Topology Control for Constant Rate Mobile Networks

LIANG ZHAO [1] ERROL LLOYD [1] S. S. RAVI [2]

***Abstract*— Controlling the topology of a wireless ad hoc network is very important from the point of view of performance. One known technique for controlling the topology is through the assignment of appropriate transmission power levels to the nodes. Such an assignment aims to minimize a specified function of the powers assigned to nodes. While this problem has been widely studied for the case of *stationary* wireless networks, few reported theoretical results for *mobile* wireless networks (MANETs). In this paper, we consider the topology control problem for MANETs from a theoretical perspective.**

We define a topology control problem under the Constant Rate Mobile Network model. In this model, all the n nodes in the network may move. Associated with each moving node are its constant moving speed and direction. The goal is to minimize the maximum power used by any network node in producing a connected network. We provide two polynomial algorithms for solving this problem: one for the decision version, the other for the optimization version.

I. INTRODUCTION

Two of the most critical issues associated with wireless ad hoc networks used in military and search-and-rescue operations are to conserve energy so as to prolong battery life and to accommodate the movement of network nodes. This paper considers these issues in the context of topology control.

A *wireless ad hoc network* consists of a collection of nodes which self-organize using communication based on radio propagation, since there is no pre-existing infrastructure. In communicating through *wireless links* each node functions, when necessary, as a relay so as to allow multihop communications. In ad hoc networks, battery power is a precious resource.

[1]Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716. Email: {zhaol, elloyd}@cis.udel.edu. Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes not withstanding any copyright notation thereon.

[2]Department of Computer Science, University at Albany - SUNY, Albany, NY 12222. Email: ravi@cs.albany.edu.

In a *mobile* wireless ad hoc network *(MANET)*, where nodes move freely, the *network topology* is formed based on the transmission ranges and the routes of the nodes. The objective in *topology control* is to maintain a specified network topology such as connected or biconnected. The desired effect of topology control is to reduce energy consumption, reduce MAC layer interference between adjacent nodes, and to increase the effective network capacity.

A widely studied method of controlling the topology of a wireless network is by adjusting the transmission powers of the nodes. The idea is to choose the transmission powers so that the graph induced by the power assignment satisfies the specified properties. Since battery power is a precious resource, the power assignment aims to optimize a suitable function of the powers assigned to nodes. In this paper, we will focus on minimizing the *maximum power* utilized by any node.

Over the last six years, the topology control problem has been studied by several groups of researchers [7], [5], [10], [3], [8], [9]. All of this work has assumed that the nodes are stationary. In this paper, we consider certain topology control problem with mobile nodes. We are the first to develop polynomial time algorithms for this problem.

II. RELATED WORK

For stationary networks, considerable work has been reported in the literature on a variety of topology control problems. For instance, several groups of researchers have studied connectivity problems (under the undirected graph model) for minimizing the total power assigned to nodes [9], [8]. Likewise, work on the connectivity problems under the directed graph model may be found in [4], [7], [5]. In most instances, the problems are shown to be **NP**-hard and the focus is on the development of approximation algorithms having either $O(\log n)$ or constant approximation ratios. For instance, [1], [2] show the **NP**-hardness of connectivity problems (under the undirected graph model) for minimizing the total power, and Calinescu *et al.* [2] present a 5/3-

approximation algorithm for that problem. For 2-node/edge-connectivity problems for minimizing the total power, two new approximation algorithms with asymptotic approximation ratios of 8 are presented in [9]. Both of the approximation ratios are improved to 4 in [3].

III. Background

A. Problem Specification

In studying topology control for MANETs, we are given a set of nodes in the plane, each of which may move as time progresses. At any given instant, for each ordered pair (u, v) of nodes, there exists a transmission power *threshold*, denoted by $\pi(u, v)$, with the following significance: A signal transmitted by node u can be received by node v if and only if the transmission power of u is at least $\pi(u, v)$. In this paper we utilize the *geometric model* in which the threshold is determined by the Euclidean distance $d(u, v)$ between u and v. Throughout this paper, the threshold $\pi(u, v)$ is taken to be $d(u, v)^\alpha$, where α is the *attenuation constant* associated with path loss [11]. The value of α is typically between 2 and 4. Note that in the geometric model threshold values are *symmetric*. That is, $\pi(u, v) = \pi(v, u)$. In the remainder of this paper, we let $\pi(u, v)$ denote both itself and $\pi(v, u)$.

At any instant in time, given the transmission powers and the positions of the nodes, an ad hoc network can be represented by an undirected graph over the nodes of the network. An edge (u, v) is in this *induced* graph if and only if the transmission powers of both u and v are at least the threshold $\pi(u, v)$.

In the context of MANETs, the main goal of topology control is to assign transmission powers to nodes so that the network is *movement-connected*. That is, at every instant in time, the undirected graph induced by the transmission powers of the nodes is *connected*. Topology control aims to achieve this connectivity while minimizing the maximum power uniformly assigned to any nodes. In this case we say that *network $\mathcal{N}$ is movement-connected under such power*.

We slice the lifetime of a mobile network into *unit time intervals* during each of which it is assumed that the movement of each node can be represented by a line segment. We study topology control in MANETs for each unit time interval.

In this paper we consider **The Constant Rate Mobile Network (CRMN) Problem** for MANETs in which there are n moving nodes. Associated with each node are its starting and ending positions in the unit time interval. It is assumed that each node moves at its own constant rate and direction throughout the time interval. The goal is to minimize the maximum power uniformly assigned to all nodes such that the network is movement-connected throughout the unit time interval. Readers are referred to [14], [13] for other relevant mobility models.

The main results of this paper provide two polynomial algorithms for solving the CRMN problem:

- An $O(n^2(\log(n))^2)$ algorithm for the decision version (i.e. given a power p and an instance $\mathcal{N}$ of CRMN, to decide whether $\mathcal{N}$ is movement-connected under p);
- An $O(n^4(\log(n))^2)$ algorithm for the optimization version (i.e. given an instance $\mathcal{N}$ of CRMN, to find the minimum power p_{min} assigned to all nodes such that $\mathcal{N}$ is movement-connected under p_{min}).

B. Some Definitions and Notations

Some definitions and notations used in the remainder of this paper are defined below:

- We let $G^p(\mathcal{N})$ denote the undirected graph induced on a stationary network $\mathcal{N}$, when transmission power p is uniformly assigned to each node. That is, in $G^p(\mathcal{N})$ an edge is present between nodes u and v if $p \geq \pi(u, v)$.
- A *threshold graph* is a complete undirected edge-weighted graph where each edge is of positive weight. The weight of each edge is its *threshold*. A *threshold graph for a stationary network* $\mathcal{N}$ is a threshold graph with the same node set as $\mathcal{N}$ and where the weight of edge (u, v) is $\pi(u, v)$.

C. Stationary Networks

The general form of topology control considered in this paper was first proposed by Ramanathan and Rosales-Hain [10]. Among the several results in that paper, they presented an algorithm for stationary networks that minimizes the maximum power assigned to any node such that the resulting network is connected. Subsequently, [9] provided a general polynomial algorithm for minimizing the maximum power assigned to any node so as to achieve a range of graph properties. Additional results for stationary networks are given in Section II on related work.

In this paper we make extensive use of the algorithm given in [10], [9] for minimizing the maximum power p such that network $\mathcal{N}$ is connected

under power p. That algorithm is based on the insight that p must come from among the threshold values associated with node pairs in $\mathcal{N}$. This permits the algorithm to do a binary search over those threshold values searching for the least p such that $\mathcal{N}$ is connected under power p. We refer to that algorithm as MINMAXGRAPH. The running time of MINMAXGRAPH is $O(n^2 \log(n))$. We will make use of MINMAXGRAPH as a subroutine throughout this paper.

IV. CONSTANT RATE MOBILE NETWORKS

In this section we give polynomial time algorithms for topology control in constant rate mobile networks: i.e. an $O(n^2(\log(n))^2)$ algorithm for the decision version and an $O(n^4(\log(n))^2)$ algorithm for the optimization version. Throughout this section for a moving node V_i, we refer to its starting position as V_i, to its ending position as V_i', and to the vector $\vec{v}_i = \overrightarrow{V_iV_i'}$ as its moving route[3]. Our approach to solving an instance of CRMN is based on partitioning the unit time interval into *time slots*. We refer to this as *time slicing*.

A. Distance Functions and Threshold Functions

Since $|\vec{v}_i|$ is the length of the moving route of node V_i in a unit time interval, it follows that $|\vec{v}_i|$ is also the speed of V_i and at any instant t in $[0, 1]$ its position is $\vec{v}_i \cdot t$.

To deal with distances and thresholds associated with multiple moving nodes we introduce the concepts of *distance functions* and *threshold functions* based on vector calculations. In a standard coordinate system, point A is denoted as $A = (x_A, y_A)$ and *point vector* $\vec{p}_A$ represents a vector $\overrightarrow{OA}$ starting at the origin O and ending at A. Hence, a moving route starting at A and ending at A' will be denoted as $\vec{v}_{AA'} = \vec{p}_{A'} - \vec{p}_A$.

Using these concepts we define a *relative moving vector* $\vec{v}_{A-B}$ to be $\vec{v}_{AA'} - \vec{v}_{BB'}$. Note that in physical terms $\vec{v}_{A-B}$ represents the movement of A relative to B (i.e., as if B does not move). Figure 1 shows an example: $\vec{v}_{AA'} = \vec{p}_{A'} - \vec{p}_A$, $\vec{v}_{BB'} = \vec{p}_{B'} - \vec{p}_B$, and $\vec{v}_{A-B} = \vec{v}_{AA'} - \vec{v}_{BB'}$.

With these definitions, we define the *distance function* between A and B to be $d_{AB}(t) = |(\vec{p}_A - \vec{p}_B) + \vec{v}_{A-B} \cdot t|$. At any time instant t, this function is the distance between moving nodes A and B.

[3]We use $\vec{v}_i$ to represent the vector $\overrightarrow{V_iV_i'}$ starting at V_i and ending at V_i'.

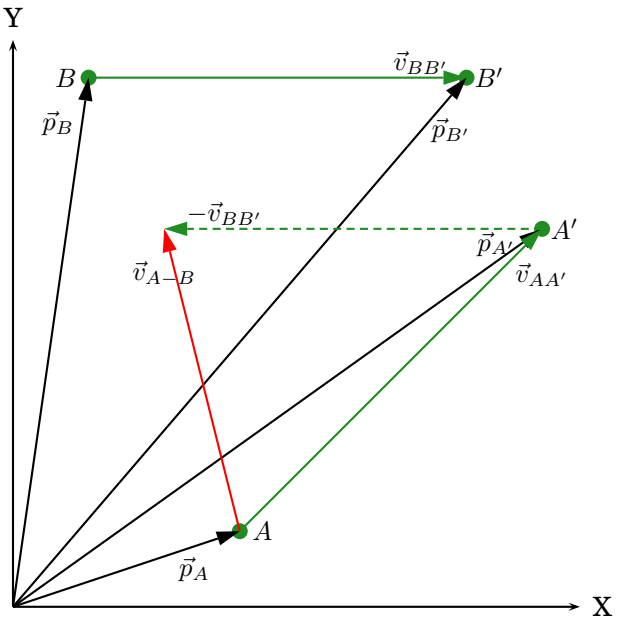

Fig. 1. Vector Notation Example

The *threshold function* between A and B is $\pi(A, B)[t] = d_{AB}^{\alpha}(t)$, where α is again the attenuation constant associated with path loss. At any time instant t, this function is the threshold between moving nodes A and B. The *threshold between A and B in a time slot* $[t_g, t_h]$ is denoted as $\pi(A, B)[t_g, t_h] = \text{MAX}(\pi(A, B)[t_g], \pi(A, B)[t_h])$.

Finally, a distance function is the square root of a quadratic function. Since there are at most two real solutions[4] to a quadratic equation, we state without proof:

Lemma 4.1: Given any two non-identical distance functions $d_{AB}(t)$ and $d_{CD}(t)$, there are at most two real solutions for t to the equation $d_{AB}(t) = d_{CD}(t)$.

Since a threshold function has the form $\pi(A, B)[t] = d_{AB}^{\alpha}(t)$, it follows that:

Corollary 4.2: Given any two non-identical threshold functions $\pi(A, B)[t]$ and $\pi(C, D)[t]$, there are at most two real solutions for t to the equation $\pi(A, B)[t] = \pi(C, D)[t]$.

B. An Algorithm for the Decision Version of CRMN

In this subsection, we describe an algorithm to solve the decision version of CRMN. That is, given a power p and an instance $\mathcal{N}$ of CRMN, to decide whether $\mathcal{N}$ is movement-connected under p.

[4]Imaginary solutions do not have any physical significance in this context.

Our algorithm is based on time slicing. For each pair of distinct nodes V_i and V_j in $\mathcal{N}$, we calculate the real solutions to the equation $\pi(V_i, V_j)[t] = p$. The resulting solutions are called *slicing points*. We collect all of the $O(n^2)$ distinct[5] slicing points into a sorted list. Adjacent points in this sorted list define a time slot. An important fact is that connectivity of two nodes changes only when the edge between them either comes into or goes out of existence, which can only occur around a slicing point. It follows that each time slot is a *constant-connectivity time slot*. Precisely, a time slot $[t_x, t_y]$ is a constant-connectivity time slot if given a power p and a graph $G = (V, E)$, the following hold:

- $\forall (V_i, V_j) \in E, \forall t \in (t_x, t_y) : \pi(V_i, V_j)[t] < p$
- $\forall (V_i, V_j) \notin E, \nexists t' \in (t_x, t_y) : \pi(V_i, V_j)[t'] \leq p$.

Note that the slicing points defining the time slot are not included in the above calculations. To see the necessity of excluding these points consider consecutive time slots $[t_{i-1}, t_i]$ and $[t_i, t_{i+1}]$, and assume some edge (V_j, V_k) does not exist at an interior point of $[t_{i-1}, t_i]$, but does exist at an interior point of $[t_i, t_{i+1}]$. Thus, this edge comes into existence at slicing point t_i. Clearly, t_i cannot be included in the constant-connectivity calculation for $[t_{i-1}, t_i]$. Fortunately, as we will show later there is no need to explicitly check connectivity at the slicing points.

Having partitioned the unit time interval into constant-connectivity time slots, our algorithm simply checks the connectivity for each time slot. Network $\mathcal{N}$ is movement-connected under p if and only if it is connected in each such time slot. The details are given in Algorithm 1.

Input: An instance $\mathcal{N}$ of CRMN, and a power p.

Output: If $\mathcal{N}$ is movement-connected under power p, return true; else return false.

Steps:

1) $T \leftarrow \emptyset$: a set of slicing points.
2) For each 2 distinct nodes V_i, V_j in $\mathcal{N}$ do
   a) Compute the real solutions to $\pi(V_i, V_j)[t] = p$, and put them into T.
3) Let ST be the sorted list of values in T, and let $\mathcal{T}_k = [t_k, t_{k+1}]$ be the k^{th} time slot of the unit time interval as defined by the adjacent values in ST.
4) For each $\mathcal{T}_k$ do
   a) $t'_k \leftarrow$ an interior time instant of $\mathcal{T}_k$.
   b) Construct a graph $G_k = (V, E)$, where V is the set of nodes in $\mathcal{N}$ and $E = \{(V_i, V_j) : \pi(V_i, V_j)[t'_k] \leq p\}$.
   c) If G_k is disconnected, then return false.
5) Return true.

Fig. 2. Algorithm 1 — SOLUTIONTODECISIONCRMN

To prove the correctness, we begin with a lemma stated without proof:

Lemma 4.3: Given a constant-connectivity time slot $[t_k, t_{k+1}]$, any edge present at an instant $t'_k \in (t_k, t_{k+1})$ is also present at any instant $t''_k \in [t_k, t_{k+1}]$.

Theorem 4.4: Algorithm 1 returns true if and only if network $\mathcal{N}$ is movement-connected under power p.

Proof: We show that network $\mathcal{N}$ is movement-connected under power p if and only if every graph G_k (constructed at step $4.b$ in Algorithm 1) is connected.

If some G_k is disconnected, then clearly $\mathcal{N}$ is not movement-connected under power p.

Conversely, if network $\mathcal{N}$ is not movement-connected under power p, then at some instant $t \in [0, 1]$ the induced graph at t under power p is disconnected. If t is not a slicing point, then t is in (t_k, t_{k+1}). Since (t_k, t_{k+1}) is a constant-connectivity time slot, it follows that G_k is disconnected. If t is a slicing point, say t_k, then recall that t_k is either the first or last instant at which some edge is present. In either case, it follows that the edge set of G_k as constructed at step $4.b$ in Algorithm 1 is a subset of the edge set present at t_k. Hence, G_k is disconnected. The theorem follows. ■

Theorem 4.5: Algorithm 1 runs in worst case time $O(n^2(\log(n))^2)$.

Proof: In Algorithm 1, step 2 takes $O(n^2)$ time, since there are $O(n^2)$ equations and each can be solved in time $O(1)$. Obviously, step 3 takes $O(n^2 \log(n))$ time. The primary concern is the running time of step 4. If step 4 is implemented directly as described in Algorithm 1 it requires $O(n^4)$ time, since there are $O(n^2)$ time slots and constructing G_k also takes $O(n^2)$ time. Fortunately, that running time can be reduced by observing that G_k and G_{k+1} for the consecutive time slots (t_k, t_{k+1}) and (t_{k+1}, t_{k+2}) differ only

[5]Note if there are identical slicing points, they are represented by a single point in this list.

by the insertion or deletion of an edge[6]. This allows us to implement step 4 using fully dynamic graph algorithms for connectivity [6], [12]. Such algorithms process a sequence of update and query operations interspersed in any order, where the *update* operations include the insertion and deletion of edges, and the *query* operation is a query about the connectivity of the graph. Using [6], these operations can be implemented in $O((\log(n))^2)$ amortized time per update and $O(\log(n)/\log(\log(n)))$ per query. Thus, step 4 can be implemented in $O(n^2(\log(n))^2)$ worst case time by using $O(n^2)$ update and query operations. Hence, the theorem follows. ■

Using a result of [12] for fully dynamic graph connectivity in which connectivity queries require $O(\log(n)/\log(\log(\log(n))))$ time while updates require $O(\log(n)(\log(\log(n)))^3)$ expected amortized time, it follows that:

Corollary 4.6: Algorithm 1 runs in expected time $O(n^2\log(n)(\log(\log(n)))^3)$.

C. An Algorithm for the Optimization Version of CRMN

In this subsection, we give an algorithm for solving an instance of CRMN. That is, given an instance $\mathcal{N}$ of CRMN, the goal is to find the minimum power p_{min} assigned to all nodes such that $\mathcal{N}$ is movement-connected under p_{min}. We begin with two definitions and two lemmas without proof:

Letting $G = (V, E)$ be the threshold graph at an instant t, then edge $e_m \in E$ is a ***MinMax edge*** at t, if $G' = (V, E')$ is connected, where $E' = \{e_i \in E : |e_i| \leq |e_m|\}$, and $G'' = (V, E'')$ is disconnected, where $E'' = E \setminus \{e_i \in E : |e_i| \geq |e_m|\}$.

Time slot $[t_x, t_y]$ is a *constant-order time slot* if there exists an ordered list E^* of the network edges such that for any instant $t \in [t_x, t_y]$, E^* is an ordered list by length of the edges in the threshold graph constructed from $\mathcal{N}$ at t.

Lemma 4.7: Given a constant-order time slot $[t_x, t_y]$ and an instant $t' \in [t_x, t_y]$, if edge e_m is a MinMax edge at t', then e_m is a MinMax edge at any instant $t'' \in [t_x, t_y]$.

Lemma 4.8: Given moving nodes A (moves from A to A') and B (moves from B to B'), $\text{MAX}(\pi(A, B), \pi(A', B'))$ is the minimum power that can be assigned to both A and B such that nodes A and B are movement-connected.

[6]We assume that there is only one slicing point at t_{k+1}. When there are multiple slicing points at t_{k+1}, the update operations described in this proof are applied to each of those slicing points.

Our algorithm for solving an instance of CRMN works by slicing the unit time interval into constant-order time slots, which is referred to as TIMESLICING. The procedure is very similar to steps 1 - 3 of Algorithm 1 except that using equation $\pi(V_a, V_b)[t] = \pi(V_c, V_d)[t]$ to replace $\pi(V_i, V_j)[t] = p$, where $\pi(V_a, V_b)[t]$ and $\pi(V_c, V_d)[t]$ are non-identical threshold functions. The resulting solutions are called *slicing points*[7], and the edges (V_a, V_b) and (V_c, V_d) are said to define the slicing points. Note that each equation generates at most two slicing points. We collect all $O(n^4)$ distinct slicing points into a sorted list. Adjacent points in this sorted list define a time slot. Here, given a threshold graph $G = (V, E)$ at an instant in such a time slot $[t_x, t_y]$, the ordering of a sorted edge list for E is invariant throughout $[t_x, t_y]$. It follows that each such time slot is a constant-order time slot. The following lemma gives the running time of TIMESLICING. The proof is omitted because of space reasons.

Lemma 4.9: TIMESLICING runs in $O(n^4\log(n))$ time.

With the constant-order time slots in hand, our algorithm (Algorithm 2) is shown in Figure 3.

Like Algorithm 1, we utilize fully dynamic graph algorithms instead of the naive method. The key ideas used to improve the running time are (1) avoiding an explicit construction of threshold graphs, and (2) computing the MinMax edge incrementally. To help achieve these two objectives, we incrementally construct the induced graph $G_k^{p_{mk}}(V)$ where p_{mk} is the threshold value of a MinMax edge for threshold graph G_k.

The implementation is as follows. Assume that for $[t_{k-1}, t_k]$ we have $G_{k-1}^{p_{m(k-1)}}(V)$ and the MinMax edge for G_{k-1}. For $[t_k, t_{k+1}]$, we update these two items. Since every time slot is a constant-order time slot, $G_{k-1}^{p_{m(k-1)}}(V)$ for $[t_{k-1}, t_k]$ differs from $G_k^{p_{mk}}(V)$ of $[t_k, t_{k+1}]$ only in whether or not e_x and e_y are present, where e_x and e_y are the two adjacent edges (in the sorted edge list) that define the slicing point at t_k. Due to space limitations, the detailed case analysis is omitted here. The update operation is assumed to be carried out by a procedure UPDATE which takes as parameters $\mathcal{T}_k$, $e_{m(k-1)}$ and $G_{k-1}^{p_{m(k-1)}}(V)$ and which returns e_{mk} and $G_k^{p_{mk}}(V)$.

Theorem 4.10: Algorithm 2 runs in worst case time $O(n^4(\log(n))^2)$ and expected time $O(n^4\log(n)(\log(\log(n)))^3)$.

[7]Note that these are not the slicing points defined in the prior subsection.

Input: An instance $\mathcal{N}$ of CRMN.

Output: The minimum power p_{min} such that $\mathcal{N}$ is movement-connected under p_{min}.

Steps:

1) $\{\mathcal{T}_k : k \in [1, w]\} \leftarrow$ TIMESLICING($\mathcal{N}$).
2) Let t'_1 be an interior time instant in $\mathcal{T}_1 = [t_1, t_2]$, and construct a threshold graph $G_1 = (V, E)$ at t'_1.
3) $p_{m1} \leftarrow$ MINMAXGRAPH(G_1).
   Construct $G_1^{p_{m1}}(V)$, the graph induced from G_1 by p_{m1}. Select from G_1 an edge $e_{m1} = (V_{m1}, V_{m'1})$ such that $\pi(V_{m1}, V_{m'1})[t'_1] = p_{m1}$. Redefine $p_{m1} \leftarrow \pi(V_{m1}, V_{m'1})[t_1, t_2]$.
4) $P \leftarrow \{p_{m1}\}$.
5) For k from 2 to w do
   a) UPDATE($\mathcal{T}_k$, $e_{m(k-1)}$, $G_{k-1}^{p_{m(k-1)}}(V)$), which returns e_{mk} and $G_k^{p_{mk}}(V)$.
   b) $p_{mk} \leftarrow \pi(V_{mk}, V_{m'k})[t_k, t_{k+1}]$, where $e_{mk} = (V_{mk}, V_{m'k})$.
   c) $P \leftarrow P \cup \{p_{mk}\}$.
6) $p_{min} \leftarrow$ the largest value in P.
7) Return p_{min}.

Fig. 3. Algorithm 2 — SOLUTIONTOCRMN

Proof: Step 1 takes time $O(n^4 \log(n))$. Step 3 takes $O(n^2 \log(n))$. Step 5 utilizes fully dynamic graph algorithms for connectivity to construct the induced graphs $G_k^{p_{mk}}(V)$ incrementally. Since there are totally $O(n^4)$ constant-order time slots, it follows that for step 5 the worst case running time is $O(n^4(\log(n))^2)$, and that the expected running time is $O(n^4 \log(n)(\log(\log(n)))^3)$. Step 6 requires $O(n^4)$ time. Thus, the theorem follows. ■

Since Algorithm 2 finds the MinMax edges for all constant-order time slots, we have:

Theorem 4.11: The value p_{min} returned by Algorithm 2 is such that network $\mathcal{N}$ is movement-connected under power p_{min}, and that p_{min} is the minimum such power.

V. CONCLUSIONS

In this paper, we provided two polynomial algorithms for topology control in constant rate mobile networks. Our algorithms for the decision and optimization versions of the problem have running times of $O(n^2(\log(n))^2)$ and $O(n^4(\log(n))^2)$ respectively.

Since these are the first theoretical results for topology control incorporating mobility, there are many open problems. In regard to the optimization problem handled in this paper, considering a model similar to CRMN in which the moving rate is not constant, the topology control problems specified under that model are open. The problem of minimizing the total power is also of interest.

Disclaimer: The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government.

REFERENCES

[1] D. M. Blough, M. Leoncini, G. Resta, and P. Santi. On the symmetric range assignment problem in wireless ad hoc networks. *Proc. 2nd IFIP International Conference on Theoretical Computer Science*, 223:71–82, 2002.

[2] G. Calinescu, I. Mandoiu, and A. Zelikovsky. Symmetric connectivity with minimum power consumption in radio networks. *Proc. 2nd IFIP International Conference on Theoretical Computer Science*, 223:119–130, 2002.

[3] G. Calinescu and P.-J. Wan. Range assignment for high connectivity in wireless ad hoc networks. *Proc. International Conference on Ad hoc and Wireless Networks*, pages 235–246, 2003.

[4] W. Chen and N. Huang. The strongly connecting problem on multihop packet radio networks. *IEEE Trans. Communication*, 37(3):293–295, March 1989.

[5] A. E. F. Clementi, P. Penna, and R. Silvestri. The power range assignment problem in packet radio networks in the plane. *Proc. 17th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2000)*, pages 651–660, February 2000.

[6] J. Holm, K. de Lichtenberg, and M. Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *Journal of the ACM*, 48(4):723–760, July 2001.

[7] L. M. Kirousis, E. Kranakis, D. Krizanc, and A. Pelc. Power consumption in packet radio networks. *Proc. 14th Annual Symposium on Theoretical Aspects of Computer Science*, 1200:363–374, February 1997.

[8] S. O. Krumke, R. Liu, E. L. Lloyd, M. V. Marathe, R. Ramanathan, and S. S. Ravi. Topology control problems under symmetric and asymmetric power thresholds. *Proc. International Conference on Ad hoc and Wireless Networks*, 2865:187–198, October 2003.

[9] E. L. Lloyd, R. Liu, M. V. Marathe, R. Ramanathan, and S. S. Ravi. Algorithmic aspects of topology control problems for ad hoc networks. *Mobile Networks and Applications*, 10(1-2):19–34, February-April 2005. (Earlier version appeared in MobiHoc'02).

[10] R. Ramanathan and R. Rosales-Hain. Topology control of multihop wireless networks using transmit power adjustment. *Proc. IEEE INFOCOM 2000*, pages 404–413, March 2000.

[11] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1996.

[12] M. Thorup. Near-optimal fully-dynamic graph connectivity. *Proc. Thirty-second Annual ACM Symposium on Theory of Computing*, pages 343–350, 2000.

[13] L. Zhao and E. L. Lloyd. Distributed topology control for stationary and mobile ad hoc networks. *Proc. IEEE MASS'06*, October 2006.

[14] L. Zhao, E. L. Lloyd, and S. S. Ravi. Topology control for simple mobile networks. *Proc. IEEE GLOBECOM'06-WASNet*, November 2006.