Using Heterogeneity to Enhance Random Walk-based Queries

Marco Zuniga, Chen Avin and Bhaskar Krishnamachari
Department of Electrical Engineering - Systems
University of Southern California, Los Angeles, CA 90089-0781
marcozun@usc.edu, avin@cs.ucla.edu, bkrishna@usc.edu

***Abstract*— We propose a push-pull mechanism to enhance the performance of random walk-based querying in heterogeneous sensor networks and analyze its performance. Using connections between random walks and electrical resistance, we obtain closed-form expressions and bounds of the querying cost for a linear topology, along with numerical solutions for more realistic topologies via Markov chain analysis of the time to absorption. We show that having even a small degree of heterogeneity can provide very significant improvements in query performance. Specifically, for linear topologies we prove that using a fraction $4/5k$ of uniformly-placed high-degree cluster-heads (where $2k$ is the degree of each such node), a query cost reduction of $\Theta(1-1/k^2)$ can be obtained. Realistic two-dimensional topologies also show a similar trend — using about 10% of the nodes as cluster-heads provides a query cost improvement between 30% and 70% depending on the coverage of the high-degree nodes.**

I. Introduction

An important approach for querying in unstructured systems is the use of random walks. This approach is gaining popularity in the networking community since random walks are intuitively simple — nodes are visited sequentially in a random order with successive nodes being neighbors in the graph [21], [22], [23], [24]. There is also a significant body of theoretical literature on random walks as querying mechanisms [9], [10], [11]. However, while this literature provides much insight into the scaling behavior of random walks on simple classes of deterministic graphs (such as 2D Torii), a major property of real-life networks, heterogeneity, was left out of discussion.

Heterogeneity on the degree distribution is a highly likely characteristic in real large-scale wireless systems, such as sensor networks, for a number of reasons. First, random deployments are inherently non-regular graphs. Second, empirical studies [2], [31] have revealed that hardware variance on the sensitivity and output power of radios lead to nodes with significantly higher degree than the average (cluster-heads). Third, some works [3] have recently proposed that for wireless sensor networks to scale, heterogenous networks consisting of highly capable and low capable devices are required.

In this work we propose the use of a push-pull mechanism to exploit the heterogeneity of the underlying communication graph to enhance the performance of random-walk-based queries. We take advantage of a well known property of the *simple* random walks: its stationary distribution $\pi(v) = d(v)/2m$, where $d(v)$ denotes the degree of node v and m the number of edges in the graph. This means that nodes with higher degree are visited more frequently by the random walk.

This work was supported in part by NSF through grants numbered CNS-0325875, CNS-0347621, CNS-0435505, and CCF-0430061.

The idea we use is intuitively simple, events are pushed towards high-degree nodes (cluster-heads) and pulled from the cluster-heads by performing a random-walk-based query. In this scenario some important questions arise: *What is the impact of heterogeneity on performance?* and *How much heterogeneity is needed?*

We use two theoretical tools to explore these questions. Our analytical results are based on the direct connection between a random walk on a graph and the *resistance* of the electrical network obtained from the graph by viewing each edge as a unit resistor [29], [30]. We bound the influence of cluster-heads on the resistance which in turn bound on the query cost. Our second tool is the use of absorption states in the transitional probability matrix of a graph G to obtain the expected number of steps in a query.

The main contribution of this work is to show that heterogeneity allows random-walk-based queries to enhance their performance. In particular, the striking result we obtain is that for a line topology where cluster-heads have a coverage k (cover k nodes to the right and k nodes to the left) and are uniformly distributed (evenly spaced), a fraction of $\frac{4}{5k}$ nodes being cluster-heads can offer a reduction in query cost of $O(1-\frac{1}{k^2})$ by using a simple distributed algorithm. In intuitive terms this translates to requiring less than 10% of the nodes being cluster-heads to obtain two orders of magnitude improvement in query cost. Through numerical analysis, we present results showing that in 2D networks also a small percentage of nodes being cluster-heads ($\sim$ 10%) can lead to significant improvements in performance (between 30% and 70% depending on the coverage of the high-degree nodes).

II. Enhancing Random Walks for Heterogeneity

In this section we present the event-push query-pull algorithm used in our work. We consider a network where two type of nodes are available: i) nodes with limited communication capability (low degree) and ii) nodes with higher communication capabilities (high degree, cluster-heads). Our focus is on infrastructure-less networks (no location nor GPS capabilities), nodes are able to communicate only with their neighbors, and they are aware of their neighbors degree (if neighbors are cluster-heads or not).

Our query mechanism is built from two parts: first an event e is generated randomly at any node in the network, and second, a random-walk-based query is issued in order to find the event. The event e can either remain at the node where it was generated or move to a cluster-head (if one exists). In a network where all nodes have the same degree, the event remains at the node where it appears. When cluster-heads are present, upon detection of an event, the event follows Algorithm 1:

Algorithm 1 Event forwarding in Heterogeneous Network

```
Require: event $e$ at node $v_i$
1: while node $v_i$ is not a cluster-head do
2:     if there is a neighbor $v_j$ with degree($v_j$) > degree($v_i$) then
3:         forward $e$ to the neighbor with the highest degree;
           break ties uniformly at random among candidates
4:     else
5:         forward $e$ to a random neighbor
6:     end if
7: end while
```

In order to find the event, a query is issued through a predefined *sink* node s. The query follows a *simple random walk* where the next node is chosen uniformly from the neighbors, until it finds the event e.

In the scenario described above there will be two costs: i) the cost of moving the event to a cluster-head, $\mathcal{C}_{\text{event}}$ and the cost of the query (i.e random walk) to find the event, $\mathcal{C}_{\text{query}}$. The total cost is the sum of both: $\mathcal{C}_{\text{total}} = \mathcal{C}_{\text{event}} + \mathcal{C}_{\text{query}}$.

Denoting L as the number of low-degree nodes and H as the number of high-degree nodes, how does $\mathcal{C}_{\text{total}}$ vary with the ratio $\frac{H}{H+L}$? Is there a value of $\frac{H}{H+L}$ that will reduce $\mathcal{C}_{\text{total}}$ significantly ?

III. Analytical Results

In this section we focus on line topologies and we are interested in analyzing the impact of cluster-heads on cost of event discover ($\mathcal{C}_{\text{total}}$). The main result is as follows:

Theorem 1: **Consider a line topology with $(n + 1)$ nodes, where cluster-heads have a degree $2k$ and are uniformly distributed. The first local minima for the maximum hitting time and for the query cost is obtained when the fraction of high-degree nodes is $\frac{4}{5k}$. And for this fraction a reduction in query cost of $\Theta(1 - \frac{1}{k^2})$ is obtained.**

As mentioned earlier, this result is obtained using bounds on the resistance of an electrical circuit related to the network graph. The remainder of the section is dedicated to the proof of this theorem, and Table I presents the notation used for the proof, which explained in detail in the next two subsections.

A. Background on Random Walks and Resistance Methods

For a graph $G(V, E)$ where $|V|$ is the number of nodes and $|E|$ is the number of edges, the following notation will be used. An element of V or E is represented by a lowercase, a subset or array is represented by a bold lowercase and the complement of a set or element will be denoted with an upper-bar, for example e represents an element, **e** an array (subset) and $\bar{e}$ and $\bar{\mathbf{e}}$ represent the complements of e and **e**, respectively.

$(n+1)$	number of nodes
k	cluster coverage
d	inter cluster-heads distance
α	overlaping of cluster-heads coverage (region 2)
$(s+1)$	number of clusters

TABLE I
Notation

The *hitting time* h_{uv} is the expected time taken by a simple random walk starting at u to reach v for the first time [27].

For a source $s \in V$ and the subset of cluster-head nodes $\mathbf{e} \subseteq V$, the average hitting time from s to **e** is:

$$h_{s\mathbf{e}} = \frac{\sum_{e \in \mathbf{e}} h_{se}}{|\mathbf{e}|} \tag{1}$$

The *commute time* C_{uv} is defined as the expected time taken by a random walk starting at u to reach v and come back to u. Note that by definition $C_{uv} = h_{uv} + h_{vu}$, but in general $h_{uv} \neq h_{vu}$. In a seminal work, Doyle and Snell [29] explored the connection between a random walk on a graph G and the resistance of an electrical network obtained from G by viewing each edge as a unit resistor. In [30], Chandra *et al.* extended this work and proved the following equality that relates the commute time C_{uv} and the effective resistance R_{uv} of the electrical network of G :

$$C_{uv} = 2mR_{uv} \tag{2}$$

Where m is the number of edges in the graph. Notice that in case of symmetry[1] h_{uv} = h_{vu}, which implies that the commute time is two times the hitting time. We will use this property in the analysis of the hitting time for line topologies.

B. Parameters of Line Topology

We consider the undirected graph $\mathcal{L}(n, k, d) = G(V, E)$ with the following parameters. The set of nodes is $V = v_0, v_1, \ldots v_n$, since the index goes from 0 to n, the number of nodes is $|V| = n + 1$, we will also denote n as the number of edges when no cluster-head has been added (when nodes communicate only with their immediate right and left neighbors). In addition to the initial n edges, each cluster-head has edges to all nodes which are less than or equal to k nodes away from it on the line. d is the inter cluster-head distance, hence, a node v_i is a cluster-head iff $i \mod d = 0$. The set of edges E for $\mathcal{L}(n, k, d)$ is defined as follows:

$$E = \{v_i v_j \mid (i \mod d = 0 \text{ and } |i - j| \leq k) \text{ or } |i - j| = 1\}$$

We will consider the case where $n \gg k > 1$. Since we are interested in the limit $n \to \infty$ we will consider only the cases

[1]The graph G' where we name u as v and vice versa is isomorphic to G

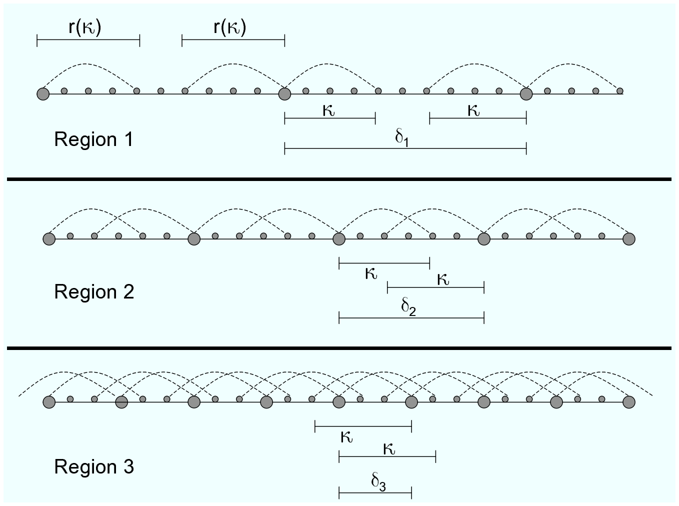

Fig. 1. Examples of line topologies. The big circles denote cluster-heads and the dashed lines their coverage k, all nodes within k hops from the cluster-head are its neighbors.

where $n = sd$, where s is an integer. Then, the total number of clusters is given by $(s+1)$.

For a given n and k we are interested in studying the impact of d on $\mathcal{C}_{\text{total}}$ via the resistance, and the analysis is divided in different regions according to the inter cluster-heads distance d:

$$\text{regions} = \begin{cases} \text{region } 1, & 2k \leq d \leq n/2 \\ \text{region } 2, & k < d < 2k \\ \text{region } 3, & 1 \leq d \leq k \end{cases} \tag{3}$$

Figure 1 presents examples of line topologies for the 3 different regions. The big circles denote cluster-heads and the dashed lines their coverage k. Notice that k is constant for all three topologies.

Later in this section we will show that $\mathcal{C}_{\text{total}}$ depends mainly on $\mathcal{C}_{\text{query}}$, specially for $d < 2(k+1)$ where $\mathcal{C}_{\text{event}}$ will be shown to be less than 1. For this reason, the resistance analysis will be focused on $\mathcal{C}_{\text{query}}$.

For the different regions we will first obtain expressions for the number of edges m and the effective resistance R between the nodes at the extremes of the line. Then, in subsections III-C and III-D we use these expressions, together with symmetry, to obtain the maximum hitting time and average query cost, $\mathcal{C}_{\text{query}}$.

1) Analysis of region 1*:* Recalling that the number of nodes in $\mathcal{L}$ is $(n+1)$, and that when no clusters are present the initial number of edges is n. Besides n, each cluster-head contributes with $2(k-1)$ new edges. Then the total number of edges is given by:

$$\begin{aligned} m_1 &= n + 2(k-1)s \\ &= n + 2(k-1)n/d \\ &= \tfrac{n}{d}(d + 2(k-1)) \end{aligned} \tag{4}$$

The coverage on each side of a cluster-head can be represented by an effective resistance $r(k)$ (Figure 1). $r(k)$ can be derived based on k using techniques to reduce resistors in parallel and series:

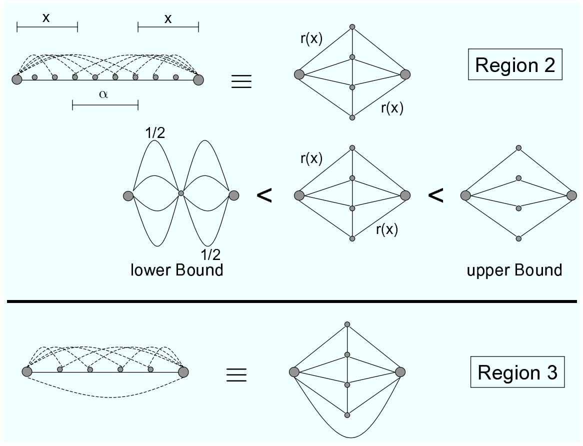

Fig. 2. Resistance calculation for regions 2 and 3.

$$\begin{aligned} k = 2 &\quad\Rightarrow\quad r(2) = 2/3 \\ k = 3 &\quad\Rightarrow\quad r(3) = 5/8 \\ k = 4 &\quad\Rightarrow\quad r(4) = 13/21 \end{aligned}$$

It can be derived that the numerator and denominator follows the fibonacci series $fib(.)$, hence:

$$r(k) = \frac{fib(2k-1)}{fib(2k)} \tag{5}$$

Since every cluster-head covers its right and left k-neighbors and the extreme vertices cover only one side, the effective resistance R_1 between the extremes of $\mathcal{L}$ is given by:

$$\begin{aligned} R_1 &= (2r(k) + d - 2k)s \\ &= (2r(k) + d - 2k)n/d \end{aligned} \tag{6}$$

Finally, using equation 2 and symmetry, the hitting time for nodes at the extreme of the line is given by:

$$h_1 = (\tfrac{n}{d})^2 \ (d + 2(k-1)) \ (d + 2(r(k) - k)) \tag{7}$$

2) Analysis of region 2*:* The number of edges is the same as equation 4 since every cluster-head that is added brings $2(k-1)$ new edges. However, the effective resistance between 2 cluster-heads is different. In this subsection we provide upper and lower bounds for this resistance.

Letting αk be the overlap between the coverage of two neighboring cluster-heads, where $0 < \alpha < 1$, then $d = k(2-\alpha)$. The resistance circuit is equivalent to the one shown in Figure 2, where $r(x) = r(k(1-\alpha))$ (the effective resistance for the non-overlapping part). Given that $r()$ converges to the inverse of the golden ratio, $1/2 < r(x) < 1$, further, considering Rayleigh's monotonicity law we can provide upper and lower bounds for the resistance as shown in Figure 2:

$$\begin{array}{lll} \frac{2}{\alpha k+2} & < r_2 & < \frac{2}{\alpha k+1} \\ \frac{2}{\alpha k+2} s & < R_2 & < \frac{2}{\alpha k+1} s \\ \frac{2}{\alpha k+2} \frac{n}{d} & < R_2 & < \frac{2}{\alpha k+1} \frac{n}{d} \end{array} \tag{8}$$

Finally, the hitting time is bounded by:

$$\begin{array}{ll} 2(\frac{n}{d})^2(\frac{4k-k\alpha-2}{\alpha k+2}) & < \\ & h_2 \\ & < 2(\frac{n}{d})^2(\frac{4k-k\alpha-2}{\alpha k+1}) \end{array} \tag{9}$$

	Maximum Hitting Time (h) [subsection III-C]	**Average Hitting Time ($\mathcal{C}_{\text{query}}$)** [subsection III-D]
region 1 (lower bound)	$(\frac{n}{k})^2(k-\frac{1}{2})$	$\frac{(2k-1)n(n+k)}{6k^2}$
region 2 (upper bound)	$2(\frac{4n}{5k})^2(\frac{13k-8}{3k+4})$	$\frac{4n(13k-8)(8n+5k)}{3(3k+4)(5k)^2}$
region 3	$6(\frac{n}{k})^2(\frac{k-1}{k+1})$	$\frac{(k-1)(2n+k)n}{(k+1)k^2}$

TABLE II
MINIMUM MAXIMUM AND MINIMUM AVERAGE HITTING TIMES PER REGION

3) Analysis of region 3*:* For this region, we will analyze only the point where $d = k$.

Contrary to the case of regions 1 and 2, neighboring cluster-heads have an edge connecting each other, hence each cluster-head brings $(2(k-1)-1)$ new edges. And the total number of edges is given by:

$$\begin{aligned} m_3 &= n + (2k-3)s \\ &= n + (2k-3)n/d \\ &= 3\tfrac{n}{k}(k-1) \end{aligned} \tag{10}$$

The resistance between two cluster-heads can be transformed to an equivalent circuit as shown in Figure 2, which leads to an equivalent resistance of $\frac{2}{k+1}$ between two neighboring cluster-heads. Hence, the effective resistance between the extremes of the line is given by:

$$\begin{aligned} R_3 &= \tfrac{2}{k+1}s \\ &= \tfrac{2}{k+1}\tfrac{n}{k} \end{aligned} \tag{11}$$

Finally, the hitting time for $d = k$ is given by:

$$h_3(d=k) \quad = 6(\tfrac{n}{k})^2\tfrac{k-1}{k+1} \tag{12}$$

Tight analytical results for the case $d < k$ are harder to obtain, but we can show the following lower bound[2]:

$$h_3 \geq \frac{2n^2}{k^2+3k} \tag{13}$$

C. Local Minimum for Maximum Hitting Time

In this section we analyze the hitting time between the sink (v_0) and the last cluster-head on the line (v_n). Table II shows the minimum value of h_1, h_2 and h_3. In region 3, we analyzed only one point ($d = k$) , hence, for region 3 the value in Table II is the one obtained in equation 12. For region 1, the minimum value is obtained for $d = 2k$, we further assume a lower bound on h_1 by setting $r(k) = 0.5$.

In the case of region 2, the hitting time depends on the overlapping α. Even though αk should take only integer values, we assume α to be real in order to differentiate h_2 (equation 9) and obtain the value of α that minimizes h_2. The values of α for the upper and lower bounds are given by (α_U and α_L):

[2]The proof of this result is not shown due to lack of space. However, it does not affect in anyway the result in Theorem 1. It is presented only on the interest of completeness

$$\alpha_U = \frac{12nk-7n+4k-16k^2}{2(2n-4k+1)k} - \frac{\sqrt{80n^2k^2-88n^2k+96nk^2-128nk^3+17n^2+48nk-16n}}{2(2n-4k+1)k} \tag{14}$$

$$\alpha_L = \frac{-8k^2+6nk-4n-2\sqrt{-8k^3n+5n^2k^2-4n^2k+8nk}}{2k(n-2k)} \tag{15}$$

For large n and k, $\alpha_U = \alpha_L = 0.7639$, hence $\alpha_{opt} \approx \frac{3}{4}$, Table II shows the upper bound of h_2 for α_{opt}.

From Table II we observe that $h_1 = \Theta(n^2/k)$, and h_2 and h_3 are $\Theta(n^2/k^2)$, hence for large k the minimum is either on h_2 or h_3. Since n and k are given, we can compare directly upper bound of h_2 and h_3 given in Table II, which leads to:

$$\lim_{k\to\infty} \frac{h_3}{h_2} = 1.0817 \tag{16}$$

Hence, the first local minima is in region 2 and the inter cluster-head distance that attains this minimum is $d = \frac{5}{4}k$, which corresponds to a ratio Ψ between high-degree nodes and the total number of nodes:

$$\Psi = (\frac{4n}{5k})(\frac{1}{n+1}) \approx \frac{4}{5k} \tag{17}$$

It is important to notice that the global minima might be in region 3, however, the reduction obtained by moving from the first local minima to the global minima is not significant, as it will be shown numerically in the next section. Furthermore, as presented in equation 13 the cost in region 3 is the same order as in region 2.

D. Local Minimum for Expected Hitting Time

For regions 1, 2 and region 3 (when $d = k$), cluster-head i is visited only after cluster-head $i-1$ has been visited. This property allows us to discard all nodes beyond a given cluster-head i when we are interested in obtaining the hitting time to i. Hence, for cluster-head i we can consider a new line topology, which is a shorter version of the original one, where the sink is at one extreme and cluster-head i is at the end of the new line. This behavior allows us to use directly the expressions derived in the previous subsections with the only change to be made in the initial number of edges n.

Recalling that the distance set between the sink and cluster-heads is given by $\{0, d, 2d, 3d...(s-1)d, sd\}$. $\mathcal{C}_{\text{query}}$ is the average hitting time ($E[h_i]$) to all clusters and for region 1 is given by:

$$\begin{aligned} E[h_1] &= \frac{\sum_{i=0}^{s} h_1(n=id)}{s+1} \\ &= \frac{\sum_{i=1}^{s} (\frac{n}{d})^2 \ (d+2(k-1)) \ (d+2(r(k)-k))}{s+1} \\ &= \tfrac{(d+2(k-1)) \ (2r(k)+d-2k)}{s+1} \sum_{i=1}^{s} i^2 \\ &= (d+2(k-1)) \ (2r(k)+d-2k) \ \tfrac{s(2s+1)}{6} \end{aligned} \tag{18}$$

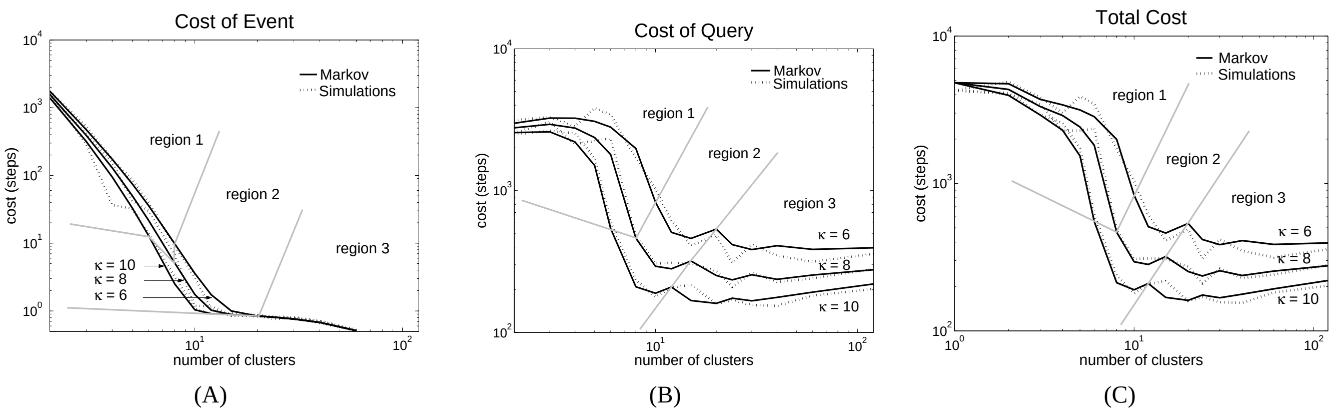

Fig. 3. $\mathcal{C}_{\text{query}}$, $\mathcal{C}_{\text{event}}$ and $\mathcal{C}_{\text{total}}$ vs the number of clusters for a line topology with 121 nodes and different values of k (6, 8 and 10). The solid lines represent the cost obtained using Markov numerical analysis and the dotted lines represent simulation results.

For Region 2, using the upper bound of equation 9 and recalling that $d = k(2-\alpha)$, the expected hitting time is given by:

$$\begin{aligned} E[h_2] &= \frac{\sum_{i=0}^{s} h_2(n = id)}{s+1} \\ &= \frac{\sum_{i=1}^{s} 2(\frac{n}{d})^2(\frac{4k-k\alpha-2}{\alpha k+1})}{s+1} \\ &= \frac{2(4k-k\alpha-2)}{(\alpha k+1)(s+1)} \sum_{i=1}^{s} i^2 \\ &= \frac{n(2n+1)(4k-k\alpha-2)}{3(\alpha k+1)(k(2-\alpha))^2} \end{aligned} \tag{19}$$

The derivative of this equation with respect to α leads to an expression that is considerably more complicated than equation 14 and it is not presented. However, the results are the same as the obtained for the maximum hitting time, i.e. $\alpha_U = \alpha_L = 0.7639$ and $\alpha_{opt} \approx \frac{3}{4}$. The closed-form expressions and values can be easily obtained by using mathematical software such as Matlab or Mathematica.

For Region 3, when $d = k$, the expected hitting time is given by:

$$\begin{aligned} E[h_3(d=k)] &= \frac{\sum_{i=0}^{s} 6(\frac{n}{k})^2 \frac{k-1}{k+1}}{s+1} \\ &= \frac{6(k-1)}{(s+1)(k+1)} \sum_{i=0}^{s} i^2 \\ &= \frac{k-1}{k+1} s(2s+1) \end{aligned} \tag{20}$$

Table II also presents the minimum value of $\mathcal{C}_{\text{query}}$ for the 3 regions: $d = 2k$, $r(k) = 0.5$ for region 1 (lower bound), $d = \frac{5}{4}k$ for region 2 (upper bound) and $d = k$ for region 3 (only point analyzed). The comparisons lead to the same result as the ones obtained for the maximum hitting time: the order of h_1 is greater than the order of h_2 and h_3, and as n and k goes to infinity the ratio of h_3 over h_2 goes to 1.0817.

Now we have all the elements to prove Theorem 1

Proof of Theorem 1:

The optimization of the maximum and average hitting times (equations 9 and 19) leads to $d = \frac{5}{4}k$, using Table II we proved that the first local minima occurs in region 2 for both the maximum and average hitting times, and that their cost are $\Theta(n^2/k^2)$. It is known that random walks on regular line topologies have a cost of $\Theta(n^2)$ [9], hence, a line topology $\mathcal{L}(n,k,d)$ with $d = \frac{5}{4}k$ leads to a cost reduction of $\Theta(1 - \frac{1}{k^2})$ for both the maximum and average hitting times. □

IV. Numerical and Experimental Results

In this section we use Markov numerical methods and simulations to study the impact of heterogeneity on the performance of random walk-based queries. First, we briefly introduce the method of using absorption states to obtain hitting times. Then, we present numerical results on a line topology that validates the analytical contributions of Section III. Finally, we present numerical results on regular grids and random geometric graphs, and simulation results on realistic graphs for wireless sensor networks.

A. Background on Hitting Times in Markov Chains

Given a graph $G(V,E)$, a random walk on the graph can be defined by a Markov chain $\mathcal{M}$, where $\mathcal{M}$ represents the transition probability matrix. The notation used for elements and subsets of V and E is the same as the one presented in subsection III-B.

Let us consider that the event is in node $e \in V$ and the sink is in node $s \in V$. As presented in [28], absorption states can be used to obtain hitting times. Let $\mathcal{M}_e$ be the matrix resulting from deleting the row and column corresponding to e in $\mathcal{M}$, and let Q_e be:

$$Q_e = (I - \mathcal{M}_e)^{-1}\mathbf{1} \tag{21}$$

Where I is the identity matrix and $\mathbf{1}$ is a column vector of ones. Q_e is an array representing the expected hitting time from each node to e (except e). Hence, in our case $h_{se} = Q_e(s)$, where $Q_e(s)$ represents the s'th element of the array.

B. Line Topologies

For a given line topology $\mathcal{L}(n,k,d)$ with transition probability matrix $\mathcal{M}$, where the sink is the extreme left vertices

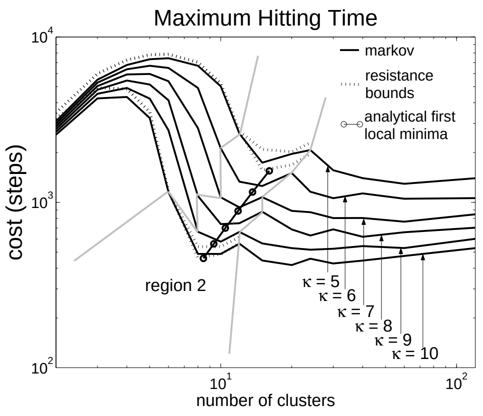

Fig. 4. Maximum hitting time for a line topology with 121 nodes and values k ranging from 5 to 10. The full lines represent Markov numerical values and the dotted lines the analytical results through resistance methods. The line with the circle markers represent the analytical values of the number of cluster heads for the first local minima.

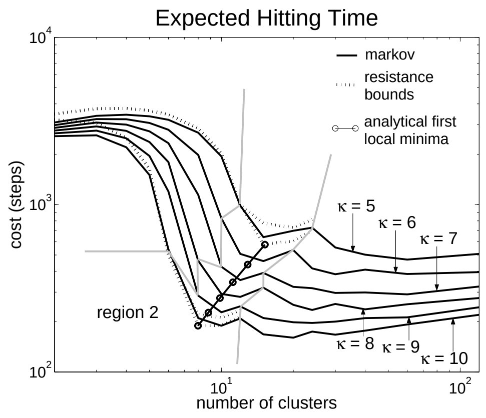

Fig. 5. $\mathcal{C}_{\text{query}}$ (expected hitting time) for a line topology with 121 nodes and values k ranging from 5 to 10. The full lines represent Markov numerical values and the dotted lines the analytical results through resistance methods. The line with the circle markers represent the analytical values of the number of cluster heads for the first local minima.

v_0. Clusters-heads are positioned at v_{id} where i=0,1, ..., s. The hitting time from v_0 to a specific cluster-head is $Q_{id}(v_0)$, where Q_{id} is given by:

$$Q_{id} = (I - \mathcal{M}_{id})^{-1}\mathbf{1} \tag{22}$$

Based on equation 1, $\mathcal{C}_{\text{query}}$ is the average hitting time over all cluster-heads and is given by:

$$\begin{aligned}\mathcal{C}_{\text{query}} &= \tfrac{d}{2n}(Q_0(v_0) + Q_{sd}(v_0)) + \tfrac{d}{n}\sum_{i=1}^{s-1} Q_{id}(v_0) \\ &= \tfrac{d}{2n}(Q_{sd}(v_0)) + \tfrac{d}{n}\sum_{i=1}^{s-1} Q_{id}(v_0)\end{aligned} \tag{23}$$

In the previous equation all cluster-heads have the same weight except for the extreme ones (v_0 and v_{sd}), this is due to the fact that at the extremes, the expected number of stored events is half of those stored at the intermediate cluster-heads. However, for large number of cluster-heads the weight can be considered similar and:

$$\mathcal{C}_{\text{query}} \approx \frac{\sum_{i=1}^{s} Q_{id}(v_0)}{s+1} \tag{24}$$

$\mathcal{C}_{\text{event}}$ will be derived according to the regions defined in 3 and algorithm 1. There are $s+1$ cluster-heads for which there is no need to move the event, hence the cost is zero.

When $d < 2(k+1)$, all nodes will be directly connected to a cluster-head and $\mathcal{C}_{\text{event}} = \frac{n-s}{n+1}$. However, when $d \geq 2(k+1)$ (most of region 1), there will be orphan nodes between any pair of consecutive cluster-heads. Due to symmetry, the cost will be the same for any subset of nodes between any two neighboring cluster-heads and for simplicity we consider cluster-heads v_0 and v_d. For these clusters, nodes between $(k+1)$ and $(d-k-1)$ are orphan. Let us define $a = (k+1)$, $b = (d-k-1)$ and $\mathcal{M}_{(\text{a:b})}$ as the $\mathcal{M}$'s sub-matrix which includes only the rows and columns between a and b. For $\mathcal{M}_{\text{a:b}}$, $Q_{a:b} = (I - \mathcal{M}_{a:b})^{-1}\mathbf{1}$ is the array containing the expected number of steps of each orphan node to reach the closest node directly connected to a cluster-head. Hence, the cost of moving orphan nodes between a and b to a cluster-head is given by:

$$\mathcal{C}_{\text{orphan}} = \sum_{i \in Q}(Q_{a:b}(i) + 1) \tag{25}$$

Where the constant 1 represents the cost of moving the event from a node connected to a cluster-head to the cluster-head.

Finally, recalling that $(n+1)$ is the total number of nodes in $\mathcal{L}$, $\mathcal{C}_{\text{event}}$ is given by:

$$\mathcal{C}_{\text{event}} = \begin{cases} \frac{2k(s+1)+s\mathcal{C}_{\text{orphan}}}{n+1}, & 2(k+1) \leq d \leq n/2 \\ \frac{n-s}{n+1}, & 2k \leq d < 2(k+1) \\ \frac{n-s}{n+1}, & \text{region 2} \\ \frac{n-s}{n+1}, & \text{region 3} \end{cases} \tag{26}$$

Figure 3 shows $\mathcal{C}_{\text{query}}$, $\mathcal{C}_{\text{event}}$ and $\mathcal{C}_{\text{total}}$ vs the number of clusters for a line topology with 121 nodes and different values of k. The solid lines represent the cost obtained using Markov numerical analysis (equations 24 and 26), and the dotted lines represent simulation results, it can be observed that the Markov method provides an accurate representation of the cost. Also it must be noted that the query cost accounts for most of the total cost, which validates the focus of the analytical section on $\mathcal{C}_{\text{query}}$.

Figures 4 and 5 compare the maximum and expected hitting time between the Markov analysis (full lines) and the expressions obtained through the resistance method (dotted lines) for a line topology with 121 nodes and values of k ranging from 5 to 10. The dotted lines show that the bounds get tighter for higher values of k in both figures. The figures also shows a line with circle markers depicting the number of clusters required to reach the first local minima according to our analysis (equation 14), which supports the results presented in Theorem 1. It is important to notice that the analytical values for the number of clusters are not necessarily integers and hence they may not match exactly the numerical ones,

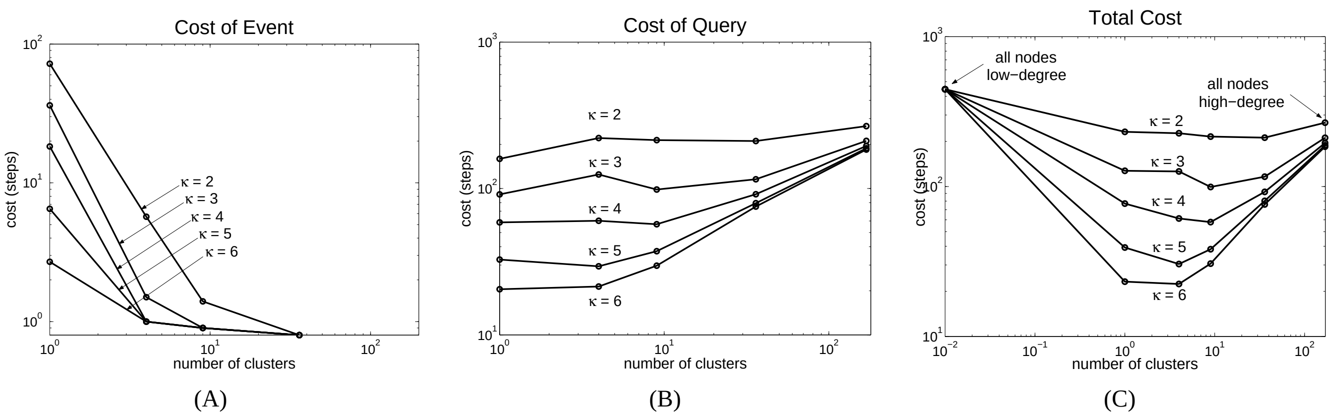

Fig. 6. (A) $\mathcal{C}_{\text{event}}$, (B) $\mathcal{C}_{\text{query}}$ and (C) $\mathcal{C}_{\text{total}}$ for a grid topology with 169 nodes and values of k ranging from 2 to 6.

specially for low values of k. However, for large values of k and n the floor or ceiling of the analytical value will not incur in significant differences. It is also important to notice that the first cluster-heads added leads to a significant cost reduction (region 1 and part of region 2), adding even more high-degree nodes beyond this point provides diminishing returns or in some cases even degrades the performance.

C. *Regular Grids and Random Geometric Graphs*

Grids and Random Geometric Graphs (RGG) are common models to study various properties and protocols for wireless systems. In the previous section, we showed that in line topologies the addition of cluster-heads can greatly reduce the cost of random-walk-based queries, do cluster-heads have the same significant effect on 2-dimensional topologies?

1) Grids: We assume that the number of cluster-heads is a perfect square and they are uniformly distributed on the grid, i.e. the grid is divided in the same number of cells as cluster-heads and each cluster-head is positioned in the node at the center of each cell.

According to the algorithm presented in 1, events appearing in a cluster-head has a cost of 0, events appearing in nodes directly connected to a cluster-head have a cost of 1 and events appearing in orphan nodes perform a simple random walk until it hits a node that is directly connected to a cluster-head. Denoting $\mathcal{M}$ as the transitional probability matrix, $w \subseteq V$ as the subset of vertices containing all cluster-heads and all the nodes directly connected to them and $\bar{w} \subseteq V$ as its complement; we define $\mathcal{M}_w$ as the sub-matrix where all the rows and columns of the vertices in w have been removed. Hence, the hitting time for orphan nodes is given by:

$$h_{\bar{w}w} = \sum_{i=0}^{|\bar{w}|} (Q_w(i) + 1) \tag{27}$$

And $\mathcal{C}_{\text{event}}$ cost for the grid topology is given by:

$$\mathcal{C}_{\text{event}} = \frac{(|w| - (s+1)) + h_{sw}}{n} \tag{28}$$

In our analysis the sink is located at the bottom-left corner of the grid (v_0). The query cost $\mathcal{C}_{\text{query}}$ is the average hitting time from the sink to the set of cluster-heads and it is given by combining equations 1 and 21.

Figure 6 presents results for $\mathcal{C}_{\text{event}}$, $\mathcal{C}_{\text{query}}$ and $\mathcal{C}_{\text{total}}$ ((A), (B) and (C) respectively) for 169 nodes deployed on a 2D grid. The y axis represent the cost and the x axis the number of clusters. There are two important observations: i) contrary to the line topology, the case when all nodes are high-degree perform significantly worse, ii) similar to the line topology, the first cluster-heads account for most of the savings (greater than 30%) and the higher k the higher the savings. Also note that, as k increases, the case where all nodes are high-degree approaches a complete graph.

Another important difference with respect to the line topology is that the event cost plays a significant role in the total cost, while the query cost does not a have the same significant impact. This may be due to several reasons one of them is that for the same number of nodes the diameter of a line ($\Theta(n)$) topology is significantly larger than a grid ($\Theta(\sqrt{n})$) topology. From the resistance method perspective, in grids we can not eliminate the edges beyond a given cluster, all edges should be considered and hence according to equation 2 this would make the query cost for different clusters more similar.

2) Random Geometric Graphs: The procedure for getting event and query costs in random geometric graphs is the same as for grids (equation 28, and a combination of equations 1 and 21). However, the interesting case in random geometric graphs is that even when only low-degree nodes are deployed there are some inherent cluster-heads due to some favorable geographical position. We further enhance the inherent cluster-heads formed by increasing their transmission range. According to the algorithm presented in 1, in these scenarios the event moves in a greedy way towards the local cluster-head.

Table III presents results for 169 nodes deployed randomly on a 1x1 square area. The results are the average over 50 runs. The initial radius is 0.12 which for this density gives a connectivity probability of $\approx$ 0.5. The table has two columns named "clustering" and "no clustering". Due to the random deployment some nodes will end up being local-clusters (their degree is higher or equal than their neighbors). For the "clustering" column, we enhance these local clusters by increasing their transmission range to the value given in the "transmission range" column, while the nodes that are not local cluster-

transmission range	clustering	no clustering	savings (%)
0.12	679.7	833.0	18.4
0.18	414.2	296.2	-39.8
0.24	171.4	225.0	23.8
0.30	88.0	202.7	56.6
0.36	52.9	193.5	72.7

TABLE III

RANDOM GEOMETRIC GRAPHS

output power (dBm)	clustering	no clustering	savings (%)
-14	263.1	428.7	38.6
-13	211.7	370.4	42.8
-12	174.6	333.4	47.6
-11	152.6	311.4	51.0
-10	132.5	278.1	52.3

TABLE IV

GRID DEPLOYMENTS IN REALISTIC ENVIRONMENTS

output power (dBm)	clustering	no clustering	savings (%)
-14	367.2	557.7	34.2
-13	250.4	432.9	42.2
-12	243.8	380.8	36.0
-11	207.9	332.9	37.6
-10	169.9	294.4	42.3

TABLE V

RANDOM DEPLOYMENTS IN REALISTIC ENVIRONMENT

heads have a range of 0.12. For the "no clustering" column all nodes have the transmission range given the "transmission range" column, but events stay in the nodes where they appear. We observe that in random geometric graphs clustering also have a significant impact on the performance of random-walk-based queries (except for r=0.18 where "no clustering" is better) . It is important to mention that for the initial r=0.12 approximately ∼11% of the nodes end up being local clusters.

D. Low-Power Wireless Graphs

Using the link layer model proposed in [31], we evaluate through simulations the effectiveness of cluster-heads in realistic graphs, which are characterized by the presence of unreliable and asymmetric links. In order to guarantee the survival of the random walk we implemented a 3-way handshake protocol. A node with the random walk issues a request to the next neighbor to receive the random walk, upon reception of the packet the neighbor acknowledges the reception of the random walk, finally upon reception of the acknowledgment, the original node sends a release packet which ends the transfer of the random walk.

Tables IV and V present the results for grid and random deployments. The presentation is similar to Table III, the "clustering" column represents networks were only the inherent local cluster-heads are enhanced by increasing their output power, while in the "no clustering" column all nodes increase their output power but events remain in the nodes where they appear. We can observe that clustering plays a significant role in reducing the cost of random walk-based queries (between 30% and 50%). On grid deployments approximately 12% of the nodes are inherent cluster-heads, while on random deployments about 8% of nodes are cluster-heads.

V. RELATED WORK

The simplest implementation of a query dissemination protocol for a sensor network is the basic flooding mechanism where a query message is forwarded by all nodes in the network. Flood-based queries (used, for instance, in Directed Diffusion [4] to set up routes from the sources to the querying node) have the advantage of simplicity, and, when used in the context of continuous data stream responses, can be justified because their costs can be amortized over the period of the response. However, for one-shot queries, other techniques are desired. Several researchers have studied the use of sequential TTL-based controlled floods (expanding rings) as unstructured query mechanisms [1], [5], [6], [7], [8]. An important approach for pull-based one-shot queries in unstructured systems is the use of random walks.

Random walks on graphs have been studied mathematically, and there is a substantial-yet-growing body of theoretical literature on the subject [9], [10], [11]. They are also finding increasing use in a wide range of protocols in the context of several networked distributed systems. For instance, they have been used in Grid-aware operating systems [12], in unstructured P2P Networks [13], [14], [15], for hybrid application overlays [16], for group membership services in mobile ad hoc networks [17], [18], for distributed model checking [19], and for index quality determination for the world-wide web [20].

Specifically in the context of unstructured wireless sensor networks, different variants of random-walk-based protocols have been proposed and analyzed by several research groups. Servetto and Barrenechea [21] proposed and analyzed the use of constrained random walks on a grid for performing load-balanced routing between two known nodes. Avin and Brito [22] have argued that even simple random walks can be used for efficient and robust querying because they are inherently load-balanced and their partial cover times show good scaling behavior. The ACQUIRE protocol [23] provides a tunable look-ahead parameter to combine random walks with controlled floods and show that such random-walk-based hybrids can outperform flooding and even expanding-ring-based approaches in the presence of replicated data. The rumor routing algorithm [24] is a hybrid push-pull mechanism that advocates the use of multiple random walks from the events as well as the sinks, so that their intersection points can be used to provide a rendezvous point. Shakkottai [25] has analyzed different variants of random-walk-based query mechanisms and concludes that source and sink-driven sticky-searches (similar to rumor routing) provide a rapid increase of query success probability with the number of steps. Most recently, Alanyali *et al.* [26] have proposed the use of random walks in energy-constrained networks to perform efficient distributed computation of a class of decomposable functions (useful in computing certain kinds of aggregates). To our knowledge, these prior studies have not investigated the impact of heterogeneous deployments on the performance of random-walk-based querying protocols.

VI. Conclusions

Our work presented an study of the impact of heterogeneous node connectivity on random walk-based queries. The main contribution of the work is showing that with a small percentage of high-degree nodes in the network ($< 10\%$) and using a simple distributed push-pull mechanism, significant cost savings can be obtained — between 30% and 70% depending on the coverage of the high-degree nodes. Our work provides interesting theoretical results for line topologies showing that when cluster-heads have a coverage k (cover k nodes to the right and left) and are uniformly distributed, a fraction of $\frac{4}{5k}$ nodes being cluster-heads can offer a reduction in query cost of $O(1 - \frac{1}{k^2})$ by using a simple distributed algorithm.

While designing a system it is important to discuss the extra cost of high-degree nodes. In wireless systems, high-degree nodes can be special nodes or nodes whose output power is increased, in both cases a high-degree node incurs a higher cost. Our work provided the impact of clustering without considering costs, however, costs can be easily inserted in our analysis by multiplying the cost for each value of delta by the extra cost incurred by the fraction of high-degree nodes utilized.

Another important issue is the one of delay. One of the drawbacks of random walks is the significant delay that they encounter. In our work, by minimizing the required number of steps on the random walk we are not only reducing the cost but also the delay. Hence, heterogeneous networks also provide an extra advantage in terms of delay. However, an accurate quantification of the savings should include specific characteristics of the protocol used at the MAC layer.

Given the performance improvement that heterogeneous networks have on random walk-based queries. It is important to highlight some other effects that may strength heterogeneity in real networks, for instance remaining battery power, antenna orientation and geographical position above ground may lead to larger cluster-heads. Also, given the distributed nature of random walks and the push-pull algorithm proposed, cluster-heads can rotate in order to avoid energy depletion, and the only nodes that need to be informed are the neighbors.

References

[1] B. Krishnamachari and J. Ahn, “Optimizing data replication for expanding ring-based queries in wireless sensor networks”, WiOpt 2006.

[2] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin and S. Wicker, “Complex Behavior at Scale: An Experimental Study of Low-Power Wireless Sensor Networks”, UCLA CS Technical Report UCLA/CSD-TR 02-0013, 2002.

[3] R. Govindan, E. Kohler, D. Estrin, F. Bian, K. Chintalapudi, O. Gnawali, S. Rangwala, R. Gummadi, T. Stathopoulos, “Tenet: An Architecture for Tiered Embedded Networks”, CENS Technical Report 56, November 10 2005.

[4] C. Intanagonwiwat, R. Govindan, and D. Estrin, “Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks”, ACM Mobicom 2000.

[5] N. B. Chang and M. Liu, “Revisiting the ttl-based controlled flooding search: Optimality and randomization”, ACM MobiCom 2004.

[6] N. B. Chang and M. Liu, “Controlled flooding search in a large network”, accepted for publication, IEEE/ACM Trans. on Networking, 2006.

[7] Z. Cheng and W. Heinzelman, “Flooding Strategy for Target Discovery in Wireless Networks”, ACM/Baltzer Wireless Networks, 2005.

[8] Z. Cheng and W. Heinzelman, “Searching Strategy for Multi-Target Discovery in Wireless Networks”, ASWN 2004.

[9] Aldous and Fill, “Reversible Markov Chains and Random Walks on Graphs”, Draft Monograph, online at http://www.stat.berkeley.edu/ $\sim$ aldous/RWG/book.html

[10] L. Lovasz, “Random Walks on Graphs: A Survey, Combinatorics, Paul Erdos is Eighty”, Vol. 2, Janos Bolyai Mathematical Society, Budapest, 1996, 353398.

[11] R. Burioni and D. Cassi, “Random walks on graphs: ideas, techniques and results”, Journal of Physics A: Mathematical and General Vol. 38 Issue 8 Article R01, March 2005.

[12] M. Fertr, E. Jeanvoine, C. Morin and L. Rilling, “Grid-aware Operating System”, Programming Parallel and Distributed Systems for Large Scale Numerical Simulation Applications 2005.

[13] E. Cohen and S. Shenker, “Replication strategies in unstructured peer-to-peer networks”, ACM SIGCOMM 2002.

[14] L.A. Adamic, R. Lukose, A. Puniyani, and B. Huberman. “Search in power-law networks”, Phys. Rev., E(64), 2001.

[15] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham,and S. Shenker, “Gia: Making gnutella like p2p systems scalable”, ACM SIGCOMM 2003.

[16] R. Tian, Y. Xiong, Q. Zhang, B. Li, B. Y. Zhao, X. Li, “Hybrid Overlay Structure Based on Random Walks”, IPTPS 2005.

[17] Dolev, S., Schiller, E., and Welch, J., “Random walk for self-stabilizing group communication in ad-hoc networks”, SRDS 2002.

[18] Z. Bar-Yossef, R. Friedman, G. Kliot, “RaWMS - Random Walk based Lightweight Membership Service for Wireless Ad Hoc Networks”, To appear in MobiHoc 2006.

[19] H. Sivaraj, and G. Gopalakrishnan, “Random Walk Based Heuristic Algorithms for Distributed Memory Model Checking”, PDMC 2003.

[20] M. R. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork, “Measuring Index Quality Using Random Walks on the Web”, 8th International World Wide Web Conference, May, 1999.

[21] S. D. Servetto and G. Barrenechea, “Constrained Random Walks on Random Graphs: Routing Algorithms for Large Scale Wireless Sensor Networks”, WSNA 2002.

[22] C. Avin and C. Brito, “Efficient and Robust Query Processing in Dynamic Environments Using Random Walk Techniques”, IPSN 2004.

[23] N. Sadagopan, B. Krishnamachari, and A. Helmy, “Active Query Forwarding in Sensor Networks (ACQUIRE)”, Journal of Ad Hoc Networks, Elsevier, Vol 3, Issue 1, January 2005.

[24] D. Braginsky and D. Estrin, “Rumor Routing Algorithm For Sensor Networks”, WSNA 2002.

[25] S. Shakkottai, “Asymptotics of query strategies over a sensor network”, IEEE INFOCOM 2004.

[26] M. Alanyali, V. Saligrama, and O. Savas, “A random-walk model for distributed computation in energy-limited networks”, 1st Workshop on Information Theory and its Applications, 2006.

[27] L. Lovasz, “Random walks on graphs: A survey”, in Combinatorics, Paul Erdos is Eighty, Vol. 2, Janos Bolyai Mathematical Society, Budapest, 1996, pp. 353398.

[28] S. Resnick, “Adventures in Stochastic Processes”, Birkhuser, Boston, 1992, pp. 102-110.

[29] P. Doyle and J. Snell, “Random walks and Electric Networks”, MAA, 1984.

[30] A. Chandra , P. Raghavan , W. Ruzzo , R. Smolensky, “The electrical resistance of a graph captures its commute and cover times”, ACM symposium on Theory of computing, 1989.

[31] M, Zuniga, B. Krishnamachari, “Analyzing the Transitional Region in Low Power Wireless Links”, under submission TOSN.