig. 2). We note however that the Stretched Walk avoids early overlaps, while it needs more time to cover the entire network (Fig. 1). This leads to the conclusion that most of the overlaps occur towards the end of the network traversal process, thus the converging tails in Fig. 2.

For more dense networks the impact of the bias factor of $\gamma$-Stretched Walk is increasing along with the value of $r$, thus eventually demonstrating the best performance for this metric. Also, as the network graph density increases, the number of revisits for the majority of the nodes is more or less the same (10-15 revisits). However, the total number of revisits is increased significantly, as the last few unvisited nodes are hard to find in a dense network.

There lies a different intuition behind each walk on how the network area is traversed, regardless of the total number of hops needed. For example the strategy of avoiding previous positions is getting less effective when the density of the network increases, while the $\gamma$ factor in Stretched walks is allowing the sink to visit many different network subregions very early in the traversal process. The proximity variance metric captures these differences. In Fig. 6 we note that the two walks which perform better in sparse graphs are the Stretched Walk (where $\gamma=1$) and the Random Walk with Memory. As there are relatively few neighbouring nodes slightly favouring distant nodes or just avoiding previous position seems to suffice. On the other hand, the $\gamma$-Stretched
5. CONCLUSIONS

In this work we address the problem of efficient data collection in wireless sensor networks via a mobile sink. We model the data collection process by random walks on Random Geometric Graphs as they better capture the geometric dependencies that characterise inter-node wireless communication in WSN’s. We use random walks as local, simple motion strategies for the sink that have very low computational and a-priori knowledge requirements. We define a new walk called Stretched Walk and evaluate two variations of it. The basic idea of the new walk is to favour distant neighbours of the current node towards avoiding visit overlaps. Also, motivated by real-life WSN’s, we define a new metric called Proximity Cover Time; this captures collecting data from all neighbour nodes of a currently visited node.

Findings show that the use of constant sized memory in order to accelerate the data collection process has limited results, particularly when the network density increases. Instead, walks that use local (immediate neighbour) information in a smart manner can demonstrate significantly better results. However, their use should be wise as they may lead to opposite results (e.g. strong bias factor in sparse networks leads to bigger cover times).

6. REFERENCES

2008.


