Forming Connected Topologies in Bluetooth Adhoc Networks

R. Guerin, S. Sarkar, E. Vergetis
Electrical and Systems Engineering Department
University of Pennsylvania
{guerin,swati,vergetis}@ ee.upenn.edu

Abstract— **This paper explores the problem of forming connected topologies, in adhoc networks built on the Bluetooth technology. Providing connectivity, when feasible, is the most basic requirement for any system aimed at allowing devices to communicate with each other. In this paper, we illustrate that this seemingly innocuous goal gives rise to many significant challenges in the context of the Bluetooth technology. The paper first provides a brief overview of Bluetooth and its operation, and then identifies some of the major problems the technology faces when it is used to build adhoc networks. The paper's contributions are in identifying some basic algorithmic problems when building connected Bluetooth networks, and in developing and evaluating several possible solutions capable of generating "good" connected topologies.**

I. Introduction

Bluetooth is a recently proposed standard for short range, low power wireless communication [10]. Initially, it is being envisioned simply as a wire replacement technology. Its most commonly described application is that of a "cordless computer" consisting of several devices including a personal computer, possibly a laptop, keyboard, mouse, joystick, printer, scanner, etc., each equipped with a Bluetooth card. There are no cable connections between these devices, and Bluetooth is to enable seamless communication between all them, essentially replacing what is today achieved through a combination of serial and parallel cables, and infrared links. However, Bluetooth has the potential for being much more than a wire replacement technology, and the Bluetooth standard was indeed drafted with such a more ambitious goal in mind. Bluetooth holds the promise of becoming the technology of choice for adhoc networks of the future. This is in part because its low power consumption and potential low cost make it an attractive solution for the typical mobile devices used in adhoc networks.

This being said, there are multiple major technical hurdles to cross before Bluetooth can fulfill its potential of becoming more than a wire replacement solution. The most basic challenge that the technology faces, and the one that is the focus of this paper, is how to organize nodes into an operational network, while satisfying the many constraints introduced by Bluetooth. There are obviously multiple possible interpretations of what operational means, and in this paper we concern ourselves primarily with connectivity. In other words, our goal is to understand when and how Bluetooth networks can be built that allow communication between all pairs of nodes. This is clearly one if not the most basic functionality one expects from a network, but as we shall see, providing it in Bluetooth networks gives rises to numerous challenges. The reasons for those difficulties are at many levels. Although Bluetooth nodes are assumed functionally equivalent, communications proceed according to a "master-slave" model, with additional constraints on the number of slaves that a master can support. This introduces a degree constraint on the graph representing connections between masters and slaves. As a result, the selection of which nodes become master or slaves and of how slaves are assigned to masters during the topology formation phase, can have a significant impact on the final connectivity.

Similarly, the synchronization process that allows Bluetooth nodes to discover their neighbors influences the order in which topology formation decisions are made, and therefore also affects the resulting connectivity. This latter aspect was partially investigated in [14], [6], and in this paper we concentrate mainly on the former issues, namely, how the basic communication model and constraints of Bluetooth affects its ability to form connected topologies.

The paper's contributions are primarily in investigating the problem from an algorithmic perspective. For example, our investigation shows that the seemingly innocuous problem of deciding whether there exists at least one connected topology that is consistent with the degree constraint of the Bluetooth technology is actually NP-hard in the most general case. However, when node locations are restricted to a two-dimensional plane, we obtain, under certain simplifying assumptions, a polynomial complexity topology formation algorithm that attains end-to-end connectivity whenever it is possible to do so. For the more realistic case of three-dimensional networks and assuming more complex node configurations, we propose other algorithms which attain end to end connectivity most of the times, and demonstrate their efficacy by empirical studies. Finally, we propose an algorithm which enables the designer to have a greater control over the topology than mere connectivity. More specifically, this algorithm has provisions of tuning the degrees of masters and slaves differentially. The algorithms we propose in this paper also cater to efficient distributed operation in a dynamic scenario.

This paper is organized as follows. We review briefly the salient features of the Bluetooth technology in Section II. In Section III, we describe technical challenges that arise when using the Bluetooth technology in adhoc networks, with a focus on topology formation issues. Section IV is a brief summary of recent research on the topic. Section V is devoted to our problem formulation and to showing that the problem of attaining end to end connectivity is NP-hard in the general case. In Section VI, we present a polynomial complexity topology formation algorithm that, under some simplifying assumptions, attains a connected topology whenever one such exists. We show that this algorithm is suitable for distributed and dynamic operation. In sections VII and VIII we address the cases when these assumptions do not hold. We present certain heuristic approaches for these cases and evaluate their performance via simulation.

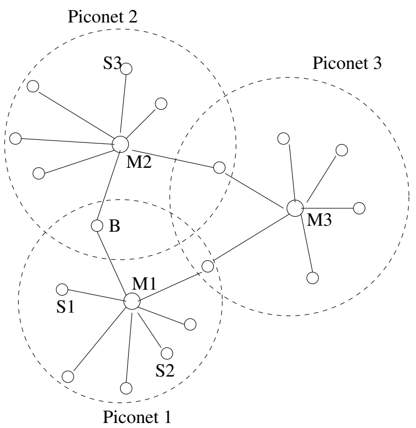

Fig. 1. An example Bluetooth topology is illustrated. The nodes are organized into 3 piconets. The masters of these piconets are M_1, M_2, M_3 respectively. The remaining nodes are the slave nodes or bridge nodes. Slave nodes S_1 and S_2 can communicate via master M_1. Nodes S_1 and S_3 can communicate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

II. A Brief Overview of Bluetooth

In this section, we briefly describe the basic features of a Bluetooth network. Bluetooth nodes are organized in small groups called *piconets*. Every piconet has a leading node called "master," and other nodes in a piconet are referred to as "slaves." A node may belong to multiple piconets, and we refer to such a node as a "bridge." A piconet can have at most 8 members. Refer to figure 1 for a sample organization. Every communication in a piconet involves the master, so that slaves do not directly communicate with each other but instead rely on the master as a transit node. The master decides the communication order (and duration) for the slaves. In other words, Bluetooth provides a half-duplex communication channel, and its usage is solely under the responsibility of the master. Communication between nodes in different piconets relies on bridge nodes. A bridge node cannot be simultaneously active in multiple piconets. It is active in one piconet and "parked" in others. Bluetooth allows different activity states for the nodes: active, idle, parked, sniffing. Data exchange takes place between two nodes only when both are active. Activity states of nodes change periodically.

Bluetooth uses frequency hopping spread spectrum in the physical layer. Different piconets use different frequency hopping sequences. The frequency hopping sequence of a piconet is derived from the node id and the clock information of the master. A node thus needs to know the identities and the clock information of the masters of all the piconets it participates in. It acquires this information from the master when it joins the piconet. Synchronization information is also exchanged periodically. The bandwidth of the Bluetooth communication channel is currently 1 Mbps. Nodes in different piconets can transmit simultaneously even if they are within transmission range of each other. This is because they use different frequency hopping patterns. However, there can be only one communication at a time within in a piconet and this communication involves the master and one slave.

Besides the operation and constraints associated with the Bluetooth communication channel, another key aspect affecting the formation of Bluetooth topologies is, as mentioned earlier, the node discovery process. It involves a well-known channel or frequency hopping pattern, that is used by nodes to periodically transmit their identity and listen for other nodes (see [10] for details). The main impact of this process is through its influence on when nodes discover each other and, therefore, proceed to form or join piconets. This was investigated in [6], and in this paper we instead concentrate on how the basic characteristics of the Bluetooth communication channel impacts topology formation.

III. Challenges and Objectives in Bluetooth Topology Formation

As mentioned earlier, communications within a Bluetooth piconet follow a master-slave model, while communications across piconets rely on the presence of nodes belonging to multiple piconets. The technology is flexible in that except for limiting the number of slaves in a piconet to 7, no other constraints exist regarding the assignment of roles to nodes. This flexibility, however, raises a number of questions, and we briefly 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 masters be allowed to be bridge nodes (slaves in other piconets)?

All the above questions are complex and important in their own right, and in the context of maximizing the likelihood of producing a connected topology, their answers should preferably avoid introducing additional constraints. However, some of them can have severe performance implications at a level such that basic connectivity alone may ultimately not be a sufficient attribute.

Specifically, consider questions 4 and 5. Because a bridge node can only be active in one piconet at the time, the greater the number of piconets to which a node belongs, the poorer the connectivity it can provide between them. This impact is compounded when the bridge node is a master in one piconet. This is because any period during which the node is acting as a slave in some piconet, corresponds to a complete communication blackout for all the slaves of the piconet for which it serves as a master. In other words, although those nodes "theoretically" have connectivity to the rest of the network, it is likely to be so limited as to be nearly impractical. Because of this potentially severe impact on basic connectivity, we will assume that the answer to question number 5 is negative, i.e., masters cannot be bridges (slaves) in other piconets. Note that this assumption was also made in [14]. As far as question number 4 is concerned, we will limit ourselves to considering it desirable for a bridge to be involved in as small a number of piconets as possible, while preserving connectivity. This is a criterion that will be incorporated in some of the algorithms we present in the paper.

The primary focus of the algorithms we develop will then be to generate connected topologies, when feasible, while satisfying the degree constraint to which masters are subject, i.e., a degree of 7 at most. In that context, the algorithms will aim at answering questions 1 and 2, while assuming that the answer to question 3 is simply based on the upper bound that the Bluetooth specification imposes. In the rest of this section, we briefly expand on what achieving those objectives means for our topology formation algorithms.

Generating a connected topology is a reasonably clear objective, but there are many cases where it may not be a feasible objective. For example, if there are two groups of nodes such that no node in one group is in the transmission range of any node in the other group, then the topology will be disconnected irrespective of which algorithm one uses. It is, therefore, also important to articulate the algorithm's goals in such cases. If global connectivity is not feasible then the logical topology will consist of "islands" of nodes such that each island includes nodes that are within transmission range of each other, but nodes in different islands can not communicate. In this case, the objective is to minimize the number of islands in the final topology.

Next, let us examine the implications of our objective of maximizing connectivity within the Bluetooth design constraints. Observe first that any Bluetooth topology must satisfy some basic properties. For one, the partitioning of nodes into masters and slaves implies that the graph associated with any Bluetooth topology is a bi-partite graph. This is because neither masters nor slaves can communicate directly, and therefore the set of nodes associated with masters only has edges to the set of nodes corresponding to slaves, and vice-versa. Similarly, as mentioned before, the constraint that a piconet cannot contain more than 7 slaves implies that all nodes associated with masters must have a degree less than or equal to 7. This also implies that if at any time the total number of masters is less than one eighth of the total number of nodes, then certain nodes will not belong to any piconet and thus the topology remains disconnected. Both of these observations provide "hints" towards designing topology formation algorithms. In particular, focusing on algorithms that generate bipartite graphs and can bound or minimize node degrees appears to be a promising direction, which we explore further in Sections VI and VII.

In addition, because adhoc networks are inherently distributed systems, with nodes making individual decisions, often based on local information, we expect that another important factor besides the choice of a role (master or slave) is *when* is that choice actually made, i.e., the order in which nodes make such decisions. In particular, because connectivity between piconets is ensured through bridge nodes and not all (slave) nodes are capable of playing such a role (the node must be able to "hear" the master of each piconet), connectivity between two piconets may be precluded if the corresponding node attempts to join one of the piconets after the piconet has become full, i.e., already has 7 slaves. This indicates that efficient algorithms will most likely need to allow for some iterations that will let nodes modify their earlier decisions, e.g., by having some slaves relinquish their membership in one piconet and move to another, or by allowing nodes to change their role altogether (change from slave to master or possibly vice-versa). Identifying when and how to allow such changes while preserving or improving connectivity or degree constraint is a challenging task, especially in the context of distributed decisions. The approach we will follow, is to first articulate a centralized algorithm capable of generating a good/optimal solution, and then describe how to transform this centralized algorithm into a distributed one. Techniques for converting centralized algorithms into distributed ones are commonly available for many standard algorithms, e.g., shortest path or minimum spanning tree, and we will take advantage of this fact.

A second important aspect in the context of adhoc networks is to allow for dynamic operation. Dynamic operation has two major dimensions: the addition and deletion of nodes as users come and go, and the possible modifications of the topology (connectivity graph) induced by mobile nodes. Designing algorithms and protocols capable of efficient dynamic operation is an entire research area in itself, where meaningful solutions typically need to carefully account for operational constraints. The algorithms we present in the paper are clearly aimed at supporting dynamic operation, e.g., by relying only on local information when attempting to optimize connectivity. However, there are many aspects of dynamic operation that we do not explore or consider, and formulating a complete solution for a dynamic environment is a topic of ongoing work. We believe that the results of this paper provide essential building blocks for developing such a solution.

IV. Related Research

In this section, we mention very briefly a number of previous works that have also been motivated by the need to extend the standard Bluetooth specifications, if the technology is to be used for building adhoc networks. Those works span three major areas associated with adhoc networks: routing, resource management or scheduling, and topology formation. The latter being clearly the area of most relevance to this paper.

In [1] Bhagwat *et al.* present a source routing mechanism for Bluetooth scatternets, i.e., networks formed from the interconnection of piconets. Das *et al.* [3] and Johanson *et al.* [9] present distributed scheduling policies for Bluetooth networks. The topology formation problem was first investigated in [14] by Salonidis *et. al.* who presented a distributed topology construction scheme in Bluetooth networks. A basic assumption in the paper is that all nodes are within transmission range of each other, i.e., the underlying connectivity graph is a full mesh. Under this assumption, nodes conduct a leader election algorithm, whose winner knows the identity of all other nodes in the network and uses this information to design the desired topology. A likely limitation of this approach is that it may not scale as the number of nodes is large. This was one of the motivations for investigating alternative approaches in this paper. In addition, the assumption of full reachability may not hold in many scenarios. The paper also makes a very interesting observation in that it shows that the average delay involved in synchronizing two nodes (the time spent in the inquiry and the page sequences before the nodes are able to exchange the clock information) is infinite if the nodes rely on a deterministic pattern of alternating between paging and paged modes. This has important implications when designing the synchronization procedure that Bluetooth nodes will use to discover their neighbors. In [4], Zaruba *et al.* present "Bluetrees," a scatternet formation algorithm for cases where the full reachability assumption does not hold. However, the Bluetree algorithm reduces the degree of the nodes by a series of re-arrangements. There is no guarantee that these re-arrangements actually terminate, and thus the resulting topology may not satisfy the degree constraints. The problem of topology formation was also the topic of [6], where we investigated the performance of a few very basic (simplistic) topology formation algorithms from the standpoint of connectivity and convergence time. The investigation was primarily empirical, i.e., relying on a detailed low-level simulation of the Bluetooth protocol, and showed that those simple-minded algorithms commonly resulted in disconnected topologies, when a connected one was actually available. This finding was another motivation for a more systematic investigation of the topology formation problem, and for identifying algorithms capable of better performance.

V. Network Model and Problem Complexity

As a first step towards a systematic investigation of the connectivity issue, we formulate a mathematical model for the system objectives and constraints. Observe that there can be two types of communication links between any two nodes. One is a *physical* (layer) link which exists between any pair of nodes that are in communication range of each other. The other is a *logical* Bluetooth link which exists if the Bluetooth topology establishes a communication link between the two nodes. This happens when one node is a master and the other is a slave in the same piconet. Clearly, a pre-condition for existence of a logical link between two nodes is the existence of a physical link between them. The scenario can be modeled by two graphs, a physical-topology graph and a logical-topology graph. Vertices in both graphs represent the nodes in the actual network. The logical topology graph is thus a sub-graph of the physical topology graph. The physical topology graph is given, while the logical topology graph is the output of the topology formation algorithm.

The logical topology graph must satisfy certain constraints. It is a bipartite[1] graph since the vertex set can be partitioned in masters and slaves and the logical links connect masters to slaves and vice-versa. Vertices in one partition, the partition associated with masters, can have a

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

maximum degree[2] of 7, and for those in the other (slave) partition it is desirable that their degree be kept as small as possible. Regular slave nodes have a degree of only 1 (the link to the master), but bridge nodes have a degree equal to the number of piconets they participate in. Because a bridge node with a degree of 7, i.e., participating in 7 piconets, would represent a major bottleneck in the system (it would provide very poor connectivity between those piconets), we assume that the degree constraint of 7 applies to the bridge (slave) nodes as well.

Connectivity is then deemed feasible if there exists a connected[3] sub-graph of the physical topology graph which satisfies the degree constraint (maximum degree of 7). The objective is to first detect whether connectivity is feasible. If connectivity is feasible, then the aim is to construct a connected logical topology graph which satisfies the desired constraint. If connectivity is not feasible, then any logical topology graph will consist of "islands" or components[4]. In this case the objective is 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[5], which satisfies the degree constraints of a logical topology graph. This is because a spanning tree of any graph is connected and bipartite [7]. Thus, if a spanning tree of the physical topology graph satisfies all the constraints of a logical topology graph, then it forms a feasible instance of a logical topology. The converse is trivially seen to be true. The partition which has a maximum degree less than or equal to 7 will be chosen as the master set, while the other with a potentially lower maximum degree will constitute the slave/bridge set. Note that if there is a connected logical topology sub-graph, then its spanning tree is also a spanning tree of the physical topology graph. as a result, we focus on constructing a spanning tree of the physical connectivity graph, which satisfies the requisite constraints.

Let the degree of a spanning tree be the maximum degree of its vertices. The challenge is then to construct a spanning tree with degree less than or equal to 7, if one exists. This is an NP-hard problem for a general graph [5]. More specifically, a spanning tree with degree 7 or less exists if and only if the minimum degree of a spanning tree in a graph is upper bounded by 7, and deciding this is an NP-hard problem [5]. It follows that *deciding whether connectivity is feasible and constructing a connected logical topology graph which satisfies the desired constraints is an NP-hard problem for a general physical topology.*

The above result not withstanding, we explore in the rest of the paper a number of polynomial time algorithms. Polynomial time algorithms are available in certain practical scenarios, where additional constraints are imposed on the underlying network graph (Section VI). Furthermore, we show how those polynomial time algorithms can be extended to provide efficient heuristics in general scenarios (Sections VII and VIII).

VI. Topology Formation Algorithms for Nodes with Identical Power Levels in a 2−Dimensional Plane

In this section, we approach the connectivity problem under certain simplifying assumptions which we describe and justify next.

1. Nodes constitute points in a two-dimensional plane. This assumption is justified if the transceivers are at similar heights, and applies to several ground based civilian and military communication networks, while ruling out applications which involve air to ground communication.
2. All nodes have the same transmission range. This happens if the propagation conditions are largely similar throughout the network and nodes have the same maximum transmission power limitation and similar reception capabilities. Let this transmission range be d units. Under this assumption, a physical link exists between any two nodes if and only if their Euclidean distance is upper bounded by d.

The connectivity problem becomes of polynomial complexity under these assumptions. The following lemma provides the cornerstone for designing a simple polynomial complexity distributed dynamic algorithm which 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 which satisfies all the desired constraints.

We present the following result obtained by Monma *et. al.* [11] which we will use in proving this lemma.

Proposition 1: Consider a complete[6] graph with nodes corresponding to points in the 2−dimensional plane and the weight of the edges being the Euclidean distance between them. Any minimum weighted spanning tree in such a graph has degree less than 6.

An intuition behind this result is provided in figure 2.

Proof of Lemma 1: Clearly a necessary condition for connectivity to be feasible is that the physical topology graph

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

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

[4] A component of a graph is a connected sub-graph which can not be expanded any further while retaining connectivity, i.e., addition of a node in a component removes the connectivity.

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

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

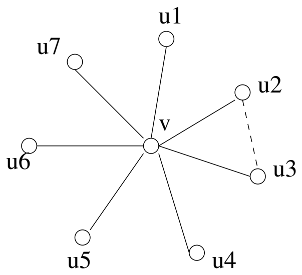

Fig. 2. We explain intuitively why the degree of a MST is no more than 6 in a complete graph with edge weights equaling the Euclidean distance between the corresponding vertices. Consider a complete graph with vertices $v, u1, \ldots u6$. Assume that a vertex v in a MST has degree 7. Let its neighbors in the MST be $\{u1, ..., u7\}$. Note that the Euclidean distance between pair $(u2, u3)$ is less than the distance between $(v, u2)$ or $(v, u3)$. Thus the MST will include the edge $(ui, ui + 1)$ rather than $(v, u2)$ or $(v, u3)$.

is connected. We will show that this condition is sufficient as well. Assume that the physical connectivity graph is connected. Consider a new graph formed by adding edges between all pairs of nodes in the physical connectivity 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. Clearly this graph satisfies the conditions of Proposition 1 and thus the degree of any minimum spanning tree in this graph is less than or equal to 6 by Proposition 1. Any minimum weighted spanning tree in the physical topology graph is also a minimum weighted spanning tree in the completion graph. This follows from the following facts: (a) all edges in the completion graph with weight less than a certain real number d belong to the physical topology graph and (b) the physical topology graph is connected. Thus any minimum weighted spanning tree in the physical topology graph has degree less than or equal to 6. Thus such a minimum weighted spanning tree satisfies the desired degree constraints, and is a bipartite graph by virtue of being a tree. Hence, any minimum weighted spanning tree in the physical topology graph is a connected logical topology graph which satisfies all the desired 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 minimum weighted spanning trees 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 minimum weighted spanning trees in each component of the physical topology graph. It is thus sufficient to show that this sub-graph satisfies all the constraints of a logical topology graph. Now, consider each component of the physical topology graph separately. Since each component is connected, by Lemma 1 the minimum weighted spanning tree in the component satisfies all the constraints of a logical topology graph. It follows that a collection of such disjoint minimum spanning trees satisfy the constraints of a logical topology graph as well. □

It follows from the results in Lemmas 1 and 2 that constructing a minimum weighted spanning tree in the physical topology graph will provide a logical topology graph which (a) is connected if connectivity is feasible and (b) consists of minimum number of components if connectivity is not feasible. It is interesting to observe that a minimum weighted spanning tree construction algorithm has a complexity of only $O(E \log V)$ if the physical topology graph has E links and V nodes, whereas the construction of the logical topology graph is NP-hard in the general case (i.e., without the assumptions made in this section). Clearly a minimum weighted spanning tree can be constructed by distributed computation at the nodes, e.g., Kruskal's algorithm [2] for constructing minimum weights spanning trees can be distributized. A node only needs to know an ordering of the weights of its incident edges. In the Bluetooth setting, a node acquires this knowledge while synchronizing with its neighbors. During this time, a node can measure 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 which is closer.

We would like to point out that the full message sequence required by the distributed construction of an MST in the context of Bluetooth still needs to be worked out. The same observation holds for addressing a dynamic scenario. For example, new nodes may join and existing nodes may leave the system. Nodes may be continuously on the move, and thus the neighbor set and the Euclidean distances between neighbors change. Thus the spanning tree needs to be updated in response to these topology alterations. There are efficient algorithms for dynamic update of spanning trees [12], and we believe that a practical solution can be devised, even if once again, formulating a complete message exchange protocol remains a topic of future work. In this paper, we focus on subsequent algorithmic challenges.

The design of a logical topology is not complete without assigning master/slave/bridge roles to the nodes. Since a minimum weighted spanning tree 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. A piconet consists of a master and the nodes which have link with the master in the spanning tree. Any node in the slave set with degree more than one is a bridge. If a bridge node has links with nodes $n_1, \ldots, n_p$ then it serves as a bridge between the piconets whose masters are $n_1, \ldots, n_p$. As mentioned before, a node should not serve as a bridge in a large number of piconets, i.e., the degree of the bridge nodes should be minimized as far as possible. The slave set can then be chosen with this objective. More formally, let the degree of a partition be the maximum degree of any vertex in the partition. The partition with the least degree can be chosen as the slave set.

We now present a number of experimental evaluations that were carried out to explore the characteristics of the topologies created by the MST algorithm, as well as a couple of other simple algorithms. The motivation for those experiments and for investigating additional algorithms besides the MST algortihm, was to examine the actual degrees of nodes in the generated topology. This is all the more important, as while the MST algorithm guarantees that the resulting topology meets the degree constraint on a master, it does not explicitly take into account the need to keep the degree k of bridge nodes as small as possible. The experiments are directed towards determining whether the degree guarantee is conservative, i.e., all nodes typically have a much smaller degree than 6, or whether bridges end-up belonging to multiple piconets in most cases. We investigate topologies generated by three different algorithms:

1. The MST algorithm, which gives guarantees on the maximum degree.
2. A depth first search (DFS) based spanning tree topology construction algorithm.
3. A breadth first search (BFS) based spanning tree topology construction algorithm

The DFS(BFS) based algorithms construct DFS(BFS) spanning trees, and subsequently the maximum degree partition is chosen as the master set. Both algorithms cater to distributed and dynamic operations. However, neither provides any analytical guarantee on the maximum degree of a node. Our goal is to compare the degrees attained by these algorithms in practice with those attained by a MST algorithm, and emerge with concrete recommendations.

In order to evaluate the performance of these algorithms, we consider networks of two types in our experiments. The first type consists of nodes uniformly distributed in a square of size 1 unit. The second type consists of a "clustered topology" of three separate clusters of nodes. The positions of the centers of the clusters are selected randomly in the square of size 1. A node may belong to one of the three clusters, or it may not belong to any cluster at all. These 4 events are taken to be equi-probable. If a node belongs to a cluster then it is uniformly distributed in the cluster square (of side 0.4) around the cluster center, else its position is uniformly distributed in the original square of size 1 unit. For each of these two types of networks, we evaluate the performance of the algorithms with different number of nodes $(25, 50, 100)$ and two different transmission radii (0.4 and 0.6 units). The results have been tabulated in Table II (uniform distribution) and Table III (clustered distribution). The symbols used in all tables in this paper have been explained in Table I.

N	Number of Nodes
d	transmission radius of each node
M	Maximum degree of the masters
B_a	Average degree of the bridges
B_m	Maximum degree of the bridges

TABLE I
LEGEND FOR SUBSEQUENT EXPERIMENTAL RESULTS

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	7	2.6	6	21	4.8	7	36	7.3	20
MST	3	2	3	3	2.3	4	4	2.23	0.4
DFS	3	2	2	3	2	2	3	2	3

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	21	2.5	6	47	5.6	7	67	7.4	9
MST	3	2.4	3	3	2.3	3	3	2.3	3
DFS	3	2	2	2	2	2	3	2	2

TABLE II
THE TABLE EXHIBITS THE DEGREES OF THE NODES IN TOPOLOGIES FORMED BY DIFFERENT ALGORITHMS, WHEN THE NODES ARE UNIFORMLY DISTRIBUTED IN A $2-$DIMENSIONAL UNIT SQUARE AND HAVE THE SAME POWER LEVEL.

The MST algorithm generates topologies where the masters and the bridges have low maximum degrees. We

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	17	4	8	32	5.6	6	44	8.3	18
MST	3	2.2	3	3	2.3	3	3	2.2	3
DFS	3	2	2	3	2.2	3	2	2	2

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	21	2.5	6	47	5.6	7	67	7.4	9
MST	3	2.4	3	3	2.3	3	3	2.3	3
DFS	3	2	2	2	2	2	3	2	2

TABLE III
THE TABLE EXHIBITS THE DEGREES OF THE NODES IN TOPOLOGIES FORMED BY DIFFERENT ALGORITHMS, WHEN THE NODES ARE CLUSTERED IN A $2-$DIMENSIONAL PLANE AND HAVE THE SAME POWER LEVEL.

observe that the maximum degree of bridges is between 2 and 3, which is substantially lower than the analytical guarantee of 6 for all nodes. As expected, the maximum degree of both masters and bridges are high in topologies generated by BFS. This is because the basic construction step of the algorithm is geared towards generating "bushy" trees, which clearly contributes to generating trees of very high degree. Thus the BFS based algorithm can not be used "as is." The DFS algorithm in contrast has a basic construction step that keeps node degrees minimal for as long as possible, and we therefore, expected its performance to be much better. Indeed, it actually attains slightly lower degrees than MST for both masters and slaves, even though it does not provide any analytical guarantee. Thus although one can easily construct examples in which DFS fails to provide a low degree spanning tree, we expect that in most practical scenarios, the degree of the tree produced by DFS will have lower degree than that of an MST algorithm. Another advantage of using the DFS algorithm is that there is no need to perform any power control in order to calculate (or rank) the Euclidean distances between the communicating nodes. We also note that the degree in the case of BFS depend heavily on the number of nodes and on the transmission radius. On the contrary, the degrees obtained by the MST algorithm and by DFS appear to be robust to the choice of these parameters. We recommend the use of DFS based topology construction in this case, if the objective is to generate connected topologies where masters have degree less than 8 and the bridges have degree around 2 or 3 (note that a bridge has degree at least 2).

If all nodes have low degrees, then the end to end path between certain nodes may be long, and this may not be desirable from delay considerations. Thus one may wish to have somewhat larger piconets (desired piconet size can be a design parameter). This calls for algorithms which can tune the degree of masters to a certain desired value, and degree of bridges to a different, and possibly lower value. None of the algorithms studied in this section allow us to distinguish between masters and slaves, and selectively decrease the degrees of the bridges further, once the universal degree constraint of 7 is satisfied. In the next two sections, we propose algorithms which can accommodate such a discriminatory treatment, and more important, are capable of generating connected topologies in cases where the assumptions of this section do not hold, i.e., higher dimensionality and relaxed power levels assumptions.

VII. Topology Formation Algorithms for Networks with Nodes in $3-$Dimensional Space

In this section, we consider networks with nodes located in a $3-$dimensional space. We still assume that every node has the same communication range. In effect, we relax the $2-$dimensional assumption made in the previous sections.

We focus on designing a topology where a node does not have degree more than 7 (i.e., a master can not have more than 7 slaves and a bridge participates in at most 6 piconets). Robins *et. al.*[13] showed that the degree of a minimum weighted spanning tree can be as large as 13. Thus unlike the $2-$dimensional case a MST based algorithm is not guaranteed to satisfy the degree constraint of the masters. The problem needs to be investigated in the framework of a minimum degree spanning tree, but as discussed in Section V this is an instance of an NP-complete problem. We will investigate heuristics and approximation algorithms for this purpose.

We first investigate the performance of a minimum weighted spanning tree in this case. As before, we consider the physical topology graph. The edge weights are Euclidean distances between the corresponding nodes. Nodes construct a minimum weighted spanning tree with the knowledge of a relative ordering of the weights of the links originating from the nodes. The maximum degree partition is chosen as the master set. We experimentally investigate the degrees of the masters and the bridges for this algorithm. In addition we also study the performance of DFS and BFS based spanning trees in this regard.

The network in the 3-dimensional scenario is constructed as follows: we first distribute the nodes on a plane exactly as described in the 2-dimensional case. Then we assign a z-coordinate to each node, which is uniformly distributed between 0 and 0.3 units. This leads to two distinct cases: nodes uniformly distributed in

3−dimensional cuboids (uniform topology) and nodes distributed in 3−dimensional clusters (clustered topology). The results have been tabulated in Table IV (uniform distribution) and Table V (clustered distribution).

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	5	2.8	8	11	4.3	13	27	6.3	17
MST	3	2.3	3	4	2.2	5	4	2.5	3
DFS	3	2.1	3	3	2.1	3	3	2	2

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	17	3.3	4	30	7.3	8	85	5.6	27
MST	3	2.3	3	4	2.5	3	5	2.2	4
DFS	3	2	2	3	2.1	3	3	2	2

TABLE IV
THE TABLE EXHIBITS THE DEGREES OF THE NODES IN TOPOLOGIES FORMED BY DIFFERENT ALGORITHMS, WHEN THE NODES ARE UNIFORMLY DISTRIBUTED IN A 3−DIMENSIONAL CUBOID ($1 \times 1 \times 0.3$) AND HAVE THE SAME POWER LEVEL.

The interesting thing to note is that although one could expect that the algorithms would result in spanning trees of much larger degree than they did in the 2-dimensional scenario, this is not the case. Although neither DFS nor the MST algorithm provide reasonable analytical guarantees on the maximum degree of the resulting spanning tree, in both cases the degree is much less than 7 (the Bluetooth degree constraint). BFS produces topologies with higher degrees. We also note that as in the 2-dimensional case, the degrees in the case of BFS depend heavily on the number of nodes and on the transmission radius, while the degrees obtained by the MST and DFS algorithms remain similar for different number of nodes.

None of the algorithms presented so far give any analytical guarantee on the degrees of the nodes in the 3− dimensional case. In addition, none of these have the potential of controlling the degrees of the masters and the bridges differentially (even for the 2−dimensional case). As discussed before, this will be essential if we are not satisfied with "any" connected topology which satisfies the degree constraints, but need to impose other performance considerations like delay etc. Now we present a topology design procedure based on an approximation algorithm which is guaranteed to generate a spanning tree with degree at most one more than the minimum possible value in any arbitrary graph[8], and which provides a "knob" to tune the degrees of masters and bridges differentially. We explain

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	16	3	7	20	4.6	7	42	8.5	10
MST	4	2.1	4	3	2.4	3	4	2.5	4
DFS	3	2	2	4	2.1	3	3	2	2

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	24	3.9	4	38	3.4	5	66	5.1	9
MST	4	2.1	3	4	2.4	3	4	2.4	4
DFS	3	2	2	2	2	2	2	2	2

TABLE V
THE TABLE SHOWS THE DEGREES OF THE NODES IN TOPOLOGIES FORMED BY DIFFERENT ALGORITHMS, WHEN THE NODES HAVE A CLUSTERED DISTRIBUTION AND HAVE THE SAME POWER LEVEL.

the analytical guarantees in this case more formally. Let the degree of the spanning tree generated by this algorithm be d. Then any other spanning tree will have a maximum degree of $d - 1$ or more. In the Bluetooth context, this means that if $d \geq 9$ then any connected logical topology will have at least one master with more than 7 slaves or one bridge participating in more than 7 piconets which violates the degree constraints, and as such connectivity is not feasible. If $d \leq 7$, then connectivity is feasible, and the spanning tree generated by this algorithm is a valid logical topology. If $d = 8$, then connectivity may or may not be feasible, and any connected logical topology will have at least one piconet with 7 slaves. Thus the "gray area" where this algorithm may fail, and yet connectivity be feasible is for only one value of d, $d = 8$. We denote this algorithm as the "MDST" algorithm. We describe it next, assuming that the physical topology graph is connected. Otherwise, connectivity is not feasible and the algorithm operates on each component of the physical topology graph. The basic approach[8] is to start with any spanning tree, and replace edges from vertices of high degree with those from vertices of low degree.

1. Find any spanning tree T of the physical topology graph.
2. Determine the maximum degree d of spanning tree T.
3. Mark all vertices of degree d and $d - 1$ as "bad." Consider a 'forest' $F = T \setminus \{$ bad vertices $\}$ (a forest is a collection of disconnected trees). The vertices in F are marked "good."
4. While there is an edge e connecting two different components of F,

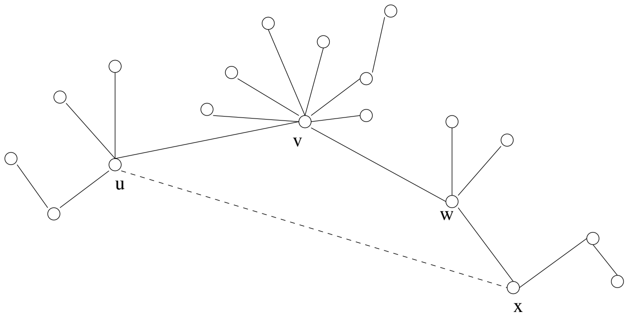

Fig. 3. We explain the operation of the MDST algorithm in this figure. Let MDST algorithm start with the spanning tree shown in the figure. Node v has degree 8 while all other nodes have degree less than 5. So, node v is marked as 'bad', and all other nodes are marked as 'good' (since their degree is less that $d-1 = 8-1 = 7$). 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).

(a) Consider the cycle C generated by spanning tree T together with e. If C has a vertex w of degree d, denote the edge in C incident on w as l, update T as follows: $T \rightarrow T \setminus \{l\} \cup \{e\}$ and go to step (2).

(b) Mark all "bad" vertices in C as good. Update F by combining the components along the cycle C and these newly marked vertices into a single component.

5. Output the final tree T.

We present an example illustrating MDST in Figure 3.

MDST runs in polynomial time[8], $O(VE\log V)$[7] MDST "almost" identifies whether connectivity is feasible and generates a connected logical topology if connectivity is feasible in most of the cases. We use the word "almost" because as mentioned before if the MDST output tree has degree 8 then connectivity may be feasible, while the output tree violates the Bluetooth degree constraint on masters. Next, we experimentally study how often the output tree has degree 8, and we also consider the effect of degree constraints on the slaves. For this purpose we first examine the performance of MDST in this regard. Recalling, MDST chooses the spanning tree partition with lower degree as the slave/bridge set. We examine the degree of nodes in this set. We consider the uniform and the clustered topologies described before in this section. The results have been tabulated in Table VI (uniform distribution) and Table VII (clustered distribution). The results show that all nodes have degrees between 2 and 3 (closer to 2 than 3).

Now we discuss how one can extend MDST to control

[7] More precisely, the run time is $O(VE\alpha(V,E)\log V)$, 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.

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
MDST	3	2.1	3	3	2.1	3	3	2	2

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
MDST	2	2	2	2	2	2	2	2	2

TABLE VI

THE TABLE EXHIBITS THE DEGREES OF THE NODES IN TOPOLOGIES FORMED BY MDST, WHEN THE NODES ARE UNIFORMLY DISTRIBUTED IN A $3-$DIMENSIONAL CUBOID $(1 \times 1 \times 0.3)$ AND HAVE THE SAME POWER LEVEL.

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
MDST	3	2.1	3	3	2	2	3	2	2

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
MDST	2	2	2	2	2	2	2	2	2

TABLE VII

THE TABLE EXHIBITS THE DEGREES OF THE NODES IN TOPOLOGIES FORMED BY MDST, WHEN THE NODES ARE DISTRIBUTED IN $3-$DIMENSIONAL CLUSTERS AND HAVE THE SAME POWER LEVEL.

the degrees of the masters and bridges differentially. Observe that MDST algorithm reduces the maximum degree of the nodes as much as possible. However, the objective is somewhat different now. *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 subsequently preferentially reduce the maximum degree of the bridges to the desired value (k).* Reducing the degree of all nodes uniformly need not attain this. MDST decreases the degrees of all vertices uniformly, and both masters and bridges have degrees close to 2 in the resulting topology.

We proceed as follows. MDST starts with the spanning tree generated by BFS, which generates spanning trees of large degrees. Note that the trees generated by DFS or MST can not be used as starting points as these provide low degrees (2 or 3) for both masters and slaves. MDST is allowed to terminate when the maximum degree is reduced to p (the desired piconet size). Imposition of the following terminating condition in step 2 will serve the purpose: If $d \leq p$ terminate. We denote this minor modification of MDST as "M-MDST." Now we propose an extension of MDST, which we call E-MDST for reducing the degrees of the bridges selectively. This algorithm starts with the spanning tree generated by the M-MDST algorithm, and subsequently reduces the degrees of the bridges without increasing that of the masters beyond the degree constraint of p. The basic difference between MDST and E-MDST is that in E-MDST the degree considerations used for marking slave nodes "bad" are different from those for marking master nodes "bad," and the edge replacement is used to decrease the degrees of bridges only, once the overall degree constraint of p is satisfied by the M-MDST algorithm.

1. Start with the output of the M-MDST algorithm, T. 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 with edges between the master and slave sets only (i.e., eliminate all master to master and slave to slave links).
2. Determine the maximum degree d_s of nodes in the slave set of spanning tree T. If $d_s \leq k$ terminate.
3. Mark all master nodes of degree 7 and slave nodes of degree d_s and $d_s - 1$ as "bad." Consider a 'forest' $F = T \setminus \{$ bad vertices $\}$. The vertices in F are marked "good."
4. While there is an edge e connecting two different components of F,
   (a) Consider the cycle C generated by spanning tree T together with e. If C has a slave node w of degree d_s, denote the edge in C incident on w as l, update T as follows: $T \rightarrow T \setminus \{l\} \cup \{e\}$, and go to step (2).
   (b) Mark all "bad" vertices in C as good. Update F by combining the components along the cycle C and these newly marked vertices into a single component.
5. Output the final tree T.

We investigate the performance of E-MDST experimentally in order to judge its efficiency in fine tuning the topology. The experimental evaluation is necessary as there is no analytical guarantee that the desired master and bridge degrees will indeed be attained. We consider the uniform and the clustered topologies described before in this section. We first set $p = 7, k = 3$ and next consider $p = 5, k = 3$. The results for $p = 7, k = 3$ have been tabulated in Table VIII (uniform distribution) and Table IX (clustered distribution). The results show that the maximum degree of masters is exactly 7. The maximum degrees of bridges is between 3 and 4, and the average degrees of bridges are between 2 and 3. The results for $p = 5, k = 3$ have been tabulated in Table XI (uniform distribution) and Table X (clustered distribution). The results show that the maximum degree of masters is exactly 5. The maximum degrees of bridges is exactly 3, and the average degree of bridges is between 2 and 3. We conclude that the algorithm is approximating its objective fairly well.

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
E-MDST	7	2.6	4	7	2.9	4	7	2.4	3

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
E-MDST	7	2.5	4	7	2.5	3	7	2.8	4

TABLE VIII

THE TABLE EXHIBITS THE DEGREES OF THE NODES IN TOPOLOGIES FORMED BY E-MDST, WHEN THE NODES ARE UNIFORMLY DISTRIBUTED IN A $3-$DIMENSIONAL CUBOID $(1 \times 1 \times 0.3)$ AND HAVE THE SAME POWER LEVEL.

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
E-MDST	7	2.3	3	7	2.4	3	7	2.6	3

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
E-MDST	7	3	3	7	2.9	3	7	3	4

TABLE IX

THE TABLE EXHIBITS THE DEGREES OF THE NODES IN TOPOLOGIES FORMED BY E-MDST, WHEN THE NODES ARE DISTRIBUTED IN $3-$DIMENSIONAL CLUSTERS AND HAVE THE SAME POWER LEVEL.

The operation of both MDST and E-MDST indicate that the computations essentially rely on local information, and as such we envision a relatively simple extension for operation in a distributed scenario. However, the exact message exchange scenario need to be designed for this purpose. Also, examining the efficiency of the algorithm in a dynamic scenario, where topology changes continuously remains an interesting topic for future investigation.

VIII. TOPOLOGY FORMATION ALGORITHMS FOR NETWORKS WITH NODES WITH MULTIPLE POWER LEVELS

Now we assume that different nodes have different transmission/reception ranges. As specified by the Bluetooth specification, devices are classified in three power classes. These correspond to maximum output powers of 100mW, 2.5mW and 1mW, and translate to transmission

N	d	M_m	B_a	B_m
25	0.4	5	2.4	3
50	0.4	5	2.5	3
100	0.4	5	2.5	3
25	0.6	5	2.5	3
50	0.6	5	2.1	3
100	0.6	5	2.3	3

TABLE X
NODE DEGREES OF E-MDST FOR THE 3-D CASE (UNIFORM NODE DISTRIBUTION)

N	d	M_m	B_a	B_m
25	0.4	5	2.3	3
50	0.4	5	2.4	3
100	0.4	5	2.5	3
25	0.6	5	2.6	3
50	0.6	5	2.5	3
100	0.6	5	2.5	3

TABLE XI
NODE DEGREES OF E-MDST FOR THE 3-D CASE (CLUSTER NODE DISTRIBUTION)

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	20	3	6	18	7.7	10	68	6.2	21
MST	3	2.3	3	3	2.1	3	4	2.3	3
DFS	3	2	2	3	2	3	3	2	2
MDST	2	2	2	2	2	2	2	2	2
E-MDST	7	2.7	3	7	2.4	3	7	2.7	3

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	16	2	5	32	6.7	20	72	4.5	13
MST	4	2.1	3	3	2.1	3	3	2.3	3
DFS	3	2	2	3	2	2	2	2	2
MDST	3	2	2	2	2	2	2	2	2
E-MDST	7	2.8	3	7	3	4	7	2.4	3

TABLE XII
THE TABLE EXHIBITS THE DEGREES OF THE NODES IN TOPOLOGIES FORMED BY DIFFERENT ALGORITHMS, WHEN THE NODES ARE UNIFORMLY DISTRIBUTED IN A $2-$DIMENSIONAL UNIT SQUARE AND HAVE TWO DIFFERENT POWER LEVELS.

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	19	3.6	5	33	3.8	12	52	8.8	19
MST	3	2.4	3	3	2.4	3	3	2.3	3
DFS	3	2	2	3	2	2	2	2	2
MDST	3	2	2	3	2	2	2	2	2
E-MDST	7	2.4	3	7	2.8	3	7	2.2	3

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	10	4.7	12	40	10	18	82	4.4	8
MST	3	2.4	3	3	2.3	3	4	2.3	3
DFS	2	2	2	2	2	2	3	2	2
MDST	2	2	2	2	2	2	3	2	2
E-MDST	7	2.6	3	7	2.3	3	7	2.5	3

TABLE XIII
THE TABLE EXHIBITS THE DEGREES OF THE NODES IN TOPOLOGIES FORMED BY DIFFERENT ALGORITHMS, WHEN THE NODES ARE UNIFORMLY DISTRIBUTED IN $2-$DIMENSIONAL CLUSTERS.

ranges of approximately 100m, 10m and 10cm respectively. We do not expect that power class 3 (i.e. 1mW or 10cm range) will prove to be particularly useful in practical applications. Therefore we will focus on the problem of having two different transmission ranges, namely 10 meters and 100 meters.

First consider the simple case where the nodes are points in $2-$dimensional plane. We furnish a counter-example in Figure 4 to show that nodes in a minimum weighted tree in the physical topology graph, where the weight of an edge is the Euclidean distance between the corresponding nodes can have degree 8 or more. Thus a MST-based logical topology design algorithm need not satisfy the degree constraints on the master. As in Section VII, we need to revert to the minimum degree spanning tree framework, which is an NP-complete problem. Thus, we examine the performance of heuristics and approximation algorithms in this case. The network topology is generated in the same way as in the previous parts, with the following difference. Now a node is assigned a transmission radius of d with probability 0.5 and a transmission radius of $10d$ with probability 0.5. We investigate for two different values of d.

We examine the performance of MST, DFS based spanning tree, BFS based spanning tree, MDST and E-MDST. Incidentally, the analytical guarantees described for MDST in the previous section, hold for arbitrary graphs, and as such apply in this case as well. Thus, the maximum degree of the output spanning tree of MDST is at most one more than the minimum possible value.

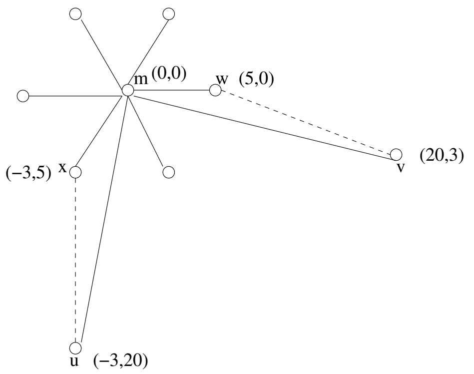

Fig. 4. We provide a counter example to establish that a MST in a physical topology graph can have degree more than 6 if nodes have two different power levels. The figure shows certain nodes in a two dimensional plane and indicates their coordinates. Suppose the transmission range of nodes m, v and u is 100 meters, while that of all other nodes is 10 meters. Thus edges (w, v) and (x, u) are not included in the physical topology graph since the vertices v, w (u, x) have a distance greater than 10 meters. Similarly nodes u and v have a distance of more than 100 meters and thus edge (u, v) is not in the physical topology graph. The figure (solid lines) shows the MST in this case. Node m has a degree 8.

Hence, MDST satisfies the master degree constraints most of the times (provided connectivity is feasible). The results have been tabulated in Tables XII and XIII. The tables demonstrate that MST, DFS and MDST satisfy the degree constraint of the masters. In fact the degrees of masters are well below 7 for all three algorithms. Bridges have degrees between 2 and 3 in all three algorithms. Degrees of nodes are somewhat less for DFS and MDST as compared to MST.

E-MDST is directed towards attaining topologies where the maximum degree of masters is 7 ($p = 7$) and the maximum degree of bridges is 3 ($k = 3$). Tables XII and XIII show that the degree of the masters is, as desired, exactly 7, while bridges have degrees between 2 and 3. Considering that the algorithm has a two-fold objective (reducing the degree of the bridges and fine-tuning the degrees of masters and bridges around specified numbers) it performs reasonably well. The maximum desired degree of bridges was set to 3 and it was well approximated by E-MDST. Also, the maximum degree of the masters was set to 7, a constraint that was easily satisfied.

Now, we consider the more general case where nodes are points in $3-$dimensional space. Clearly determining connectivity and designing a logical topology is a harder problem in this case, and subsequently the previous complexity result apply. We study the performance of all the previously mentioned algorithms below. The results have been tabulated in Table XIV (uniform distribution) and Table XV (clustered distribution).

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	16	4	5	30	6.7	6	67	5.8	11
MST	4	2.3	3	4	2.4	3	4	2.6	3
DFS	3	2	2	2	2	2	3	2	2
MDST	3	2	2	2	2	2	2	2	2
E-MDST	7	2.6	3	7	2.5	3	7	2.9	4

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	7	2.8	3	7	2.9	4	7	2.7	3
MST	4	2	3	4	2.2	3	4	2.5	4
DFS	3	2	2	3	2	2	2	2	2
MDST	2	2	2	2	2	2	2	2	2
E-MDST	7	2.8	3	7	2.9	4	7	2.7	3

TABLE XIV
NODE DEGREES OF 3-D TOPOLOGY FORMATION ALGORITHMS FOR 2 POWER LEVELS (UNIFORM NODE DISTRIBUTION)

	d=0.4								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	10	4	5	42	4.5	8	50	6.9	15
MST	3	2.3	3	4	2.3	3	4	2.4	4
DFS	3	2	2	3	2	2	3	2	2
MDST	3	2	2	2	2	2	2	2	2
E-MDST	7	2.5	3	7	2.5	3	7	2.9	4

	d=0.6								
N	25			50			100		
	M	B_a	B_m	M	B_a	B_m	M	B_a	B_m
BFS	22	2	5	29	6	12	50	10.8	33
MST	3	2.5	3	4	2.4	3	4	2.2	4
DFS	3	2	2	3	2	2	2	2	2
MDST	2	2	2	2	2	2	2	2	2
E-MDST	7	2.8	3	7	2.4	3	7	2.9	4

TABLE XV
NODE DEGREES OF 3-D TOPOLOGY FORMATION ALGORITHMS FOR 2 POWER LEVELS (CLUSTER NODE DISTRIBUTION)

In the 2-power-level 3-dimensional scenario, the performance of the algorithms is similar to that in the previous case. Again, DFS and MST succeed in reducing the degrees of both masters and slaves to values between 2 and 4 (with DFS attaining slightly lower degrees), while MDST achieves even lower degrees (equal to 2 in most cases). In this case the objective of the E-MDST was to attain topologies where the maximum degree of masters is 7 ($p = 7$) and the maximum degree of bridges is 3 ($k = 3$). The results show that this objective is mostly attained, with one exception where the maximum degree of the bridges is 4.

IX. Conclusion

We summarize the results of the investigation in this paper. The MST algorithm is the only one that is guaranteed to automatically satisfy all degree constraints in the 2-D scenario with one power level. However, as the simulations show, the degree constraint is easily met by DFS as well in $2 - D$, $3 - D$ and the two power level cases, although no analytical guarantees can be given. From a simple connectivity standpoint DFS satisfies the degree constraints for both masters and bridges. In addition, it has certain benefits not provided by other algorithms. It does not use the Euclidean distances and therefore does not require any power control. DFS has running time is smaller than that of MST or MDST and finally it is very easy to implement. However, from a delay/throughput point of view it need not always be the case that minimal degrees for both masters and slaves is desirable. Therefore, the introduction of E-MDST provides us with a means for tuning the degrees of both masters and bridges, and therefore controlling the structure of the resulting network.

References

[1] P. Bhagwat and R. Seigal. A routing vector method (RVM) for routing in Bluetooth scatternets. In *IEEE International Workshop on Mobile Multimedia Communications (MoMuC'99)*, San Diego, CA, November 1999.

[2] T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 1990.

[3] A. Das, A. Ghose, A. Razdan, H. Saran, and R. Shorey. Enhancing performance of asynchronous data traffic over the Bluetooth wireless ad-hoc network. In *Proceedings of INFOCOM'2001*, Anchorage, Alaska, April 2001.

[4] S. Basagni G. Zaruba and I. Chlamtac. Bluetrees-scatternet formation to enable bluetooth based ad-hoc networks. *ICC*, 2001.

[5] M. R. Garey and D. S. Johnson. *Computers and Intractability*. Freeman, 1979.

[6] R. Guerin, E. Kim, and S. Sarkar. Bluetooth technology: Key challenges and initial research. *Conference on Network and Distributed Simulations*, 2002.

[7] F. Harary. *Graph Theory*. Addison-Wesley, Reading, MA, 1969.

[8] D. Hochbaum. *Approximation Algorithms for NP-hard Problems*. PWS Publishinga Company, 1995.

[9] N. Johansson, U. Korner, and L. Tassiulas. A distributed scheduling algorithm for a Bluetooth scatternet. In *Proceedings of ITC'2001*, Salvador da Bahia, Brazil, December 2001.

[10] B. Miller and C. Bisdikian. *Bluetooth Revealed: The Insider's Guide to an Open Specification for Global Wireless Communications*. Prentice-Hall, 2000.

[11] C. Monma and S. Suri. Transitions in geometric minimum spanning trees. *Discrete and Computational Geometry*, 8(3), 1992.

[12] K. Siu P. Narvaez and H. Tzeng. New dynamic algorithms for shortest path tree computation. *IEEE/ACM Transactions on Networking*, 8(6), 2000.

[13] G. Robins and J. Salowe. On the maximum degree of minimum spanning trees. *Proceedings of ACM Symposium on Computational Geometry*, 1994.

[14] T. Salonidis, P. Bhagwat, L. Tassiulas, and R. Lamaire. Distributed topology construction of Bluetooth personal area networks. In *Proceedings of INFOCOM'2001*, Anchorage, Alaska, April 2001.