Spreadable Connected Autonomic Networks (SCAN)

Joshua Reich[a]
reich@cs.columbia.edu

Vishal Misra[a]
misra@cs.columbia.edu

Dan Rubenstein[b]
danr@ee.columbia.edu

Gil Zussman[b]
gil@ee.columbia.edu

[a]CS / [b]EE Department, Columbia University, New York, NY, USA

ABSTRACT

A Spreadable Connected Autonomic Network (SCAN) is a mobile network that automatically maintains its own connectivity as nodes move. We envision SCANs to enable a diverse set of applications such as self-spreading mesh networks and robotic search and rescue systems. This paper describes our experiences developing a prototype robotic SCAN built from commercial, off-the-shelf hardware, to support such applications. A major contribution of our work is the development of a protocol, called SCAN1, which maintains network connectivity by enabling individual nodes to determine when they must constrain their mobility in order to avoid disconnecting the network. SCAN1 achieves its goal through an entirely distributed process in which individual nodes utilize only local (2-hop) knowledge of the network's topology to periodically make a simple decision: move, or freeze in place. Along with experimental results from our hardware testbed, we model SCAN1's performance, providing both supporting analysis and simulation for the efficacy of SCAN1 as a solution to enable SCANs. While our evaluation of SCAN1 in this paper is limited to systems whose capabilities match those of our testbed, SCAN1 can be utilized in conjunction with a wide-range of potential applications and environments, as either a primary or backup connectivity maintenance mechanism.

1. INTRODUCTION

Ensuring network connectivity is a fundamental problem in the wireless networking domain. The problem of placing and scheduling predominantly immobile nodes so as to form a connected network is fairly well understood, as is that of enabling end-to-end connectivity in mobile networks, so long as they maintain physical connectivity. However, there has been significcnatly less work exploring techniques that *control node movement specifically to enforce physical layer connectivity* of networks comprising mobile wireless nodes. In this paper we examine this third issue, defining a new network class, the *Spreadable Connected Autonomic Network (SCAN)* - a mobile wireless network that automatically maintains its own connectivity by regulating the movements of its constituent nodes.

SCANs have several potentially useful application domains. Here, we point out two:

- **Self-Deploying Mesh Network Infrastructure**: In this application a SCAN spreads over some area to provide wireless coverage for previously disconnected, wireless (perhaps stationary) end nodes. Once it has covered these nodes, the SCAN can provision them with point-to-point connectivity. By ensuring that mobiles remain connected, a self-spreading mesh network infrastructure can provide wireless services in a more robust way than could a sporadically connected network. Moreover a system that maintains connectivity inherently prevents nodes from disconnecting - thereby preventing them from wandering off and becoming lost and unused.

- **Search and rescue**: A wirelessly-communicating robotic search and rescue team deployed during an ongoing disaster must move to locate victims, but it must also remain connected to base-station points to immediately notify first responders when a victim has been found, in addition to providing ongoing feedback about the search environment.

This paper explores how to enforce the connectivity requirement of a SCAN within a set of communicating mobile nodes. Generally, the mechanism that enforces the procedure will depend heavily on the mobility requirements of the application, the movement capabilities of the nodes, and the various technologies the nodes can employ to sense their environment. Our focus is on techniques that are *robust*, in that they can be deployed across a wide range of SCAN scenarios. Our solutions are therefore generally deployable, though not necessarily optimal for particular applications, and thus can serve as a useful baseline for subsequent solutions that are fine-tuned for the particular scenario.

Our solutions also apply in the real-world prototype SCAN that we have developed from "off-the-shelf" components. We use a set of mobile Roomba floor cleaning robots, and task them with standard 802.11 communicating devices. Roombas are able to move forward, rotate, and freeze, but have

1

very limited sensing capabilities and know little about their surrounding environment, such as their current position and current direction. Effectively, they are driving "blind". A pair of Roombas can directly communicate when they are sufficiently close to one another, but they can easily move out of one another's communication range. Furthermore, we have found that in realistic environments with background noise and obstacles, 802.11 transmissions are not a good predictor of an upcoming imminent disconnect. Hence, if one wants to ensure that two communicating Roombas do not disconnect from one another in the near future, they must **freeze** their movement in their current positions.

To maintain connectivity, we propose a novel algorithm, the first Spreadable Connected Autonomic Network algorithm (SCAN1), that enables individual nodes to determine when they must constrain their mobility (in our setting, freeze). Most notably, SCAN1 achieves our global connectivity goal through an entirely distributed process in which individual nodes utilize only local knowledge (2-hop) of the network's topology and does so without the need for any specialized hardware (e.g. GPS) beyond standard 802.11. SCAN1's loose requirements on node mobility patterns and node capabilities ensures that it is an effective, simple mechanism that can be deployed in *any* SCAN in which nodes have the ability to freeze.

The contributions of our work are as follows:

- We define a new class of mobile wireless networks, the Spreadable Connected Autonomic Network (SCAN), that can autonomically maintain its own connectivity while allowing its member nodes significant leeway to spread across an area as they individually choose.

- We propose SCAN1 as a mechanism for enforcing physical layer connectivity in a SCAN (Section 3).

- Through analysis and simulation, we evaluate SCAN1's ability to cover an area as a function of how frequently network partitions arise. We also identify a phase-transition point as a function of the number of mobiles and the size of the containing region in which a SCAN operates as to whether the network will entirely freeze.

  We construct an analytic model under which we analyze the properties of SCAN1 in larger networks in more expansive topological settings than our testbed permits (Section 4).

- We implement SCAN1 on our mobile robotic network testbed as proof-of-concept, and evaluate its performance in two very different real-world (albeit simplified) potential SCAN applications: a self-deploying mesh network system and collaborative search and rescue scenario (Section 5).

2. PROBLEM SETTING

In this section, we begin by motivating our general problem of interest, and then describe the details of our testbed that motivated our baseline SCAN model, which is described at the conclusion of the section.

As cataloged in [13], there are many different types of systems regard to function, mobility, node density, awareness - that might utilize mobility to achieve some network goal such as global connectivity. Our general problem, stated simply, is "How can we ensure a currently connected network of mobile nodes remains connected as these nodes move?"

How one answers this question depends heavily on the objectives of the mobile nodes, their movement capabilities, and what information these nodes can obtain and utilize about their environment, such as their positions, directions of movement, future directions of movement, and effect of their movement on their communication capabilities.

Our interest is to design techniques that can be utilized in a large variety of SCAN configurations. Hence, the fewer assumptions one makes about node abilities, the more robust the resulting solution. This also permits our solution to serve as a baseline for comparison, since it is easily deployable in settings where node movements are more restrictive (assuming nodes can still freeze), and as certain capabilities, such as access to GPS or predicting future connectivity, become unavailable. We use our mobile robotic networking testbed as this baseline.

2.1 Our Mobile Robotic Networking Testbed

The mobile robotic networking testbed we use for our implementation and experiments is similar to recent systems such as [24], [5], and [34]. Each mobile node in our testbed is composed of an iRobot Roomba Create mobility and sensing platform (Figure 1). On top of each Roomba we affix a Linksys WRTSL54GS wireless router running a build of OpenWrt Linux. The Linksys router provides an integrated unit featuring a Broadcom 4704 processor running at 266MHz, 8MB flash, 32MB RAM, an integrated Broadcom wireless 802.11g radio, BCM5325 switch, and a USB 2.0 port. The WRTSL54GS provides communication, computation and memory, while the Roomba provides power, movement, and environmental sensing. The Roomba platform and the router are connected by a modified serial cable that allows the router to both poll the Roomba's sensors and control the Roomba's motors and actuators. The serial cable also provides a direct unregulated power feed to the Roomba's battery which we use to power the router. Typically a node can run without a recharge for several hours.

This setup allows us to experiment with real mobile nodes whose broadcast and mobility decisions we can specify utilizing standardized programming languages.

Utilizing standard commercial equipment does have drawbacks. Both the capabilities of and our access to the router's wireless card is limited. The Broadcom wireless card in our routers does not support distance estimation, and the driver is proprietary. Consequently, we cannot access much of the link-layer which means all of our communication routines must run at or above the IP layer. We have attempted to

Figure 1: A Testbed Node

access some link-layer functionality through OpenWrt's wl package. However, the most useful information we could obtain was a global and not very stable RSSI reading of the channel which, because of the plenitude of 802.11 crosstalk, did not provide much information about access signal strength on a per-peer basis.

Our testbed nodes also do not have integrated GPS (although this may be a possibility for the future) and could not have used it in our experimental environment which was indoors and thus GPS-denied. Finally, the odometric information available through the Roomba's sensors is fairly low quality. Without some supplemental system for localization, such as Cricket [21], a node's positional estimate degrades very quickly. Hence our nodes are essentially driving "blind".

These constraints led us to explore techniques that can be implemented with local communication[1], utilizing only information obtainable through inter-neighbor communication; specifically, knowledge of the current local connectivity structure. They also led us to explore freezing node position as our mechanism for ensuring the integrity of critical connections.

It bears mentioning that there are several excellent pieces of research which address localization [16, 21] and internodal distance inference [19] utilizing basic wireless broadcast capabilities, and that is worth considering in future work how these localization schemes could be utilized to bring nodes who recently disconnected back toward one another to enable re-establishment of their connection, or, with good estimates of the distance where they are likely to disconnect, to keep them from reaching this distance.

2.2 Connectivity in Our Testbed

The first challenge in implementing SCAN1 on our testbed was in determining when two Roombas should be considered to be "connected". Consider two Roombas, which we label A and B. If A and B can receive (a majority of) one another's transmissions, then clearly they should be connected,

---

[1]we employed a lightweight UDP based message exchange protocol

and if A and B cannot hear one another's transmissions, they should not be connected. But what about cases in between? In particular:

- A could hear B, but the reverse need not hold true.

- A might hear from B only intermittently, their connection being too sporadic to allow for communication that consumes any significant bandwidth.

As discussed in Section 2.4, we consider nodes to be connected only when they can directly, mutually, and consistently communicate over a wireless channel. In order to determine whether this was the case we utilized the following scheme for neighbor detection.

2.2.1 Neighbor Detection Scheme

The protocol run by a node to determine connectivity repeatedly transmits over a fixed period of time we refer to as a *broadcast cycle*. We make no attempt to synchronize the start time of a cycle across nodes, or to eliminate drift across node clocks, as such tight synchronizations are not needed for our use of the broadcast cycle. The remainder of this subsection discusses the details of how we determine when two nodes are connected within a broadcast cycle, and how a node learns of both its neighbors and 2-hop neighbors (i.e., its neighbors' neighbors). This discussion can safely be skipped by the reader who is not concerned with these details.

In our testbed, a broadcast cycle lasts 1.5 seconds. In each cycle, a node A classifies a node B whose transmissions it hears during the previous or current cycle into one of three possible classes: *heard*, *reciprocally-heard*, and *confirmed* as a neighbor. B is *heard* by A if A receives transmissions from B. B is *reciprocally-heard* if in addition, A receives a broadcast from B indicating A is currently heard by B. Finally, B is confirmed as a neighbor if A has reciprocally-heard from B in two consecutive broadcast cycles. Once B is confirmed as a neighbor, B remains in A's set of neighbors until such time as A fails to reciprocally hear B for two consecutive broadcast cycles.

The messages sent in a broadcast cycle, a node broadcasts a sequence of 3 *HEARD* messages followed by a sequence of 3 *CONFIRMED* messages. A *HEARD* message begins with the prefix "2" followed by a list of IP addresses nodes heard by the sender in its previous broadcast cycle. A *CONFIRMED* message begins with the prefix "3" and is followed only by a list of IP addresses belonging to the sender's confirmed neighbors. Either *HEARD* or *CONFIRMED* messages received can be used to classify a node as being reciprocally heard. However, only nodes whose addresses appear in a *CONFIRMED* message can be used for SCAN1's calculations.

We found this scheme to be very effective in both locally communicating the information used by SCAN1 and providing neighbor sets that were fairly stable against random

3

fluctuations on the wireless channel - without artificially restricting the set of neighbors detected.

2.3 Node Mobility Pattern

Nodes decide at the end of each broadcast cycle whether they should move or freeze for the next broadcast cycle. Their view of the network is effectively fixed between cycles, so there is little reason for them to change their mind at some point in the middle.

Aside from sometimes requiring the Roombas to freeze, we place no other explicit restrictions on their movement. Essentially, a Roomba moves straight until it bumps into an obstacle. It has two sensors which allow it to determine whether the obstacle is off to the side or in front of it. If the obstacle is to a side, it rotates a random angle from the obstacle and continues. If in front, it backs up slightly, rotates a random angle, and proceeds. The important point to take away is that *node movement is oblivious to any network connectivity requirement.*

This algorithm ran as a separate thread on each of our nodes. At the end of each broadcast cycle each node would reassess the SCAN1 criterion. If its mobility state changed from freezing to moving, the main thread would notify the mobility thread to begin again. Conversely, if the mobility state changed from moving to frozen, then the thread would be paused and a freeze command sent to the Roomba platform.

2.4 Network Model

Armed with a solid practical definition of node connectivity by a link, we can now view a SCAN in a more mathematical form that will facilitate the description of the SCAN1 algorithm.

We view the network as a graph in which nodes are mobiles. We say two nodes u and v are *neighbors* and are *directly connected* via a *link* if they can directly, mutually, and consistently communicate over a wireless channel (if u can receive v's broadcast but not visa-versa then u and v are not directly connected). We refer to two directly connected nodes as *neighbors.*

We define $N(u)$ to represent the set of nodes that are node u's neighbors with $u \notin N(u)$, and assume, as is the case in our testbed, that each node u knows $N(u)$, and is also informed of $N(v)$ for each of its neighbors $v \in N(u)$.

We say u and v are *connected* if there is a *path* from u to v across a series of links. A network is globally *connected* when every two nodes u and v in the network are connected.

With a fixed period T, each node assesses its current local connectivity, and, based on this state decides to either *move* or *freeze* until its next assessment. We do not require that nodes make their assessments simultaneously, nor do their respective T's need to match exactly (clock drift is permissible).

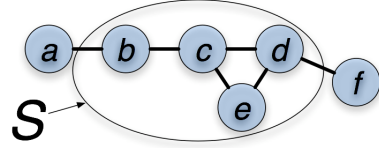Over time, new links can *form* and existing links can *fail*.

Figure 2: a **and** b **are neighbors.** a **and** d **are connected by path** (a, b, c, d)**.**

However, the state of a link between two nodes u and v can change only if at least one of the nodes is moving. If both u and v are frozen, then an existing link between them cannot fail, nor can a link be added when there is none. In other words, only a pair of nodes' mobility significantly alters whether they can or cannot communicate within a broadcast cycle.

3. SCAN1

In this section, we describe the our main algorithm, SCAN1, that nodes use to enforce SCAN connectivity by telling individual nodes when to freeze. We also describe a simpler algorithm that will be used as a point of comparison during our simulations, whose ability to keep the network connected while permitting mobility is comparatively poor.

SCAN1 should allow nodes freedom to move as much as possible, so long as they do not disconnect the network. Note that a node's view of the network is a local one, comprising simply its neighbors and its neighbors' neighbors. Since we cannot access information regarding the relative fragility of the various connections were the node to move, the graphical representation presented in Section 2 is a good description of the node's vantage point in the practical setting we explore.

To get a feel for when, from this perspective we may wish to freeze a node, consider the example in Figure 3. Node a
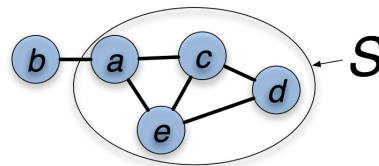

Figure 3: A network with both robust and fragile connections

is connected to 3 neighbors. To disconnect a from any of the neighbors to its right (or in fact any of node to which it is connected) at least 2 links in S must be broken. In contrast, only a single link needs to fail to disconnect node a from node b. If links fail infrequently (e.g., 1 failure during a broadcast cycle), then if a were only concerned about the nodes in the set S, it could continue to move. However, there is a high likelihood that any movement could cause the single link between a and b to fail, ending the connection

with b, thereby partitioning the SCAN. To keep the network partition-free, both a and b should freeze.

But what if, during a broadcast cycle, $k \geq 1$ links can be expected to fail? How do we then determine whether nodes must freeze to ensure SCAN connectivity is maintained? It is this question that SCAN1 is designed to address.

Formally, SCAN1 is pre-configured with a parameter k, such that a node u is allowed to move as long as $|N(u) \cap N(v)| \geq k$ for every $v \in N(u)$. In other words, u moves if it shares k neighbors with each of its neighbors v - information we have already shown is easily obtained. If this property does not hold for even a single neighbor, u must freeze.

Algorithm 1: SCAN1's Mobility Criterion

---

if $|N(u) \cap N(v)| \geq k, \ \forall v \in N(u)$ **then**
  $\llcorner$ move;
else
  $\llcorner$ freeze;

3.1 Global Network Connectivity Proof

Intuition for why SCAN1 is desirable can further be extrapolated from the following Lemmas and Theorem:

LEMMA 1. *In any graph, if u and v are directly connected and $|N(u) \cap N(v)| = m$, if fewer than m links fail, then there is a path of at most 2 hops from u to v.*

PROOF. Clearly, if u and v remain directly connected, then there is a 1-hop path from u to v. To remove 2-hop paths, a link must fail between each node in $N(u) \cap N(v)$ and either u or v. Hence, at least $m + 1$ links must fail to remove all 1 and 2-hop paths. $\square$

For the following Lemma, we remind the reader that, as stated in Section 2.4, links cannot fail between a pair of frozen nodes.

LEMMA 2. *Consider a network that is connected at some time t, with all nodes using SCAN1 with parameter k. If at most k links can fail near a node during its broadcast cycle, then the network remains connected for the duration of the broadcast cycle.*

PROOF. We note that links may form during the period in which k links fail. If we ignore these forming links and show the network remains connected, then clearly the network remains connected when we add these forming links back in.

The simplest proof is by contradiction. Let t be the time that the graph partitions, and let the partition result from the edge connecting u and v failing. This means that either u was moving or v was moving under SCAN1. WLOG, assume u was moving at time t. This implies that $|N(u) \cap N(v)| \geq k$ at the start of u's broadcast cycle that contains time t. Since at most k links can fail in a broadcast cycle, at most k links can fail between the start of this broadcast cycle and time t. By Lemma 1, u and v must be connected (within

at least 2 hops) even after k links fail, and hence cannot reside in separate partitions at time t. $\square$

Lemma 2 states that, by using SCAN1, even if k links suddenly drop simultaneously (and are connected to at least one moving node), the way that SCAN1 chooses nodes to move and freeze ensures that these k dropped links, each of which must drop between two nodes where at least one is moving, will not disconnect the network.

Finally, we can state our main theorem:

THEOREM 1. *If a node reconsiders its decision to move or freeze at the end of each of its broadcast cycles, and at most k links can fail within its broadcast cycle, then a SCAN that is initially connected will always remain connected.*

PROOF. The proof is by induction on the iteration of the broadcast cycle. Our assumption is that for the initial broadcast cycle, the network is connected. Using the inductive assumption, assume the network remains connected by the end of the ith broadcast cycle of node u. u reassesses its local connectivity at the end of the cycle, and moves in the $i + 1$st broadcast cycle only if $|N(u) \cap N(v)| \geq k$ for all $v \in N(u)$. We simply apply Lemma 2 with t equal to the start time of the $i + 1$st broadcast cycle to complete the inductive step. $\square$

3.2 Choosing k

The best value of k depends upon how many neighboring links are expected to fail (i.e., mobiles move out of communication range) within a broadcast cycle. As one increases the speed of a node, decreases the range of transmission, or increases the broadcast cycle time, a larger k is needed to ensure connectivity. In general, as k increases, it becomes less likely that the network will partition, but the expected time that nodes spend moving is reduced as well, which can slow progression of a SCAN achieving its goal for which mobility of nodes is required In Section 4 we see that for network whose nodes move slowly $k = 2$ is more than sufficient while for more volatile settings $k > 4$ appears extremely robust.

3.3 Neighbor Density Algorithm

While SCAN1 utilizes a fairly simple mechanism to determine when nodes should freeze, there exist simpler mechanisms. As a point of comparison, we consider a second algorithm, the *Neighbor-density algorithm* (ND) as a naive parameterizable heuristic solution to the connectivity problem. ND utilizes nodal density to achieve connectivity: if a node has more than k neighbors it can move, fewer it must freeze. While ND is not practically of much use of small networks (as we see in simulation it requires k in excess of our largest fielded system to produce very good results), its performs moderately well in larger, dense networks. In Section 4.2.2 we compare the performance of SCAN1 and ND

in simulation, showing SCAN1 to be both more resilient and less constraining than ND.

Algorithm 2: ND's Mobility Criterion

if $|N(u)| \geq k$ **then**
  ∟ move;
else
  ∟ freeze;

Both ND and SCAN1 share the common assumption that, while the strength of the wireless channel may fluctuate unpredictably over space, it will remain relatively constant over time. This is to say, a small relative movement between two nodes can easily change whether a direct connection exists between these nodes, but the progression of time will be unlikely to change the state of direct connectivity between a pair of frozen nodes. In situations where the strength of the wireless channel is fluctuating wildly over time (e.g., significant and varying external interference, radiation storms) more specialized techniques will be required. Mild to moderate fluctuations over time should have relatively little effect on SCAN1's actual performance as it is an inherently conservative. In fact this assertion was borne out in both our hardware experiments and simulation. The same holds true for ND with larger values of k.

4. SIMULATION AND ANALYSIS

In this section we seek to answer three essential questions regarding the behavior of networks running SCAN1, utilizing both simulation and analysis.

- How many nodes are required as a function of transmission radius and topology size to avoid having the network freeze entirely?

- How well does a SCAN, when frozen, cover an area?

- How frequently does the SCAN partition?

The first question is relevant to search-and-rescue operations, where we want to ensure enough nodes are deployed so that the network does not spread so far as to have all nodes freeze. The second question is applicable to the mesh coverage setting, where the objective is to cover as much area as possible. Our last question is important in both settings, as we wish to minimize the frequency with which nodes are disconnect from the network. We provide answers to all of these questions as functions of the number of nodes N, mean transmission range r, value of k (for SCAN and ND), and the size of the space in which the SCAN is deployed A.

All simulation discussed in this section was conducted on NetLogo 4.0.2 [33], a combined Logo-like language and simulation platform - ideal for modeling a distributed protocol whose behavior influences and is influenced by the topology of the dynamically evolving network upon which it is running.

To provide additional realism to our simulations, we varied the range of broadcast stochastically. In our simulation two neighbors are connected if the distance between them is less than or equal to a normal random variable. Unless otherwise noted we investigated two cases:

- low-stochasticity link variation (STD = 5%) - where the standard deviation of the links was 5% of the mean link length

- high-stochasticity link variation (STD = 20%) - where the standard deviation of the links was 20% of the mean link length

Each pair of potential pair of neighbors had its own independently chosen random variable, and these random variables were regenerated at a rate equal to the frequency of the movement decision made by the nodes.

Nodes, when moving, moved in random directions unless they hit a boundary, and periodically recomputed the direction in which they moved. Their speeds were fixed at a constant rate.

4.1 The Freeze Phase-Transition

If our nodes are confined to a very small space relative to their transmission radii, then they will never freeze. As the size of the space is increased, and they are able to spread further apart, the likelihood of freezing increases. As the size of the space grows to ∞, we eventually will reach a point where, with probability 1, all nodes will eventually freeze simultaneously. At this point, since all nodes are frozen, they cannot obtain new neighbors, and the network remains in a frozen configuration. Here, we investigate the point of the phase-transition: for a given k and N, what is the ratio of the size of the space to the transmission radii where beneath the value, the network is forever moving, and above whose value the network always reaches a freezing configuration?

By knowing where this phase transition occurs, a wireless practitioner can adjust the size of the network so as to achieve his or her goal, whether it be ensuring the network reaches a stable frozen configuration while covering the space with high probability (e.g., self-spreading mesh network) or allowing nodes to continually move while maintaining connectivity (e.g., a search and rescue or intrusion detection system).

We will first build a model to predict this point and then verify our model's accuracy through simulation.

4.1.1 Minimum Bounding Area

Our model will be composed of two components: a regular spatial pattern in which nodes can be laid out in a frozen configuration and the minimum scaling of this spatial pattern below which additional links will form. If we choose our model appropriately, the minimum area needed by this model will approximate the minimum area needed for such a system to freeze. Our main task is identifying a regular spatial pattern that is more dense than almost any frozen configuration we can expect to encounter and then delivering a closed formula for its size as a function of k, N and r.

We begin by considering the simplest case $k = 1$ and a simple topology, a perfectly square space of area A. In order for a network to freeze, we must find some spatial configuration of the nodes such that (a) no node has any neighbors in common with any other neighbor, (b) the nodes are configured as densely as possible, and (c) the configuration is regular enough to analyze easily. The final criterion leads us to explore *regular tessellations*. A *tessellation* is created when a shape is repeated over and over again covering a plane without any gaps or overlaps. A *regular tessellation* is simply a tessellation composed of *regular polygons* - polygons for which all sides are the same length s. As it turns out, our search will be relatively simple since there are only three regular polygons which tessellate in the euclidean plane: the triangle, square, and hexagon [4].

(a) Triangles (b) Squares (c) Hexagons
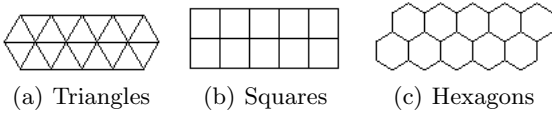
Figure 4: The Only Regular Tessellations

Of these (a) rules out utilizing triangles since every neighbor automatically is neighbors with at least one other. Of the two remaining options, the hexagon allows for a tighter packing. However, it is moderately more difficult to work with than the square, which as we see will prove more than adequate to provide a reasonable approximation of the phase-transition.

(a) square with diagonal r (b) A 5-by-5 grid with 25 nodes (c) A 5-by-5 grid with 37 nodes
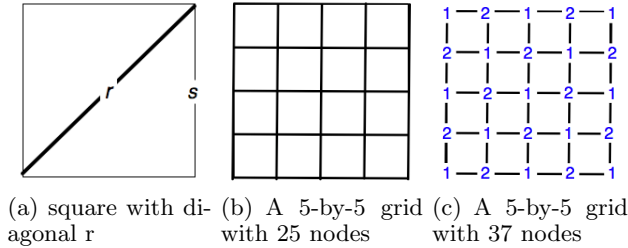
Figure 5: Geometry of Our Bounding Area Model

Examining Figure 5(a) we see that the maximum size for s in a square, the length of whose diagonal is $> r$ is $\frac{r}{\sqrt{2}}$ since $r^2 = 2s^2$. We consider the situation in which N is a perfect square (for other values of N we can substitute $\lceil \sqrt{N} \rceil$). We can then place one node on each of the grid points on a $\sqrt{N} X \sqrt{N}$ grid as seen in Figure 5(b). In general such a grid will take up an area of

$$A = \frac{r^2}{2} \left(\left\lceil \sqrt{N} \right\rceil - 1 \right)^2$$

and the ratio of the broadcast area of a single node to the

bounding area will be

$$\frac{\pi r^2}{A} = \frac{\pi r^2}{\frac{r^2}{2} \left(\left\lceil \sqrt{N} \right\rceil - 1 \right)^2} = \frac{2\pi}{\left(\left\lceil \sqrt{N} \right\rceil - 1 \right)^2}$$

Extending this model to cover larger k proves not overly difficult. For $k = 2$ instead of placing a single node at each grid intersection we place two nodes at every other intersection as seen in Figure 5(c). In this way, each node shares precisely one node with any of its neighbors, whether it is alone on its grid point or sharing it. Then for a given N we only need $\left\lceil \sqrt{\frac{2N}{3}} \right\rceil - 1$ grid lines on each side, requiring an area of $\frac{r^2}{2} \left(\left\lceil \sqrt{\frac{2N}{3}} \right\rceil - 1 \right)^2$. $k = 3$ is even easier requiring us to put two nodes at each grid intersection. We can extend this strategy to arbitrary $k \geq 1$ obtaining:

$$A = \frac{r^2}{2} \left(\left\lceil \sqrt{\frac{2N}{k+1}} \right\rceil - 1 \right)^2 \tag{1}$$

and

$$\frac{\pi r^2}{A} = \frac{\pi r^2}{\frac{r^2}{2} \left(\left\lceil \sqrt{\frac{2N}{k+1}} \right\rceil - 1 \right)^2} = \frac{2\pi}{\left(\left\lceil \sqrt{\frac{2N}{k+1}} \right\rceil - 1 \right)^2} \tag{2}$$

$\square$

Equation 1 provides a closed form for the minimum bounding area for continuous movement, while Equation 2 gives this as the ratio of an individual node's broadcast area to the total area. As we will see in Section 4.1.2, our model prediction tightly bounds the actual freezing behavior seen in simulation.

4.1.2 Ratio of Broadcast Area to Bounding Area in Simulation

Our simulation results were obtained through a binary search for the largest ratio of individual node broadcast area to bounding area that would result in a frozen configuration. Clearly we could not verify a particular ratio would never freeze since that would require infinite time. Instead we measured the average convergence times from our experiments in the unbounded space and allowed our system to run in excess of 10 times the maximum convergence times taken there. Because this experiment took several days of computational time on the system we had available, if a given trial had been running for some time without at any time experiencing a significant (percentage-wise) decrease in the proportion of system nodes frozen, that experiment was terminated early and assumed not to lead to a frozen configuration.

We explored scenarios with $N = 12, 25, 50, 100, 200, 400$ and $1 \leq k \leq 10$ over the course of up to 5000 time-steps. Each combination of N and k received 10 trials. To obtain a clearer correspondence with our model, in this trial alone we did not stochastically vary the connection lengths. At the end of each trial that did not result in a freezing configuration, the ratio was decreased by half its current value for

the subsequent trial. Conversely, whenever a trial ended in a freezing configuration the ratio would be increased by half. The first trial began with a ratio above the point where freezing could occur, but not too far. This was determined by a set of preliminary experiments.

The results of our exploration for $k = 1, 4$ are shown in Figure 6, as are the model predictions from Equation 2 (other k bound similarly but were omitted for graphical clarity). In this figure the y-axis measures the ratio of an individual node's broadcast area to the total bounding area while the x-axis measures the number of nodes N, each plot point representing the highest such ratio found at which a network of N froze.
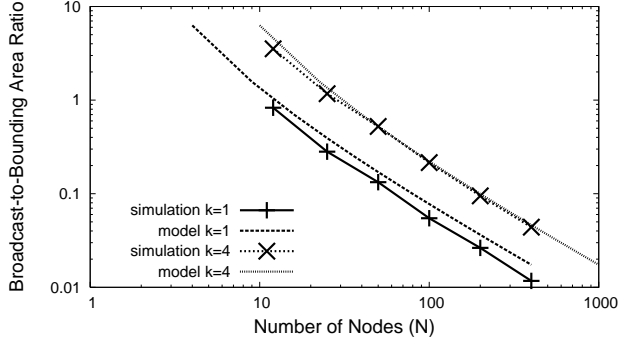
Figure 6: Our model for a lower bound on the phase-transition point along with corresponding data from our simulations

The behavior of the phase-transition point for all k can be characterized roughly as for every doubling in the number of nodes, the ratio between node broadcast area and bounding area decreases by slightly less than half. Moreover, increasing values of k lie at a relatively constant log-scale distance above one another. This is unsurprising as at higher levels of k a given number of nodes with a given broadcast area can fit in a smaller bounding space while still being able to reach a freezing state as argued in our model above.

4.2 Coverage of SCAN1 in Unbounded Spaces

Here, we explore properties of the freezing configuration when SCAN1 and ND are applied in an unbounded space, where the configuration is guaranteed to eventually freeze.

In order to measure the freedom of mobility allowed to nodes, we will use ratio of the area covered to the sum of the disjoint node broadcast areas. We call this the *coverage-ratio*. This is a useful measure because it not only characterizes how far nodes can spread, but it gives us a measure scaled to N, allowing a quick comparison of the latitude of movement allowed, without needing to rescale for the size of the network.

4.2.1 *SCAN1's Frozen Topology Coverage for $k = 1$*

We begin by computing an analytical upper bound on the coverage-ratio. For the sake of tractability of mathematical analysis, we will assume that the transmission range of all nodes have identical broadcast discs bounded by a fixed radius r, [2] and validate the results with simulation where we allow broadcast distance to vary stochastically.

We bound the maximum area a network running SCAN1 can cover for $k = 1$ (and consequently for all $k \geq 1$, although the bound becomes progressively less tight as k increases). We first note it is possible to bound this area trivially as $N\pi r^2$ where N is the number of nodes in the network. However, we can give a much tighter bound for $k = 1$ by examining the minimally overlapping line topology shown in Figure 7. To calculate this area we first must determine the
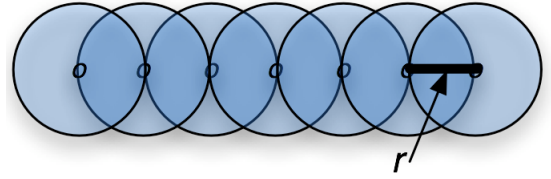

Figure 7: The connected topology covering maximal area for $k = 1$

overlap between two nodes at distance r from one another.

Consider some neighbor v of u. v will be positioned at some distance s from u. This distance determines the shaded area of overlap as seen below in Figures 8(a) and 8(b). We

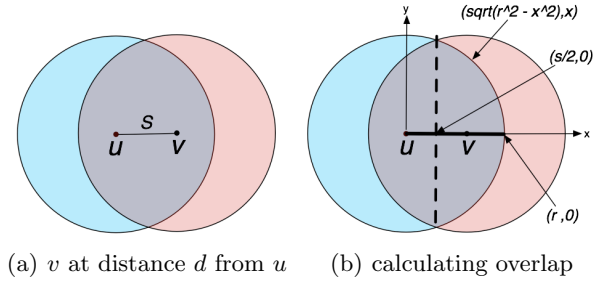(a) v at distance d from u (b) calculating overlap

Figure 8: Overlap Calculations

calculate this shaded area S by noting that in general it is four times the size of segment circumscribed by u's perimeter and the dashed line and the x-axis. Since both circles have identical radii r and are centered at u and v respectively, by symmetry the dashed line lies halfway between them. This implies that $s/2 \leq x \leq r$ and $0 \leq y \leq \sqrt{r^2 - x^2}$

---

[2]Although it has been recently shown that in some cases other models are required in order to capture issues such as collision and wireless interference [17],[22], the model still provides a reasonable abstraction. Extending the results to general SINR-based constraints is a subject for further research.

in the area of interest.

$$S = 4 \int_{s/2}^{r} \int_{0}^{\sqrt{r^2-x^2}} dy dx$$

$$= 4 \int_{s/2}^{r} \sqrt{r^2 - x^2} dx$$

$$= 4 \left[\frac{x}{2} \sqrt{r^2 - x^2} + \frac{r^2}{2} \sin^{-1} \left(\frac{x}{r} \right) \right]_{s/2}^{r}$$

Since in this case $s = r$ we have:

$$S = \frac{2\pi}{3} r^2 - \frac{\sqrt{3}}{2} r^2$$

Finally, we calculate the total area covered in a straight line configuration is

$$A = \pi r^2 + (N-1)(\pi r^2 - S) = \left[\pi + (N-1) \left(\frac{\pi}{3} + \frac{\sqrt{3}}{2} \right) \right] r^2 \tag{3}$$

$\square$

4.2.2 Properties of SCAN1's Frozen Topology, $k \geq 1$

To further the explore properties of SCAN1's frozen topology for values of $k \geq 1$, we turn to simulation. In this set of simulation experiments we also assess how successful SCAN1 and ND were in producing a frozen, connected network. Individual nodes were considered to be connected only, if they possessed a connected path back a base station positioned at the origin point of all nodes. Consequently the coverage area of the nodes itself was dependent not only on how far the nodes were able to spread from the origin, but also on how well they maintained the network connectivity.

We ran our experiments for $N = \{25, 50, 100, 200\}$. Movement was reassessed at the time interval in which a node could more $\frac{r}{10}$. SCAN1 was tested for $1 \leq k \leq 8$, while ND used $5 \leq k \leq 12$ (lower values of k for ND produced almost entirely disconnected networks). For succinctness we will refer to SCAN1 run with $k = i$ as SCAN1$_i$ and likewise for ND.

4.2.3 Coverage

Figure 9 plots the size of the coverage area (y-axis) as a function of k (x-axis) when using SCAN1 on a network whose connection lengths varied with a STD of 5%. The different curves depict differing numbers of nodes in the network, with each node's communication range averaging a unit distance. Not surprisingly, the coverage area increases in proportion with the size of N, and is a decreasing convex function with the size of k, where nodes are required to maintain larger collections of neighbor sets. Additionally, it is worth noting that for $k = 1$ the benefit of increased mobility in providing greater coverage is more than offset by the decrease in connectivity. While for $k > 2$, the frequency of any network partition does decrease but (Figures 13 and 14), this proves increasingly costly from a coverage standpoint.
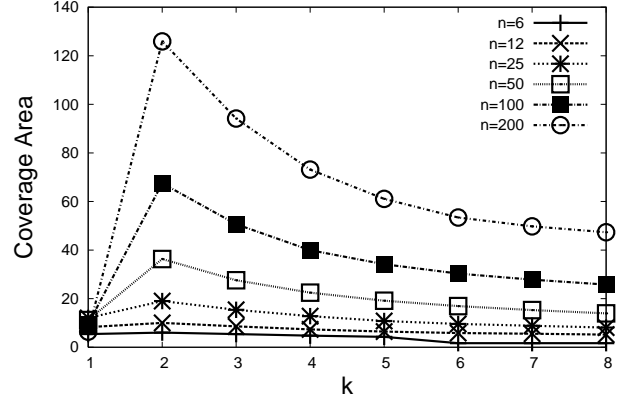

Figure 9: The Size of the Attained Coverage Area Under SCAN1 with $r = 1$ as a Function of k for STD=5%

Figure 10 provides the identical plot for an experiment in which the connections were varied with a standard deviation of 20% from the mean. In this case we can see that the same trends apply, however the optimal point for k has increased by 1, to $k = 3$. This is because with more significantly varying connection lengths, the probability of multiple broken connections in a single time-step increases. Consequently, the additional redundancy provided by a higher value of k proves more valuable than the increased mobility for a slightly higher value than when connection variations are more stable.
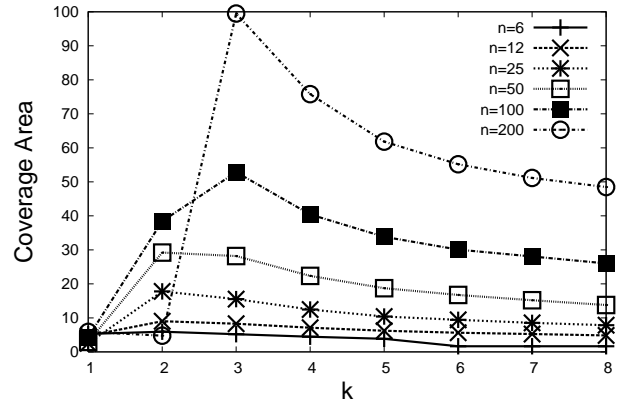

Figure 10: The Size of the Attained Coverage Area Under SCAN1 with $r = 1$ as a Function of k for STD=20%

Figures 11 and 12 provides comparable plots for ND for standard deviations of 5% and 20% respectively. The same trends discussed for SCAN1 are apparent, although the performance of optimally parameterized ND is inferior to opti-

mally parameterized SCAN1 in both cases. The advantage of SCAN1 over ND, does decrease as the broadcast channel becomes increasingly unstable. This is unsurprising as SCAN1's strength lies in its ability to conduct a more complex inference process on current local topology information than can ND. As this current information becomes increasingly less predictive of future performance, SCAN1's comparative advantage decreases.
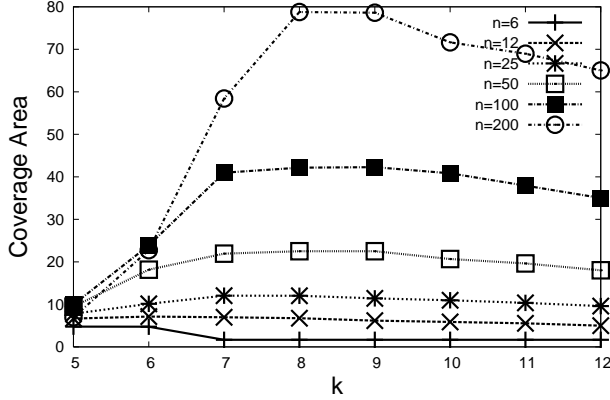

Figure 11: The Size of the Attained Coverage Area Under ND with $r = 1$ as a Function of k for STD=5%
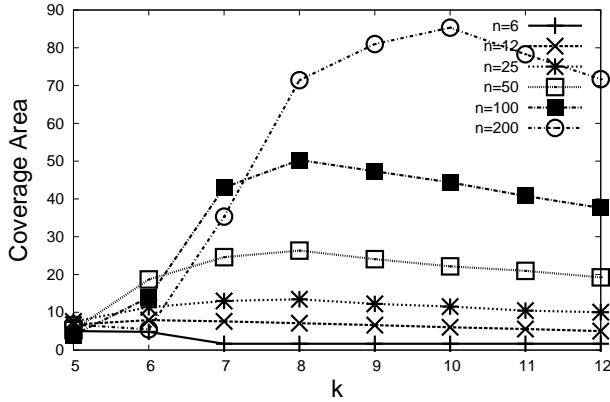
Figure 12: The Size of the Attained Coverage Area Under ND with $r = 1$ as a Function of k for STD=20%

4.2.4 Frequency of Network Partition

Figures 13 and 14 plot the fraction of time that the SCAN1 graph is disconnected (y-axis) as k is again varied on the x-axis and different curves plot differing values of N for STD of 5% and 20% respectively. Here, we see that the rate of disconnection is relatively unaffected by the size of N, but

drops dramatically as a function of k. Note that even for values of $k = 2$ and $k = 3$, a significant number of disconnections occur as connections become less stable. This is due to the variability in the size of the transmission radius: nodes will form neighbor relationships when the radius is large, and then suddenly lose them, even when frozen, when the radius is small. By increasing k, nodes have a sufficiently large set of neighbors to offset the stochastic disconnections.
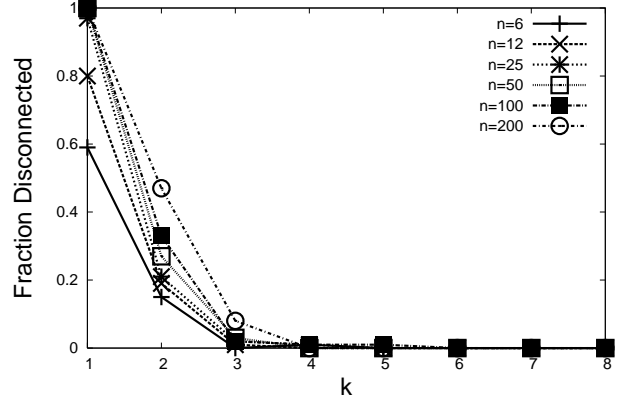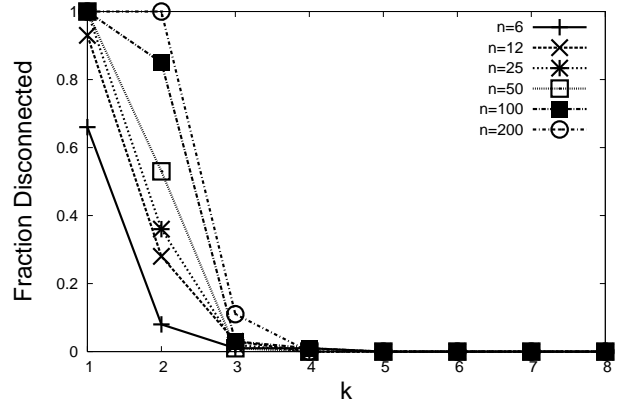

Figure 13: Fraction of Any Disconnections

Figure 14: Fraction of Any Disconnections

Figures 15 and 16 show the same plots and trends for ND. In striking comparison to SCAN1, ND's convergence towards a zero-partition frequency is both much more gradual, and incomplete. ND also has a non-zero partition rate, even for $k = 12$ and 5% connection variablilty.

4.2.5 Performance Comparison Between SCAN1 and ND

Simulation Figures 17 and 18 compare the performance of SCAN1 with that of ND. We fix the number of nodes N at 25 in Figure 17 and at 100 in Figure 18. The y-axis plots the fraction of disconnections, while the x-axis plots the *coverage ratio*, which is the ratio of the area actually covered to
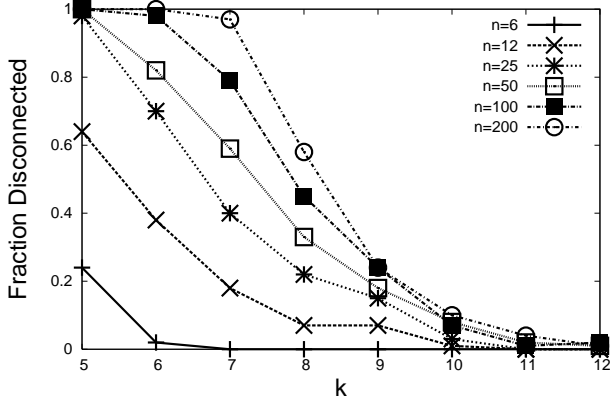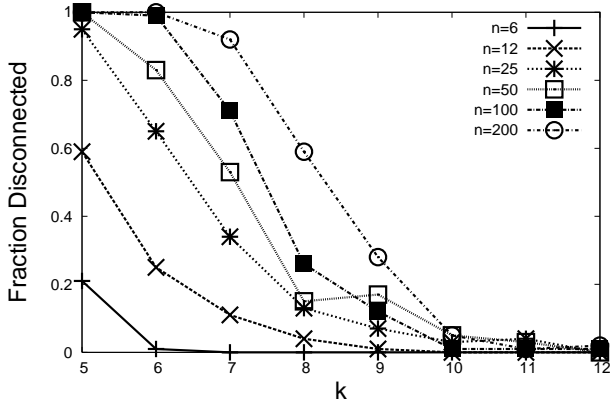
Figure 15: Fraction of Any Disconnections

Figure 16: Fraction of Any Disconnections

Figure 17: Coverage Ratio vs. Fraction of Disconnection: N=25
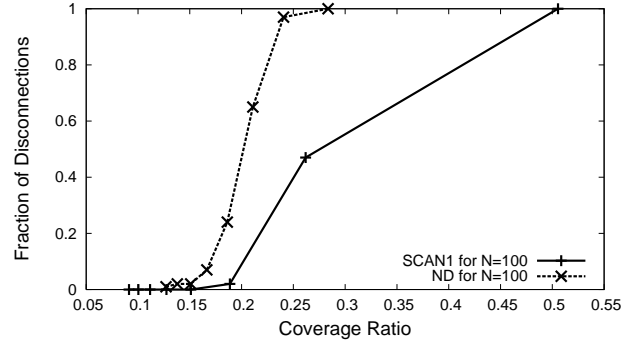
Figure 18: Coverage Ratio vs. Fraction of Disconnection: N=100

what the set of nodes could cover were there no connectivity requirement and there was no region of overlap. The coverage ratio is intended to normalize results with respect to the number of nodes, N, which, as we saw in Figures 13 and 14, for fixed k, the region of coverage grows linearly in the number of nodes.

The different points are obtained by varying k. Comparing ND and SCAN1 for a fixed k proves to not be a useful comparison, since ND generally offers better coverage ratio, but at a higher fraction of disconnections. However, these curves permit a direct comparison. The "better" algorithm is the one that yields a lower fraction of disconnections for a given coverage ratio, or, in graphical terms, the "better" algorithm's curve is to the right and below the other algorithm's curve. Here we see that SCAN1 is the better of the two algorithms for achieving good coverage with lower fraction of disconnections.

4.2.6 Target Detection

To assess the performance of SCAN1 for a target detection mission, we ran a simulation experiment in which nodes
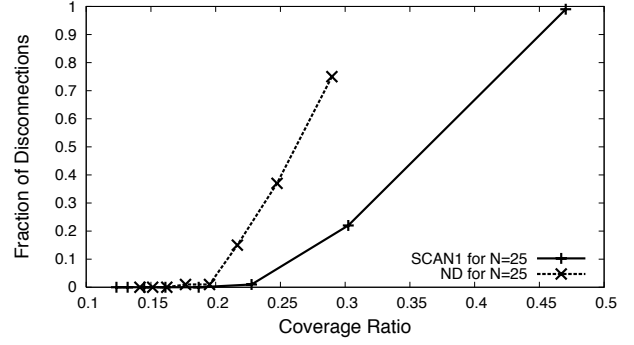
detected target when they were within $\frac{1}{10}^{th}$ of their mean broadcast radius. To gain an understanding of the interplay between connectivity and exploration targets were only registered as detected by the system, if the detecting node was connected to the base station through some path in the network at time of detection.

Targets were located uniformly throughout the space with a density of 2 per square mean broadcast radius. Performance was examined for the 5% STD broadcast case.

Figure 19 shows the performance of SCAN1 at the target detection task plotting the number of targets detected (y-axis) against k (x-axis). From this we might conclude that connectivity is of little value in the target detection task, even though the version we investigate requires some level of connectivity to detect targets! However, this would be a mistaken conclusion as comparison to the corresponding Figure 20 for ND shows. When no provisioning for connectivity is supplied ND_0 we see significantly poorer perfor-

mance than that of SCAN1$_1$ as can be seen in the direct comparison of optimally parameterized variants of SCAN1 and ND shown in Figure 21 (y-axis - percentage improvement of SCAN1 over ND, x-axis - network size). The conclusion to be gained is that intelligent maintenance of some minimal level of connectivity is substantially helpful for the target detection task (above and beyond any other potentially advantage provided by connectivity maintenance) although overmuch and/or unintelligent connectivity provisioning will decrease performance.
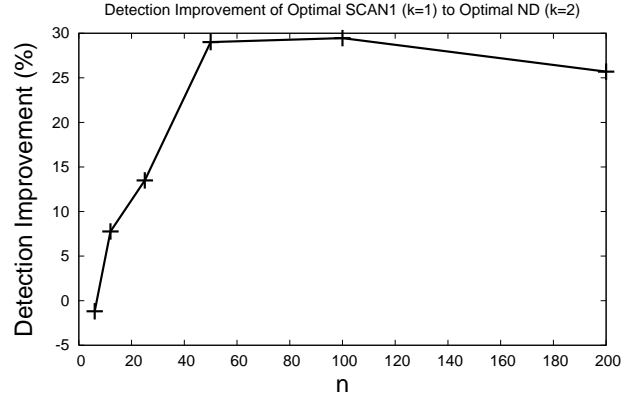

Figure 21: Percentage Improvement of SCAN1 over ND

networks freeze. There are two interesting trends. The first is a significant steepening of the curves as k increases. As k grows larger, not only does the network freeze more quickly, but the vast majority of the nodes halt their movement within a relatively short time window. The second interesting trend is that all distributions have relatively thin tails. There tends to be a long period before the network freezes during which only very few of the network's nodes are moving. This trend is most pronounced for small k.
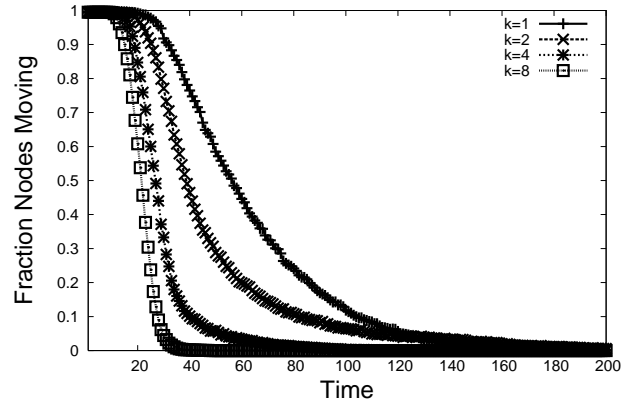

Figure 19: Number of Targets Detected Vs. k for SCAN1

Figure 22: Percentage of Nodes Moving Vs. Time for $N = 200$, $STD = 5\%$

When we consider the PDF (y-axis) of the fraction of nodes moving (x-axis) in Figure 23 we can see the above trends clearly. For $k \geq 2$ the PDF spikes for very large numbers of nodes and very small numbers of nodes. Most nodes stop during a relatively short period of time, but the last few nodes take a significantly longer time to halt. For $k = 1$ the PDF actually peaks significantly before this point - it is currently unclear to us why this happens.
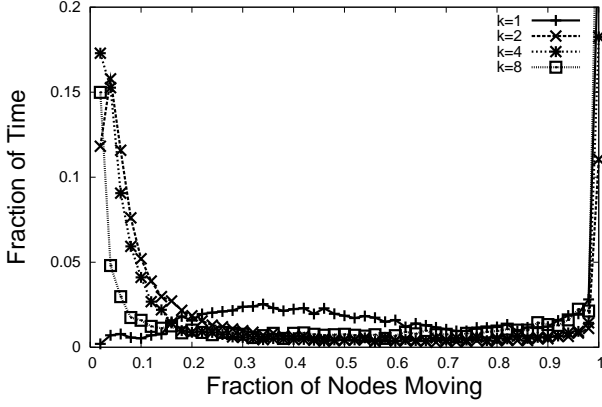

Figure 20: Number of Targets Detected Vs. k for ND

4.2.7 Movement of Nodes

We now examine the impact on SCAN1 varying values of k on nodal movement for $N = 200$ considering first the more stable broadcast scenario.

Figure 22 shows the fraction of nodes moving (y-axis) plotted against time (x-axis). As is expected, eventually all

Figure 23: PDF of Node Movement for $N = 200$, $STD = 5\%$

Figure 24: Percentage of Nodes Moving Vs. Time for $N = 200$, $STD = 20\%$

The curves for the 20% connection STD case differ significantly in the particulars of their movement trends as can be seen in the corresponding Figures 24 and 25. The slope of the decrease in number of nodes moving is notably less steep for the higher variance connection environment. While the PDF topology for $k = 1$ is significantly more pronounced and now has a second hump sharp hump in right around 10% of the nodes are moving. These changes in morphology are likely due to contribution of instability of connectivity. When the network begins to get close to a freezing configuration, the topological changes provoked by instability in the broadcast channel begin to dominate those arising through the evolution of the topology induced by nodal movement. Hence a relatively long series of slight nodal movements are required in a significant fraction of the node population before the network finds a state in which the network topology is consistent despite time-wise broadcast variations. This can actually be seen in Figure 24 in which there is an extended period of time during which around 10% of the nodes are moving for $k = 1$, explaining the inversion of the curves for $k = 1$ and $k = 2$ between Figures 22 and 24. A similar line of reasoning may also explain the less noticeable spikes in the PDF curves for $k = 2$.

4.3 Summary of Results

In this section, we identified the freezing phase transition point: the size of the space which when reached will result in the SCAN freezing, and below which it will not freeze entirely, finding that our analytical approximation is a good fit for our simulated results. We then focused our attention to unbounded regions, finding that SCAN1 outperforms ND, offering a lower disconnection rate when covering similar-sized regions. For SCAN1, coverage and disconnection rate drop quickly with k, whereas disconnection rate is relatively insensitive to N, while coverage area is roughly proportional to N.
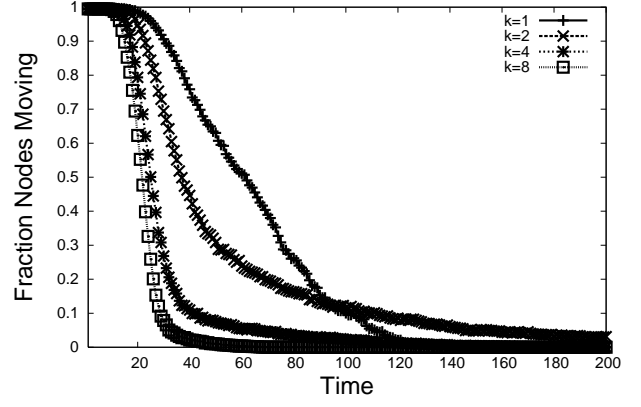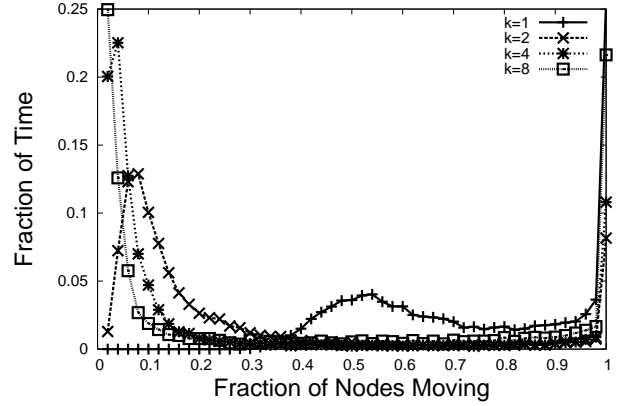

Figure 25: PDF of Node Movement for $N = 200$, $STD = 20\%$

5. TESTBED EXPERIMENTS

One of our work's goals was to produce an algorithm that could maintain the connectivity of a real-world system. We have implemented SCAN1 on our Roomba robotic testbed and run several hours of experiments to validate our ideas in a practical setting.

We found that SCAN1 could provide connectivity in a remarkably robust fashion. Out of 273 minutes and 50 seconds of experiments our network remained connected in all but 2 minutes and 23 seconds. Moreover the network partitions we encountered were comprised of one node separating from the network, with the sole exception of a 15 second period during which a pair of nodes partitioned themselves as a connected component. This result is shown graphically in 26 which compares the total time (y-axis) during which zero, one, or two nodes partitioned from the main networks (x-axis). Figure 27 shows the partition frequency (y-axis) bro-
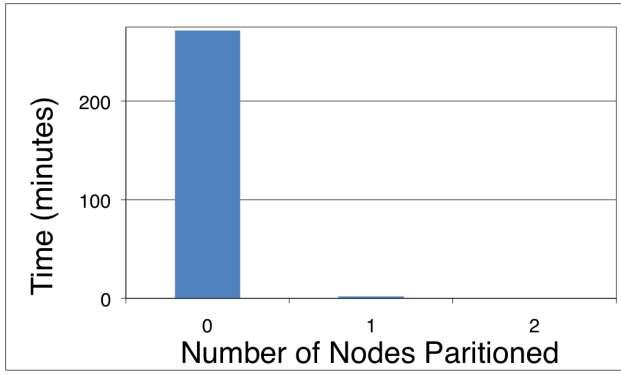
Figure 26: Run Time Vs. Size of Partition

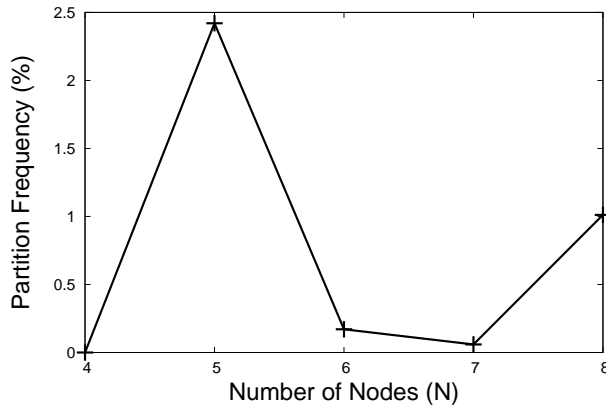ken down by the number of nodes in experiments (x-axis).

Figure 27: Percentage of Time Network Partitions Vs. Number of Nodes

5.1 Experimental Setup

Our testbed experiments were designed for two primary purposes:

1. To assess whether SCAN1 works to maintain network connectivity when implemented on actual hardware, and measure it's performance.

2. To assess whether in a real-world setting SCAN1 provides sufficient latitude of movement to be used by a system with particular application needs.

Regarding this latter goal: to assess the appropriateness of SCAN1 as a mechanism for ensuring the connectivity of a self-spreading mesh network and a robotic search and rescue system, we examined whether SCAN1 provided our nodes sufficient latitude of movement to achieve a spatial configuration associated needed by these applications. As mentioned in Section 2.1, our nodes were not easily capable of directed movement without an external localization mechanism. Clearly, if our randomly moving nodes could stumble

into a successful configuration, nodes with more robust mobility routines tailored for a particular application could do so as well.
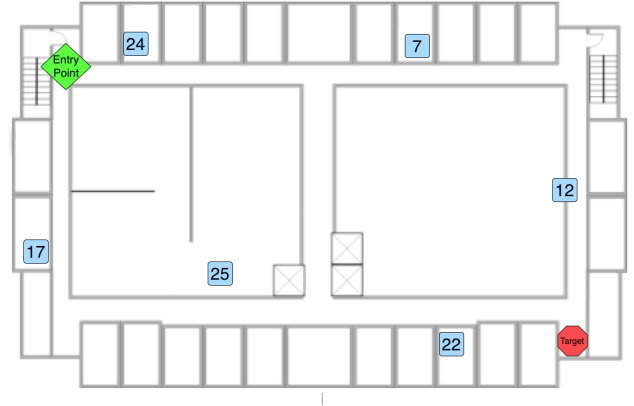

Figure 28: the indoor space used for experiments

Our experiments were run on one floor of a research building[3], covering approximately 20,000 square feet. A dozen or so wireless networks were competing for use on this particular floor providing a moderate level of interference. Each experiment measured two tasks:

- **Mesh-Coverage**: providing simultaneous wireless coverage to all nodes, corresponded to the self-spreading mesh-network. In this part of the experiment, we assessed whether the mobile nodes spread far enough from the start area to simultaneously provide wireless coverage to all of the client nodes shown in Figure 28, with client locations represented by the numbered boxes.

- **Locate-Target**: This corresponds to the search and rescue system. Our metric of success was whether at least one of the nodes would reach the target area before the network froze or the 20 minutes elapsed (this timeout was reached in only one of the 25 trials).

While the experimental space was moderately large and subject to both wireless interference from competing networks, as well as broadcast obstacles, our nodes could still broadcast a good proportion of its length. To conserve power and emulate a transmission-limited environment, we dialed down the broadcast power used by our nodes to the minimum level supported in software 0.25 dB and did not restrict the shielding of client nodes (e.g., if they were behind doors, in far corners, or on the ground). As can be seen in Figure 29 which plots the percentage of the time a given node was covered (y-axis) against the value of k used (x-axis), these efforts were partially successful. Certain clients were always covered, while others were often quite difficult to cover.

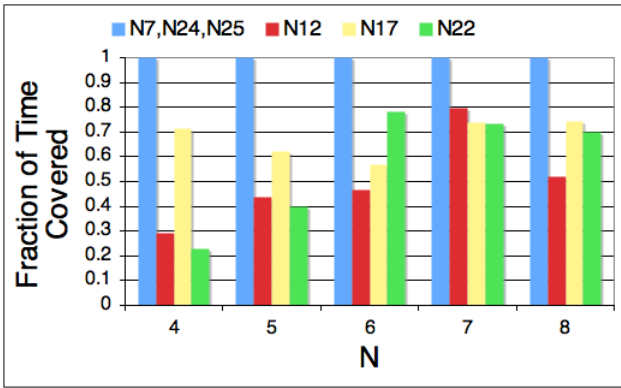[3]identity concealed for the purposes of double-blind review

Figure 29: Fraction of Time Clients Were Covered During Experiments

Figure 30: Number of Nodes Moving Vs. Time for $N = 6$

Our experiments tested networks of size $N = \{4, 5, 6, 7, 8\}$ using $k = 2$. We followed the logic laid out in Section 3.2 to choose this value of k: the Roombas moved slow enough such that two links simultaneously failing in a broadcast cycle was sufficiently unlikely. Each data point represents the average of 10 trials

5.2   Correlation with Simulation

The experimental design for the hardware experiments conducted on our testbed was significantly different from those used on our simulation experiments. This was both because of differences in goal (our simulation was designed to explore asymptotic proprieties and extend modeling, while our hardware experiments examined the function of SCAN1 and its suitability for use in a real-world setting) and natural constraints (our experimental space limited the number of nodes we could reasonably test and the density of monitoring nodes we could deploy, while a simulation environment can only provide a rough approximation of the vagaries of wireless broadcast such as multi-path fading, interference from competing systems, stochastic performance, especially in a physically complex environment).

Consequently, we might expect that SCAN1's behavior on our hardware testbed would show little in common with it's behavior in our simulator (and analytic models). However, when we compared how the number of nodes moving evolved with time as SCAN1 was run in identically parameterized versions of both our simulation and experiment $N = 6, k = 2$, we found a striking degree of correlation which can be seen in Figure 30 which compares the number of nodes moving (y-axis) for both the testbed and simulation against time (x-axis). [4] The degree to which movement patterns correlate between our simulation and testbed experiments indicate that the general trends of the results produced in each experimental domain bear significant applicability to

---

[4]We compare with the higher variability simulation link model with link length STD = 20% of mean length and appropriate time rescaling.
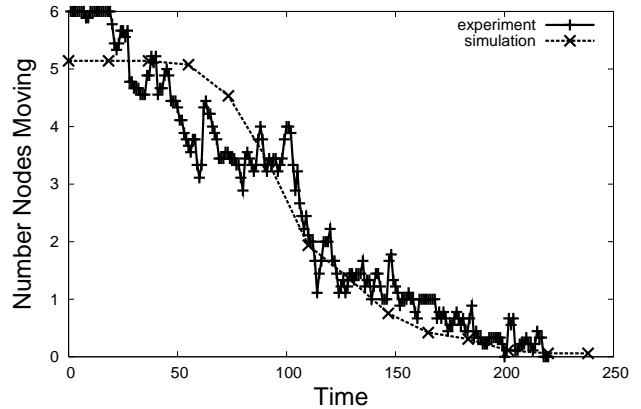
the other.

5.3   Experimental Results

In Figure 31 we can see the results for Mesh-Coverage. In this figure the left y-axis measures the percentage of trials in which coverage was achieved, the right y-axis the time in minutes, and the x-axis the number of nodes. Networks of 4 nodes were unable to ever fully cover all nodes at the same time, although the nodes were able to move far enough that every client was covered for at least some significant proportion of the experiment as seen in Figure 29. For $N = 5$ and greater, we see a significantly difference. In this space a network of 5 nodes seems sufficient to provide simultaneous coverage to all nodes, although it takes a relatively long 12 minutes on the average to do so. We see a continuing decrease in the time taken until all nodes are covered as N increases but the difference between $N = 5$ and $N = 6$ is by far the most striking. The small kink in success rate at $N = 7$ is most likely do to a high variance resultant from the statistically smaller number of trials run. Thus, SCAN1 coupled with the most rudimentary of mobility mechanisms was able to self-organize a small number of mobile nodes into a mesh network providing coverage to all the clients. We reiterate this was done in the presence of significant interference from competing networks, as well as dramatic variations in signal strengths due to a typical office setting with walls and corners.

The Locate-Target task proved much more difficult for our nodes. In part this was due to the difficulty an essentially randomly moving node would have in reaching a specific location in a circuitous environment with many small obstacles (e.g., waste baskets). But strikingly in only one out of 50 trials did the nodes run out of time before they either all froze or the target was reached by at least one node, and in that trial $N = 8$. The main constraint appeared to be that smaller networks simply lacked the number of nodes needed to maintain robust network connectivity as they continued
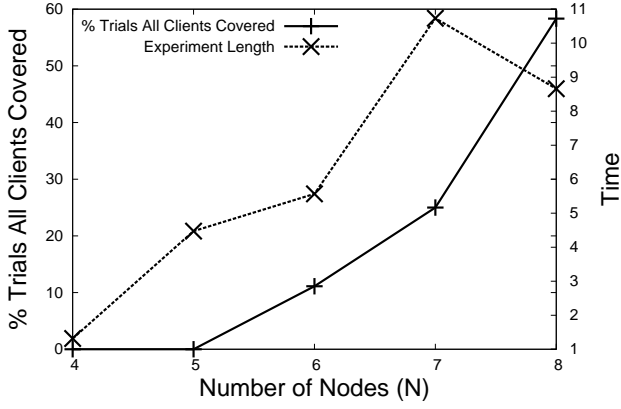
Figure 31: Percentage of Trials Coverage Achieved and Time Taken To Achieve Coverage

spreading out towards the target area (for $k = 4$ even one link breaking was enough to freeze the network).
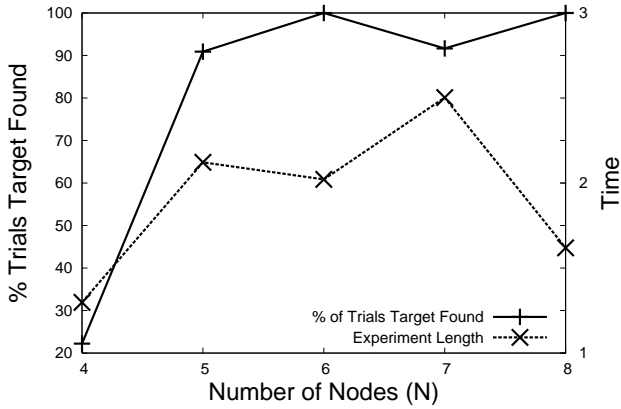

Figure 32: Percentage of Time Target Area Reached and Experiment Length

The experiment length displayed in Figure 32 is worth explaining. In this figure, the left y-axis measures the percentage of trials in which the target was reached, the right y-axis the time in minutes, and the x-axis the number of nodes. In contrast to the Mesh-Coverage experiments which finished more quickly as the N increases (as can be seen Figure 31), here we can see the opposite trend. This difference is due to the different stopping criteria for these experiments. In Mesh-Coverage, greater N meant an earlier time for total coverage, while in Locate-Target, the total experiment length was determined by the freezing time of the network or (mostly applicable to $N = 8$) the time needed to reach the target. Having more nodes means network will take longer to freeze, hence the increasing curve for the stopping time of the Locate-Target experiment for the values of N we tested (greater N would eventually reverse this trend since trials

would end in successful locations of the target, not frozen networks). In fact, if only freezing times were taken into account $N = 8$ would plot significantly higher, at around 18 minutes. We suspect $N = 8$ is close to the phase transition point at which a network running SCAN1 is contained in a space small enough, continual movement is guaranteed (Section 4.1).

6. RELATED WORK

The presence, absence, and quality of physical layer connectivity has played a fundamental and evolving role in wireless networks research. Early on, research into sensor networks examined how wireless sensors should be positioned and scheduled to maximize some metric (e.g. lifetime) under the constraint that these produce a connected network (e.g., [3]). MANET research subsequently addressed how spontaneously arising physical layer connectivity across some set of nodes could be utilized so as to produce a functioning network providing for routing, admission control, etc.

The obvious problem MANETs and related ad hoc wireless systems ran into was that physical layer connectivity could disintegrate as easily as it had arisen. This prompted work to assess how likely and under what conditions physical layer connectivity might be maintained. [18] studied the asymptotic properties of connectivity under various mobility models. [7] provided a method for assessing network connectivity through use of a graphical model, proposing a complementary framework to ours and [32] showed how predicting future network topology from current mobility patterns can be used to improve the performance of routing algorithms.

Alternately, Delay Tolerant Networks (DTN) were proposed in [8] as a network class that would be able to function *despite* loss of physical layer connectivity. [28] presented a DTN in which Data MULEs move around and collect information from stationary sensors. [1] investigated the connectivity requirement for DTN infrastructure composed of a set of Throwbox nodes used as drop points at which information can be stored and forwarded.

Concurrently researchers began to explore the use of mobility in various wireless networking scenarios as a potential asset as opposed to a limitation. In their seminal paper [11], Grossglauser and Tse showed that mobility can potentially increase the capacity of a wireless network while [13] formalized the concept of a MORPH network which uses mobility to help realize some desirable network property/goal. [14] proposed to extend the lifetime of an already connected network comprising predominantly immobile nodes by using mobile nodes to redistribute the routing burden.

To the best of our knowledge previous work on networked systems allowing active mobility have dealt with the problem of ensuring needed physical layer connectivity in one of two ways. Either they have framed the problem as one in which connectivity is relatively easy to guarantee, or they have required mobiles to travel along completely and homo-

geneously determined trajectories, or often both.

Examples of the former include [12] and [20] which present algorithms for distributed self-deployment of mobile sensors under the assumption that transmission areas are identical and spherical for all nodes, with nodes possessing strong localization capabilities (e.g., GPS coordinates, absolute relative position) - [16] and [19] provide additional detail on how such information may be obtained. [12] attempts to maximize the network lifetime by adjusting the network's topology while [20] looks to maximize the coverage area under the constraint that every node has at least some threshold K number of neighbors. [23] examines the joint mobility and routing problem for such a network when all energy costs are known.

Various hierarchical wireless networking approaches in which some nodes are more capable than others are presented in [27] and studied analytically in [31]. In such networks, the more capable nodes can serve as Mobile Backbone Nodes, providing the infrastructure over which end-to-end communication can take place. Using graph theoretic tools, [10] provides a similar approach in which cluster heads follow mobile nodes.

In the search and rescue domain, [26] examines systematically searching a bounded space while maintaining connectivity as an optimization problem. [25] runs against this grain by assuming very little in terms of nodal localization capabilities (nodes can tell when they are getting closer to or further from a given neighbor and have no other localization knowledge) considering a hybrid system of immobile wireless sensors and mobile wireless robots. Here connectivity is provided by a reliance on dense deployment of the cheap immobile sensors, which then support the robots' search.

A variety of work from the field of of Control Theory addresses the assurance of physical layer connectivity as a somewhat more primary problem, utilizing motion planning algorithms to enforce physically layer connectivity ([15] provides a review). In such approaches, the connectivity algorithm typically provides for connectivity while maximizing some target function by completely determining the movement of each node. Such approaches typically assume a deterministic nodal broadcast radius. In [29], Spanos and Murray uses geometric methods to provide for local connectivity and maximal coverage, showing that under certain mild conditions this will lead to global connectivity (while still maximizing coverage). They then extend this concept to motion planning for continuously connected group movement from one area to another [30]. In a series of papers, Zavlanos and Pappas consider the use of potential fields to supply centralized [36], distributed [37], and centralized double integrator [38] schemes for ensuring connectivity while maximizing some metric of interest. The assumption of deterministic nodal broadcast radii is relaxed by[6] and [9] which consider a fairly realistic broadcast model, feeding their main challenge: maximizing SNR performance. However to achieve this [6] considers only straight line topologies while [9] considers a DTN scenario in which connectivity maintenance isn't required.

To the best of our knowledge, our work is the first to provide a connectivity maintenance mechanism that can perform under realistic broadcast assumptions (and without sophisticated localization capabilities), while providing for complex network topologies, and additionally allowing constituent nodes significant latitude in determining their own movement. Notably, our work is the only work addressing connectivity maintenance to have been successfully implemented and tested on hardware ([5],[34],[24] provide examples of robotic mobile network testbeds).

Finally we note that our current work does not address situations in which nodes are malicious, malfunctioning, or otherwise deviate from conforming with the cooperative constraints, although exploring the creation of a SCAN comprised of such nodes would be a fascinating direction for future work. Likely, several of the security techniques used for protecting the multihop network connectivity/routing in MANETs discussed in [35], [2] will be applicable to such problems in the SCAN domain as well.

7. FUTURE WORK

As we note in Section 1, the mechanisms utilized by SCAN1 could potentially be enhanced significantly through the incorporation of additional information. For example, if we can obtain a high-quality reading of per-peer RSSI then potentially nodes could be allowed to move until both the current SCAN1 criterion is reached and at least one of the current links is beginning to weaken. Potential sources of information we plan to consider include: GPS coordinates, neighbor distances, relative direction (e.g., compass heading), and RSSI. Along these same lines, it could be potentially interesting to explore what might be done to extend our techniques in heterogeneously equipped networks (e.g., one of every 20 nodes has a GPS).

SCAN1 exploits knowledge gathered from a 2-hop radius. It may prove interesting to investigate whether a sufficiently low-overhead mechanism utilizing information in a 3-hop radius exists and if it might provide significant improvement upon SCAN1, or an interesting trade-off in accuracy versus overhead.

In our current work, we focus on systems composed only of mobile nodes. Our techniques can be trivially extended to a network comprising a mixture of mobile and immobile nodes. However, it is likely that there are non-trivial optimizations of our techniques that could be used to address such mobility-heterogeneous networks.

Finally, inspired by the work done on easily emplacable immobile wireless infrastructure nodes such as Throwboxes [1], we are examining how our techniques may be extended towards mobility-heterogeneous networks in which mobile nodes distribute immobile ones.

8. ACKNOWLEDGEMENTS

9. REFERENCES

[1] N. Banerjee, M. D. Corner, and B. N. Levine. An energy-efficient architecture for DTN throwboxes. In *Proc. IEEE INFOCOM'07*, May 2007.

[2] S. Bhargava and D. Agrawal. Security enhancements in aodv protocol for wireless ad hoc networks. In *Proc. IEEE VTC'01 Fall*, 2001.

[3] M. Cardei and J. Wu. Energy-efficient coverage problems in wireless ad-hoc sensor networks. *Computer Communications*, 29(4):413–420, 2006.

[4] D. Chavey. Tilings by regular polygons—ii: A catalog of tilings. *Computers and Mathematics with Applications*, (17):147–165, 1989.

[5] P. De, A. Raniwala, R. Krishnan, K. Tatavarthi, J. Modi, N. A. Syed, S. Sharma, and T. C. Chiueh. MiNT-m: an autonomous mobile wireless experimentation platform. In *Proc. ACM MobiSys'06*, 2006.

[6] C. Dixon and E. Frew. Controlling the mobility of network nodes using decentralized extremum seeking. *Decision and Control, 2006 45th IEEE Conference on*, pages 1291–1296, 13-15 Dec. 2006.

[7] L. J. Dowell and M. L. Bruno. Connectivity of random graphs and mobile networks: validation of monte carlo simulation results. In *Proc. ACM symposium on Applied computing (SAC'01)*, Mar. 2001.

[8] K. Fall. A delay-tolerant network architecture for challenged internets. In *Applications, Technologies, Architectures, and Protocols for Computer Communication*. ACM / SIGCOMM, 2003.

[9] E. Frew, T. Brown, C. Dixon, and D. Henkel. Establishment and maintenance of a delay tolerant network through decentralized mobility control. *Networking, Sensing and Control, 2006. ICNSC '06. Proceedings of the 2006 IEEE International Conference on*, pages 584–589, April 2006.

[10] J. Gao, L. J. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Discrete mobile centers. *Discrete and Computational Geometry*, 30(1):45–63, May 2003.

[11] M. Grossglauser and D. Tse. Mobility increases the capacity of ad hoc wireless networks. In *IEEE/ACM Transactions on Networking*, volume 10, pages 477–486, 2002.

[12] N. Heo and P. K. Varshney. A distributed self spreading algorithm for mobile wireless sensor networks. In *Proc. IEEE WCNC'03*, Mar. 2003.

[13] A. Kansal, M. Rahimi, D. Estrin, W. Kaiser, G. Pottie, and M. Srivastava. Controlled mobility for sustainable wireless sensor networks. In *Proc. IEEE SECON'04*, Oct. 2004.

[14] J. Luo and J.-P. Hubaux. Joint mobility and routing for lifetime elongation in wireless sensor networks. In *Proc. IEEE INFOCOM'05*, Mar. 2005.

[15] S. Martinez, J. Cortes, and F. Bullo. Motion coordination with distributed information. *IEEE Control Systems*, 27(4):75–88, 2007.

[16] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust distributed network localization with noisy range measurements. In *Proc. ACM SenSys'04*, 2004.

[17] T. Moscibroda, R. Wattenhofer, and A. Zollinger. Topology Control Meets SINR: The Scheduling Complexity of Arbitrary Topologies. In *Proc. ACM MOBIHOC'06*, May 2006.

[18] P. Nain, D. Towsley, B. Liu, and Z. Liu. Properties of random direction models. In *Proc. IEEE INFOCOM'05*, Mar. 2005.

[19] C. Peng, G. Shen, Z. Han, Y. Zhang, Y. Li, and K. Tan. A beepbeep ranging system on mobile phones. In *Proc. ACM SenSys'07*, New York, NY, USA, Nov. 2007.

[20] S. Poduri and G. Sukhatme. Constrained coverage for mobile sensor networks. *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, 1:165–171 Vol.1, 26 April-1 May 2004.

[21] N. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket Location-Support System. In *6th ACM MOBICOM*, Boston, MA, August 2000.

[22] L. Qiu, Y. Zhang, F. Wang, M. K. Han, and R. Mahajan. A general model of wireless interference. In *Proc. ACM MOBICOM'07*, Sept. 2007.

[23] R. Rao and G. Kesidis. Purposeful mobility for relaying and surveillance in mobile ad hoc sensor networks. *IEEE Trans. on Mobile Computing*, 3(3):255–231, July-Aug. 2004.

[24] J. Reich, V. Misra, and D. Rubenstein. Roomba madnet: a mobile ad-hoc delay tolerant network testbed. *To appear in ACM Sigmobile, MC2R: Mobile Computing and Communications Review*, 2008.

[25] J. Reich and B. Sklar. Robot-sensor networks for search and rescue. In *In IEEE International Workshop on Safety, Security and Rescue Robotics*, Gaithersburg, MD, August 2006.

[26] M. Rooker and A. Birk. Combining exploration and ad-hoc networking in robocup rescue. In D. Nardi, M. Riedmiller, and C. Sammut, editors, *RoboCup 2004: Robot Soccer World Cup VIII*, volume 3276 of *Lecture Notes in Artificial Intelligence (LNAI)*, pages 236–246. Springer, 2005.

[27] I. Rubin, A. Behzad, R. Zhang, H. Luo, and E. Caballero. Tbone: A mobile-backbone protocol for ad hoc wireless networks. In *Proc. IEEE Aerospace*

Conference, Mar. 2002.

[28] R. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: modeling a three-tier architecture for sparse sensor networks. In *Proc. ISPN'03*, may 2003.

[29] D. Spanos and R. Murray. Robust connectivity of networked vehicles. *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, 3:2893–2898 Vol.3, 14-17 Dec. 2004.

[30] D. Spanos and R. Murray. Motion planning with wireless network constraints. *American Control Conference, 2005. Proceedings of the 2005*, pages 87–92, June 8-10, 2005.

[31] A. Srinivas, G. Zussman, and E. Modiano. Mobile backbone networks - construction and maintenance. In *Proc. ACM MOBIHOC'06*, May 2006.

[32] W. Su, S.-J. Lee, and M. Gerla. Mobility prediction and routing in ad hoc wireless networks. *International Journal of Network Management*, 11(1):3–30, Jan.-Feb. 2001.

[33] S. Tisue and U. Wilensky. Netlogo: A simple environment for modeling complexity. In *Proc. International Conference on Complex Systems*, 2004.

[34] I. Tsigkogiannis, R. Balani, J. Carwana, J. Friedman, D. Lee, C.-C. Han, R. Shea, R. K. Rengaswamy, M. Petralia, L. Corman, E. Wittenmeier, E. Kohler, and M. Srivastava. Dynamically configurable robotic sensor networks. In *Proc. ACM SenSys'05*, 2005.

[35] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang. Security in mobile ad hoc networks: challenges and solutions. *IEEE Wireless Communications*, 11(1):38–47, Feb. 2004.

[36] M. Zavlanos and G. Pappas. Controlling connectivity of dynamic graphs. *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, pages 6388–6393, 12-15 Dec. 2005.

[37] M. Zavlanos and G. Pappas. Potential fields for maintaining connectivity of mobile networks. *Robotics, IEEE Transactions on [see also Robotics and Automation, IEEE Transactions on]*, 23(4):812–816, Aug. 2007.

[38] M. M. Zavlanos and G. J. Pappas. Distributed connectivity control of mobile networks. *Decision and Control, 2007 46th IEEE Conference on*, pages 3591–3596, 12-14 Dec. 2007.