Can Bluetooth Succeed as a Large-Scale Ad Hoc Networking Technology?

Evangelos Vergetis, *Student Member, IEEE,* Roch Guérin, *Fellow, IEEE,* Saswati Sarkar, *Member, IEEE* and Jacob Rank

***Abstract*— We investigate issues that Bluetooth may face in evolving from a simple wire replacement to a large-scale ad hoc networking technology. We do so by examining the efficacy of Bluetooth in establishing a connected topology, which is a basic requirement of any networking technology. We demonstrate that Bluetooth experiences some fundamental algorithmic challenges in accomplishing this seemingly simple task. Specifically, deciding whether there exists at least one connected topology that satisfies the Bluetooth constraints is NP-hard. Several implementation problems also arise due to the internal structure of the Bluetooth protocol stack. All these together degrade the performance of the network, or increase the complexity of operation. Given the availability of efficient substitute technologies, Bluetooth's use may end-up being limited to small ad hoc networks.**

***Index Terms*— Wireless ad hoc networks, Bluetooth, topology formation, scatternets, performance.**

I. INTRODUCTION

BLUETOOTH, a short-range low-power communication protocol, was initially envisioned as a wire replacement solution. Bluetooth uses a design paradigm that is fundamentally different from that of competing technologies like IEEE 802.11. This motivates an examination of the extent to which Bluetooth can be used in a networking context, and in particular large ad hoc networks. IEEE 802.11 is a simple distributed protocol, where a node can transmit whenever it senses a free channel. The resulting collisions however waste bandwidth and power. On the other hand, Bluetooth is partly distributed and partly centralized. It has a hierarchical organization where the nodes are organized in groups denoted as piconets. In each group, a master node controls the transmissions of other nodes. This local control eliminates collisions and is therefore expected to offer high throughput and low power consumption. We however demonstrate that this organization introduces significant complexity in establishing a connected topology in large and dynamic ad hoc networks. Given Bluetooth's difficulty in fulfilling the simplest of all networking tasks, that of attaining connectivity, its use is likely to be limited to small ad hoc networks.

The difference between 802.11 and Bluetooth is analogous to that between Ethernet and Token Ring. Token Ring offered a higher throughput but was more complex. The increase in transmission speeds more than compensated for Ethernet's throughput inefficiency. Although bandwidth constraints are greater in the wireless setting, we believe that the choice between 802.11 and Bluetooth will also be guided by simplicity of operation. This is because operational complexity seriously undermines the operation of large dynamic ad hoc networks as nodes have limited processing power and computations cannot be offloaded to an infrastructure.

This work was supported by NSF grants ITR-0085930, ANI-0106984, ANI-9902943 and ANI-9906855, and by grant NCR-0238340.

We focus on the basic aspect of topology formation as it illustrates the problems that Bluetooth encounters when used as a networking technology. First, we investigate this problem from an algorithmic perspective to gain a basic understanding of its fundamental complexity. In Section II, we describe the technical challenges related to topology formation in ad hoc networks using Bluetooth. Although Bluetooth nodes are functionally equivalent, communications proceed according to a "master-slave" model, with a constraint on the number of slaves that a master can support. This introduces a degree constraint on the resulting topology graph. Furthermore, the topology formation algorithms need to determine which nodes will be masters and appropriately assign slaves to those masters. In Section III, we show that decisions have a significant impact on connectivity. Specifically, deciding whether there exists at least one connected topology that satisfies the degree constraint of Bluetooth is NP-hard. This explains why forming a Bluetooth topology in a short time while satisfying all the Bluetooth constraints has been a topic of extensive research for several years.

Next, we explore topology formation algorithms of different complexity (Sections IV and V). We present a polynomial complexity topology formation algorithm that, under some simplifying assumptions, yields a connected topology whenever one such exists. We then present several heuristics that produce good results when these simplifying assumptions do not hold, including an efficient and natively distributed algorithm. In Section VI, we develop a detailed emulator of the Bluetooth stack, and use it to evaluate the performance of our most promising solution. Our investigation reveals that in spite of several simplifying assumptions that made for a "best case" evaluation, performance, and in particular the time it takes to form a stable connected topology, is poor and in some cases (large networks) unacceptable. We confirm that this disappointing showing is not specific to our algorithm through a comprehensive comparison with previously proposed algorithms. This comparison helps highlight key properties and assumptions that are important when evaluating Bluetooth's performance, and leads us to conclude that the performance-minded design choices that were behind Bluetooth's specifications make it difficult if not impossible for it to be successful in large-scale ad hoc networks. We examine some related works in Section VII, and conclude in Section VIII.

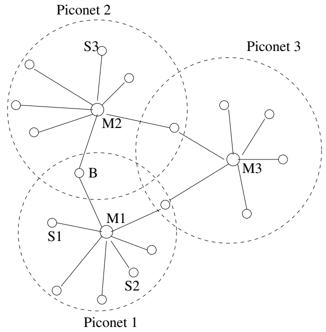

Fig. 1. An example of a Bluetooth topology is illustrated. The nodes are organized into 3 piconets. The masters of these piconets are M_1, M_2, M_3 respectively. The remaining nodes are slave or bridge nodes. Slave nodes S_1 and S_2 can communicate via master M_1. Nodes S_1 and S_3 can communicate via master M_1, bridge B and master M_2.

II. CHALLENGES AND OBJECTIVES IN BLUETOOTH TOPOLOGY FORMATION

We first describe the basic features of the Bluetooth technology that are relevant to topology formation. Bluetooth nodes are organized in small groups called *piconets*. Every piconet has one "master" node and up to 7 "slave" nodes. Refer to Figure 1 for a sample organization. Slaves in a piconet do not directly communicate with each other, but rely on the master as a transit node. Communication between nodes in different piconets relies on bridge nodes that belong to multiple piconets. A bridge node can only be active in one of the piconets it is connected to at a time. Bluetooth allows different activity states for nodes: active, idle, parked, and sniffing. However, data exchange takes place between two nodes only when both are active, and nodes periodically change their activity state. This combination of flexibility and constraints on which Bluetooth is based raises a number of questions and challenges. We list below those that are most relevant to topology formation.

1) How should nodes select their role (master or slave)?
2) Which piconet(s) should a (slave) node join?
3) How many slaves should a master accept (below the specified maximum of seven)?
4) How many piconets should a bridge node belong to?
5) Should a master serve as a slave in other piconets?

When Bluetooth is used as a wire-replacement technology, the above questions have trivial answers. There is only one piconet and one obvious choice for the master, *e.g.*, the computer rather than the keyboard, or the cell phone rather than the head set. The master accepts new slaves as long as the maximum number of 7 has not been reached. In ad hoc networks consisting of a small number of piconets, answering the above questions may not incur significant additional complexity. Bluetooth is ideally suited for such simple scenarios. Power consumption is low, and resources can be allocated more efficiently due to the masters' local control.

In a large distributed environment, however, appropriately answering the above questions introduces significant added complexity that can affect network connectivity[1]. For example, the answer to question 2 depends on how busy the node is, how well connected the topology is, whether the node can play a dual role, *etc*. Also, answers to the above questions can seriously affect a number of network attributes, *e.g.*, throughput. For example, consider questions 4 and 5. Since a bridge node can only be active in one piconet at a time, the greater the number of piconets to which a node belongs, the poorer the data rate it can provide between them. Thus, it is desirable for a bridge node to be involved in as small a number of piconets as possible, while preserving connectivity. The impact on throughput is compounded when a bridge node is a master in one piconet. This is because all slaves in the piconet are in a communication blackout, when the master is active in other piconets. Thus, it is desirable for a master not to be a slave in other piconets, provided that this does not substantially complicate forming and modifying topologies.

Since nodes select their roles based on local information, efficient algorithms will most likely allow nodes to modify their earlier decisions, *e.g.*, by allowing some slaves to leave one piconet and join another piconet, or by allowing nodes to change their role from slave to master or vice-versa. Identifying when and how to allow such changes while preserving the degree constraint or improving connectivity is a challenging task, especially when assuming distributed decisions.

There are several other difficulties above and beyond the development of "clever" topology formation algorithms that introduce additional challenges when using Bluetooth in large ad hoc networks. First, during topology formation, nodes might need to exchange information with each other, and this means establishing a connection where one node will act as the master and the other as a slave. This is easy when neither node belongs to a piconet, but introduces significant complexity when either one or both nodes are engaged in some piconet. For example, a slave and a master can communicate only after they negotiate a time window, called a "sniff" window. In the sniff period, a slave must communicate with or listen to its master. If the slave is not there during this time, then the master terminates the connection (see [1], vol. 2, pp. 163–164). We demonstrate next through a simple example that determining sniff windows can introduce significant complexity when nodes try to establish a new connection.

Suppose slave S in piconet P_1 with master M_1 is trying to join piconet P_2 with master M_2. Let M_2 have several other slaves, and let the only available sniff window overlap with the sniff window that S has already established with M_1. Now, either S and M_1 have to negotiate a different sniff window, or M_2 has to move the sniff window of one of its existing slaves. This incurs additional complexity. Now, if neither M_1 nor the slaves of M_2 have any other available sniff windows, then the changes in sniff windows can propagate over the whole network! Furthermore, if the master of a piconet is also the slave in some other piconet, determining sniff windows becomes increasingly complex. Thus, topology formation becomes a stumbling block even when we do not

[1]Note that these issues do not arise in 802.11, which highlights the trade-off associated with different design criteria and their different impact in different environments.

consider mobility, or nodes periodically turning their power on or off. Note again that the determination of sniff windows is not an issue in a wire replacement setting, and less likely to be a problem when the number of piconets is small.

The inquiry and page modes used in Bluetooth to allow nodes to discover each other pose yet another challenge. Suppose two nodes A and B receive an "inquiry response" message from node C at roughly the same time. Then A and B will both page C repeatedly and their "page" messages will collide. Although this may be solved via randomization, it can introduce a delay in the node discovery process and in the formation of connections.

In conclusion and as we quantify later, by focusing extensively on controlling the use of resources, Bluetooth ends-up violating a basic design principle in networking - *simplicity of operation* - which is critical in large distributed systems. The complexity it introduces in providing connectivity more than offsets any resource optimization capabilities it may afford.

III. NETWORK MODEL AND PROBLEM COMPLEXITY

We formulate next a mathematical model for the system's objectives and constraints. There can be two types of links between any two nodes. One is a *physical* (layer) link that exists between any pair of nodes that are in communication range of each other. The other is a *logical* Bluetooth link that exists if the Bluetooth topology establishes an actual communication link between the two nodes. The physical topology graph is determined by the positions and the transmission radii of the nodes, while the logical topology graph is generated by the topology formation algorithm.

The logical topology graph must have certain properties. According to the Bluetooth specification, vertices that will be assigned the role of a master can have a maximum degree[2] of 7. For the vertices that will serve as slaves, it is desirable that their degree be kept as small as possible. Regular slave nodes have a degree of only 1, but bridge nodes have a degree equal to the number of piconets they participate in. Since a bridge node with a degree more than 7 would provide poor data rate between the piconets it connects, we assume that the degree constraint of 7 applies to the bridge (slave) nodes as well. We choose the number 7 as this will give the same degree constraint for master, slave and bridge nodes. The logical topology graph is bipartite[3] when the desirable condition that a master is not a slave in another piconet holds.

Connectivity is then deemed feasible if there exists a connected[4] sub-graph of the physical topology graph which satisfies the degree constraint (maximum degree of 7). If connectivity is feasible, then we want to construct a connected logical topology graph that satisfies the desired degree constraint. Otherwise, any logical topology graph will consist of "islands" or components[5], and we then seek to minimize the number of components in the logical topology graph.

Note that a connected logical subgraph exists if and only if the physical topology graph has a spanning tree[6] that satisfies the degree constraint of a logical topology graph. This is because a spanning tree of any graph is connected and bipartite [2]. In a spanning tree, the partition that has a maximum degree less than or equal to 7 is chosen as the master set, while the other with a potentially lower maximum degree forms the slave/bridge set.

Let the degree of a spanning tree be the maximum degree of its vertices. A spanning tree with degree less than or equal to 7 exists if and only if the maximum degree of a spanning tree in a graph is upper bounded by 7, and deciding this is NP-hard [3]. Thus, *deciding whether connectivity is feasible and constructing a connected logical topology graph which satisfies the desired degree constraint is NP-hard.*

Nevertheless, polynomial time algorithms are available in certain practical scenarios, where additional constraints are imposed on the underlying network graph (Section IV-B). Furthermore, we show how those polynomial time algorithms can be extended to provide efficient heuristics in general scenarios (Section IV-C). Many of these algorithms are centralized, but the basic intuition behind them motivates a fully distributed and dynamic approximation (Sections V and VI).

IV. EXPLORING THE RANGE OF POSSIBLE SOLUTIONS

We explore the range of algorithms that are capable of forming the desired topologies. We start with a naïve algorithm, continue with algorithms for nodes on a plane, and finally present algorithms that operate in 3-dimensional space.

A. A naïve algorithm for topology formation

We first consider a naïve algorithm where a node randomly chooses its role as either master or slave [4]. Then, if it is a slave, it accepts every connection request up to the limit of 7, and if it is a master, it pages slave nodes until it forms 7 connections. Here, using the emulator described in Section VI, we quantify how often this algorithm generates a disconnected topology, even when a connected one exists.

When 100 nodes are uniformly placed on a square of size 1 unit and the transmission radius of each node is 0.25 units, the algorithm forms a connected topology with probability 0.39. However, a connected topology exists with probability 0.86. Thus the algorithm fails to form a connected topology about 55% of the time. We simulated various other combinations of numbers of nodes (10, 25, 50 and 100) and transmission radii (0.1, 0.17, 0.25, 0.32, 0.4, 0.5, 0.6 and 0.75 units). The algorithm failed to construct a connected topology in more than 20% of the cases. Moreover, in many cases this failure probability is much higher than 0.5 (see Figure 2 for the 50 node case). These results motivate us to develop "smarter" topology formation algorithms.

[2]The degree of a vertex is the number of edges originating from the vertex.

[3]A bipartite graph is one where the vertex set can be partitioned in two sets such that there is no edge connecting two vertices in the same set.

[4]A graph is connected if there is a path between any two nodes.

[5]A component of a graph is a connected sub-graph that cannot be expanded any further while retaining connectivity.

[6]A spanning tree is a connected subgraph which does not have a cycle and spans all vertices in the graph.

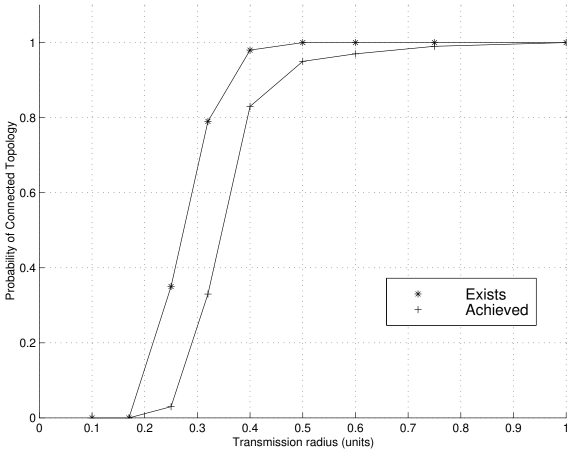

Fig. 2. The line denoted by * corresponds the probability that a connected topology exists. The line denoted by + corresponds to the probability that a connected topology is actually achieved by the naïve algorithm.

B. Topology formation algorithms for nodes with identical power levels on a plane

We now approach the connectivity problem under certain simplifying assumptions, which we describe and justify next. First, we assume that nodes constitute points on a plane. This assumption is justified in several ground-based civilian and military communication networks where the transceivers are at similar heights and there is no air to ground communication. Second, we assume that nodes have the same transmission range, d. This happens if the propagation conditions are similar throughout the network and nodes have the same maximum transmission power limitation and similar reception capabilities. Now, a physical link exists between any two nodes if and only if their Euclidean distance is upper bounded by d.

Under these two assumptions, the connectivity problem becomes of polynomial complexity. The following Lemma provides the cornerstone for designing a simple polynomial complexity, distributed algorithm that generates a connected logical topology whenever connectivity is feasible.

Lemma 1: Connectivity is feasible if and only if the physical topology graph is connected. A minimum weighted spanning tree (MST) in the physical topology graph, with the weight of an edge equaling the Euclidean distance between the nodes, is a connected logical topology graph that satisfies the constraints.

We first present the following result obtained by Monma *et. al.* [5], which we will use in proving this lemma.

Proposition 1: Consider a complete[7] graph with nodes corresponding to points on a plane and the weight of the edges being the Euclidean distance between them. Any MST in such a graph has degree less than or equal to 6.

The intuition behind this proposition is provided in Figure 3.

Proof of Lemma 1: Clearly, a necessary condition for connectivity to be feasible is that the physical topology graph be connected. We will show that this condition is sufficient as well. Assume that the physical topology graph is connected.

[7]A graph is complete if it has edges between any pair of vertices.

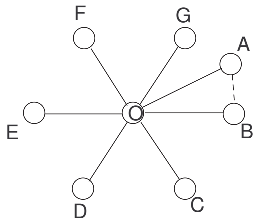

Fig. 3. We explain intuitively why in a complete graph with edge weights equaling the Euclidean distance between the corresponding vertices, the degree of an MST is no more than 6. Consider a complete graph with vertices O, A,. . . , G. Assume that vertex O in an MST has degree 7. Let its neighbors in the MST be $\{A,\ldots,G\}$. Note that the Euclidean distance between nodes A, B is less than the distance between (O, A) or (O, B). Thus, the MST will include the edge (A, B) rather than (O, A) or (O, B).

Consider a new graph formed by adding edges between all pairs of nodes in the physical topology graph. This graph is referred to as the completion of the physical connectivity graph. The weights of the new edges equal the Euclidean distance between the nodes. The physical topology graph is a sub-graph of this completion graph consisting of all edges of the completion graph with weight less than d. From Proposition 1, the degree of any MST in the completion graph is less than or equal to 6. Any MST in the physical topology graph is also an MST in the completion graph. This follows from the following facts: (a) all edges in the completion graph with weight less than d belong to the physical topology graph and (b) the physical topology graph is connected. Thus, any MST in the physical topology graph has degree less than or equal to 6. Therefore, such an MST satisfies the degree constraint, and is a bipartite graph by virtue of being a tree. Hence, any MST in the physical topology graph is a connected logical topology graph which satisfies the required constraints. □

Next, we consider the case when connectivity is not feasible. This happens only when the physical topology graph is disconnected. The objective in this case is to construct a logical topology graph with the minimum number of components. The following lemma gives the basis for the procedure we follow.

Lemma 2: The sub-graph of the physical topology graph consisting of the MSTs in each component of the physical topology graph is a logical topology graph with the minimum number of components.

Proof of Lemma 2: Since a logical topology graph is a sub-graph of the physical topology graph, the former has at least as many components as the latter. Thus the logical topology graph has at least as many components as the sub-graph consisting of MSTs in each component of the physical topology graph. It is thus sufficient to show that this sub-graph satisfies the degree constraint of a logical topology graph. Now, consider each component of the physical topology graph separately. Since each component is connected, then by Lemma 1, the MST in it satisfies the degree constraint of a logical topology graph. Thus, a collection of such disjoint MSTs satisfies the degree constraint of a logical topology graph. □

Lemmas 1 and 2 show that constructing an MST in the *physical topology graph will provide a logical topology graph which (a) is connected if connectivity is feasible and (b) consists of the minimum number of components if connectivity is not feasible.*

Let the physical topology graph have E links and V nodes. Then, an MST can be constructed in a centralized manner with time complexity $O(E \log V)$ [6]. A distributed construction has time complexity $O(V \log V)$ and exchanges $O(V \log V + E)$ messages [7].

The design of a logical topology is not complete without assigning master/slave/bridge roles to the nodes. Since an MST is a bipartite graph, and all nodes have degree less than or equal to 6, any one partition can be selected as the master set, and the other partition as the slave set. Since we would like to minimize the degree of the bridge nodes, the partition with the smallest degree can be chosen as the slave set.

The MST-based algorithm has, however, some disadvantages. First, if all nodes have low degrees, which is typically going to be the case in an MST, then the end-to-end path between certain nodes may be long, and this causes large end-to-end delay. Thus, the piconet size can be a design parameter. We need to tune the degree of masters to a certain desired value, and the degree of bridges to a different, possibly lower value. The MST algorithm does not allow us to selectively decrease the degrees of the bridges, once the universal degree constraint of 7 is satisfied. We next propose algorithms that can accommodate such a discriminatory treatment, and more importantly, are capable of generating connected topologies when the simplifying assumptions of this section do not hold.

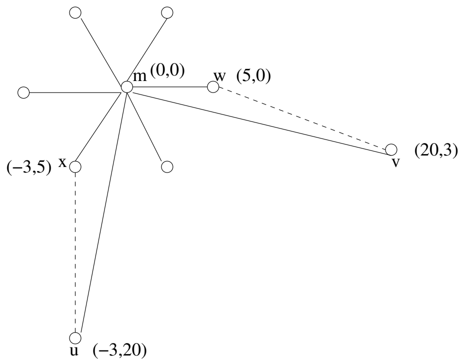

Fig. 4. The figure shows an example where an MST in a physical topology graph has a degree of 8. Here, nodes are on a 2-dimensional plane and nodes m, v and u have transmission ranges 100 meters, while all other nodes have transmission range 10 meters. The solid lines show the MST. Node m has a degree 8.

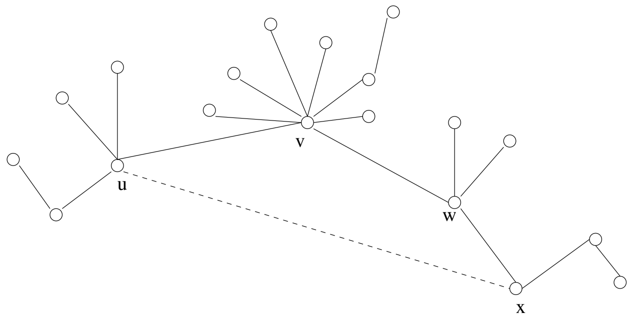

Fig. 5. We explain the operation of the MDST algorithm in this figure. Let the MDST algorithm start with the spanning tree shown in the figure. Node v has degree 8 while all other nodes have degree less than 5. Node v is marked as "bad," and all other nodes are marked as "good" (since their degrees are less than $d - 1 = 8 - 1 = 7$). Now the algorithm considers the cycle generated when edge (u,x) is added to the tree. The degree of node v can now be reduced by including edge (u,x) in the tree and deleting one of the edges (u,v) or (v,w).

C. Topology formation algorithms for networks with nodes in 3-dimensional space

We assume that nodes are located in 3-dimensional space and can have different communication ranges. Robins *et al.* [8] showed that in a 3-dimensional scenario the degree of an MST can be as large as 14, even when all nodes have the same communication range. As a result, enabling an MST-based algorithm to find a connected topology in a 3-dimensional space requires that we relax Bluetooth's constraint to allow up to 15 (instead of 7) active slaves in a piconet. Similarly, when communication ranges are different, even in the 2-dimensional case, the degree of an MST can exceed 7 (Figure 4). Hence, the problem needs to be investigated in the framework of a *minimum degree* spanning tree, which is an NP-hard problem (Section III). We therefore investigate heuristics and approximation algorithms.

We next present a topology design procedure that provides a "knob" for separately tuning the degrees of masters and bridges. This is based on an approximation algorithm (Minimum Degree Spanning Tree "MDST") guaranteed to generate a spanning tree with degree at most one more than the minimum possible value in any arbitrary graph (see [9], pp. 272-276). Thus, MDST generates a connected logical topology in "most" of the instances in which connectivity is feasible. Specifically, the only exception occurs when MDST generates a spanning tree of degree 8 and there exists a connected logical topology with degree 7. MDST starts with any spanning tree, and replaces edges from vertices of high degree with those from vertices of low degree. Refer to Figure 5 for an illustrative example. MDST runs in polynomial complexity $O\left(VE \log V\right)$[8].

We now discuss how to extend MDST to separately control the degrees of the masters and bridges. The goal is to first satisfy a degree constraint of, say, p for all vertices (where p is the desired maximum number of slaves in a piconet and $p \leq 7$), and then reduce the maximum degree of the bridges to a desired value, k. For this, we use MDST to decrease the degree of a spanning tree generated by Breadth First Search (BFS) to p. Now, edges originating from slaves with degree greater than k are removed from the spanning tree, and replaced by those originating from the masters with degree less than p and slaves with degrees less than $k - 1$. The pseudocode for this extension, which is referred to as "E-MDST" (Extended-MDST), follows.

1) Execute MDST on a spanning tree generated by BFS.

[8]More precisely, the run time is $O\left(VE\alpha(V, E) \log V\right)$, where α is the inverse of Ackermann's function and grows slowly. For all practical purposes, $\alpha(V, E)$ can be treated as a constant [9].

2) Let MDST output a spanning tree T with degree p.
3) The partition with a larger maximum degree is the master set and the other partition is the slave/bridge set. Consider the physical topology graph G' with edges between the master and slave sets only.
4) Terminate if the maximum degree d_s in T of the slave set is less than or equal to k.
5) Mark all master nodes of degree p and all slave nodes of degree d_s and $d_s - 1$ as "bad." A vertex is marked "good" if it is in the "forest" $F = T \setminus \{\text{bad vertices}\}$.
6) While there exists an edge e of G' that connects two different components of F.
   a) Consider the cycle C generated by spanning tree T together with e.
   b) If C has a slave node w of degree d_s, then denote the edge in C incident on w as l, update T by $T \leftarrow T \setminus \{l\} \cup \{e\}$, and go to step (4).
   c) If C does not have any slave node w of degree d_s then mark all "bad" vertices in C as good. Update F by combining the components along C and these newly marked vertices into a single component.
7) Output T.

We tested MST, MDST and E-MDST in networks with nodes whose x and y coordinates are uniformly distributed in a square of size 1 unit and z coordinates uniformly distributed between 0 and 0.3 units. We also consider "clustered networks" where the z coordinates of the nodes are selected as above, but the x and y coordinates are clustered. Three square clusters of size 0.4 each are placed randomly in a square of size 1. A node may belong to one of the three clusters, or it may not belong to any cluster. These four events are equi-probable. If a node belongs to a cluster then its x and y coordinates are uniformly distributed in the corresponding square, otherwise these are uniformly distributed in the original square of size 1 unit. For each of these two types of node distributions, we evaluate the performance of the algorithms for different number of nodes (25, 50, 100) and two different transmission radii (0.4 and 0.6 units), averaging the results over 100 runs. In all scenarios node degrees remain well below 7. Table I shows the results with transmission radius of 0.4 units. The average degree of the masters (M_a) indicate that E-MDST achieves its objective of generating a "bushier" topology, while at the same time attaining a small average degree for the bridges (around 2.7). The results remain similar in the 2-dimensional case and for other node distributions and transmission radii in the 3-dimensional case .

We conclude that all algorithms (MST, MDST and E-MDST) easily achieve the degree bound imposed by Bluetooth, even in the 3-dimensional case for which MST could possibly yield a degree larger than 7. Hence, the added complexity of MDST over MST does not appear warranted. However, when comparing MST and E-MDST, we see that the latter yields much more compact trees. This motivates considering E-MDST, despite its greater complexity, given that long trees can significantly degrade the network's performance [4].

V. TOWARDS DISTRIBUTED AND DYNAMIC ALGORITHMS

In this section, we first illustrate how an MST-based algorithm can be extended to operate in a distributed and dynamic setting. Since this extension is complex, even for an algorithm as simple as MST, we then introduce an algorithm that is inherently distributed and provides similar albeit somewhat weaker analytical guarantees.

A. Distributizing an MST-based algorithm

An MST can be constructed by distributed computation at the nodes. Gallager *et al.* [7] show how Prim's algorithm (see [6], p.505) for constructing an MST can be distributized. A node only needs to know an ordering of the weights of its incident edges. In the Bluetooth setting, a node can acquire this knowledge by measuring the signal strength of the synchronization messages sent by its neighbors. If all nodes transmit these messages at the same power level, the signal will be stronger for a neighbor that is closer.

The logical topology needs to be constantly updated due to changes in the physical topology. These changes occur because nodes move and new nodes join and existing nodes leave the system. The spanning tree needs to be updated in response to these topology alterations. See [10], [11] for efficient algorithms for the dynamic update of MSTs.

The complexity of a distributed and dynamic version of the MST algorithm can however be high. In the distributed implementation, nodes are initially singletons, and they gradually merge to form fragments which again merge in order to finally yield an MST. The nodes need to maintain and broadcast a fragment ID, as well as certain information about their outgoing edges in order to decide in a distributed manner which edges to add next [7] [10]. Sniff windows must be established and continuously updated for enabling this exchange of information. But as discussed in Section II, this is a complicated task. Moreover, because of the distributed operation, some nodes will be assigned dual roles. Consider for example two fragments F_1 and F_2 that are trying to merge by forming a link between nodes A (which belongs to F_1) and B (which belongs to F_2). If both A and B are masters (or slaves) in their piconets, then forming the link AB means that one of the two nodes will have to assume a dual role, or invoke a complex role switching operation for all the nodes in one of the two fragments.

All these complications motivate the consideration of a simpler distributed algorithm that provides weaker analytical guarantees than an MST, but may offer a better trade-off between performance and complexity.

B. A fully distributed and dynamic algorithm

We now describe a fully distributed and dynamic algorithm, that results in a topology known as the relative neighborhood graph (RNG) in computational geometry [12]. We refer to this algorithm as the RNG algorithm. RNG adds links as and when they are discovered. Let $|AB|$ denote the Euclidean distance between nodes A and B. RNG adds a link between two nodes

N	25				50				100			
	M_a	M_m	B_a	B_m	M_a	M_m	B_a	B_m	M_a	M_m	B_a	B_m
MST	2.2	3	2.3	3	2.3	3	2.1	3	2.4	4	2.3	3
MDST	1.9	2	2	2	2	2	2.1	3	2	2	2.1	3
E-MDST	5.9	7	2.7	3	6.1	7	2.4	3	6.1	7	2.7	3

TABLE I

DEGREE STATISTICS FOR THE THREE PROPOSED ALGORITHMS. N IS THE NUMBER OF NODES; M_a IS THE AVERAGE DEGREE OF MASTERS; M_m IS THE MAXIMUM DEGREE OF MASTERS; B_a IS THE AVERAGE DEGREE OF BRIDGES; B_m IS THE MAXIMUM DEGREE OF BRIDGES.

N	E	D_1	D_2	D_3	D_4	D_5	M_a	B_a	M/S	$D_{M/S}$
100	117.3	9.1	50.2	37.8	2.9	0.002	2.4	2.5	16.9	2.6
500	616.4	24.8	236.5	219.7	18.9	0.02	2.5	2.6	93.7	2.7
1000	1246.9	41.2	464.1	454.4	40.2	0.05	2.5	2.6	192.2	2.7

TABLE II

EVALUATION OF THE RNG ALGORITHM IN A 3-D CLUSTERED TOPOLOGY. N IS THE NUMBER OF NODES; E IS THE NUMBER OF EDGES IN THE RESULTING TOPOLOGY; D_i IS THE AVERAGE NUMBER OF NODES WITH DEGREE i; M_a IS THE AVERAGE DEGREE OF MASTERS; B_a IS THE AVERAGE DEGREE OF BRIDGES; M/S IS THE NUMBER OF NODES WITH A DUAL ROLE; $D_{M/S}$ IS THE AVERAGE DEGREE OF DUAL ROLE NODES.

A and B in the logical topology if and only if A and B are in each others transmission range and $|AB| \leq \max(|BC|, |AC|)$ for any other node C which is in A's and B's transmission ranges ("RNG rule"). After RNG has added AB, if a node C that violates the above condition is discovered, then RNG deletes AB. Figure 6 illustrates this rule.

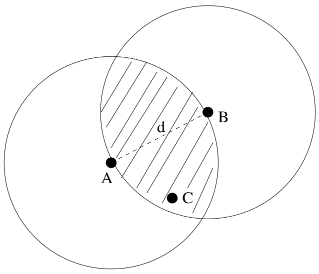

Fig. 6. The 2 circles in the figure have radii $|AB|$ and centers A and B respectively. RNG would add link AB, if there is no other node in the intersection of the circles (shaded area). Link AB is not added in this case as node C is in the shaded area.

We assume that each node knows its neighbors in the physical topology graph (G). A node also knows an ordering among the Euclidean distances between its neighbors from power measurements and subsequent information exchange with its neighbors. Observe that the addition and/or deletion of a link do not affect any other link additions or deletions, and depend only on local information. Hence, there is no need to broadcast any information throughout the graph. Thus, RNG exchanges fewer messages and is simpler than the distributed MST algorithm.

Lemma 3: RNG generates a topology that is a superset of the MST.

Proof of Lemma 3: For simplicity, we assume that there exists a unique MST. Let there exist an edge AB that is chosen by the MST algorithm but not by the RNG algorithm. Thus there exists a node C such that $|AC|$ and $|BC|$ are less than or equal to $|AB|$. Note that in the MST at least one of the paths, A to C, or B to C must use the link AB (else there is a cycle). Let the path between B and C use this link. Thus, edge BC does not exist (else there is a cycle). Thus, add edge BC to the MST. The earlier path from C to B forms a cycle with edge BC, and this cycle contains edge AB (as AB is in the path between C and B by assumption). Remove edge AB from the MST, to construct a spanning tree whose weight is not more than that of the earlier MST (since $|AB| \geq |BC|$). □

Observe that the above proof holds for any link weights $|AB|$ (not just for Euclidean distances). Thus, the lemma holds for all graphs. The following corollary follows directly from Lemma 3.

Corollary 1: RNG generates a connected logical topology.

Unlike in an MST, there may be multiple paths between any two nodes in an RNG. Thus, an RNG has better connectivity than an MST.

Now assume that nodes are on a plane and have equal transmission radii. Then, we prove that RNG satisfies the degree constraint of Bluetooth in most cases.

Lemma 4: Let all nodes be on a plane and have equal transmission radii. Let different pairs of nodes have distinct Euclidean distances. Then, the degree of any node in the logical topology generated by RNG is at most 6.

Proof of Lemma 4: Let the degree of a node O in the logical topology generated by RNG exceed 6 (refer to Figure 3 for an illustration). Then, there exist at least two nodes A, B such that RNG selects edges OA and OB, and the angle $A\hat{O}B$ (θ) is less than or equal to $2\pi/6 = \pi/3$. Without loss of generality, let $r_2 = |OB| < |OA| = r_1$. Note that $|OB| \neq |OA|$ by assumption. Let $p = r_1 - r_2$, where by our assumptions $0 < p < r_1$. From standard geometry we have that $|AB| = \sqrt{r_1^2 + r_2^2 - 2r_1r_2\cos\theta}$. Since $\theta \leq \pi/3$, we have that $\cos\theta \geq 1/2$, and using the relation $r_2 = r_1 - p$, we get $|AB| \leq \sqrt{r_1^2 - pr_2} < r_1$. Thus, $|OA| > \max(|AB|, |OB|)$. Since RNG selects edge OA and all nodes have equal transmission ranges, B is in both A's and in O's transmission ranges. Thus, RNG will not select edge OA, which contradicts our assumption that node O has degree greater than 6. □

Note that different pairs of nodes have distinct Euclidean distances with probability 1 if the x and y coordinates of the nodes are independent continuous random variables with arbitrary density functions. Thus, this condition is satisfied in many practical instances. However, if different pairs of nodes have equal Euclidean distances, then a slight modification of RNG still satisfies the degree bound of 6 [13].

The overall RNG algorithm works as follows. Two nodes A and B that have recently discovered each other, first decide as per the RNG rule[9] whether to form a connection (*i.e.*, add the link between them to the logical topology). If a connection is to be formed, A and B next decide on their respective roles as masters and/or slaves, according to the following rules. Let A have a higher ID than B.

1) When a node powers on, it has an unassigned state.
2) Let A and B have unassigned states when they discover each other. Then, A becomes master, and B becomes a slave in A's piconet.
3) When one node is unassigned and the other is a master, the unassigned node becomes a slave in the piconet of the master if the piconet has less than 7 slaves.
4) When one node is unassigned and the other is a slave, the unassigned node becomes the master of a new piconet, and the other node joins the piconet as a slave (bridge).
5) If neither A nor B is unassigned, then we consider the following cases separately.
   a) If both are masters, then B becomes A's slave (bridge).
   b) If one is a master and the other is a slave in a different piconet, then the slave becomes a bridge between the two piconets.
   c) If both are slaves, then A becomes the master and B becomes the slave (bridge).

Thus, some nodes assume dual roles, *i.e.*, they are both a master and a slave (cases 5a and 5c). This can not be avoided as the resulting graph is not necessarily bipartite. This situation is not desirable even though the Bluetooth standard allows it. We therefore assess the percentage of dual role nodes.

We evaluated the RNG algorithm in the same topologies we considered for MST, MDST and E-MDST (see Section IV-C). In all scenarios, the degrees of the nodes are below 6 (Table II). The percentage of nodes that have to play a dual role is approximately between 17 and 19 percent of the total number of nodes, but their average degree is still low (around 2.7). The simplicity of the RNG algorithm together with its ability to meet the degree constraint that Bluetooth imposes, make it appealing for practical implementation. However, as we show in the next section, implementing even this simple algorithm in a realistic setting is challenging, and more importantly its performance may not be adequate.

[9] A link is not added if one of the incident nodes has a degree of 7. This situation may arise when nodes are in 3-dimensional space or have unequal transmission ranges.

VI. INVESTIGATING THE RNG ALGORITHM FURTHER

This section is devoted to investigating the behavior of the RNG algorithm in terms of its ability to form connected topologies in reasonably large ad hoc networks. Our focus is two-fold. First, we want to assess RNG's performance in a realistic setting and for a variety of scenarios. Second, we want to compare the RNG algorithm with several existing topology formation algorithms that have been proposed by others. Our main purpose for performing such a comparison is to establish that the conclusions we reach based on the performance of the RNG algorithm, extend to systems using other algorithms as well. Specifically, while be believe that the combination of a native distributed operation, minimum reliance on external information (*i.e.*, only the relative distance between nodes is needed), and strong algorithmic guarantees make the RNG algorithm an ideal candidate for topology formation in Bluetooth, we also want to ensure that this is not achieved at the cost of significantly lower performance (*i.e.*, much larger topology formation times) when compared to other alternatives.

For the purpose of evaluating the performance of the RNG algorithm in a realistic setting, we developed a low level emulator of the Bluetooth protocol stack. The neighbor discovery process, *i.e.*, the inquiry/inquiry_scan and page/page_scan modes, is modeled as described in the Bluetooth specifications [1]. The emulator also includes a limited version of the HCI layer, the interface that allows the control layer to communicate with the lower layers of the stack. The emulator controls the operation of the nodes and gets the information needed for topology formation via specific HCI commands and events (see [1], vol. 2, pp. 373–579). For example, the emulator computes the distances between the devices from the strength of the received signal, which can be measured by using the the Read_RSSI command of HCI. Other commands allow the control layer to instruct nodes to switch between Inquiry and Inquiry_scan modes, create a connection, accept a connection request, *etc*.

We test the performance of the RNG algorithm in several different scenarios and for different numbers of nodes. In all scenarios, nodes are powered on at random times that are uniformly distributed in an interval between 0 and 3 seconds, and node positions are generated as described in Section IV-C. During our initial experiments, we allow nodes to conduct device discovery and topology formation in parallel. When two nodes discover each other, they decide whether to form a logical link as per the RNG rule. However, for simplicity, if two nodes decide to form a logical link, they follow the default Bluetooth behavior, namely, the node performing inquiry becomes the master and the node performing inquiry-scan becomes the slave (or bridge). This differs from the role selection rule specified in Section V-B, which would have required the implementation of a more complex role switching capability. Our goal in following the default Bluetooth behavior is to evaluate the percentage of dual role nodes it would produce, and therefore better assess the need for implementing a more complex approach that would also affect the time

	Time (sec.)			Degree Statistics							
N	T_{conn}	T_{enter}	$T_{converge}$	Average	M_a	M_m	B_a	B_m	D_a	D_m	% of Dual
10	10	11	41	2.3	1.3	3	2.0	3	3.4	4	19
25	12	15	146	2.9	1.8	3	3.6	4	3.1	4	44
50	11	16	223	2.8	2.1	5	2.9	4	3.6	5	49
100	14	21	404	3.1	2.5	6	2.4	5	3.2	6	49

TABLE III

EVALUATION OF THE RNG ALGORITHM IN A DYNAMIC SCENARIO WITHOUT DATA TRANSFER

	Time (sec.)			Degree Statistics							
N	T_{conn}	T_{enter}	$T_{converge}$	Average	M_a	M_m	B_a	B_m	D_a	D_m	% of Dual
10	11	11	61	2.4	2.1	4	2.3	3	2.4	3	20
25	11	15	172	3.1	2.7	3	2.8	4	3.7	5	41
50	13	16	231	2.9	2.5	4	2.3	4	3.1	5	47
100	17	22	434	3.2	2.3	5	2.2	4	3.6	5	45

TABLE IV

EVALUATION OF THE RNG ALGORITHM IN A DYNAMIC SCENARIO WITH DATA TRANSFER

required to form a stable, connected topology.

In addition to basic topology statistics such as average and maximum degrees of nodes of different types and percentage of dual role nodes, we also track several other parameters of interest. The first is the average time $T_{converge}$ required to *converge* to the final topology. Convergence to a stable topology is obviously important, as it affects the time taken by routing algorithms to converge and effectively deliver information. We also consider the average time T_{conn} required to form a connection (measured from the start of Inquiry until the connection is formed) and the average time T_{enter} a node requires to establish its *first* link. The latter should be representative of the time it would take a new node to connect to an existing network. We consider both the case of nodes spending all their time doing topology formation (Table III), and of nodes that spend 15% of their time[10] in data transmission mode (Table IV) during which they are, therefore, not available for topology formation.

In conformance with the results of Section V-B all experiments produce a connected topology (when one exists), and the degrees of all nodes are kept below 7. Due to the device discovery scheme of Bluetooth, the average time to connect T_{conn} is around 10 seconds, while a node may have to wait about 20 seconds (T_{enter}) before entering an existing topology. We next investigate $T_{converge}$ and observe that even in the 10 node case, it takes about 1 minute to form a *stable* topology. This time increases to nearly 7 minutes when the number of nodes goes up to 100. When nodes are allowed to spend 15% of their time in "data transfer" mode, as expected, the time increases even further (Table IV). In addition and consistent with our expectations, the percentage of nodes that assume a dual role is substantially higher in this version of RNG than for the one presented in Section V-B (Table II). This is caused by the difference in the master-slave role selection rules between the two versions of RNG. Allowing role switching as proposed in Section V-B can help lower the number of dual role nodes down to about 20% (see Table II), which while still high, it may be worth the added complexity.

Those results indicate that even under relatively benign conditions, *e.g.*, no node mobility, homogeneous transmission ranges, *etc.*, and using a simple distributed algorithm such as RNG, forming stable connected topologies in large (of the order of 100 nodes or more) ad hoc Bluetooth networks may take too long to be practical.

Our next step is to confirm those conclusions by comparing the results obtained for the RNG algorithm to data available for other algorithms. Several scatternet formation algorithms have been previously proposed in [15], [16], [17], [18], [19], and [20], and from the results reported on their performance (see Table VI[11]) it appears that they yield significantly lower topology formation times. It is therefore important to determine whether this difference is attributable to deficiencies in the RNG algorithm. Upon investigating the characteristics of the above algorithms, and more importantly the operating assumptions used when evaluating them, it appears that there are two main reasons behind the reported differences in performance. The first one is a different model for how node discovery and topology formation are carried out, and the second is a different definition of topology formation time. Specifically, the results are obtained by *sequentially* carrying out node discovery for a *fixed* amount of time, and only then initiating the topology formation part. In particular, Basagni *et al.* [14] assumed in their evaluation of several different algorithms that topology formation was preceded by a fixed 20-second period of node discovery. In addition, the times

[10] This is approximately the time that a slave spends in data transmission if its master has 7 slaves. Obviously, masters and bridges will in general spend more time transmitting data. Our model corresponds, therefore, to an optimistic scenario for topology formation.

[11] Table VI reports the topology formation times for most of those algorithms, as well as additional information regarding their main features. Statistics for LSBS [21], BlueTrees [18], Bluenet [19] and BlueStars [20] were taken from [14] as it provides a detailed comparison of those algorithms. Statistics for the remaining algorithms were taken from the original papers. Note that in several instances, those statistics were obtained using simulators instead of low level emulators, and thus the resulting estimates may be somewhat optimistic.

N	$T_{connected}$ (sec.)	$T_{converge}$ (sec.)
10	33	27
25	63	32
50	75	36
100	119	43

TABLE V

EVALUATION OF THE RNG ALGORITHM: TIME TO FORM THE FIRST CONNECTED TOPOLOGY (USING THE ORIGINAL DEVICE DISCOVERY SCHEME) AND TIME TO FORM A STABLE TOPOLOGY WHEN FOLLOWING THE DEVICE DISCOVERY SCHEME OF [14]

reported for topology formation are not always the times until a stable topology has formed, and instead often measure the time it takes to *first* form a connected topology.

Those two differences, and especially the first one, *i.e.*, sequentially performing the node discovery and topology formation, introduces several significant limitations. First, because device discovery is executed only for a fixed amount of time, not all nodes and links are discovered. Thus, it is possible that some nodes are ultimately unable to communicate. Second, since the full *physical* topology is not discovered, the analytical guarantees offered by the MST algorithm (Lemma 1), the RNG algorithm (Lemma 4) and several other algorithms, *e.g.*, [15], [17], [19], [21], no longer hold. Specifically, Basagni *et al.* [14] show that for a network of 110 nodes, after 20 seconds of device discovery, a node only discovers about 88% of its neighbors. If connectivity depends on the remaining 12% of the neighbors, then the algorithms will obviously fail to construct a connected topology. Both of those issues are probably not of much significance in small ad hoc networks, *i.e.*, around 10 nodes, where a discovery phase of 20 or even 10 seconds will typically be sufficient to discover all nodes, but are likely to result in much more severe problems in large-scale networks. Third but not least, the enforcement of a fixed discovery period that precedes topology formation is difficult, if not impossible, to be implemented in a dynamic environment where nodes power on and off or are mobile.

We believe that the definition of topology formation time and the methodology (parallel and ongoing node discovery and topology formation, progressive power-up of nodes, *etc.*) used in our experiments with the RNG algorithm, provide for a more realistic and meaningful assessment of topology formation in large Bluetooth ad hoc networks. Nevertheless, in order to allow for a consistent comparison of RNG and the algorithms of Table VI, we perform additional experiments. The first set of experiments still uses our original assumption of parallel and ongoing node discovery and topology formation, but instead of measuring the time it takes for RNG to form a stable topology, we instead track the time $T_{connected}$ it takes to *first* form a connected topology. The second set of experiments reproduces the operating conditions of [14], and tracks the time $T_{converge}$ for RNG to form a topology in such a setting. Those results are shown in Table V, where the column labeled $T_{connected}$ reports on the first set of experiments, and the column labeled $T_{converge}$ on the second.

¿From the values reported for $T_{connected}$ in Table V, we see that the time taken by RNG to first form a connected topology is much smaller[12] than the time $T_{converge}$ (from Tables III and IV) it takes for this topology to stabilize. This is because the "device discovery" process constantly discovers new nodes and links, which occasionally modifies the topology. As discussed earlier, it is difficult for routing to converge until the topology has settled, which may affect reliable data delivery. Thus, we believe that $T_{converge}$ is a more realistic measure of the time it would take before an ad hoc network forms and becomes operational. Turning to the second column of Table V, we see that when evaluating RNG in a manner consistent with that used to evaluate other algorithms, it yields similar topology formation times. This confirms our initial assessment that the larger topology formation times we had initially observed for RNG are essentially caused by the different operating assumptions we used. As discussed earlier, we believe that our assumptions are more representative of a realistic environment. It should also be pointed out that there are other differences between RNG and some of the algorithms of Table VI. In particular, several of them assume that all nodes are within communication range, which essentially eliminates the connectivity constraint but is unlikely to hold in large scale networks.

Finally, we want to point out that there are several additional difficulties in forming stable, connected topologies in large ad hoc networks that neither our Bluetooth emulator nor any of the other simulation results mentioned in Table VI have meaningfully incorporated. One of these factors is the establishment of compatible sniff windows across piconets. Another aspect is node mobility, which would require constant changes to the topology and possibly frequent renegotiations of sniff windows in the different piconets. Both of these are likely to increase topology convergence times, so that the reported figures should probably be considered "best case scenarios," especially for large numbers of nodes. These, together with long topology formation times and the emergence of a relatively large number of dual role nodes, are the bases for our general conclusion that the deployment of Bluetooth as a core technology for building large-scale ad hoc networks is unlikely, especially given the availability of seemingly more suitable alternatives such as 802.11.

VII. RELATED RESEARCH

We briefly mention a number of previous works that have been motivated by the ambition to use Bluetooth in ad hoc networks. They span two related areas: (i) assessing the

[12]And much closer to the topology formation times of algorithms in Table VI.

Algorithm	Ref.	# Nodes	Time (sec.)	Percent of dual nodes	Connected topology guaranteed?	Degree constraint guaranteed?	Devices within range?	Comments
BlueTrees	[18]	110	36	50	Yes	No	Yes	Needs connected topology for "blueroot" election.
Bluenet	[19]	110	33	78	No	Yes	No	
TSF	[16]	64	14	See note (1)	Yes	No	Yes	Elects a coordinator within each subtree.
BlueStars	[20]	110	24	22	Yes	No	No	
BlueMesh	[17]	120	See note (2)	12-17	Yes	Yes	No	Degree constraint guaranteed only in 2-D.
Law *et al.*	[15]	128	42	See note (3)	Yes	Yes Note (4)	Yes	Leader election in each component.
LSBS	[21] [20] [14]	110	34	21	Yes	Yes	Yes	Degree constraint guaranteed only in 2-D. Needs node locations.
RNG	[22]	100	43 Note (5)	47	Yes	Yes	No	Degree constraint guaranteed only in 2-D Needs Euclidean distance (RSSI).

TABLE VI
COMPARISON OF VARIOUS TOPOLOGY FORMATION ALGORITHMS

potential of Bluetooth in comparison to other technologies and (ii) developing algorithms for forming and maintaining network topologies.

Johansson *et al.* investigate the suitability of Bluetooth as a networking technology [23], [24]. The authors identify Bluetooth's potential in building Personal Area Networks [23]. They compare Bluetooth to IEEE 802.11 and conclude that in small Personal Area Networks Bluetooth is better suited than IEEE 802.11. Their conclusions do not apply to large ad hoc networks as they do not consider topology formation and the effects of device discovery.

A few authors have already acknowledged that building Bluetooth-based networks is complex. Basagni *et al.* [14] identify several problems that the Bluetooth technology gives rise to. They observe that the device discovery process consumes a lot of time, and propose modifications to the Bluetooth standard that may make its operation more efficient. They conclude that forming scatternets is still a formidable task. Liu *et al.* [25] present an on-demand approach for building a path between Bluetooth devices. However, the delay incurred in their route discovery process is large. Moreover, their results suggest that scatternets face scalability problems. Zheng *et al.* [26] briefly comment on the complexity of Bluetooth when comparing it to other technologies. Law *et al.* [15] mention that the problem of collisions of paging messages becomes significant when the number of nodes exceeds 64. Salonidis *et al.* [27] prove that the average delay involved in synchronizing two nodes is infinite if the nodes rely on a deterministic pattern of alternating between paging and paged modes. This is another issue that is irrelevant when Bluetooth is used as a wire replacement technology, but that is important in a networking context. Chiasserini *et al.* [28] consider procedures to handle topology changes in an already existing Bluetooth network. Kallo *et al.* [29] also consider topology maintenance.

BTCP [27] describes a leader election process to control the topology formation process. It requires all nodes to be in each others transmission range in order to carry out the leader election. This condition is unlikely to hold in general, and also means that the leader election approach of BTCP is not truly a distributed algorithm since all nodes have access to global information to elect a leader. Barrière *et al.* [30] have proposed a dynamic and distributed algorithm that is capable of achieving not only connectivity, but also of controlling the size of piconets as well as the desired degrees of masters and slaves. However, it requires that all nodes be capable of communicating with each other, which will often not hold. Finally, Marsan *et al.* [31] formulate an integer linear program for computing the "optimal" Bluetooth topology. The complexity of the proposed algorithm is however high. Furthermore, the integer linear program can only be solved in a centralized manner.

As discussed earlier, several topology formation algorithms have been proposed and evaluated, and Table VI summarizes their main properties and performance. Additional comments and clarification regarding the properties of the different algorithms are provided in the notes that accompany Table VI. None of the distributed algorithms listed in Table VI are guaranteed to produce connected topologies that satisfy the degree constraints of Bluetooth in general settings. This can be explained by our result that satisfying both requirements is an NP-hard problem. Like the MST and RNG algorithms we presented in Sections IV-B and VI, BlueMesh [17] and

LSBS [21] satisfy both these requirements only when nodes are on a plane and have equal transmission ranges. In addition, LSBS assumes that each node knows its own and its neighbors' locations. This requires additional hardware, *e.g.*, a GPS receiver and is therefore not consistent with Bluetooth's design goal of providing low cost energy-efficient transceivers. By using the relative neighborhood graph structure, which is a subset of the geometric structure (Delauney Triangulation) that LSBS uses, the RNG algorithm achieves similar connectivity and degree constraint guarantees as LSBS, but does not require nodes to be location aware.

Notes on Table VI.

1) Tan *et al.* [16] do not provide statistics on the percentage of nodes that assume a dual role. However, since components (subtrees) merge only from their root nodes, some roots will have to assume a dual role. The nodes can can subsequently switch their roles, but switching roles would require network-wide changes.
2) Petrioli *et al.* [17] mention that on average about 4 iterations are required to complete the scatternet formation process for 120 nodes. However, there is no information on how much time each iteration takes. Moreover, there is no information on how much time the first phase (topology discovery) of the protocol takes. Given the results of [14], it is likely that each node takes more than 20 seconds to discover its one- and two-hop neighbors.
3) Since each leader executes SEEK (*i.e.*, Inquiry) or SCAN (*i.e.*, Inquiry_Scan) using a randomized procedure, some nodes will have dual roles. However, no statistics on the percentage of nodes that assume a dual role are available.
4) The authors focus on the case where all devices are within range. However, in Section 8.3 (page 11) they discuss a scenario allowing out of range devices, and mention that in this case the degree constraint is not guaranteed.
5) For uniform comparison (as in [14]) we report the running time of the algorithm when devices discover each other for 20 seconds (Table V).

VIII. CONCLUSION

In this paper, we investigate the feasibility of using Bluetooth as the base communication technology in large-scale ad hoc networks, a task that significantly exceeds its initial scope of a "wire replacement" technology. Our investigation is motivated by Bluetooth's design paradigm that is fundamentally different from that of competing technologies such as IEEE 802.11. We focus on the basic aspect of topology formation as it illustrates the problems that Bluetooth encounters when used as a networking technology. We first investigate this problem from an algorithmic perspective to gain a basic understanding of its fundamental complexity, and establish that deciding whether there exists at least one connected topology that satisfies the degree constraint of Bluetooth is NP-hard. This explains why forming a topology in a short time while satisfying all the Bluetooth constraints has remained elusive even after several years of extensive research. However, we also prove that an MST-based algorithm is guaranteed to satisfy Bluetooth's constraints under some simplifying assumptions. We also propose several heuristics that satisfy Bluetooth constraints under most conditions and do not rely on those assumptions. Some of these heuristics can differentially control the degrees of masters and slaves, and thereby attain a better delay/throughput trade-off. These results provide the foundation for an in-depth investigation of Bluetooth's implementation complexity and operational overhead when used as an ad hoc network technology.

For a comprehensive and realistic investigation of Bluetooth's implementation complexity we designed a detailed low-level emulator of the Bluetooth stack, and used it to examine the convergence time and complexity of a simple, distributed algorithm (RNG) that is capable of satisfying Bluetooth guarantees in most environments. Our findings are that although the algorithm succeeds in forming connected topologies, the time required to generate a stable topology in the presence of a large number of nodes is large enough that it is unlikely to be practical. Furthermore, the presence of a large percentage of dual role nodes substantially impacts the network throughput. These already poor results would only worsen if all the other constraints imposed by the Bluetooth protocol, *e.g.*, sniff window negotiations, handling of node mobility and topology adjustments, *etc.*, were taken into account. Several topology formation algorithms proposed by other authors also perform similarly. As a result, we believe that in spite of the significant attention it has received over the past few years and the many interesting proposals and results it has generated, Bluetooth's inherent complexity as a *networking* protocol makes it unlikely that it will be widely used in building large ad hoc networks. Nevertheless, it is certainly possible for Bluetooth to be successfully used in building small ad hoc networks, where the issue of topology formation is of much lesser concern.

ACKNOWLEDGEMENT

The authors would like to thank Dr. Murali Kodialam, currently at Bell Laboratories, Lucent Technologies, Holmdel, NJ, for directing us to Relative Neighborhood Graphs.

REFERENCES

[1] Bluetooth SIG, *Specification of the Bluetooth System, Version 1.2*, November 2003.
[2] F. Harary, *Graph Theory*. Addison-Wesley, 1969.
[3] M. R. Gary and D. S. Johnson, *Computers and Intractability*. Freeman, 1979.
[4] R. Guérin, E. Kim, and S. Sarkar, "Bluetooth technology: Key challenges and initial research," in *Proc. Commun. Netw. & Dist. Sys. Model. & Simul. Conf. (CNDS)*, San Antonio, TX, January 2002.
[5] C. Monma and S. Suri, "Transitions in geometric minimum spanning trees," *Discrete and Comput. Geometry*, vol. 8, no. 3, 1992.
[6] T. Cormen, C. Leiserson, and R. Rivest, *Introduction to Algorithms*. MIT Press, 1990.
[7] R. Gallager, P. Humblet, and P. Spira, "A distributed algorithm for minimum-weight spanning trees," *ACM Trans. Program. Lang. & Sys.*, vol. 5, no. 1, January 1983.
[8] G. Robins and J. Salowe, "On the maximum degree of minimum spanning trees," in *Proc. ACM Symp. Comput. Geometry*, Stony Brook, NY, June 1994.
[9] D. Hochbaum, *Approximation Algorithms for NP-hard Problems*. PWS, 1995.
[10] C. Cheng, I. Cimet, and S. Kumar, "A protocol to maintain a minimum spanning tree in a dynamic topology," in *Proc. ACM Symp. Commun. Arch. & Prot.*, Stanford, CA, August 1988.
[11] P. Nárvaez, K.-Y. Siu, and H.-Y. Tzeng, "New dynamic algorithms for shortest path tree computation," *IEEE/ACM Trans. Networking*, vol. 8, no. 6, December 2000.
[12] G. T. Toussaint, "The relative neighborhood graph of a finite planar set," *Pattern Recognition*, vol. 12, no. 4, pp. 261–268, 1980.

[13] X. Li, Y. Wang, P. Wan, and O. Frieder, “Localized low weight graph and its applications in wireless ad hoc networks,” in *Proc. INFOCOM’04*, Hong Kong, China, March 2004.

[14] S. Basagni, R. Bruno, G. Mambrini, and C. Petrioli, “Comparative performance evaluation of scatternet formation protocols for networks of Bluetooth devices,” *Wireless Netw.*, vol. 10, no. 2, March 2004.

[15] C. Law, A. K. Mehta, and K.-Y. Siu, “A new Bluetooth scatternet formation protocol,” *Mobile Netw. and Apps.*, vol. 8, no. 5, October 2003.

[16] G. Tan, A. Miu, J. Guttag, and H. Balakrishnan, “An efficient scatternet formation algorithm for dynamic environments,” in *Proc. IASTED Intl. Conf. Commun. & Comput. Netw.*, Cambridge, MA, November 2002.

[17] C. Petrioli, S. Basagni, and I. Chlamtac, “BlueMesh: Degree-constrained multihop scatternet formation for Bluetooth networks,” *Mobile Netw. and Apps.*, vol. 9, no. 1, February 2004.

[18] G. Záruba, S. Basagni, and I. Chlamtac, “Bluetrees-scatternet formation to enable Bluetooth-based ad hoc networks,” in *Proc. IEEE Intl. Conf. Commun. (ICC)*, vol. 1, Helsinki, Finland, June 2001.

[19] Z. Wang, R. Thomas, and Z. Haas, “Bluenet - a new scatternet formation scheme,” in *Proc* 35^{th} *Hawaii Intl. Conf. Sys. Sciences (HICSS-35)*, Big Island, HI, January 2002.

[20] C. Petrioli, S. Basagni, and I. Chlamtac, “Configuring BlueStars: Multihop scatternet formation for Bluetooth networks,” *IEEE Trans. Comput., Special Issue on Wireless Internet*, vol. 52, no. 6, June 2003.

[21] X.-Y. Li, I. Stojmenovic, and Y. Wang, “Partial Delaunay triangulation and degree limited localized Bluetooth scatternet formation,” *IEEE Trans. Parallel Dist. Sys.*, vol. 15, no. 4, April 2004.

[22] R. Guérin, J. Rank, S. Sarkar, and E. Vergetis, “Forming connected topologies in Bluetooth adhoc networks - An algorithmic perspective,” in *Proc.* 18^{th} *Intl. Teletraffic Congr.*, Berlin, Germany, September 2003.

[23] P. Johansson, M. Kazantzidis, R. Kapoor, and M. Gerla, “Bluetooth: An enabler for personal area networking,” *IEEE Network*, vol. 15, no. 5, September/October 2001.

[24] P. Johansson, R. Kapoor, M. Kazantzidis, and M. Gerla, “Personal area networks: Bluetooth or IEEE 802.11?” *IntL. J. Wireless Infor. Netw.*, vol. 9, no. 2, April 2002.

[25] Y. Liu, M. Lee, and T. Saadawi, “A Bluetooth scatternet-route structure for multihop ad hoc networks,” *IEEE J. Select. Areas Commun.*, vol. 21, no. 2, February 2003.

[26] J. Zheng and M. Lee, “Will IEEE 802.15.4 make ubiquitous networking a reality?: A discussion on a potential low power, low bit rate standard,” *IEEE Commun. Mag.*, vol. 42, no. 6, June 2004.

[27] T. Salonidis, P. Bhagwat, L. Tassiulas, and R. LaMaire, “Distributed topology construction of Bluetooth personal area networks,” in *Proc. INFOCOM’01*, Anchorage, AK, April 2001.

[28] C. Chiasserini, M. Marsan, E. Baralis, and P. Garza, “Towards feasible topology formation algorithms for Bluetooth-based WPANs,” in *Proc.* 36^{th} *Hawaii Intl. Conf. Sys. Sciences (HICSS-36)*, Big Island, HI, January 2003.

[29] C. Kalló, C.-F. Chiasserini, R. Battiti, and M. Marsan, “Reducing the number of hops between communication peers in a Bluetooth scatternet,” in *Proc. IEEE Wireless Commun. & Netw. Conf. (WCNC)*, Atlanta, GA, March 2004.

[30] L. Barrière, P. Fraigniaud, L. Narayanan, and J. Opatrny, “Dynamic construction of Bluetooth scatternets of fixed degree and low diameter,” in *Proc. of the* 14^{th} *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, Baltimore, MD, January 2003.

[31] M. Marsan, C. Chiasserini, A. Nucci, G. Carello, and L. D. Giovanni, “Optimizing the topology of Bluetooth wireless personal area networks,” in *Proc. INFOCOM’02*, New York, NY, July 2002.

PLACE PHOTO HERE

Evangelos Vergetis (S’00) received a Bachelor of Science in Computer Science and Electrical Engineering from Cornell University, Ithaca, NY, in 2001, and an M.S.E. in Electrical Engineering from the University of Pennsylvania, Philadelphia, PA, in 2002. He is currently working towards his Ph.D. in the department of Electrical and Systems Engineering at the University of Pennsylvania.

PLACE PHOTO HERE

Roch Guérin (F ’01 / ACM ’98) received an engineer degree from ENST, Paris, France, in 1983, and M.S. and Ph.D. degrees in EE from Caltech in 1984 and 1986, respectively. He joined the Electrical and Systems Engineering department of the University of Pennsylvania in 1998, where he is the Alfred Fitler Moore Professor of Telecommunications Networks. Before joining the University of Pennsylvania, he spent over twelve years at the IBM T. J. Watson Research Center in a variety of technical and management positions. From 2001 to 2004 he was on partial leave from Penn, starting a company, Ipsum Networks, that pioneered the concept of protocol participation in managing IP networks. His research has been in the general area of networking, with a recent focus on developing routing and traffic engineering solutions that are both lightweight and robust across a broad range of operating conditions. Dr. Guérin served as the editor of the ACM SIGCOMM technical newsletter, CCR, from 1998 to 2001. He chaired the IEEE Technical Committee on Computer Communications from 1997 to 1999, and was an editor for the Journal of Computer Networks, the IEEE Communications Surveys, the IEEE/ACM Transactions on Networking, the IEEE Transactions on Communications, and the IEEE Communications Magazine, and a guest editor of a JSAC issue on Internet QoS published in December 2000. He served as member-at-large of the Board-of-Governors of the IEEE Communications Society from 2000 to 2002, as General Chair of the IEEE INFOCOM’98 conference, and as Technical Program co-chair of the ACM SIGCOMM’2001 conference. In 1994 he received an IBM Outstanding Innovation Award for his work on traffic management in the BroadBand Services Network Architecture. He is on the Technical Advisory Board of France Telecom and Samsung Electronics, and has consulted for numerous companies in the networking area. His email address is: `guerin@ee.upenn.edu.`

PLACE PHOTO HERE

Saswati Sarkar (S’98, M’00) received a Master of Engineering in Electrical Communication Engineering from the Indian Institute of Science in 1996 and a Ph.D. in Electrical and Computer Engineering from University of Maryland, College Park in 2000. She is currently an Assistant Professor in the department of Electrical and Systems Engineering in University of Pennsylvania. Her research interests are in resource allocation and performance analysis in communication networks. She received the Motorola gold medal for the best masters student in the division of electrical sciences at the Indian Institute of Science and a National Science Foundation (NSF) Faculty Early Career Development Award in 2003. She has been an associate editor of IEEE Transaction on Wireless Communications from 2001.

PLACE PHOTO HERE

Jacob Rank received a B.S.E. in computer and telecommunications engineering from the University of Pennsylvania, Philadelphia, PA, in 2003. He is currently working as a consultant in the product engineering group at CGI-AMS in Fairfax, VA. His research interests include genetic programming and alternative energy sources.