Distributed Mobile Disk Cover – A Building Block for Mobile Backbone Networks

Anand Srinivas, Gil Zussman, and Eytan Modiano
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, MA 02139
{anand3, gilz, modiano}@mit.edu

***Abstract*— The novel hierarchical architecture of Mobile Backbone Networks has been recently studied by a few different approaches. An important subproblem related to the design and operation of such networks is the problem of constructing and maintaining a Geometric Disk Cover (GDC) under mobility. While from the context of static nodes and centralized solutions the GDC problem has been extensively studied, the Mobile GDC problem did not receive much attention. We present two new algorithmic approaches for the solution of this problem. These approaches significantly differ from previously presented approaches. In order to analyze the worst case performance of the algorithms, we develop a novel graph-based analysis technique. Then, we develop a methodology to compute bounds on the average case performance of recently presented strip-based algorithms. We use these bounds along with simulation results to evaluate the performance of the algorithms. It is shown that the proposed algorithms perform very well under mobility, thereby providing an important building block for the construction and maintenance of Mobile Backbone Networks.**

I. Introduction

Wireless Sensor Networks (WSNs) and Mobile Ad Hoc Networks (MANETs) can operate without any physical infrastructure (e.g. base stations). Yet, it has been shown that it is sometimes desirable to construct a *backbone* over which reliable end-to-end communication can take place [3]. In particular, if some of the nodes are more capable than others, these nodes can be dedicated to providing the backbone. A novel hierarchical approach for a *Mobile Backbone Network* operating in such a way was recently proposed and studied by Rubin et al. and by Gerla et al. (see [15],[18] and references therein).

Based on [15], and [18], a Mobile Backbone Network was defined in [16] as composed of two types of nodes. The first type includes static or mobile nodes (e.g. sensors or MANET nodes) with limited capabilities. These nodes are refered to as *Regular Nodes* (RNs). The second type includes mobile nodes with superior communication, mobility, and computation capabilities as well as greater energy resources (e.g. Unmanned-Aerial-Vehicles). These nodes are termed *Mobile Backbone Nodes* (MBNs). The main purpose of the MBNs is to provide a mobile infrastructure facilitating network-wide communication.

Fig. 1 illustrates an example of the architecture of a Mobile Backbone Network. The set of MBNs has to be placed and *mobilized* such that (i) every RN can directly communicate with at least one MBN, and (ii) the network formed by the MBNs is connected. We assume a *disk* connectivity model, whereby two nodes can communicate if they are within a certain range. We also assume that the communication range of the MBNs is significantly larger than the communication range of the RNs. The problem of placing the minimum number of MBNs has been termed in [16] as the Connected Disk Cover (CDC) problem. A similar problem has been recently also formulated in [17].

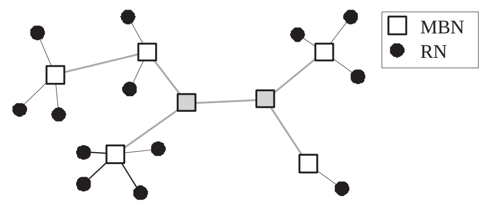

Fig. 1. A Mobile Backbone Network in which every Regular Node (RN) can directly communicate with at least one Mobile Backbone Node (MBN). All communication is routed through a connected network formed by the MBNs.

The algorithms in [16] focus on *controlling the mobility* of the MBNs in order to provide a backbone for reliable communication. These algorithms are based on the fact that the CDC problem can be decomposed into the Geometric Disk Cover (GDC) problem and the and the Steiner Tree Problem with Minimum Number of Steiner Points (STP-MSP). It was shown in [16] that if the GDC and STP-MSP subproblems are solved separately by γ and δ-approximation algorithms, the approximation ratio of the joint solution is γ+δ.

Motivated by this decomposition result, in this paper, we focus on the GDC subproblem. This problem can be stated as: given a set of points in the plane, place the minimum number of disks such that all points are covered. Due to the our focus on decentralized operation in a mobile environment, we aim to develop *distributed* algorithms that maintain a disk cover under *mobility*. It follows from the decomposition result that any improvement in the approximation ratio of the GDC problem (γ) immediately improves the approximation ratio of the overall CDC solution. Hence, the developed algorithms are an important building block for any decomposition-based Mobile Backbone Network algorithm.

The Mobile GDC problem also stands alone as an important problem and has several applications in MANETs [5],[11], and in WSNs. For example, a possible application is in the area of point coverage in sensor networks (e.g. [13]), where sensors have to track or follow a set of moving targets. Hershberger [8] points out applications in databases, where clustering can support queries regarding time-varying data. Finally, in the context of Mobile Backbone Networks, assuming that MBNs can communicate with each other over long distances ensures that the MBNs' network is always connected and reduces the CDC problem to GDC problem.

The *static* GDC problem has been extensively studied in the past. Hochbaum and Maass [10] provided a Polynomial Time Approximation Scheme (PTAS) for the problem. However, their algorithm is impractical for our purposes, since it is centralized and has a high running time for reasonable approximation ratios. Several other *centralized* algorithms have been proposed. For example, Gonzalez [7] presented an algorithm based on dividing the plane into strips, whose approximation ratio has been recently shown to be 6 [16]. Franceschetti et al. [4] developed an algorithm that places disks only on vertices of a mesh. A table comparing the various centralized GDC algorithms can be found in [4].

As mentioned above, the properties of wireless networks call for *distributed* disk cover algorithms that deal with RNs *mobility*. However, only a few recent works have focused on algorithms that maintain coverage under mobility (i.e. solve the Mobile GDC problem) and even fewer proposed distributed algorithms. We note that *clustering* given nodes to form a hierarchical architecture has been extensively studied in the context of wireless networks (e.g. [1],[2],[6]). However, the idea of deliberately controlling the motion of *specific* nodes in order to maintain some desirable network property has been introduced only recently (e.g. [12],[16]).

In the specific context of the Mobile GDC problem, [11] present a 7-approximation distributed algorithm. Hershberger [8] presents a *centralized* 9-approximation algorithm for a slightly different problem: the mobile geometric *square* cover problem. Gao et al. [5] study a closely related problem in which the centers have to be selected from the set of points (i.e. RNs). Finally, [16] presented a number of distributed approximation algorithms for the Mobile GDC problem.

Similarly to [16], we assume that all nodes can detect their position via GPS or a localization mechanism. This assumption allows to take advantage of location information in designing distributed algorithms. However, in [16] the Mobile GDC problem is solved by dividing the plane into strips, solving the GDC problem locally within strips, and finally combining these solutions to form an overall solution. One of the advantages of this type of a *strip-based* algorithm is that the optimization is easier within a narrow strip, as opposed the whole plane. Another advantage is that the computation is localized to within strips, yielding a sort of spatial decentralization of both computation and communication.

However, a drawback of this approach is the fact that cross-strip optimization cannot be exploited. A typical example of the resulting inefficiency is depicted in Fig. 2, in which a strip-based algorithm uses two MBNs to cover two RNs that could obviously be covered by a single MBN. In this paper we present and analyze a number of new *planar-based* distributed algorithms that do not use strips. Yet, we show they are still able to distributedly solve the Mobile GDC problem while providing good performance guarantees.

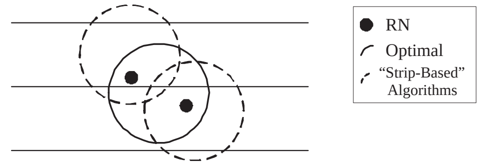

Fig. 2. Example of basic inefficiency of strip-based algorithms.

We start by presenting a novel family of algorithms that periodically merge neighboring MBNs (if possible) and spatially separate groups of neighboring MBNs (if required). Analyzing the worst case performance of these algorithms requires developing a novel graph-based technique. We use this techniques to obtain the approximation ratios of the algorithms. We later show via simulation that on average the algorithms perform better than the strip-based algorithms.

We then present a very simple 5-approximation algorithm that is based on an *overlooked* observation regarding the relation between the GDC problem and the maximal independent set problem. We show that placing the MBNs (i.e. the disk centers) on top of some of the RNs (points) yields a restricted GDC problem, which is equivalent to a minimum dominating set problem in a unit disk graph. We show that we can find an approximate solution to the unrestricted problem by finding a maximal independent set in the unit disk graph. This simple observation is important, since it immediately provides a 5-approximation distributed algorithm for the static and mobile GDC problems, whereas in the past much effort has been dedicated to developing centralized algorithms with higher complexities and approximation ratios (see the table in [4]).

In order to evaluate the performance of the algorithms, we revisit two of the algorithms presented in [16]. We present a new methodology for *average case* performance analysis of these algorithms. Based on this methodology, we show that their average approximation ratios are bounded by 3, whereas their worst case approximation ratios were shown to be 4.5 and 6. Then, we evaluate the performance of the algorithms via simulation. We start by studying the performance under mobility and by comparing the performance of the planar algorithms, presented in this paper, to a number of previously presented Mobile GDC algorithms. Then we compare average case and simulation results of the different algorithms.

To summarize, our main contribution is the development and analysis of distributed algorithms for the Geometric Disk Cover problem in a mobile environment. These algorithms may operate on a stand-alone basis or provide an important building block for the Mobile Backbone Network algorithms.

This paper is organized as follows. In Section II we formulate the problem. The new distributed planar algorithms are presented and analyzed in Sections III and IV. The average case analysis of the algorithms described in [16] is presented in section V. In Section VI we evaluate and the performance of the algorithms via simulation. We summarize the results and discuss future research directions in Section VII.

II. Problem Formulation

We consider a set of *Regular Nodes* (RNs) distributed in the plane and assume that a set of *Mobile Backbone Nodes* (MBNs) has to be deployed to cover them. We denote by N the collection of *Regular Nodes* $\{1, 2, \ldots, n\}$ and by $M = \{d_1, d_2, \ldots, d_m\}$ the collection of MBNs. The locations of the RNs are denoted by the $x - y$ tuples (i_x, i_y) $\forall i$ and d_{ij} denotes the distance between nodes i and j.

We assume that the RNs and MBNs have both a communication channel (e.g. for data) and a low-rate control channel. For the communication channel, we assume a disk connectivity model. Namely, an RN i can communicate bi-directionally with another node j (e.g. an MBN) if the distance between i and j, $d_{ij} \leq r$. We denote by $D = 2r$ the diameter of the disk covered by an MBN communicating with RNs. For the control channel, we assume that both RNs and MBNs can communicate over a much longer range than their respective data channels. Since given a fixed transmission power, the communication range is inversely related to data rate, this is a valid assumption.

For this work, we assume that the number of available MBNs is not bounded (e.g. if necessary, additional MBNs can be dispatched). Yet in our analysis, we will try to minimize the number of MBNs that are actually deployed. We formulate the Geometric Disk Cover (GDC) problem [10], as follows:

Problem GDC: Given a set of RNs (N) distributed in the plane, place the smallest set of MBNs (M) such that for every RN $i \in N$, there exists at least one MBN $j \in M$ such that $d_{ij} \leq r$.

The Mobile GDC problem is implicit in the above formulation, as the goal is to maintain a valid GDC under RN mobility. We assume there exists some sort of MBN routing algorithm, which routes specific MBNs to their new locations. The actual development of such an algorithm is beyond the scope of this paper.

Before proceeding, we introduce additional notation required for the presentation and analysis of the algorithms. Note that in the formulation of the Mobile GDC problem it is required that every RN is connected to at least one MBN. We assume that even if an RN can connect to multiple MBNs, it is actually assigned to exactly one MBN. Thus, we denote by P_{d_i} the set of RNs connected to MBN d_i. We denote by d_i^L, d_i^R, d_i^B and d_i^T the leftmost, rightmost, bottommost, and topmost RNs connected to MBN d_i. Their (x, y) co-ordinates are denoted with $x - y$ subscripts, e.g. $(d_i^L)_x, (d_i^L)_y$.

III. Planar Merge-And-Separate Algorithms

In this section we present and analyze a family of distributed algorithms for the Mobile GDC problem. We refer to these algorithms as the Planar Merge-And-Separate (PMAS) algorithms. These algorithms build upon the ideas presented in the development of the in-strip Merge-And-Separate (MAS) algorithm of [16]. However, as mentioned in Section I, the PMAS algorithms are planar-based as opposed to the strip-based algorithms of [16]. The advantage of this approach is that it avoids inherent inefficiencies resulting from dividing the plane into strips and takes advantage of possible cross-strip optimizations.

A. Distributed Algorithms

Our presentation is in the form of a generic algorithm, with three versions[1]: (i) Square-Cover with Rectangular Separation (SC) (ii) Disk-Cover with Rectangular Separation (DCR), and (iii) Disk-Cover with Circular Separation (DCC). The two disk-cover versions, i.e. DCR-PMAS and DCC-PMAS, constitute *distributed* algorithms for the Mobile GDC problem. The Square Cover Planar MAS (SC-PMAS) is a distributed algorithm that places the minimum number of $D \times D$ *squares* to cover the RNs. Note that the SC-PMAS algorithm is *not* applicable to the Mobile GDC problem. It is presented here solely to serve as a simple demonstration of the analysis technique that is developed for analyzing the DCR-PMAS and DCC-PMAS algorithms.

Algorithm 1/2/3 SC-PMAS, DCR-PMAS, DCC-PMAS algorithms (at MBN d_i, RN q)

Disconnection Rule (at RN q)
if q uncovered **then**
  place MBN d_i, set $P_{d_i} \leftarrow q$
Merge Rule (at MBN d_i)
call Chk-Sqr-Merge(d_i**)**, or **Chk-Dsk-Merge(**d_i**)**
Separate Rule (at MBN d_i)
call SC-Separate(), **DCR-Separate()**, or **DCC-Separate()**

Procedure Chk-Sqr-Merge(d_i**)**
for all MBNs d_j within $3\sqrt{2}D$ of d_i **do**
  if $P_{d_i} \cup P_{d_j}$ coverable by a single $D \times D$ square **then**
    merge d_i and d_j
Procedure Chk-Dsk-Merge(d_i**)**
for all MBNs d_j within $2D$ of d_i **do**
  if $P_{d_i} \cup P_{d_j}$ coverable by a single disk **then**
    merge d_i and d_j

Procedure SC-Separate() (see Fig. 3(a))
if $\exists$ 9 MBNs $a_1, \ldots, a_9$ (including d_i) such that all RNs $q \in \cup_{j=1}^{9} P_{a_j}$ lie within a $3D \times 3D$ area **then**
  separate and **reorganize** $a_1, \ldots, a_9$
Procedure DCR-Separate() (see Fig. 3(b))
if $\exists$ 17 MBNs $a_1, \ldots, a_{17}$ such that all RNs $q \in \cup_{j=1}^{17} P_{a_j}$ lie within a $3D \times 3D$ area **then**
  separate and **reorganize** $a_1, \ldots, a_{17}$
Procedure DCC-Separate() (see Fig. 3(c))
if $\exists$ 14 MBNs $a_1, \ldots, a_{14}$ such that all RNs $q \in \cup_{j=1}^{14} P_{a_j}$ lie in a circular area of diameter $3D$ **then**
  separate and **reorganize** $a_1, \ldots, a_{14}$

[1] In the description of the algorithm, it should be clear which procedure applies to which algorithm version.

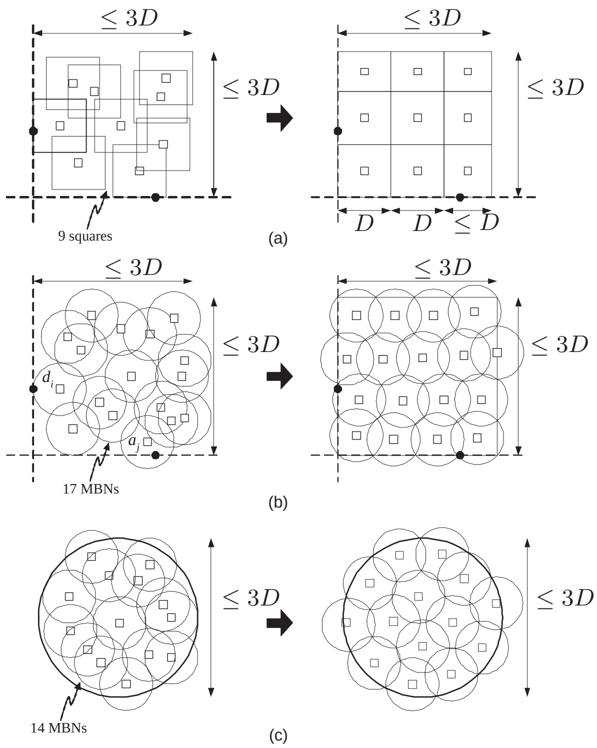

Fig. 3. Planar MAS separation rules: (a) SC-PMAS, (b) DCR-PMAS, and (c) DCC-PMAS.

The generic PMAS algorithm is simple, and the basic idea is that we periodically enforce a *merge rule* and a *separate rule* at each MBN d_i. Additionally, a *disconnection rule* is enforced at each RN q. Namely, if at any time q is not covered by any MBN, assign a new MBN to cover q.

Initially, we assume that there is an MBN covering each individual RN (i.e. as per the disconnection rule). The *merge rule* states that if there exists another MBN d_j that can be merged with d_i (i.e. $P_{d_i} \cup P_{d_j}$ coverable by a single MBN), then *merge* d_i and d_j. The *separate rule* states that if the point-sets of too many mutually non-mergeable MBNs simultaneously converge on a sufficiently small area, then these MBNs should be *separated* (i.e. the MBNs relocated and their point-sets reassigned), as illustrated in Fig. 3. The reasoning behind the choice of the numbers defining *too many* and *sufficiently small* area (e.g. 17 and a $3D \times 3D$ square for DCR-PMAS) will become clear in the next section, when we bound the worst case performance of the algorithms.

For correctness of the algorithm, we assume that both the merge and separate operations can be executed atomically (i.e. without any interrupting operation). We also use the convention that an MBN can be placed arbitrarily within its coverage disk, as long as it is within distance r from all the RNs it is covering. For square MBNs (i.e. for the SC-PMAS), we assume simply that the MBNs are placed somewhere in the $D \times D$ coverage square. Finally, we assume that if at any time an MBN does not cover any RNs (e.g. after a separation operation), it is released.

Note that in the description of the PMAS algorithm, the separate rules are described in general terms, as opposed to an explicit implementation. The reason for this is that there are several possible ways to implement the algorithm, and our goal is to convey the general idea. An example of a distributed implementation of the DCR-PMAS separation rule at MBN d_i could be as follows. MBN d_i starts by detecting all the MBNs (including itself) d_j within distance $4\sqrt{2}D$, and for which d_j^L, d_j^R, d_j^B and d_j^T all lie within an $x-y$ range of $\left[(d_i^L)_x, (d_i^L)_x + 3D\right]$, $\left[(d_i^T)_y + 3D, (d_i^B)_y - 3D)\right]$. Next, these detected MBNs are sorted by ascending bottommost point y-coordinate, yielding a sorted list, denoted by $\{a_1, a_2, \ldots, a_Q\}$. Now, d_i can sequentially check whether $(a_{j+8}^T)_y - (a_j^B)_y \leq 3D$. If this condition holds, then it can conclude that all of the RNs covered by these 9 disks $a_j, \ldots, a_{j+8}$ lie in a $3D \times 3D$ area. At this point, a separate operation can be initiated by sending messages to the appropriate MBNs to move to their new coordinates, and reassign RNs as illustrated in Fig. 3-b. Note that the reassignment of RNs would require additional messages in order to inform each RN of its new covering MBN. The points of reference for the separation are $(d_i^L)_x$ and $(a_j^B)_y$, which are shown in the figure. In particular, the left-bottommost corner of the $3D \times 3D$ area in Fig. 3-b would be $\left[(d_i^L)_x, (a_j^B)_y\right]$.

B. Worst Case Performance

We now analyze the worst case performance of the PMAS algorithms. The induction-based methodology used in the analysis of the strip-based algorithms in [16] cannot be extended to 2-dimensions, since there is no left-to-right directionality that can be exploited. Thus, we develop a novel graph-based analysis technique, which we demonstrate by first analyzing the Square Cover version of the PMAS algorithm (SC-PMAS). We then show how this can be straightforwardly applied to the Disk-Cover versions of the PMAS algorithm.

We use $OPT = \{d_1, d_2, \ldots, d_{|OPT|}\}$ to denote an optimal solution and $ALGO = \{a_1, a_2, \ldots, a_{|ALGO|}\}$ for an SC-PMAS solution. Let P_{d_i} and P_{a_i} represent the sets of RNs covered by the OPT square d_i and the ALGO square a_i, respectively. We define the notion of a_i *touches* d_i (or vice versa) as if and only if there exists at least one RN q, such that $q \in P_{a_i}$ *and* $q \in P_{d_i}$. Finally, define the notion of the PMAS algorithm being in *steady state* if there are no merge or separate actions currently pending.

Lemma 1: In steady state, no more than 8 SC-PMAS ALGO squares can touch a single OPT square d_i.

Proof: Suppose 9 ALGO squares each covered at least one point from P_{d_i}. However, if this was the case then all of the points covered by these 9 squares must lie in a $3D \times 3D$ area, and would have been reorganized as per the separation rule illustrated in Fig. 3(a). Once reorganized, an OPT square can clearly touch at most 4 ALGO squares, which is a contradiction. ■

Lemma 2: In steady state, at most one SC-PMAS ALGO square a_i can exclusively touch a single OPT square d_j (i.e. $P_{a_i} \subseteq P_{d_j}$).

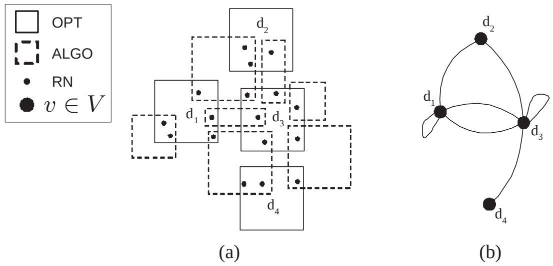

Fig. 4. Demonstration of a graph transformation: (a) original network and square cover, and (b) transformed graph.

Proof: Suppose there existed 2 ALGO squares a_1, a_2 that exclusively touched a single OPT square d_j (i.e. $P_{a_1} \cup P_{a_2} \subseteq P_{d_j}$). However, by definition this means that the set of RNs covered jointly by a_1 and a_2 could be covered by a single square. It follows that in steady state a_1 and a_2 would have been merged as per the merge rule, which is a contradiction. ■

We are now ready to prove the performance guarantee of the SC-PMAS algorithm.

Theorem 1: In steady state, the SC-PMAS algorithm is a 4.5-approximation algorithm.

Proof: We construct an undirected graph $G = (V, E)$ as follows. Define a vertex $v \in V$ for each of the OPT squares. For each $ALGO$ square a_i, we associate *exactly one edge* according to two cases: (i) if a_i only touches a single OPT square d_j, define a self-loop edge (d_j, d_j), and (ii) if a_i touches multiple OPT squares $d_p, d_q, \ldots$, then pick two of these OPT squares (arbitrarily) and define an edge between them (e.g. (d_p, d_q)). Note that there could be both self-loops and parallel edges in the resultant graph. An example of the graph transformation is depicted in Fig. 4.

Finally, since we have associated exactly one $ALGO$ square with one edge, we have that $|V| = |OPT|$ and $|E| = |ALGO|$. Using the standard formula for counting the number edges in an undirected graph with self-loops we have that by lemmas 1 and 2,

$$\begin{aligned} |E| &= \sum_{v \in V} \left(\frac{d(v) - s(v)}{2} + s(v) \right) \\ &\leq \sum_{v \in V} \left(\frac{7}{2} + 1 \right) = \frac{9}{2}|V|, \end{aligned}$$

where $d(v)$ represents the degree of node v, and $s(v)$ the number of self-loop edges at v. ■

At this point, the reasoning behind the exact numbers defining the PMAS separation area (denoted A), and the number of MBNs that must converge on A before separation (e.g. 9 and $3D \times 3D$ square for the SC-PMAS), can be more clearly understood. In turn with Lemma 1, A is defined to be a minimal area satisfying the following: Consider some optimal square (disk) d. For any algorithm square a to touch d, it must only cover RNs which lie in A. Furthermore, a valid separation and reorganization can only be ensured if the squares involved can compactly cover the separation area, so as to ensure all RNs within A are still covered after the separation. Therefore, the number of separated PMAS MBNs (e.g. 9, 17 and 14 respectively) represent the minimum number of MBNs required to compactly cover their respective separation areas.

We are now ready to analyze the disk cover versions, starting with the DCR-PMAS. To do so, we can use the exact same analysis as for the square cover version. To start, we restate lemmas 1 and 2 (whose proofs are identical, except reapplied to disks) in the context of disks, followed by the approximation ratio theorem.

Lemma 3: In steady state, no more than 16 DCR-PMAS ALGO disks can touch a single OPT disk d_i.

Lemma 4: In steady state, at most one DCR-PMAS ALGO disk a_i can exclusively touch a single OPT disk d_j (i.e. $P_{a_i} \subseteq P_{d_j}$).

Theorem 2: In steady state, the DCR-PMAS algorithm is a 8.5-approximation algorithm.

Proof: Using the same definitions and graph transformation as from the proof of Lemma 1, we have that, $|E| \leq \sum_{v \in V}(15/2 + 1) = 8.5|V|$. ■

For the DCC-PMAS algorithm, the proof is identical and thus we simply state the result.

Theorem 3: In steady state, the DCC-PMAS algorithm is a 7-approximation algorithm.

C. Complexity

When discussing the complexity of the distributed algorithms presented in this paper, we will use two standard measures, both with respect to the complexity expended in reaction to a single RN movement. The first is the *time complexity*, which we define as the number of communication rounds and the second is the *local computation complexity* at each MBN, which for a viable algorithm should be negligible compared to a communication round length.

The local computation complexity of the DCR-PMAS algorithm is a periodic $O(C(n))$ to evaluate the merge rule, where $C(n)$ is the running time of the decision 1-center subroutine used. Various efficient algorithms exist that solve the decision 1-center problem, an example being an $O(n \log n)$ algorithm in [9]. The separate rule can be evaluated in $O(1)$, since a packing argument can be used to show that at most 48 MBNs (i.e. a constant number) need be detected by an MBN d_i before there *must* exist 17 MBNs whose points all lie within a $3D \times 3D$ area. Since all point transfers are local (i.e. only take place between adjacent MBNs), the time complexity (number of rounds) is $O(1)$. Hence, this algorithm is implementable in realistic scenarios.

While the merge rule of the DCC-PMAS algorithm also entails a local, periodic $O(C(n))$ computation, implementing the separation rule is much more complex. An example implementation could be examining all circumcircles defined by pairs and triplets of RNs whose ensuing radii are at most $3D/2$, and testing whether the point-sets of 14 MBNs lie within. Note however, that this entails a *centralized* $O(n^3 C(n))$ computation (e.g. by collecting all RN location

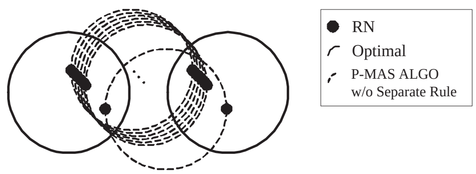

Fig. 5. A pathological example of arbitrarily bad performance of a PMAS algorithm without the separate rule.

information at some MBN), which is much too high to implement frequently.

Fortunately, an important note regarding the PMAS algorithms is that the merge rule is far more important than the separate rule. It turns out the merge rule is the one that ensures good average performance, whereas the separate rule protects against the rare, pathological yet theoretically possible cases of extreme inefficiency. An example of such a pathological situation is shown in Fig. 5, in which an arbitrarily large number of mutually non-mergeable MBNs cover points coverable by 2 optimal MBNs. However, such situation would almost never occur in any practical scenario and thus the separate rule need only be implemented very rarely, perhaps making the DCC-PMAS also a viably implementable algorithm in certain scenarios.

IV. Cluster Cover Algorithm

In this section we present the Cluster Cover (CC) algorithm which, like the PMAS algorithms, distributedly solves the Mobile GDC problem without the use of strips. The advantage of the CC algorithm over the PMAS algorithms is that it is simpler to implement, and has a lower computational complexity. Furthermore, we show that the approximation ratio of the CC algorithm is lower than that of the PMAS algorithms. Yet, as will be shown via simulation, on average the PMAS algorithms perform significantly better than the CC algorithm.

Before describing the algorithm we present the following definitions. Given an undirected graph $G(V,E)$, a *dominating set* is as a subset $Q \subseteq V$ such that $\forall i \in V$, either $i \in Q$ or $\exists (i,j) \in E$ for some $j \in Q$. An *independent set* is defined as a subset $Q \subseteq V$ such that $\forall i,j \in Q$, $\nexists (i,j) \in E$. Finally, given N points (RNs) distributed in the plane, a *unit disk graph* $G = (V,E)$ is defined such that $V = N$ and $(i,j) \in E \Leftrightarrow d_{ij} \leq r$.

The CC algorithm is based on an *overlooked* observation regarding the relation between the GDC problem and the Maximal Independent Set (MIS) problem. Before describing this relation, we note that restricting the locations of the MBNs (i.e. the disk centers) to the locations of the RNs (points) yields a restricted version of the GDC problem. This restricted GDC problem is equivalent to a *Minimum Dominating Set* (MDS) problem in a unit disk graph. Hence, $|GDC_{OPT}| \leq |MDS_{OPT}|$, where $|GDC_{OPT}|$ and $|MDS_{OPT}|$ are the cardinalities of the optimal solutions to the unrestricted GDC problem and to the MDS problem in a unit disk graph.

An MIS is by definition a dominating set. Therefore, finding an MIS provides an approximate solution to the MDS problem. An MIS can be found in linear time by a simple centralized algorithm that adds nodes to the set and then deletes their neighbors from the graph. It was shown in [14, Theorem 4.8] that in unit disk graphs the cardinality of an MIS is at most 5 times the cardinality of the MDS. Namely, $|MIS| \leq 5|MDS_{OPT}|$.

We now show that an MIS in the unit disk graph of the RNs is a valid solution to the unrestricted GDC problem and that its cardinality is *at most 5 times the cardinality of the optimal GDC solution*. Namely, $|MIS| \leq 5|GDC_{OPT}|$. Hence, an MIS algorithm operating on a unit disk graph provides a 5-approximation not only to the MDS problem in the unit disk graph but also to the *unrestricted* GDC problem in the plane. Notice that this relation is *not* directly implied by the above inequalities.

An MIS in the unit disk graph of the RNs is a feasible solution to the GDC problem, since all RNs are within distance r from an MBN. However, in general it is not an optimal solution. This results from the fact that for the GDC problem, MBNs can be placed anywhere in the plane. On the other hand, in the unit disk graph problem, MBNs are constrained to lie on top of RNs. As shown below the approximation ratio obtained by finding an MIS can be easily bounded.

Lemma 5: An MIS algorithm in the unit disk graph of RNs is a 5-approximation algorithm for the GDC problem.

Proof: Let OPT and $ALGO$ represent an optimal and algorithmic GDC solutions (the algorithmic solution is an MIS). As mentioned earlier, the algorithm maintains the invariant that no two disk (MBN) centers are within distance r from each other. Similarly to [14], it can be shown that this implies that at most 5 disk *centers* can lie in a circular area of radius r. Namely, at most 5 $ALGO$ disk centers can lie inside the area covered by an OPT disk. Since all $ALGO$ disk centers are placed on top of points (RNs) that are covered by the optimal solution, all $ALGO$ disk centers are contained within *some* OPT disk. Since the number of $ALGO$ disk centers is same as the number of $ALGO$ disks, $|ALGO| \leq 5|OPT|$. ■

A distributed implementation of the the Cluster Cover (CC) algorithm that finds an MIS in a unit disk graph of the RNs can be based on an algorithm developed by Baker and Ephremides [1] for clustering in a mobile wireless network. The local computation complexity of the CC algorithm is $O(1)$ since at each iteration simple decisions need to be taken. However, the time complexity (number of rounds) is $O(n)$. We note that several more efficient distributed implementations of MIS algorithms exist and can be easily adapted to our scenario.

V. Strip-Based Algorithms

A. Strip Cover Algorithm

For brevity, we only discuss the centralized versions of the strip cover algorithm of [16] (the analysis applies also to the distributed versions). We focus on the following algorithms:

(i) Strip Cover with Rectangles (SCR), and (ii) Strip Cover with Disks (SCD). Both algorithms act upon RNs within strips of width αD, $\alpha < 1$ and work by iteratively locating the leftmost yet uncovered RN i, and placing an MBN that greedily covers as many RNs as possible given that i is also covered. The solutions for each strip in the plane are then combined for an overall solution.

The difference between the two algorithms is the manner in which the RNs are greedily covered. The SCR algorithm simply treats MBNs as $\alpha D \times \sqrt{1-\alpha^2}D$ rectangles, and therefore, covers all RNs with x-coordinate in $[i_x, i_x + \sqrt{1-\alpha^2}D]$. By contrast, the SCD algorithm utilizes the full disk coverage area and covers as many RNs as possible such that there do not exist any uncovered RNs to the *left* of the rightmost covered RN.

The resulting worst-case approximation ratios, derived in [16] are 6 and 4.5 for the SCR and SCD, respectively. However, in general worst-case performance bounds are not a good indicator of typical performance. Accordingly, in the next section we provide a probabilistic analysis to show that the expected performance of both algorithms is actually much better than their worst-case performance ratios would indicate.

B. Average Case Performance Analysis

For our analysis, we assume that the RNs are randomly distributed according to a two dimensional Poisson process of density $\lambda\ nodes/unit^2$. Note that a key property of such a distribution is that when the number of RNs is a-priori given, their positions are independent and each is *uniformly distributed* in the plane. Due to the random locations of the RNs, the number of MBNs placed by an optimal algorithm, $|OPT|$ is a random variable. Similarly, we define $|SCR|$ and $|SCD|$ as random variables corresponding to the number of disks placed by the SCR and the SCD algorithms. We define the *average approximation ratios* β_{SCR} and β_{SCD} as,

$$\beta_{SCR} = \frac{E[|SCR|]}{E[|OPT|]},\ \beta_{SCD} = \frac{E[|SCD|]}{E[|OPT|]}. \tag{1}$$

It should be noted that β_{SCR} differs from the expected value of the approximation ratio (e.g. $E[|SCR|/|OPT|]$). Yet, it provides a good measure of the average performance.

The main result of this section is the following theorem and corollary, which bound the average approximation ratios of both the SCR and SCD algorithms (since SCD always outperforms SCR).

Theorem 4: Given RNs distributed in the plane according to a two dimensional Poisson process with density λ,

$$\beta_{SCD} \leq \beta_{SCR} \leq \frac{D^2\lambda + 2D\sqrt{\lambda} + 1}{\alpha\sqrt{1-\alpha^2}D^2\lambda + 1}. \tag{2}$$

Corollary 1: If $q = \frac{D}{\sqrt{2}}$, then $\beta_{SCD} \leq \beta_{SCR} \leq 3$.

The consequence of the above is that even though the worst case approximation ratios of the SCR/SCD algorithms are 6 and 4.5 (respectively), selecting a specific strip width results in an average approximation ratio which is bounded by 3. We spend the remainder of the section proving these results, for which the following 2 lemmas, which respectively upper bound $E[|SCR|]$ and lower bound $E[|OPT|]$, are required.

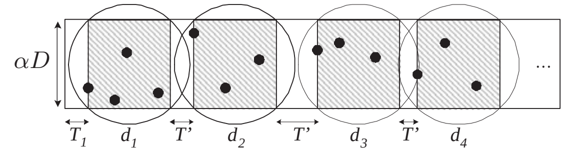

Fig. 6. Probabilistic analysis of the performance of the SCR algorithm within a strip.

Lemma 6: Given a strip width $q = \alpha D$, and an $L \times K\alpha D$ planar area,

$$E[|SCR|] \leq \frac{\lambda\alpha DLK}{\lambda\alpha\sqrt{1-\alpha^2}D^2 + 1}. \tag{3}$$

Proof: Consider a single strip S and recall that the SCR algorithm iteratively places disks by identifying the leftmost uncovered RN i and fully covering the x-range between i_x and $i_x + \sqrt{1-\alpha^2}D$ along the strip. The RNs are distributed in the plane according to a two dimensional Poisson process with density λ.

Therefore, the horizontal (x-coordinate) distance between RNs is exponentially distributed with average $\frac{1}{\lambda\alpha D}$. Thus, the expected distance to the location of the first disk is $E[T_1] = \frac{1}{\lambda\alpha D}$ (see Fig. 6). Furthermore, once a disk is placed, the expected distance between the end of its coverage and the start of the next disk is $E[T']$. Due to the memoryless property of the exponential random variable, we can conclude $E[T'] = \frac{1}{\lambda\alpha D}$.

It therefore follows that the expected number of disks used by the SCR algorithm within a strip is the total length of the strip (less the initial space) divided by the expected distance between the start of one disk and the start of another. Namely,

$$\begin{aligned} E[|SCR|_S] &= \frac{L - \frac{1}{\lambda\alpha D}}{\sqrt{1-\alpha^2}D + \frac{1}{\lambda\alpha D}} \\ &\approx \frac{\lambda\alpha DL}{\lambda\alpha\sqrt{1-\alpha^2}D^2 + 1}, \end{aligned} \tag{4}$$

where $E[|SCR|_S]$ is the expected number of disks used by the SCR algorithm within strip S, and we assume that $L >> \frac{1}{\lambda\alpha D}$. The expected total number of disks used by the algorithm over the entire plane is therefore this number multiplied by the total number of strips in the plane, giving us the desired result. ■

Lemma 7: Given an $L \times K\alpha D$ planar area,

$$E[|OPT|] \geq \frac{KL\alpha D}{D^2 + \frac{1}{\lambda} + \frac{2D}{\sqrt{\lambda}}}. \tag{5}$$

Proof: We divide the plane into D-spaced horizontal strips of width q as shown in Fig. 7. We can lower bound the expected number of disks used to cover RNs in a single strip S by an optimal algorithm by noting that the area coverable by each OPT disk is no more than a rectangle of size $q \times D$. Thus, using a similar argument to when we upper bounded

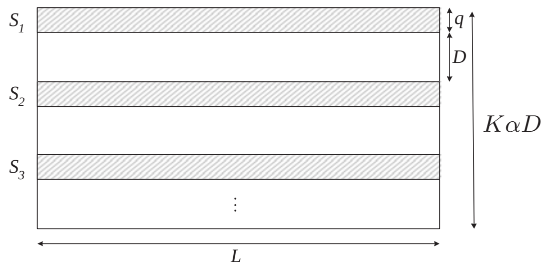

Fig. 7. Dividing the plane into strips so as to lower bound $E[|OPT|]$

the number of SCR disks required to cover a strip, we have that,

$$E[|OPT|_S] \geq \frac{L}{D + \frac{1}{\lambda q}}, \tag{6}$$

where $E[|OPT|_S]$ is the expected number of disks used by the optimal solution within strip S. Next we note that an upper bound on the expected number of OPT disks used to cover RNs in the whole plane can be achieved by summing over the disks used to cover each of the individual strips. The reason we can do this is that since there is a distance D between strips, it is impossible for a single OPT disk to simultaneously cover RNs from two different strips. We therefore have that,

$$\begin{aligned} E[|OPT|] &\geq \left(\frac{L}{D + \frac{1}{\lambda q}}\right) \cdot \left(\frac{K\alpha D}{D + q}\right) \\ &= \frac{KL\alpha D}{D^2 + \frac{1}{\lambda} + \left(Dq + \frac{D}{\lambda q}\right)}. \end{aligned} \tag{7}$$

Next, since we have control over the strip size q, and want to find the tightest possible lower bound, we can select q so as to maximize $E[|OPT|]$, i.e. minimize the bracketed quantity in the denominator of (7). It turns out that setting $q = \sqrt{1/\lambda}$ achieves this. Substituting this into (7), yields our desired result. ■

Proof of Theorem 4: Combining the results of lemmas 6 and 7, as well as (1) gives us our desired upper bound on β_{SCR}, i.e.,

$$\begin{aligned} \beta_{SCR} &\leq \left(\frac{\lambda\alpha DLK}{\lambda\alpha\sqrt{1-\alpha^2}D^2 + 1}\right) \cdot \left(\frac{\lambda D^2 + 2\sqrt{\lambda}D + 1}{\lambda\alpha DLK}\right) \\ &= \frac{D^2\lambda + 2D\sqrt{\lambda} + 1}{\alpha\sqrt{1-\alpha^2}D^2\lambda + 1}. \end{aligned} \tag{8}$$

■

Proof of Corollary 1: We derive the maximum value of (8) by differentiating with respect to λ. Upon doing so and plugging this value of λ into (8) gives us,

$$\beta_{SCR}\,|_{\lambda=\lambda_{max}} \leq \frac{\alpha\sqrt{1-\alpha^2} + 1}{\alpha\sqrt{1-\alpha^2}}, \tag{9}$$

which is independent of D. For $\frac{1}{2} \leq \alpha < 1$, (9) is minimized when $\alpha = 1/\sqrt{2}$, at which point it attains a value of exactly 3. ■

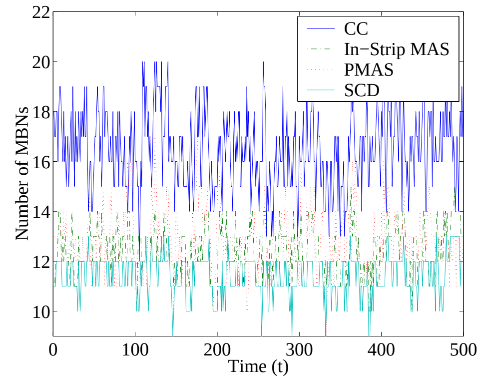

Fig. 8. The number of MBNs used by the GDC algorithms during a time period of $500s$ in a network of 80 RNs.

VI. Performance Evaluation

In this section we evaluate the performance of the algorithms via simulation. The results have been obtained by a model of our algorithms, developed in Java.

We start with the mobile RN scenario, comparing the performance of the planar GDC algorithms developed in this paper to some of the strip-based algorithms developed in [16]. Figures 8 and 9 illustrate simulation results for a network with mobile RNs. The mobility model used is the Random Waypoint Model in which RNs continually repeat the process of picking a random destination in the plane and moving there at a random speed in the range $[V_{min}, V_{max}]$. We used a plane of dimensions $600m \times 600m$, with $V_{min} = 10m/s$ and $V_{max} = 30m/s$, and set the RNs communication range as $r = 100m$. Finally, each simulation was performed for $1000s$ from which we discarded the first $500s$.

Fig. 8 illustrates the evolution of the algorithms over a $500s$ time period, with 80 RNs. It can be seen that the simplest and least computationally complex algorithm, the CC algorithm, has the poorest performance. Fig. 9 shows the average number of MBNs used over a $500s$ time period as a function of the number of RNs. Each data point is averaged over 10 instances. As can be seen in the figure, when the number of RNs is low, the PMAS is the best performing algorithm. However, for a larger number of RNs, both of the strip-based algorithms perform better. The reason for this is that when the configuration of RNs is sparse, cross-strip optimization is more important, since scenarios such as those depicted in Fig. 2 can frequently occur. By contrast, as the configuration of RNs grows more dense, MBNs will have to be used in all strips regardless. Thus, in this case, the fact that both the SCD and In-Strip MAS algorithms perform better within a strip than the PMAS explains their superior performance.

For a network with static RNs, Fig. 10 presents the the average ratios between the solutions obtained by both the planar and strip-based algorithms, and the optimal solution. We used a plane of dimensions $1000m \times 1000m$ and set the RNs communication range as $r = 100m$. For each data point, the average was obtained over 10 different random

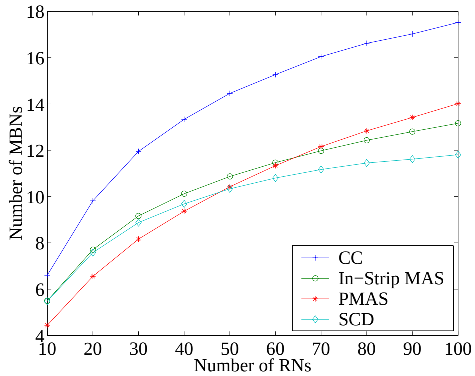

Fig. 9. The average number of MBNs used by GDC algorithms over a time period of $500s$.

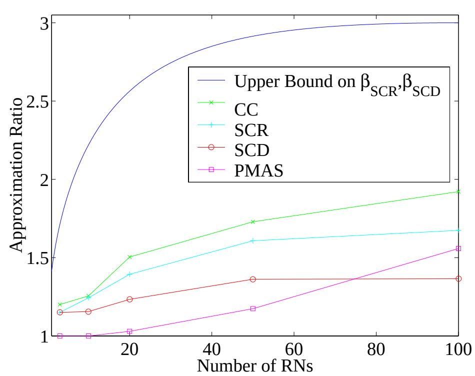

Fig. 10. Ratios between the solutions by the SCD and SCR algorithms and the optimal solution, and an upper bound on average approximation ratios.

instances in which the RNs are uniformly distributed in the plane. The optimal solutions were obtained by formulating *each instance* of the GDC problem as an Integer Program and solving it using CPLEX. From the figure, it can be seen that although the worst case performance ratios of the CC, SCR, PMAS and SCD algorithms are 5, 6, 8.5 and 4.5, their average performance ratios attained in simulation are closer to 2, 1.7, 1.5 and 1.4, respectively. Furthermore, the trend observed in the mobile scenarios, in which the PMAS outperforms the SCD for sparse RN configurations and vice versa for dense RN configurations, still holds.

Fig. 10 also presents the upper bound on the average approximation ratios (β_{SCR} and β_{SCD}) derived in Theorem 4. The large gap between the bound on the average approximation ratios and the actual ratios indicates that the bound is somewhat loose.

VII. Conclusion

The architecture of a hierarchical Mobile Backbone Network has been presented only recently. Such an architecture can significantly improve the performance, lifetime, and reliability of MANETs and WSNs. In this paper, we concentrate on placing and mobilizing backbone nodes, dedicated to maintaining connectivity of the regular nodes. Specifically, we focus on the important subproblem of Mobile Geometric Disk Cover. We have proposed a number of *distributed* planner-based algorithms for this problem and bounded the worst case performance of two of them using a new methodology. In addition, we analyzed the average case performance of two algorithms recently presented in [16]. Finally, we studied the performance under mobility via simulation.

A major future research direction is to generalize the model to other connectivity constraints and objective functions. For instance, we intend to extend the results to connectivity models that are more realistic than the disk connectivity model. Moreover, we intend to consider the energy resources and the communication requirements of the RNs when making the mobility decisions.

Acknowledgments

This research was supported by Draper Laboratories, by DARPA/AFOSR through the University of Illinois grant no. F49620-02-1-0325, and by a Marie Curie International Fellowship within the 6th European Community Framework Programme.

References

[1] D.J. Baker and A. Ephremides, "The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm," *IEEE Trans. Commun.*, 29, 11, 1694-1701, 1981.
[2] S. Basagni, "Distributed Clustering for Ad Hoc Networks," *Proc. I-SPAN'99*, June 1999.
[3] B. Das, R. Sivakumar, and V. Bharghavan, "Routing in Ad-Hoc Networks Using a Virtual Backbone," *Proc. IEEE ICCCN'97*, 1997.
[4] M. Franceschetti, M. Cook, and J. Bruck, "A Geometric Theorem for Approximate Disk Covering Algorithms," *Technical Report ETR035*, Caltech, 2001.
[5] J. Gao, L. Guibas, J. Hershberger, L. Zhang, and A. Zhu, "Discrete Mobile Centers," *Disc. Comput. Geom.*, 30, 1, pp. 45-63, 2003.
[6] M. Gerla and J. T. Tsai, "Multicluster, Mobile, Multimedia Radio Network," *ACM/Baltzer WINET*, 1, pp. 255-265, 1995.
[7] T. Gonzalez, "Covering a Set of Points in Multidimensional Space," *Inf. Proc. Letters*, 40, 4, pp. 181-188, 1991.
[8] J. Hershberger, "Smooth Kinetic Maintenance of Clusters," *Proc. ACM SoGC'03*, 2003.
[9] J. Hershberger and S. Suri, "Finding Tailored Partitions," *J. Algorithms*, 12, pp. 431-463, 1991.
[10] D.S. Hochbaum and W. Maass, "Approximation Schemes for Covering and Packing Problems in Image Processing and VLSI," *J. ACM*, 32, 1, pp. 130-136, 1985.
[11] H. Huang, A. Richa, and M. Segal, "Approximation Algorithms for the Mobile Piercing Set Problem with Applications to Clustering in Ad-Hoc Networks," *ACM/Kluwer MONET*, 9, pp.151-161, 2004.
[12] A. Kansal, M. Rahimi, D. Estrin, W.J. Kaiser, G. J. Pottie, and M. B. Srivastava, "Controlled Mobility for Sustainable Wireless Sensor Networks," *Proc. IEEE SECON'04*, 2004.
[13] M. Lu, J. Wu, M. Cardei, and M. Li, "Energy-Efficient Connected Coverage of Discrete Targets in Wireless Sensor Networks," *Proc. ICCNMC'05*, Springer, Aug. 2005.
[14] M.V. Marathe, H. Breu, H.B. Hunt III, S.S. Ravi, D.J. Rosenkrantz, "Simple Heuristics for Unit Disk Graphs," *Networks*, 25, pp. 59-68, 1995.
[15] I. Rubin, A. Behzad, R. Zhang, H. Luo, and E. Caballero, "TBONE: a Mobile-Backbone Protocol for Ad Hoc Wireless Networks," *Proc. IEEE Aerospace Conf.*, 6, 2002.
[16] A. Srinivas, G. Zussman, and E. Modiano, "Mobile Backbone Networks - Construction and Maintenance," *Proc. ACM MOBIHOC'06*, May 2006.
[17] J. Tang, B. Hao, and A. Sen, "Relay Node Placement in Large Scale Wireless Sensor Networks," *Computer Commun.*, 29, 4, pp. 490-501, 2006.
[18] K. Xu, X. Hong, and M. Gerla, "Landmark Routing in Ad Hoc Networks with Mobile Backbones," *J. Parallel Distrib. Comput.*, 63, 2, pp. 110-122, 2003.