International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-1 Issue-5, October 2012

Research on Bluetooth Technology

Aanchal Chanana, Harvinder Singh, Dipesh Miglani, Rahul Khemani, Siddharth Kathuria

Abstract: This paper introduces a number of problems faced by Bluetooth technology when attempting to use it for building ad hoc networks. The paper provides a brief overview of Bluetooth and describes some of the major issues that need to be addressed, if it is to be successful as a network technology. Some important objectives that any solution must meet are also introduced and motivated. An initial exploration of some key issues such as topology formation and throughput maximization is also provided.

Keywords: Bluetooth, network, piconet, ad hoc networks, piconet size, nodes.

I. INTRODUCTION

Bluetooth is a recently proposed standard for short range ,low power wireless communication.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, sceneries., 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 many major technical hurdles to cross before this promise can be realized. This paper describes some of the key technical challenges that the Bluetooth technology faces and needs to overcome, if it is to fulfill its potential of becoming more than a wire replacement solution.

Although the paper includes some initial research results in this area, it is primarily intended as an overview and possible road map of some of the major issues that must be tackled.

This paper is organized as follows. We briefly describe the salient features of the Bluetooth technology in Section II. We describe key technical challenges that need to be addressed for its successful deployment in large scale ado networks in Section III.

We discuss certain design objectives in Section IV, and briefly review the existing research in Section V. We Describe our research approach in overcoming these challenges and provide some initial results in Sections VI and VII. We conclude the paper in Section VIII.

II. BLUETOOTH OPERATION

In this section we, briefly describe the features of Bluetooth network.Nodes are organized in small groups called piconets. Every piconet has a leading node called "master," andother nodes in a piconet are referred to as "slaves." A nodemay belong to multiple piconets, and we refer to such a nodeas a "bridge." A piconet can have at most 7 members. Every communication in apiconet involves the master, so that slaves do not directly communicate with each other but instead rely on the master as atransit node. In other words, Bluetooth provides a half-duplex communication channel. Communication between nodes in different piconets must involve the bridge nodes. A bridge nodecannot be simultaneously active in multiple piconets. It is activein one piconet and "parked" in others.

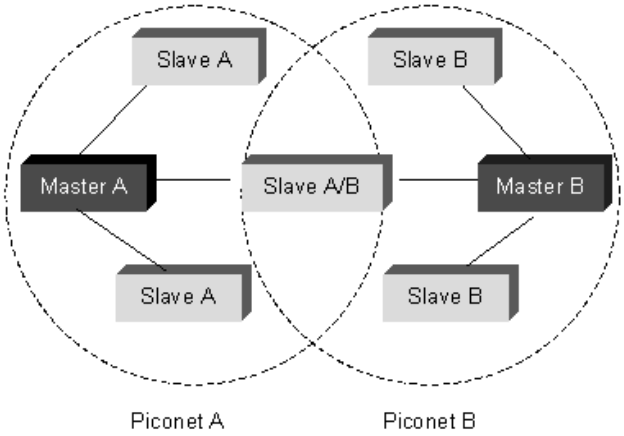

Fig.1. An example in Bluetooth topology is illustrated. The nodes are organized into 2 piconets. The masters of these piconets are master-A and master-BM1; respectively.The remaining nodes are the slave nodes or bridge nodes. Slave nodes (A AND B) can communicate via master-A and master-B respectively.

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.

Revised Manuscript Received on 30 October 2012
*****Correspondence Author(s)
Aanchal Chanana*, Department of Computer Science and Engineering. Dronacharya College of Engineering, Gurgaon, India.
Harvinder Singh, Department of Mechanical Engineering. Dronacharya College of Engineering, Gurgaon, India.
Dipesh Miglani, Department of Mechanical Engineering. Dronacharya College of Engineering, Gurgaon, India.
Rahul Khemani, Semester, Department of Computer Science and Engineering. Dronacharya College of Engineering, Gurgaon, India.
Siddharth Kathuria, Department of Computer Science and Engineering. Dronacharya College of Engineering, Gurgaon, India.

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. The master decides the communication order (and duration) for the slaves. Besides the operation and constraints associated with the Bluetooth communication channel, another key aspect in the context of ad hoc networking is the piconet formation process. A node discovers the nodes in its vicinity through the periodic use of an inquiry process that involves transmission using a well-known frequency hopping sequence. The inquiry process has two main states: A transmit state and a scan state. In the transmit state, a node continuously transmits its identity as it hops on the different frequencies of the inquiry hopping pattern. Transmission on each frequency is followed by a short listening period to determine if any node is responding to the inquiry. Nodes will be able to respond if at the time they have their receivertuned to the frequency of the hopping sequence currently used by the transmitter. However, because there is no coordination between nodes, there is no guarantee that two nodes engagedin the inquiry process will be able to hear each other. For one,they could both be in either transmit or scan mode. Furthermore,even when one is transmitting and the other listening, their lack of clock synchronization means that they may not be using the same frequency at the same time. Thus, in order to facilitate synchronization, the sender hops through the frequencies of the frequency hopping sequence faster than the receiver.Once the receiver has learned the identity of a new node as aresult of the inquiry process, it transmits its own identity. Subsequently,if either one of the two nodes decides to involve the other in a piconet in which it is the master, it pages for the other node. The paging message is transmitted on a frequency hopping sequence intended for paging, and is derived from the address of the desired recipient. If the paged node is scanning the same frequency as that on which the paging node is transmitting, then the two synchronize and the recipient receives the information required to join the piconet. Once again the transmitter switches frequencies at a faster rate than the receiver to facilitate the synchronization. Once two nodes belong to the same piconet, their clocks are synchronized and they use the same frequency hopping sequence to exchange information.

III. CHALLENGES IN BLUETOOTH DESIGN

The Bluetooth specifications have left several design issues open to implementation, when it comes to its use as a networking technology. The objective is to allow designers flexibilityso as to cater to the individual network requirements. However for adapting the technology towards large scale deployment in adhoc networks it is imperative that there be a systematic proce procedure for attaining some of the most common design objectives.

We first examine the open issues and then discuss why these need to be carefully “nailed down” in order to satisfy certain universal design objectives.A predominant open issue is how to decide which nodes become masters, slave and bridges. In Bluetooth, nodes are assumed physically equivalent with respect to their Bluetooth capabilities, so that the master and slave states are purely logical.This is a useful feature in the context of adhoc networks where nodes will likely be reasonably homogeneous, but it also introduces several problems. This is because the decision for a node to become slave or master affects the connectivity that will be available to other nodes. In addition, a node needs to decide the number of piconets it should join, and when multiple choices are possible, which subset of piconets to choose. This latter issue arises because a node may have several masters within its communication range. Note that the master of one piconet can participate as a slave in another one.

There are multiple facets to the decision of how many piconets a node should join. On one hand, bridge nodes that belong to multiple piconets improve connectivity, which reduces the number of communication hops needed to transfer data between any two nodes and can, therefore, improve overall throughput.

On the other hand, the larger the number of piconets a node joins, the larger the associated processing, storage, and most important, communication overhead. This is because a node needs to store certain information about each of the piconets it participates, and furthermore can only be active in one piconet at the time. Specifically, at any one time a node can be active in one piconet and must be parked in the other piconets to which it belongs. Switching from one piconet to another involves a non-negligible processing overhead. In addition, while involved in communications in one piconet, a node is unavailable for communications in all the other piconet. This can also affect throughput, albeit this time negatively, as the participation of one node in multiple piconets proportionally reduces the capacity available for communications between any two of the piconets to which it belongs. Note that the impact of this constraint also depends on whether the node is involved in piconets only as a slave, or whether it is the master of one of the piconets. In the latter case, any period during which the node is acting as a slave in some piconet, corresponds to a communication blackout for all the slaves of the piconet for which it serves as a master. Intuitively, this is an undesirable effect, even if its magnitude depends on the number of nodes involved in the affected piconet. As a matter of fact, the number of slaves that a piconet should have is itself an open issue. The Bluetooth specification imposes an upper bound on this number (7), but performance considerations should also be taken into account. For one, as discussed above, the number of piconets in which a master participates should be different from that of a slave. In general, even in the absence of any other constraints, e.g., assuming all nodes are capable of communicating with all other nodes, the best (throughput wise) configuration in terms of masters, slaves, and bridges is unclear. Having as few masters as possible can increase the number of nodes that are reachable either directly or in a small number of hops. However, it also means that more nodes are sharing the communication channel associated with each master. Similarly, the number of bridge nodes that should exist between different piconets is also unclear. Many bridges can facilitate load distribution and improve connectivity,

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

but this comes at the cost of increasing the complexity of synchronizing communication schedules an added overhead when switching from piconet to piconet (recall that a node can be active in only one piconet at the time). Assuming that some initial answers can be given to the problems we have just outlined, it is unlikely that static solutions will be sufficient in the context of adhoc networks. As a result, yet another level of complexity gets added when trying to determine when and how node states should change over time. Some of the factors that need to be taken account include the activity status of nodes themselves, e.g., a node may suspend its activities for some time and enter a "sleep" state in order to save power. Another aspect that needs to be considered is the dynamic nature of adhoc networks, where the number and position of nodes involved is likely to continuously vary. Obviously, node configuration and network topology need to evolve in response to such changes. For example, if a master moves out of the transmission range of its slaves, then the piconet must identify a new master, presumably from the existing members of the piconet, and this calls for a change of state from slave to master in one of the nodes. The difficulty in making such decisions, even in the simple example just outlined, is in their distributed nature. Deciding when and which node state to change requires a significant amount of information exchange and processing, which besides its intrinsic cost can also translate into substantial latency. As a result, any feasible solution calls for some trade-off between "optimality" and its cost and responsiveness to changes. For Bluetooth to succeed as a technology on which adhoc networks can be built, it is not only essential to find light-weight solutions to the above problems, but those solutions must be fully distributed. In other words, they should not assume the existence of a central entity with access to the entire system/network state, and nodes decisions should only be based on information about their own state and that of their "neighbors." However, the definition of what a node's neighborhood consists of is itself not clear. Does it consist only of nodes belonging to the

same piconet(s), or does it also include other nodes within communication reach? More generally, a neighborhood could be defined as all nodes that are k or less "hops" away (hop count corresponds to the number of masters/piconets that need to be traversed). Clearly there is a trade-off between the accuracy (or optimality) of the decisions that can be made under different scenarios. In general, the more information is available, the better the decisions. However, this comes at the cost of a higher latency, a higher processing cost, and a higher control overhead. It is, therefore, important to identify a design point that is both implementable and capable of providing a reasonably efficient operational solution. One of our goals is to start exploring the space of potential solutions to identify the range of available options.

Finally, another aspect that is left unspecified in the Bluetooth standard specifications, is how a node splits its time between actual data transmission, i.e., within a piconet, and neighbor discovery and piconet formation (inquiry and page states).

In a dynamic environment such as that of an adhoc network both phases are needed, but because they are mutually exclusive this again involves a design trade-off. One option is to assume two transceivers at each node, one for control and one for data transmissions, but this obviously imposes a significant cost penalty. Assuming a single, shared transceiver, it is then necessary to specify a strategy for controlling how long a node spends in each state and when it switches between them. If a node is only infrequently or for short periods of time in inquiry and page states, the latency for discovering a new node will be high, e.g., see [6] for an investigation of the many problems faced by this discovery process. On the other hand, requiring frequent and/or extended excursions in those control states will severely affect the data throughput available to a node or its neighbors (if the node is a master or a bridge). Again, this is an area where identifying practical and efficient policies is required before the Bluetooth technology can be effectively used in adhoc networks. In this paper, we focus on an initial exploration of some of the above issues that are associated with the problem of "topology formation,"when attempting to build an adhoc network based on the Bluetooth technology. These are, however, not the only issues that one would need to address in the context of a Bluetooth adhoc network, and there are many other interesting questions dealing with actual data transmission. For example, how does a master decide the order of data transmission among slaves? Also, as discussed earlier a node can be active in only one piconet at one time. How does a bridge node decide its order of participation in different piconets. The scheduling should be designed so that a master completes its communication with a bridge node while it is active in its piconet. This requires giving priority to bridge nodes as compared to ordinary slaves, and the priority of a bridge node should also depend on the number of piconets it participates in. These issues are closely related to administering different quality of service to different end nodes.

IV. DESIGN OBJECTIVES

In this section, we describe some of our design objectives in deciding how to best form Bluetooth topologies, and subsequently discuss the challenges involved in satisfying these objectives while exploiting the flexibility offered by the Bluetooth specifications. We are primarily concerned with three major objectives:

1. Connectivity,
2. Distributed operation and low overhead,
3. Throughput maximization.

Next, we briefly expand on those three objectives, and what it takes to achieve them. Maintaining end to end connectivity whenever feasible, i.e., when there exists a selection of node states (slave, bridge, master) that forms a connected topology, is obviously a desirable feature. Let us examine the challenges involved in achieving this objective 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. Similarly, 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 7This 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.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

These are constraints that any topology formation algorithm must take into account. In addition, it is not only the choice of role, i.e., master, slave, or bridge, that is important in determining connectivity, but the order in which nodes are assigned their role is also a key factor. 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 7slaves. This can possibly be fixed by having some slaves relinquish their membership in the piconet, but identifying when this is needed, e.g., connectivity might still exist between the piconets through a multi-hop path, and which node should leave the piconet, is a complex problem. Achieving connectivity is, therefore, a complex and possibly unachievable task, but it provides a benchmark against which heuristics can be evaluated.

In Section VII, we briefly review how some very basic algorithms perform as we vary a number of system characteristics. Our second design objective, namely a distributed operation and low overhead, is a must for any practical solution. Aspointed out earlier, node state changes should be triggered in response to changes in the physical topology. Figure 2 gives an example of how the roles of existing nodes need to be changed to accommodate the arrival of a new node and maintain connectivity. In many instances, detecting and adjusting to topological changes is likely to require a certain amount of communications between nodes. One approach to minimizing overhead is to seek algorithms that rely only on local information, and hence have minimal communication overhead. However, it is unlikely that such simplistic algorithms will be able to efficiently accommodate all possible scenarios. As a result, they will need to incorporate additional design objectives to compensate for their limited decision horizon. For example, a simple strategy would be to seek topologies that have significant redundancy, e.g., connectivity between piconets is achieved through multiple bridges or by having nodes serving as bridges between multiple piconets. Similarly, trying to keep piconet sizes small can improve the odds of success of local strategies. Our third design objective of maximizing throughput, while obviously desirable, unfortunately adds complexity of its own to an already complex problem. For example, the size of piconets, which plays a role in both determining connectivity and the overhead of any algorithm responsible for maintaining connectivity, also affects the throughput of the network. Consider a piconet with k slaves, and whereevery slave generates a traffic of intensity r per unit time. In such a configuration, the master needs to support a load of 2kr per unit time assuming it itself does not generate any traffic (the load on the master increases if the master generates traffic). IfT he master has a bandwidth of B, then we must have 2kr _ B and thus the nodal throughput r that the piconet supports is inversely proportional to the number of members in the piconet. This would call for keeping k small, and hence building a topology with many small piconets. On the other hand, a large numberof small piconets will lead to long end to end routes, and this in turn may overload the transit piconets and, therefore, also limit the feasible nodal throughput. In general, the selection of the "right" size for piconets depends on how traffic is distributed between nodes and where nodes are located. For example, it is obvious that if nodes A and B are within communication range of each other and need to exchange a significant amount of traffic, then they should be assigned to the same piconet. However, other simple configurations do not necessarily yield similarly simple answers. For example, assuming a set of N nodes all capable of communicating with each other and a uniform traffic pattern, the best topology for such a configuration is not obvious. Section VI provides some initial result to this problem for some basic network topologies. Another factor affecting throughput is the number of piconets a node participates in, and as discussed earlier this number should be different for masters and slaves. There are many possible options to consider, but for the sake of simplicity we propose that a master participate in only one piconet, and that a slave participate in up to k piconets, where k is, therefore, the only remaining design parameter. Realistic values for k are probably 2 or 3: This introduces further constraints on the topology construction algorithm, but they are expected to ensure minimum throughput levels in the network.

V. 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 specifications, if the Bluetooth technology is to be used in building adhoc networks. Salonidis *et. al.* presents a distributed topology construction scheme in Bluetooth networks [6]. The basic assumption behind the scheme is that all nodes are within transmission range of each other. The nodes conduct a leader election algorithm. The winner knows the identity of all nodes and uses this information design the desired topology. Thus the algorithm is not scalable if the number of nodes is large. This paper also 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 have a deterministic sequence of switching between inquiring and inquired (or paging and paged) modes. Bhagwat *et al.* presents a source routing mechanism for Bluetooth networks [1]. Das *et al.* [2] and Johanson *et al.* [4] present distributed scheduling policies for Bluetooth networks.

VI. RELATION BETWEEN THROUGHPUT AND PICONET SIZE

In this section, we ponder over the following basic design challenge: given a certain end to end throughput requirement can we design a topology which attains the desired throughput and satisfies the Bluetooth topology constraints. In general, this is a complicated combinatorial problem, as there are several degrees of freedom. We will thus investigate certain simple cases in order to gain some insight into this complex problem. An important design parameter in Bluetooth is the size of the piconets. Bluetooth specifications upper bound this size at 8, but it is not clear whether the size of the piconets should be close to 8 or lower. As discussed earlier, small piconets translate into more piconets, which increases the number of hops traversed by packets and thus the overall network load. Hence, small piconets increase the external or transit load on the piconetmaster, while decreasing its internal load, i.e., the load associated with communications between nodes belonging to the same piconet.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

This tradeoff implies that for a given number of nodes and traffic pattern, there exists an "optimal" configuration in terms of number of masters and slaves. We investigate this tradeoff for a linear and symmetric topology,assuming again that all nodes are capable of hearing all other nodes, i.e., there are no communications constraints on the topology formation. More formally, consider a scenario with N nodes and k equal-sized piconets. Piconets are connected linearly (i.e., piconet i shares a bridge node with piconets i □ 1 and i + 1; if 1 < i < k). Our sole design parameter is then the number of piconets k, which can range from 1 to bN=2c (we assume that a piconet must have at least one slave). We assume shortest path routing and uniform traffic demands, i.e., every node wishes to send traffic at rate r to every other node. Our goal is to determine the optimum value for the number k of piconets, and therefore the size of an individual piconet, so that the feasible node-to-node throughput r is maximized. The first step towards identifying the optimal value of k is to derive an expression, function of k, for the maximum node-tonode throughput rmax, which such a network can support. This function can then be differentiated to give the optimum value of ; which in turn specifies the optimum size of a piconet. Due to lack of space we simply sketch out the steps of this derivation and its ultimate result. The node-to-node traffic r directly determines the total traffic load carried by a master, and from [3] we know that a load is feasible as long as it is less than the associated channel capacity.

Hence, the maximum node-to-node traffic rmax corresponds to the maximum value of r for which the total traffic load on the master remains smaller than its channel capacity. Under the assumption of a linear topology and uniform traffic pattern, the traffic load that a master needs to carry can be decomposed into two main components: (a) traffic that originates from or is destined to nodes in the piconet; (b) transit traffic that is routed trough the piconet. Since traffic demands are uniform and all piconets have the same size, the first quantity is the same for all piconets. Furthermore, in a linear symmetric topology, the piconet in the middle can be shown to be the one carrying the maximum amount of transit traffic. Thus, in order compute the maximum node-to-node throughput, it is sufficient to focus on this middle piconet and the impact of varying its size.

As discussed earlier, decreasing the size of the middle piconet decreases the amount of traffic that originates fromwithin the piconet and increases the amount of transit traffic that the piconet needs to carry. However, it can be shown that the increase is less than the corresponding decrease. The increase in transit traffic corresponds to traffic to and from the nodes that have been removed from the middle piconet, but not all the traffic that these nodes generate or received ends up crossing the reduced middle piconet. In particular, traffic to and from nodes that are now in the same "half" of the network as the nodes that have been removed, will not transit through the middle piconet. In contrast, when the nodes belonged to the middle piconet, all traffic to and from them consumed capacity on the transmission channel of the piconet's master. This means that the best linear topology is one in which every piconet has only two slaves, both acting as bridges to the neighboring piconet (except for the "end" piconets). This result can be established more formally and can also be shown to hold for "circular" topologies. The result, even if established only for a specific topology, indicates that small sized piconets enhance the network through put in general. However, throughput is not the only factor to consider, and other performance metrics such as end to end delay are important as well.

Many small-sized piconets lead to long end-to-end paths, and this can increase the end-to-end delay. In addition, because the operation of masters is more complex than that of other nodes, small sized piconets can also lead to a higher overall network complexity. These factors should be taken into account in practice, when choosing a piconet size.

VII. DISTRIBUTED TOPOLOGY FORMATION ALGORITHM

This section is intended as a first exploration of a possible topology formation algorithm. Our goal is not to construct a sophisticated algorithm, but instead to evaluate how a simple and lightweight solution performs under different configurations. The proposed algorithm operates using only local information, and can adapt to changes rapidly. By evaluating its performance, we seek to gain a better understanding of when and why more sophisticated solutions may be needed. Our investigation is carried using a detailed simulation model of the Bluetooth communication channel. Special attention was given to accurately account for the operation of the inquiry process and how nodes alternate between transmit and scan state. As recommended in [6], some randomization was introduced to determine when nodes switch from one state to the other. This randomization was selected so as to ensure that each spends on
average the same amount of time in transmit and scan modes. A node can have one of the following states: (i) unassigned, (ii) master, (iii)slave, and (iv) bridge. We assume that every node as a unique ID. The topology formation algorithm operates as follows:

1. Initially all nodes have unassigned states.
2. When two nodes synchronize for the first time and both are unassigned, the one with the highest ID becomesmaster, and the other node becomes a slave in the piconet of this master.
3. When two nodes synchronize and one is unassigned while the other is a master, the unassigned node joins the piconet of the master if it has less than 7 slaves.
4. When two nodes synchronize and one is unassigned while 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 unless it is already a bridge node in b piconets.
5. If two nodes discover each other and neither is unassigned, then we consider the following cases separately. If both are masters, then neither changes state. If one is a master and the other is a slave in a different piconet, then the slave joins the other piconet and becomes a bridge between the two piconets, provided the slave does not belong to b piconets. Optionally, the master may refuse the new slave if it is already has a bridge to the slave's piconet.

This simple algorithm satisfies all the topology constraints mentioned in the previous section. However, it is not clear how effective it is at meeting the performance objectives outlined in Section IV. In the rest of this section, we report some initial simulationbased results on its performance when it comes to end-to-end connectivity.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

We consider networks of two broad types for experimental performance evaluation. The first consists of nodes uniformly distributed in a square of size 1 unit ("uniform topology"). The second consists of a "clustered topology" consisting of 3 separate clusters of nodes. The position of cluster heads are selected randomly in a square of size 1 unit. A node can belong to one of the three clusters or may not belong to any cluster at all. Each of these 4 events are equi-probable. If a node belongs to a cluster, its position is uniformly distributed in a square of side 0:2 around the cluster head, else its position is uniformly distributed in the overall square of size 1 unit. For each of these cases we evaluate the performance with different number of nodes and two different transmission radii, e.g., 0:1 unit and 0:01 unit. Our performance metric is end to end connectivity or rather the number of (disconnected) components in the logical Bluetooth topology formed by the above algorithm. This number is obviously lower bounded by the number of components in the "physical topology" graph which contains an edge between a pair of nodes as long as they are within each other transmission range. Ideally, if the physical topology is connected, one would like the logical topology generated by the topology formation algorithm to to also be connected. In general, an important goal of any topology formation algorithm is to generate a number of components that is as close as possible to that of the underlying physical topology.

Thus, an important performance measure is the difference in the number of components of the logical topology and the physical topology. The smaller this difference, the better the algorithm, at least in terms of connectivity. We present our experimental results next. We first start with a scenario that consist of only 10 nodes uniformly distributed in our square of size 1 and with a transmission range of 0:1. For this scenario, the average number of components is 8:3 for both physical and logical topologies. This indicates that both these topologies display a high degree of "disconnectivity," and often include multiple nodes that are not connected to any other node. When the transmission range was further decreased to 0:01, the number of components increased to 10 for all the simulations that were run.

This is expected as the combination of a small number of nodes and small transmission rangesmakes it very likely that no node is in the connection range of some other node. As a result, such scenarios are not really meaningful data points for evaluating the performance of a topology formation algorithm. The next set of scenarios attempt to remedy this by increasing the number of nodes to 25 nodes. For the case of a transmission range of 0:1 the average number of components becomes 15:33 and 14:83 for the logical and the physical topologies, respectively, while with a transmission range of 0:01 both are equal to 24:5, i.e., most individual nodes remain disconnected. As the number of nodes continue to grow, the difference between the physical and logical topologies do increase, but less so than when considering other distributions of nodes, and in particular the clustered topology that we investigate next.

We expect the clustered topology to yield different results because the close proximity of many nodes is likely to stress the degree constraints that exist for piconets, i.e., the limitation of no more than seven slaves. As a result, while connected topologies that do not violate this constraint may exist, it appears likely that the "greedy behavior" of our simple topology formational gorithmmay not be sufficient to find them. This expectation was confirmed by the simulation results, especially for the case of a long transmission range of 0:1, which is therefore the one we investigate more extensively. For scenarios involving 100 nodes, we find that the average number of components is 29 and 11:75 for the logical and the physical topologies, respectively. This represents a significant differences in terms of the connectivity, or rather lack thereof, of the topology ultimately constructed by the topology formation the average number of components becomes 17 and 9:26 for the logical and the physical topologies, respectively, which is again significant. For 25 nodes, the average number of logical and physical components is 10:21 and 7:28. This smaller difference indicates that as the density of nodes in the area becomes low enough, the impact of the degree constraint becomes less significant, i.e., there are not enough nodes to stress it. This expected behavior is confirmed by the results for a scenario with only 10 nodes, for which the average number of components is 6:3 and 5:7 for the logical and the physical topologies, respectively. A similar result is obtained when the transmission range is decreased from 0:1 to 0:01; as this again limits the number of nodes that will be in close proximity and, therefore, avoids stressing the Bluetooth piconet degree constraint. For example, for 50 nodes the average number of components is 45:77 and 45:69 for the logical and the physical topologies, respectively. A very minor difference indeed.

In general, the results indicate that the distribution of nodes in an area can play a very significant role. In particular, simple topology formation algorithms such as the one we presented, will not fare well with distributions such as the clustered topology, which are likely to increase the likelihood of many close by nodes competing for connectivity. For such distributions, algorithm capable of "intelligently" selecting which connections to make seem to be needed. The same conclusion holds even when the transmission range is increased. A greater transmission range might offset the potential penalty of making greedy topology formation decisions, because of the broader number of connectivity options it allows.

However, from our initial results, it appears as if this potential effect is relatively minor compared to the impact of node distribution and density. Investigating algorithms that are capable of generating "good" topologies even in those more demanding conditions and without too much of an increase in complexity is a topic we are currently investigating.

VIII. CONCLUSION

This paper was intended as a brief introduction to the many challenges that the Bluetooth technology faces if it is to succeed as a technology for building adhoc networks. We have described many of the issues that need to be tackled and that have been left unspecified by the current standards. We identified a number of objectives that any solution should aim at meeting, and provided an initial investigation of some of these problems. This is obviously preliminary work, and we are actively investigating many of the problems outlined in this paper. We hope that the paper will also entice others in exploring what we feel is a promising and rich research area.

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. 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, AK, April 2001.
3. B. Hajek and G. Sasaki. Link scheduling in polynomial time. IEEE Transactions on Information Theory, 34(5), 1988.
4. N. Johansson, U. Korner, and L. Tassiulas. A distributed scheduling algorithm for a Bluetooth scatternet. In Proceedings of ITC'2001, Salvador, Brazil, december 2001.
5. B. Miller and C. Bisdikian. Bluetooth Revealed: The Insider's Guide to an Open Specification for Global Wireless Communications. Prentice-Hall, 2000.
6. T. Salonidis, P. Bhagwat, L. Tassiulas, and R. Lamaire. Distributed topology construction of Bluetooth personal area networks. In Proceedings of INFOCOM'2001, 2001.

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)