Naps: Scalable, Robust Topology Management in Wireless Ad Hoc Networks

P. Brighten Godfrey
pbg@cs.berkeley.edu

David Ratajczak
dratajcz@cs.berkeley.edu

Department of Electrical Engineering and Computer Science
University of California, Berkeley
Berkeley, CA 94720-1776

ABSTRACT

Topology management schemes conserve energy in wireless ad hoc networks by identifying redundant nodes that may turn off their radios or other components while maintaining connectivity. We present *Naps*, a randomized topology management scheme that does not rely on geographic location information, provides flexibility in the target density of waking nodes, and sends only a periodic heartbeat message between waking neighbors; thus it is implementable even on modest hardware. We formally analyze the connectivity of the waking graphs produced by *Naps*, showing that these graphs have nearly complete connectivity even at relatively low densities. We examine simulation results for a wide range of initial deployment densities and for heterogeneous and mobile deployments.

Categories and Subject Descriptors

C.2.1 [**Computer-communication Networks**]: Network Architecture and Design—*distributed networks, network topology, wireless communication*

General Terms

Algorithms, Performance, Theory

Keywords

Wireless ad hoc networks, sensor networks, topology management, percolation theory, simulation

1. INTRODUCTION

In wireless ad hoc networks, nodes are distributed in a geographic region and are able to communicate only with other nodes within a limited radius; long-range communication is made possible by forwarding messages in a multi-hop fashion toward their targets. Nodes are typically battery-powered and hence energy is a scarce resource. Furthermore, on most hardware, a substantial portion of energy is consumed by the radio, irrespective of whether it is sending, receiving, or merely listening for packets [1, 2, 3]. In high density deployments, however, not all nodes may be required to forward traffic in order to maintain connectivity in the network. In such cases, we may wish to select a fraction of nodes to turn their radios or other components off ("nap") for a calculated interval to reduce their energy consumption, thus prolonging the useful lifetime of the entire system. Such a technique can also increase the capacity of the network [4] by producing a "waking" connected backbone of lower density that assumes all forwarding responsibilities, and to which all "napping" nodes need communicate only infrequently or with reduced transmission power.

In either case, we wish to select a *nearly maximal* set of napping nodes, under the constraint that *nearly all* waking nodes are in a single connected component, and *nearly every* napping node can transmit to a waking node. This is a significantly weakened form of the NP-complete *connected dominating set* problem [5], which requires selecting the smallest connected backbone such that all nodes are either in or adjacent to the backbone. We feel that these relaxed requirements are sufficient since connectivity may be compromised anyway due to node and network unreliability or an initially disconnected placement of nodes.

Our proposed algorithm, *Naps*, decides when nodes should nap and wake up to achieve a target density of waking nodes in expectation, *without requiring that nodes know the initial density*. Intuitively, the fraction of time that a node should be awake is inversely proportional to its degree and proportional to the target density. The algorithm continually changes the subset of waking nodes to allow all nodes to periodically nap. *Naps* is extremely simple, does not require location information, does not require clock synchronization, and only requires a node to wake and broadcast a heartbeat to its waking neighbors every time period T (a tunable parameter of the algorithm).

We prove that the waking subgraphs produced by *Naps* exhibit a phase transition: if the initial and target densities are above a *critical threshold*, then after running *Naps* on an infinite graph with the initial density, any particular node is in an infinite connected component of waking nodes with non-zero probability. This general phenomenon is called *percolation* and has been widely studied [6]. In the finite setting, the expected fraction of waking nodes in the largest connected component approaches the percolation

IPSN'04, April 26–27, 2004, Berkeley, California, USA.

probability as the area is increased [7]. This is in contrast to the logarithmic density required to ensure complete connectivity [8].

We show empirically that the waking subgraphs are almost completely connected even for relatively small values of the target density. For example, in a network of at least 500 nodes with an initial density of $12/\pi$ and a target density of $6/\pi$ (such that a node expects 6 waking and 12 total neighbors), we expect more than 98% of nodes to be either waking and in the largest connected component, or napping and adjacent to that component. This remains true even while nodes wake and nap.

Naps can be used for any application that requires selecting a set of nodes with a specified target density and which can adapt to dynamic changes in the topology. Note that such adaptive algorithms have been developed for routing [9, 10], and adaptivity is already a requirement for most practical applications due to unreliability and mobility. *Naps* does not require napping nodes to receive messages – allowing them to leave their radio in the sleep state – and the number of messages received is proportional to the target density as opposed to the initial density. This is a significant improvement over some previous algorithms, and is one of the primary reasons that *Naps* shows significant energy conservation even at arbitrarily high initial densities, especially when the energy consumption of a napping node is small.

Finally, while our analysis assumes that nodes are distributed according to an infinite Poisson point process, we show through simulation that the algorithm is applicable even when the network is finite, the underlying distribution is not uniform, and when nodes are mobile.

This paper makes the following contributions: it presents a simple algorithm for "thinning" an ad hoc network by selecting nodes to nap and wake at calculated intervals; it presents the first such algorithm that maintains good connectivity and energy conservation while scaling to very high node densities and without using geographic information; and it presents a formal and an empirical analysis of the algorithm demonstrating its scalability and robustness.

The rest of this paper is organized as follows: Section 2 discusses related work; Section 3 describes our system model and environmental assumptions; Section 4 describes the *Naps* algorithm itself; Section 5 contains a theoretical analysis of the asymptotic properties of the waking subgraphs produced by *Naps*; Section 6 examines the empirical behavior of the algorithm under a wide range of test conditions; and Section 7 concludes the paper.

2. RELATED WORK

Algorithms that turn off radios by exploiting redundancy have been dubbed *topology management schemes* [1, 2, 3, 11], and several solutions have been proposed placing different assumptions on hardware and applications [1, 12, 11].

AFECA [12] is similar in spirit to *Naps* in that nodes are given a sleep interval that is related to the number of neighbors they have. However, each AFECA node is awake for a fraction of time which is roughly $2/(2+N)$ (assuming each node has an accurate measurement of its neighborhood size N). By the results of Section 5, this yields a waking subgraph with many small components. Since AFECA assumes a store-and-forward routing mechanism, messages must wait for nodes to wake before making further progress. Hence AFECA trades energy conservation for significantly increased routing latency.

GAF [11] is similar to *Naps* in its goals, and provides the application with an effectively static topology. However it relies on geographic information, limiting its use to hardware and environments where such information can be reliably obtained. We demonstrate that *Naps* provides good performance even without such features.

Span [1] is designed to select a connected backbone for the purposes of energy conservation, and does not assume geographic location information. Span produces a (not necessarily minimal) *connected dominating set*. However Span is comparatively complex, and requires that every node periodically receive topology information from each of its neighbors in the original graph, so that the fraction of time that a node's radio is idle or receiving (as opposed to sleeping) will increase as the initial density of the network increases. This property limits Span to only a factor of 2 improvement in system lifetime even at high initial densities. Span has no flexibility in the density of waking nodes, limiting its use for applications other than connectivity. TMPO [13] shares many of the same traits as Span, though with greater emphasis on performance in a mobile setting.

STEM [3] turns off nodes even more aggressively than the previous schemes by exploiting the fact that many applications only generate traffic sporadically. Similar to AFECA, it trades energy conservation for increased routing latency. It further assumes a separate low-power radio operating at a lower duty cycle.

There are a number of papers that propose turning off radios at the MAC layer when the radio is not in use; see Zheng et al. [14] and the references therein. These and other node-level power saving techniques can complement *Naps* in order to further conserve energy.

Gupta and Kumar [4, 8] and others have studied the problem of varying the radius of transmission in amenable hardware while preserving connectivity. This technique may be useful in improving the capacity of the network by reducing radio interference, though it is unlikely to yield significant energy conservation since the extra energy cost of transmission is small compared to the ambient cost of an idle radio [3].

Turning off nodes has been considered with the objective of maintaining complete (or adequate) sensing coverage as opposed to mere connectivity. Both Booth et al. [15] and Tian and Georganas [16] consider various algorithms for this problem, all of which rely on geographic information.

Distributed algorithms for approximating connected dominating sets have been proposed that require a constant number of rounds of communication and yield logarithmic factor approximations [17]. However these are difficult to implement, typically requiring some form of synchronization or leader election. Moreover, they do not evenly distribute responsibility across all nodes over time.

The percolation result of Section 5 follows a fairly standard proof technique described by Grimmett [6], and is similar to a result by Dousse et al. [18] on the impact of interferences on connectivity of ad hoc networks.

3. MODEL

We consider a wireless ad hoc network in which nodes are distributed in the plane $\mathbb{R}^2$ or a finite region thereof. In our formal analysis, we assume that nodes are distributed

according to an *infinite Poisson point process* in the plane with spatial intensity (density) λ. That is, for any region R of the plane of area A and any $n \geq 0$, $\Pr[n \text{ nodes in } R] = (A\lambda)^n e^{-A\lambda}/n!$. Furthermore, for any two disjoint regions R_1 and R_2, the number of points in R_1 is independent of the number of points in R_2.

We also assume that each node has a unit transmission radius, so that it can communicate reliably and instantaneously with any other node at a Euclidean distance at most 1; radii other than 1 are modeled by suitably rescaling the density (and the area, in the finite case). When applied to the infinite Poisson point process of density λ, these communication links produce an *infinite random geometric graph* $G(\lambda)$. A fundamental property of $G(\lambda)$, first established by Zuev and Sidorenko [19], is that there is a *critical threshold* λ^* such that for all densities $\lambda > \lambda^*$, there is a non-zero probability $p_\infty > 0$ that a particular node is in an infinite connected component, or that "percolation occurs." Furthermore, for all densities $\lambda < \lambda^*$, $p_\infty = 0$. While an exact value is not known, simulations [20] predict that $\lambda^* \approx 1.44$.

In our simulations, we consider a *uniform point process* in which $A\lambda$ nodes are distributed independently and uniformly at random within a square of area A, thus producing a density of λ. It is known that the expected fraction of nodes in the largest component quickly approaches p_∞ as $A \to \infty$; see Chapters 10 and 11 of Penrose [7] for a rigorous treatment. Thus we require $\lambda > \lambda^*$ to enable multi-hop routing even in modestly sized deployments.

We assume that each node is able to broadcast a message to all listening nodes within its transmission radius, and will receive a broadcast message whenever it is awake. In our simulations and analysis, we assume that message delivery is reliable, and messages do not collide. No clock synchronization is necessary, but clock skews should be negligible compared to the length of one iteration of the algorithm, which we envision as being at least on the order of minutes. Lastly, the node must be able to wake from sleep mode to perform an operation after a specified interval.

4. THE Naps ALGORITHM

The *Naps* algorithm, executed at each node v in the network, proceeds as follows.

```
Naps(Neighbor threshold c ∈ ℤ⁺, time period T ∈ ℝ⁺)
  Wait random time t_v chosen uniformly from [0, T)
  while true do
    Broadcast HELLO message
    Start timer t
    i ← 0
    while t < T and i < c do
      Increment i for each HELLO message received
    Nap until t = T
```

In other words, each node v first waits a random amount of time and thereafter operates in time periods of length T. At the start of each period, v broadcasts a "HELLO" message. It then listens for HELLO messages from other nodes. If fewer than c messages are received before the end of the period, v remains awake during the entire period. Otherwise, v naps from the time the c^{th} message is received until the end of the period. Note that for $t \geq T$, the graphs occurring at times $t, t+T, t+2T, \ldots$ are equivalent.

In our simulations, we augment this algorithm in two ways. First, nodes use a random period length chosen from $[0, T)$ every 10th iteration. Without this re-randomization, poor random choices are repeated every period and can lead to certain nodes remaining awake for a disproportionate amount of time. Second, we effect non-integral values of c by having each node choose a neighbor threshold of either $\lceil c \rceil$ or $\lfloor c \rfloor$ randomly (with appropriate weighting) in each period. In our analysis, we do not consider these variants.

4.1 Design intuition

If each node were given both the existing density λ of the network and a target density $\hat{\lambda} < \lambda$, then we could simply have each node remain awake a fraction $\hat{\lambda}/\lambda$ of the time independently at random. By the Poisson thinning lemma [7], the waking graph would be a random graph from $G(\hat{\lambda})$ with the desired density. However, this limits the algorithm's flexibility since it requires that the nodes know λ.

Instead, each node v adaptively estimates its local density using an estimate of its degree ($\deg(v)$). As we show in Section 5, v will remain awake for an expected fraction $c/(\deg(v)+1)$ of each period. Hence as the density increases, nodes will have more neighbors and thus nap for longer periods, keeping the density of waking neighbors nearly constant. If every node has nearly the same degree, then we expect that c nodes will remain awake among the $\deg(v)+1$ nodes in each node v's radius. This yields an overall density of approximately c/π.

4.2 Parameter selection

The neighbor threshold c controls the target density of waking nodes; the time period T controls the rate of turnover of these waking nodes. The choice of these parameters is left to the application using *Naps*, though these constants must be known to all nodes in the system.

T should be chosen large enough that nodes are awake for intervals of time that are significant for the purposes of the application. T should also be chosen small enough that a substantial number of time periods occur during a node's lifetime to spread load evenly among nodes. While this may mean that the optimal T depends on the initial density, a choice of T on the order of minutes will likely suffice for most applications and densities. While there are naturally ways to adapt c and T over time, we do not examine them here.

As noted previously, the neighbor threshold c produces a waking node density of approximately c/π. In Section 5, we show that percolation occurs for $c > c^*$, and the experiments of Section 6 imply that $4 < c^* < 5$. Thus we envision that almost all applications will want to have $c \geq 5$ so that a large fraction of waking nodes are in the largest connected component. Beyond that, the desired density may vary greatly based on the application's particular needs.

5. ANALYSIS

In this section we prove that the *Naps* waking graphs exhibit a percolation threshold similar to that of random geometric graphs. We consider the waking graphs produced at time $t \geq T$ by running *Naps* on an initial geometric random graph G from $G(\lambda)$. The resulting waking subgraph, G', has distribution $G_t(\lambda, c, T)$, where c and T are the algorithm parameters. Note that the randomness of this distribution is over both the initial node placement and the nodes' start times $t_v \in [0, T)$ chosen by *Naps*.

LEMMA 1. *In any time period $[t, t+T)$, for $t \geq 0$, every node v broadcasts exactly one HELLO message at a time t'_v independently and uniformly distributed in the period.*

PROOF. Each node v emits a HELLO at times $t_v, t_v + T, \ldots$, where the t_v are independent and identically distributed in $[0, T)$. For some integer k, we have $t \leq kT < t + T$. Thus $t'_v = t_v + kT$ if $t_v < t - (k-1)T$, else $t'_v = t_v + (k-1)T$. This is a measure-preserving bijection between t_v and t'_v. □

LEMMA 2. *Consider $G \in G(\lambda)$ and any node v. After running* Naps *with random start times $t_v \in [0, T)$ until time $t \geq T$,*

$$\Pr[(v \text{ is awake at time } t) | \deg(v) = d] = \begin{cases} 1 & \text{if } d < c \\ \frac{c}{d+1} & \text{otherwise.} \end{cases}$$

PROOF. By Lemma 1, in the interval $[t-T, t)$, there are $d+1$ HELLO messages emitted by node v and its neighbors, independently and uniformly distributed at times $t_1 \leq t_2 \leq \cdots \leq t_{d+1}$. Disregarding the event of probability 0 that any two nodes have the same start times, node v is awake at time t if and only if its HELLO message is among the most recent c messages. Since any ordering is equally likely, this occurs with probability $\frac{c}{d+1}$. □

As a result, the distribution $G_t(\lambda, c, T)$ does not depend on t or T other than the requirement that $t \geq T$. Without loss of generality, we refer to the distribution as $G(\lambda, c)$. By running *Naps* on a random $G \in G(\lambda)$, the waking subgraph at time $t \geq T$ is a random $G' \in G(\lambda, c)$.

THEOREM 3. *Let $P_{\lambda,c}$ be the event that percolation occurs in $G(\lambda, c)$ – that is, that a particular waking node g is connected to an infinite connected component of waking nodes. For any $0 \leq p < 1$, there exist λ' and c' such that for any $\lambda \geq \lambda'$ and any $c \geq c'$, $\Pr[P_{\lambda,c}] \geq p$.*

The proof of the theorem translates the presence of continuum percolation on a graph from $G(\lambda, c)$ into the presence of bond percolation on a discrete lattice; for G chosen from $G(\lambda)$, we define a lattice on the plane and label the edges of the lattice *open* or *closed* based on the placement of nodes in G so that an infinite connected component of open edges in the lattice will imply an infinite connected component in G'.

Figure 1 shows the lattice construction. First, we construct a square lattice L of edge length $\sqrt{1/5}$ centered at an arbitrary point. Note that any two nodes in adjacent squares must be connected. Let L' be the dual lattice, which has a node at the center of each face of L and an edge e' perpendicular to each edge e of L . When we refer to a node, unless explicitly stated otherwise, we mean a node of the original graph G, not a node of L or L'. A waking node refers to a node of G'.

DEFINITION 4. *An edge e of L is* open *if there exist waking nodes (nodes in G') in each of the two squares adjacent to e. Otherwise, e is* closed. *An edge e' of L' is open if the corresponding edge of L is open; otherwise, e' is* closed. *A path (or component) is* open *when all of its edges are open; it is* closed *when all of its edges are closed; otherwise, it is neither open nor closed.*

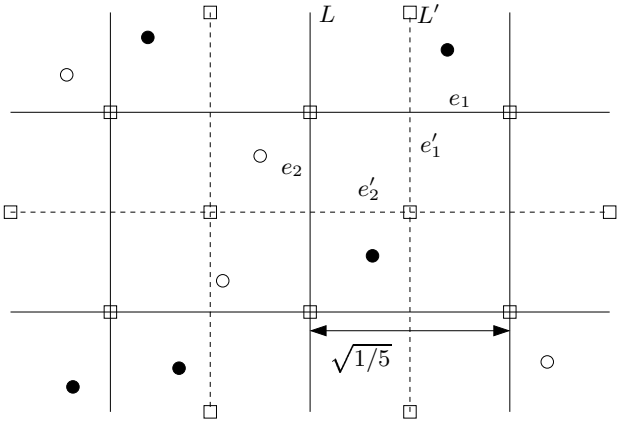

Figure 1: Nodes of L and L' are shown as squares and nodes of G are circles, filled when waking. Here e_1 and e'_1 are open, while e_2 and e'_2 are closed.

LEMMA 5. *If there is an infinite open path $P_\infty = \{e_1, e_2, \ldots\}$ in L' then there is an infinite connected component in G'.*

PROOF. Each edge e_i of P_∞ spans some adjacent faces f_i, f_{i+1} of L. Since e_i is open, there is a node v_i of G' in f_i. By our construction of L, nodes in adjacent faces are connected, so $\{v_1, v_2, \ldots\}$ is an infinite connected component in G'. □

The proofs of the next two lemmas follow the proof of Theorem 3.

LEMMA 6. *Let q be the probability that a particular edge of L is closed. Then for any $q' > 0$, there exist λ' and c' such that for $\lambda > \lambda'$ and $c > c'$, we have $q < q'$ for a random waking graph $G' \in G(\lambda, c)$.*

LEMMA 7. *Let P be a path of length n in L and let q be as in Lemma 6. Then $\Pr[P \text{ is closed}] \leq q^{n/247}$.*

PROOF. (of Theorem 3) Let v be the node of L' which is in the same face of L as g. By Lemma 5, it is sufficient to show that v is in an infinite component of open edges in L' with probability at least p. We then use the following fact: in any realization of $G(\lambda, c)$, if v's open component in L' is not infinite then it is finite, which by planar duality implies that there is a cycle of closed edges in L encompassing v. Thus, we have

$$\begin{aligned} &\Pr[g \text{ in inf. comp. in } G(\lambda, c)] \\ &\quad \geq \Pr[v \text{ in infinite open component in } L'] \\ &\quad = 1 - \Pr[v \text{ in finite open component in } L'] \\ &\quad = 1 - \Pr[\exists \text{ closed cycle in } L \text{ around } v] \\ &\quad \geq 1 - \sum_{n=1}^{\infty} \Pr[\exists \text{ closed cycle of length } n \text{ around } v] \\ &\quad \geq 1 - \sum_{n=1}^{\infty} \rho(n)(\Pr[\text{a cycle of length } n \text{ is closed}]), \end{aligned}$$

where $\rho(n)$ is the number of cycles of length n in L that surround v, and is upper-bounded by $\rho(n) \leq 4n \cdot 3^{n-2}$, as described in [6] (pps. 15-18). By Lemma 7, we can upper-bound the second factor by $q^{n/247}$, where q is the probability

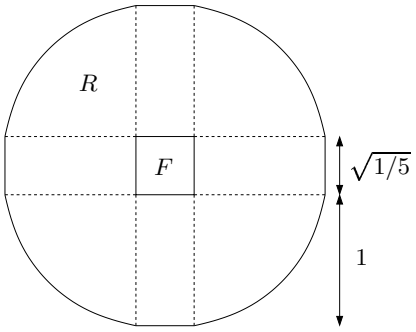

Figure 2: A face F of the lattice L and the region R, all points within distance 1 of F excluding F itself.

that a particular edge is closed (independent of all other edges). Continuing, these two bounds give us that

$$\begin{aligned}\Pr[g \text{ in inf. comp. in } G(\lambda,c)] &\geq 1-\sum_{n=1}^{\infty} 4n\cdot 3^{n-2}q^{n/247}\\ &= 1-\frac{4}{9}\sum_{n=1}^{\infty} n\cdot(3q^{1/247})^n\\ &= 1-\frac{4}{9}\cdot\frac{3q^{1/247}}{(1-3q^{1/247})^2},\end{aligned}$$

which is $\geq p$ when $q \leq \left(\frac{11-9p-2\sqrt{10-9p}}{27(1-p)}\right)^{247}$, and by Lemma 6 we can make q arbitrarily small as long as λ and c exceed some minimal values. □

Proof. (of Lemma 6) Let X be the event that a particular face F of L does not contain any waking nodes. We will show that we can make $\Pr[X]$ arbitrarily small by choosing λ and c sufficiently large. Since $q \leq 2\cdot\Pr[X]$ (by a union bound over two faces), this will imply that we can make q arbitrarily small.

Let R be the region of the plane containing all points of distance at most 1 from F, not including F. Let N_F and N_R be the number of nodes of G in F and R, respectively, and let A_F and A_R denote the area of F and R, respectively. These are depicted in Figure 2.

Let Y be the event that

$$\left(\frac{1}{k}\cdot \mathrm{E}[N_F] \leq N_F\right) \wedge \left(N_R \leq k\cdot \mathrm{E}[N_R]\right),$$

for a chosen k. We will use the fact that $\Pr[X] \leq \Pr[X|Y] + \Pr[\neg Y]$.

Since N_F and N_R are independently distributed Poisson random variables with $E[N_R] = A_R\lambda$ and $E[N_F] = A_F\lambda$, for any $k > 1$, there is a λ' such that $\lambda > \lambda'$ implies $P[\neg Y] < q'/4$.

Thus we may now restrict ourselves to examining $\Pr[X|Y]$. In this case

$$N_R \leq \left(k^2\cdot\frac{\mathrm{E}[N_R]}{\mathrm{E}[N_F]}\right)N_F = \left(k^2 A_R/A_F\right)N_F. \qquad (1)$$

Recall that our random graph $G(\lambda,c)$ is a snapshot of an arbitrary point in time t in the execution of the *Naps* algorithm. Consider the sequence of HELLO messages sent just prior to this moment by nodes in F and R. If any one of the last c messages was sent by a node v in F, then there must be a waking node in F, since v has seen fewer than c messages. Thus, if there are no waking nodes in F, then the past c messages must all have been sent by nodes in R. Note also that in the time period of length T ending at time t, exactly $N_F + N_R$ messages will be sent and these are uniformly and independently distributed in the period by Lemma 1. Thus, letting M be the number of messages sent by nodes in R between the most recently sent message from F and time t, and pessimistically assuming that $\deg(i) \geq c$ for all i, we have

$$\begin{aligned}\Pr[X|Y] &\leq \Pr[M \geq c|Y]\\ &= \frac{N_R}{N_F+N_R}\cdot\frac{N_R-1}{N_F+N_R-1}\cdot\cdots\cdot\frac{N_R-c}{N_F+N_R-c}\\ &\leq \left(\frac{N_R}{N_F+N_R}\right)^c\\ &\leq \left(\frac{(k^2A_R/A_F)N_F}{N_F+(k^2A_R/A_F)N_F}\right)^c \quad \text{(by Eqn. 1)}\\ &= \left(\frac{k^2A_R}{A_F+k^2A_R}\right)^c,\end{aligned}$$

which is a monotonically decreasing function of c. Hence there exists c' such that $c > c'$ implies $\Pr[X|Y] < q'/4$. Hence for sufficiently large λ and c, $q \leq 2\cdot\Pr[X] \leq 2(\Pr[\neg Y] + \Pr[X|Y]) < q'$. □

Proof. (of Lemma 7) Let X_e be the event that edge e is closed. Lemma 6 gives an upper bound on $\Pr[X_e]$ for a particular edge, but dependencies between edges make examining the entire path problematic. To handle this, we will pick a set of edges $S = \{e_1,\ldots,e_k\} \subseteq P$ such that the events $X_{e_1},\ldots,X_{e_k}$ are independent. Then we will have

$$\Pr[P \text{ is closed}] = \Pr[\bigwedge_{e\in P} X_e] \leq \Pr[\bigwedge_{e\in S} X_e] = (\Pr[X_e])^{|S|} = q^{|S|}.$$

We now need only find a set S of size $n/247$ that satisfies the above independence property. Consider a particular edge e of L. X_e is a deterministic function of the (random) waking/napping state of the nodes in the two faces adjacent to e. Furthermore, since each node's communication radius is 1, the state of these nodes is independent of the state of nodes at distance greater than 1 from them. Thus, X_e and $X_{e'}$ are independent if their adjacent faces are more than distance 2 apart. By laborious counting, we find that there are 246 edges within distance 2 of e's adjacent faces (excluding e). Thus each time we pick an edge out of P and put it in S, we bar ourselves from putting at most 246 other edges of P into S. Thus, we can pick S to be of size at least $n/247$. □

6. EMPIRICAL RESULTS

In this section, we measure and analyze the empirical performance of *Naps* under several different scenarios. Our measurements are based on a simple simulator. Initially $A\lambda$ nodes are placed uniformly at random (u.a.r.) in a square of area A. As per the *Naps* algorithm, each node v has a random start time $t_v \in [0,T)$, and the simulator "wakes" v at times $t_v + kT$ for $k \in \{0,1,2,\ldots\}$. We gather statistics about the n waking subgraphs occurring in $[T,2T)$. Specifically, we measure the *maximum component accessibility* (MCA) of these subgraphs, defined as the fraction of nodes that are are either in or have an edge to the largest connected

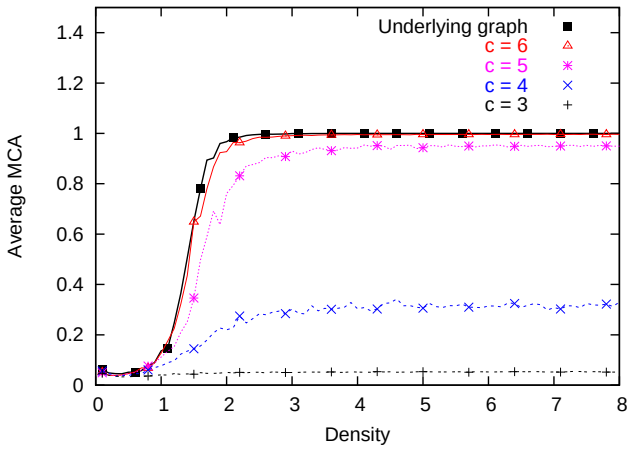

Figure 3: Average MCA as a function of density, for $A = 625$ and various c.

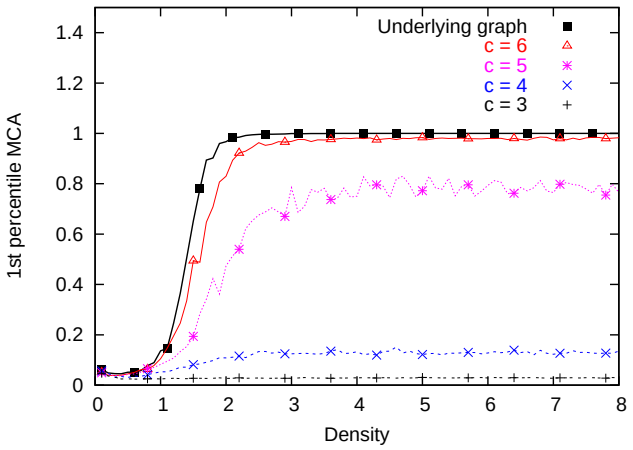

Figure 4: 1st percentile MCA as a function of density, for $A = 625$ and various c.

component of the waking subgraph. This appropriately captures the dual requirement that the waking subgraph be well-connected, and that almost every napping node should have an edge to it. Note that a connected dominating set is a waking subgraph with MCA of 1.

Our empirical results are as follows. We verify the asymptotic properties predicted by the analysis of Section 5 and demonstrate that the MCA is high for $c \geq 6$, except in small networks (Section 6.1). We show that *Naps* saves power effectively, especially at high densities, with network lifetime scaling almost linearly in λ (Section 6.2). *Naps* is robust in an obstructed environment, but may require a higher c and λ (Section 6.3). In a random waypoint model of mobility, *Naps* performs better than in the nonmobile case (Section 6.4).

6.1 Threshold Selection

We analyze the behavior of *Naps* for a large range of initial densities and for several choices of the *neighbor threshold* c. For each initial density λ on the x-axis, we run 20 trials. In each trial we generate a new random graph on which we simulate *Naps*. Based on samples at 100 random times in the period $[T, 2T)$, we calculate the average MCA, 1st percentile MCA, and average fraction of waking nodes for these sample points, and then average these values over the 20 trials. The choice of T does not affect the measurements of this section.

In Figure 3, we have chosen $A = 625$ and plotted the average MCA for densities between 0.1 and 8 in increments of 0.1 and integral neighbor threshold values between 3 and 6. We have also plotted the connectivity of the initial graph (MCA for $c = \infty$), which experiences a phase transition between densities 1 and 2 corresponding to the known critical threshold near 1.44 [20]. For $c \leq 4$, the MCA is qualitatively low in the resulting subgraph, even at high initial densities. On the other hand, for $c \geq 6$, the average MCA closely follows the largest connected component of the underlying graph. Figure 4 shows that the 1st percentile MCA, while lower than the average, is also qualitatively high for $c \geq 6$ and low for $c \leq 4$. For $c = 5$, while the average MCA is fairly high, it does produce some subgraphs with appreciably lower connectivity than the underlying graph as seen by the low measured value in this plot.

How many nodes do we expect to be awake at any given time? Recall that in the Poisson distribution, a node has degree d with probability $(\pi\lambda)^d e^{-\pi\lambda}/d!$. The probability that a node with degree d is awake is $c/(d+1)$ if $d \geq c$ or 1 otherwise, as given by Lemma 2. Thus,

$$\begin{aligned}
&\Pr[\text{node } v \text{ awake}] \\
&= \Pr[\deg(v) < c] + \sum_{d=c}^{\infty} \frac{c}{d+1} (\pi\lambda)^d e^{-\pi\lambda}/d! \\
&= \Pr[\deg(v) < c] + \sum_{d=c}^{\infty} \frac{c}{\pi\lambda} (\pi\lambda)^{d+1} e^{-\pi\lambda}/(d+1)! \\
&= \Pr[\deg(v) < c] + \frac{c}{\pi\lambda} \cdot \Pr[\deg(v) > c],
\end{aligned}$$

which tends from above to $c/(\pi\lambda)$ as $\lambda \to \infty$. Since the uniform distribution of our simulations approaches the Poisson distribution as $A \to \infty$, this suggests that the fraction of waking nodes will be slightly greater than $c/(\pi\lambda)$.

The simulation of Figure 5 verifies this rough calculation. There is diminishing utility for $c > 6$, since the number of waking nodes increases linearly with c, however the average MCA does not significantly increase above $c = 6$.

Figure 6 corroborates the result of Theorem 3. Fixing sufficiently large constants c and λ and increasing area, the 1st percentile MCA tends to a limit close to 1. However, for $c \geq 5$ connectivity is worse for small A, suggesting that one may need a neighbor threshold greater than 6 in small networks.

6.2 Energy Conservation

The preceding measurements show that the fraction of waking nodes decreases as the initial density increases, suggesting greater energy conservation. In this section we concretely measure the lifetime of a system running *Naps*.

For our simulations of the network over time, we give each node energy to survive awake for time $100T$. We parameterize the *sleeping-to-waking power ratio* r, so a node can sleep for $100/r$ time periods. We do not consider the power used in turning on and off the radio and sending the HELLO message, but we expect this cost to be negligible for large T. In this setting, the MCA is the fraction of nodes that

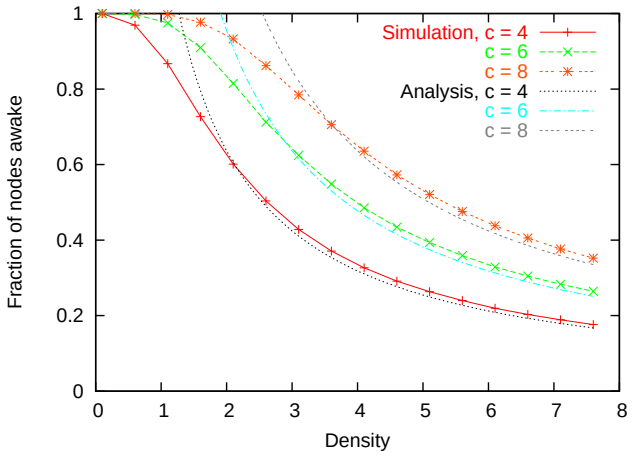

Figure 5: Average fraction of nodes awake as a function of initial density, for $A = 625$ and various c.

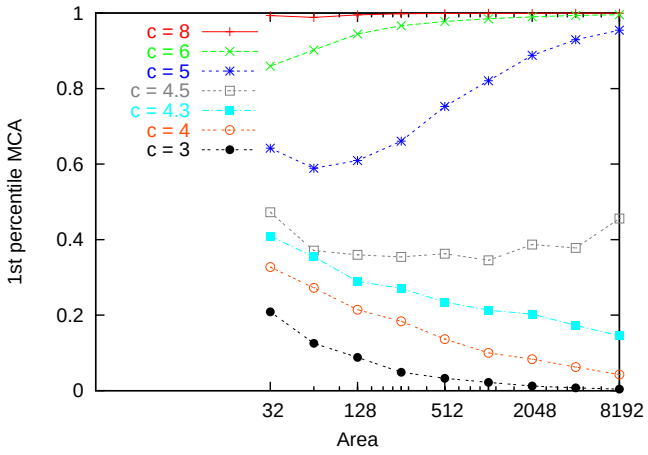

Figure 6: 1st percentile MCA as a function of area, for $\lambda = 5$ and various c.

are *not dead* (i.e. have energy remaining) and are either in or connected to the largest waking connected component. Notice that an MCA of p implies that at least a fraction p of nodes are alive, but fewer may be awake. We sample the MCA once in each time period T.

Figure 7 shows the MCA of the network over time for $r = 0.1$. The increase in network lifetime is sublinear in density since lifetime is limited to $100T/r$ regardless of density. The sharp decline towards the end of the system's lifetime implies that *Naps* effectively distributes energy consumption among nodes.

Figure 8 summarizes the power-saving results of *Naps*. We define *network lifetime* to be the amount of time that the MCA is at least 0.9. For various densities and sleeping-to-waking power ratios r, we plot the *factor increase in network lifetime* obtained by using *Naps* as compared to using no topology management. The simulation shows that *Naps* achieves arbitrarily large conservation for small enough r and high enough density. In particular, in the ideal case $r = 0$, network lifetime is linear in the initial density. Note that in the low density case $\lambda = 2$, the MCA occasionally falls below 0.9, as shown in Figure 4.

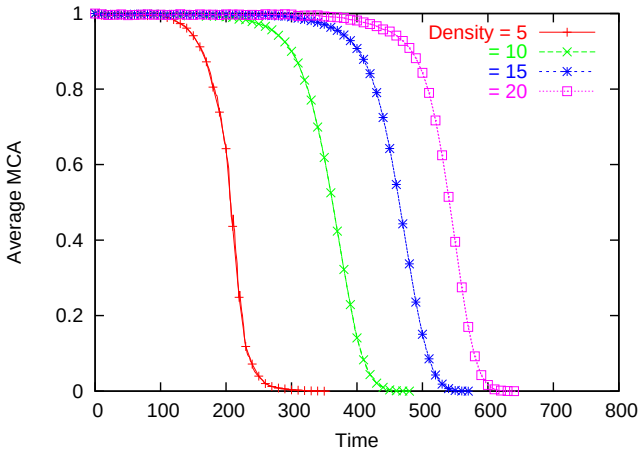

Figure 7: MCA as a function of time for various initial densities, averaged over 20 trials. $A = 625$, $c = 6$, $r = 0.1$, and a waking node survives for time $100T$.

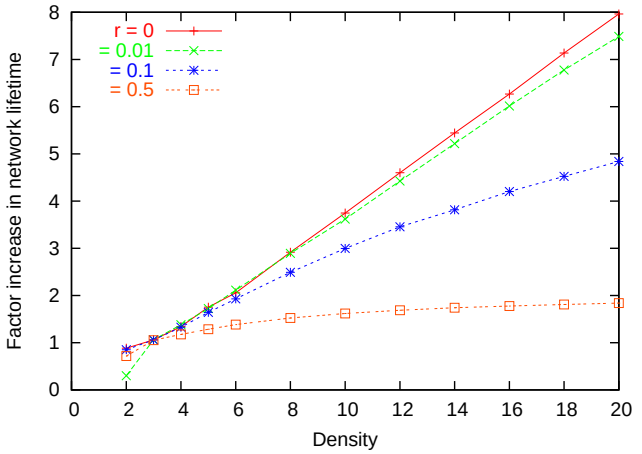

Figure 8: Factor increase in network lifetime vs. density for various r, averaged over 5 trials. $A = 900$, $c = 6$, and a waking node survives for time $100T$.

As a practical example, a Medusa II sensor node uses 22.06 mW with its microcontroller unit (MCU) on and radio in idle mode (and more if the radio is transmitting or receiving), but just 0.02 mW with the MCU sleeping and the radio off [2]. Thus $r \leq 0.001$ so our simulations show that *Naps* would increase network lifetime by at least a factor of 3.75 with density $\lambda = 10$ or a factor 7.87 with $\lambda = 20$ (with A, c, and T set as in Figure 8).

6.3 Robustness

To model a particular case of nonuniform node deployment, we place several rectangular obstructions in the square and distribute nodes u.a.r. outside these obstructions. Statistics are gathered as in Section 6.1. Figure 9 shows *Naps* run on such a graph.

Figure 10 plots average MCA as a function of density for various neighbor thresholds in this obstructed environment. We see that the obstructions negatively impact the connectivity of both the underlying graph and the waking sub-

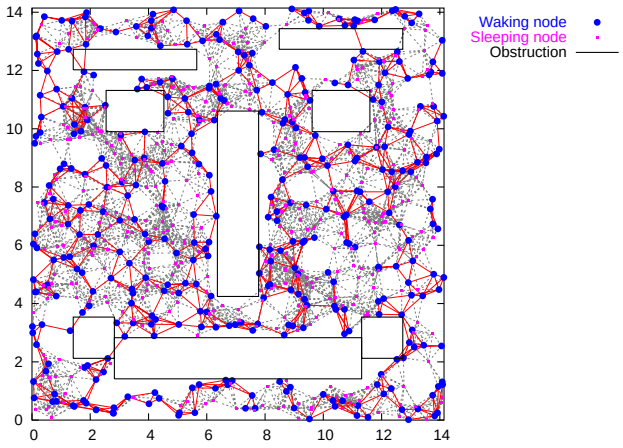

Figure 9: A snapshot of *Naps* run in a deployment with obstructions, $c = 6$, $\lambda = 4$, and $A = 200$.

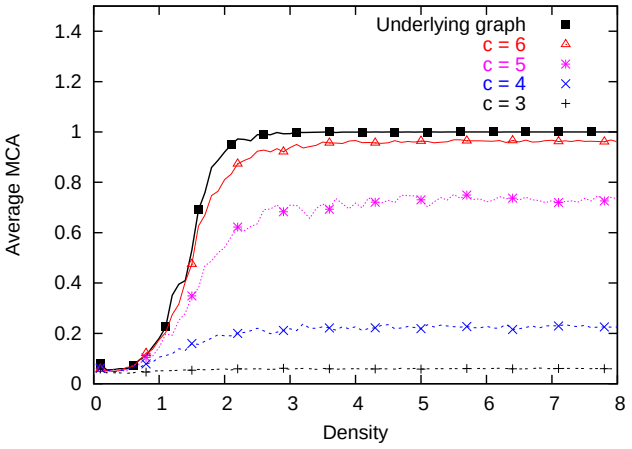

Figure 10: Average MCA as a function of density in a node deployment with obstructions, for $A = 625$ and various c.

graphs. For example, in comparison with the unobstructed case shown in Figure 3, for $\lambda = 2$ and $c = 6$ the average size of the largest component in the original graph drops from 0.97 to 0.93 and the average MCA drops from 0.93 to 0.81. Thus, one must consider deployment environment when choosing c and when choosing deployment density, regardless of whether or not a topology management algorithm is used.

This illustrates a potential enhancement to Naps. Ideally, nodes in well-connected regions would have lower c, and those in poorly connected regions would choose higher c. A mechanism for setting the neighbor threshold adaptively would allow one to avoid setting it to the highest value needed in any region of the network, thus offering further power savings in heterogeneous environments.

6.4 Mobility

To simulate mobility we use a random waypoint model. Each node begins in a location chosen u.a.r., as before. It then iteratively chooses a destination u.a.r., moves there at a speed chosen u.a.r. from $[s_{min}, s_{max}]$ units per time period

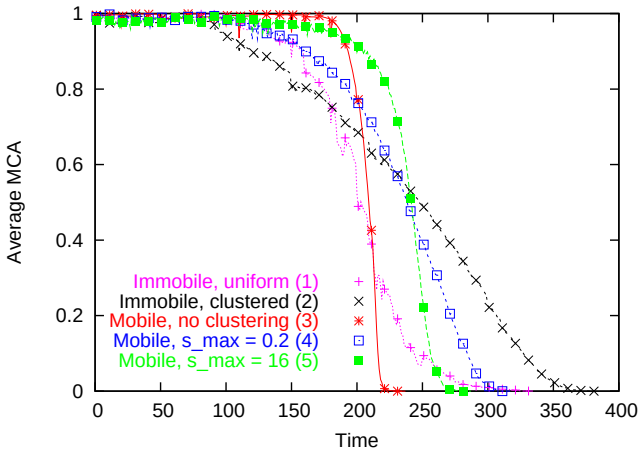

Figure 11: Average MCA vs. time under various deployment and mobility models, as described in the text. $A = 256$, $\lambda = 5$, $c = 6$, $r = 0.1$, and a waking node survives for time $100T$. Averaged over 5 trials.

T, and pauses for time p before repeating. Statistics are gathered as in Section 6.2.

Two properties of the random waypoint model have a significant effect on *Naps*. First, the movement of nodes over time improves load balance: it is less likely that a particular node is forced to remain awake for a disproportionately large amount of time due to a poor local topology. Second, nodes cluster around the center of the deployment region [21] since a node is more likely to move through the center of the region than through the edges on the way to its destination. Thus, nodes near the center spend more time napping.

To observe the effect of mobility independent of clustering, we set a very high s_{min} and s_{max} so that nodes essentially reach their destinations instantly, and we use a pause time of $p = 10T$. This has the effect of choosing a new location for each node u.a.r. every 10 time periods. To observe the effect of clustering independent of mobility, we generate a clustered deployment of nodes by taking a snapshot of the locations of nodes which have been moving according to the random waypoint model for some time. We then begin an immobile simulation at these locations. To observe the effects of mobility and clustering simultaneously, we use the random waypoint model with $s_{min} = 0$ and $p = 0$, and plot the results of $s_{max} = 0.2$ and of $s_{max} = 16 = \sqrt{A}$. Thus in the latter case, the fastest nodes can move the entire width of the deployment square in one time period. We choose $p = 0$ since it will cause the greatest fluctuation in neighbors [21] and is therefore an extreme mobility scenario.

Figure 11 exhibits the effects of mobility and clustering independently and in combination. We show for reference an immobile network with our standard uniform-random node deployment in Line (1). Line (2) shows that clustering in an immobile setting causes the nodes around the edges of the network to fail earlier since the local density is lower, but the majority of nodes are in the higher-density center and survive longer. Line (3) shows that mobility without clustering significantly improves load balance. Lines (4) and (5) show the combination of these factors in the random waypoint mobility model. Even when nodes are moving slowly ($s_{max} = 0.2$), load balance is significantly improved. In

particular, almost all nodes now benefit from the clustering since almost all nodes eventually travel through the center. As s_{max} increases, load balance improves further, though with diminishing returns.

7. CONCLUSION

We have presented *Naps*, a scalable, robust local algorithm for topology management in wireless ad hoc networks. The major advantage of the algorithm is its simplicity; nodes need only broadcast a single message every period and "nap" for the remainder of the period after receiving a threshold number of messages. Napping nodes may turn off hardware components as necessary to conserve energy, as the subgraph of waking nodes is very likely to be well-connected and may assume responsibility for forwarding traffic. All topology management algorithms, *Naps* included, require that higher layers of software can adapt to a changing topology without consuming inordinate amounts of energy. Initial work in this direction has been pursued in the context of routing [10], and we suspect that this will be an active and fruitful area of research. Due to its limited reliance on topology, *Naps* incurs no extra overhead as a result of mobility, and actually performs better due to improved load balance.

Because *Naps* provides only probabilistic guarantees, more complex algorithms may be necessary for some applications. We suspect that in practice, however, node and communication unreliability will force applications to cope with probabilistic guarantees anyway, so *Naps* may provide a simple and viable alternative.

Acknowledgment

The authors would like to thank David Aldous, Alex Fabrikant, Dick Karp, and Ion Stoica for useful comments and suggestions.

8. REFERENCES

[1] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," in *Mobile Computing and Networking*, 2001, pp. 85–96.

[2] V. Raghunathan, C. Schurgers, S. Park, and M. Srivastava, "Energy aware wireless microsensor networks," *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 40–50, March 2002.

[3] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava, "Optimizing sensor networks in the energy-latency-density design space," *IEEE Transactions on Mobile Computing*, vol. 1, no. 1, pp. 70–80, January-March 2002.

[4] P. Gupta and P. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, March 2000.

[5] Michael R. Garey and David S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman & Co., 1979.

[6] Geoffrey Grimmett, *Percolation*, Springer, 2nd edition, 1999.

[7] Mathew Penrose, *Random Geometric Graphs*, Oxford University Press, 2003.

[8] P. Gupta and P. Kumar, "Critical power for asymptotic connectivity in wireless networks," in *Stochastic Analysis, Control, Optimization and Applications: A Volume in Honor of W.H. Fleming*, W.M. McEneaney, G. Yin, and Q. Zhang, Eds., pp. 547–566. Birkhauser, Boston, 1998.

[9] Brad Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Mobile Computing and Networking*, 2000, pp. 243–254.

[10] Ananth Rao, Christos Papadimitriou, Scott Shenker, and Ion Stoica, "Geographic routing without location information," in *Proceedings of the 9th annual international conference on Mobile computing and networking*. 2003, pp. 96–108, ACM Press.

[11] Y. Xu, S. Bien, Y. Mori, J. Heidemann, and D. Estrin, "Topology control protocols to conserve energy in wireless ad hoc networks," submitted for review to IEEE Transactions on Mobile Computing, January 2003.

[12] Y. Xu, J. Heidemann, and D. Estrin, "Adaptive energy-conserving routing for multihop ad hoc networks," Tech. Rep. 527, USC/Information Sciences Institute, 2000.

[13] Lichun Bao and J. J. Garcia-Luna-Aceves, "Topology management in ad hoc networks," in *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*. 2003, pp. 129–140, ACM Press.

[14] R. Zheng, J. C. Hou, and L. Sha, "Asynchronous wakeup for ad hoc networks," in *Proceedings of the Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2003.

[15] L. Booth, J. Bruck, M. Franceschetti, and R. Meester, "Covering algorithms, continuum percolation, and the geometry of wireless networks," *Annals of Applied Probability*, vol. 13, no. 2, May 2003.

[16] D. Tian and N.D. Georganas, "A coverage-preserving node scheduling scheme for large wireless sensor networks," in *Proc. ACM Workshop on Wireless Sensor Networks and Applications*, Atlanta, May 2003.

[17] Fabian Kuhn and Roger Wattenhofer, "Constant-time distributed dominating set approximation," in *Proceedings of the 22nd ACM Symposium on Principles of Distributed Computing*, 2003.

[18] O. Dousse, F. Baccelli, and P. Thiran, "Impact of interferences on connectivity of ad hoc networks," in *Proc. IEEE Infocom*, San Francisco, April 2003.

[19] S. A. Zuev and A. T. Sidorenko, "Continuous models of percolation theory i, ii," *Theoretical and Mathematical Physics*, vol. 62, pp. 51–58,171–177, 1985.

[20] S. Quintanilla, S. Torquato, and R. M. Ziff, "Efficient measurement of the percolation threshold for fully penetrable discs," *Journal of Physics A*, vol. 33, no. 42, pp. L399–L407, October 2000.

[21] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, no. 5, pp. 483–502, 2002.