

BLUETOOTH SCATTERNET FORMATION USING PROXIMITY INFORMATION OF AN ELECTION PROTOCOL

VIVEK BHATNAGAR AND GEORGE KESIDIS
(bhatnaga@cse.psu.edu, kesidis@engr.psu.edu)

CSE and EE Departments, The Pennsylvania State University

University Park, PA, 16802

Abstract: Bluetooth devices are typically arranged as piconets, which are further organized into scatternets. The formation methodology of such scatternets, to a large extent, dictates their efficiency. In this paper, we discuss and evaluate two “tree-based” algorithms for scatternet formation. Both these algorithms use an election mechanism to form master-slave relations, enabling piconet and scatternet formation. We do not assume that all Bluetooth devices are in pair-wise communication range with each other or that the nodes are turned on simultaneously. We also compare the two algorithms and discuss their impact on the performance of the resulting Bluetooth network. We observe that the two proposed algorithms are similar in performance. While algorithm 1 (multiple election phases) requires lesser time than the algorithm 2 (single election phase), the former outperforms the latter in terms of the network diameter of the resultant scatternet.

1 Introduction

Bluetooth is a wireless communication technology that permits communication between Bluetooth-enabled devices. Bluetooth operates in the ISM band. The communication range of a Bluetooth device is between 10m and 100m, but more commonly limited to 10m-20m owing to channel noise and limited power of a typical device. Bluetooth devices within communication range can set up an ad-hoc network called a “piconet” consisting of one master that controls the piconet and a maximum of 7 slaves. The master and a slave communicate using Time Division Duplex (TDD) slots. Typically, though not necessarily, the master talks to the slave in a time-slot (625 μ s) and the slave replies to the master in the very next time-slot. The two consecutive slots where the master and one particular slave communicate with each other are called a “Bluetooth frame”. Each Bluetooth frame can be thought of as a polling epoch of the corresponding master-slave connection. The total capacity for communication for a Bluetooth piconet is about 1 Mbps.

A Bluetooth device can participate in more than one piconet. When it participates in two piconets, it may act as a bridge between them. A system of thusly interconnected piconets is called a scatternet. Several proximal piconets can simultaneously and independently operate because of Bluetooth’s frequency-hopping mechanism in which each piconet uses a different phase (offset) for its

pseudo-random frequency hopping sequence. Inter-piconet communication within a scatternet is accomplished using the bridge nodes. Bridges are either slave-slave (a slave role in every piconet to which it belongs) or master-slave (a master of exactly one piconet and a slave in the other(s)). Of course, bridges and other shared slaves must be able to synchronize to the frequency-hopping sequence of multiple piconets; they do this with the knowledge of the different sequence offset of each piconet to which they belong.

2 Issues Addressed

A primary issue in any Bluetooth scatternet formation algorithm is to be able to run that protocol in a distributed manner and at the same time take into account any asynchronous behavior of the devices (e.g., not all of them may be powered on at any given time). Other aspects that shape a scatternet formation protocol include the various constraints that the Bluetooth specification imposes on the devices (as discussed below), limited power available for communication, short communication range, and occasional mobility of these devices. Finally, there are factors of scalability to be considered when a Bluetooth scatternet formation protocol is designed, for in a dynamic environment the size of the total Bluetooth population in a scatternet may substantially vary and this should not result in any overhead propagating across the whole scatternet system. Changes to Bluetooth device population should only affect the local coordinator where the change occurs. To address the scalability issues in the scattered formation and on-going topological management protocols, we assume a hierarchical approach for both scatternet formation and data routing. So, an ideal Bluetooth scatternet formation protocol should be distributed and scalable and be able to account for device asynchronicity and occasional mobility. For example, the mobility of a node can be advertised (explicitly flagging) and then, whenever possible, it can be disallowed from becoming a piconet master to save reconfiguration time in the event of its movement.

Other important issues to be considered in a Bluetooth scatternet formation protocol include the trade-off between the maximum number of slave devices per piconet and the number of piconets in the scatternet. Finally we consider the difference in the formation issues and the routing issues for a given set of nodes, while emphasizing the former and just broaching the latter.

3 Previous work

The problem of scatternet formation in Bluetooth networks has received some attention in recent publications [1][2][6][7][8]. Some of these approaches do not make use of any additional proximity information besides what is immediately

available to the Bluetooth devices [7], while others merge the problem of scatternet formation with that of routing [8]. Many papers also impose greater restrictions on the number of slaves that a master is allowed, e.g., [7] and [8] consider piconets with masters having no more than 5 slaves. Lowering the maximum number of active slaves from 7 per piconet, results in a scatternet with higher average hop count for routes between nodes. Increasing route diversity by increasing hop-counts (route length) is a less desirable tradeoff in a wireless context with limited-power devices, such as Bluetooth. However, piconets may be limited to 6 slaves to accommodate future mobile users, such that these positions can be used by mobile nodes that can enter the piconet and leave periodically. We further illustrate how conducting a multi-phased scatternet formation provides us with additional proximity information (unlike [7]) resulting in diversity of routes among the nodes. This route diversity can be used after the principal scatternet formation is completed to form more robust scatternets from a node fault and mobility management perspective as well.

Other papers look into the concept of “clustering” in wireless (sensor) networks [10]. We believe these approaches can be extended to Bluetooth networks to improve their robustness and, more importantly, their scalability. RETRI [11] introduces the concept of transaction density, which is defined as *“the average number of transactions that occur at the same place (connectivity-wise) and at the same time in a system”*. We contend that the “transaction density” is a valuable metric to measure the scalability of a system – the transaction density of a system should remain approximately unaffected when the total population in Bluetooth devices changes in a given context.

Finally, none of the mentioned work on scatternet formation uses an “incremental” approach that gains information as successive protocol phases are executed. In the algorithms described herein, each phase consists of a group of pairwise election contests [2]. We believe that such an incremental approach is very important in a dynamic and asynchronous Bluetooth environment, due to temporal changes in the spatial orientation of the devices vis-à-vis each other. We now describe our approach in greater detail.

4 Our Approach

In this study, we describe a distributed algorithm for Bluetooth scatternet topology discovery and formation, a preliminary version of which was first presented in [1]. We also briefly discuss how information gained about the topology of the Bluetooth devices during scatternet formation can be used for efficient routing among them. We present two variations of the scatternet formation algorithm that we evaluate and compare. Both these election algorithms use the

election process described in [2] as a fundamental operation. A set of Bluetooth nodes in a given domain (room) is considered and the methods described here are applied to obtain an optimal scatternet that conforms to the stipulations of the Bluetooth standard.

We describe and employ the technique of representing small groups of proximal Bluetooth devices (piconets) as nodes of a spanning tree, the root of which is the master of the piconet. These small piconet trees become sub-trees of a larger tree that, in typical cases, represents the scatternet that these algorithms seek to discover. We also demonstrate how trees resulting from multiple election runs are incrementally pruned and merged to form successively larger scatternets. Simple tree operations are used to obtain a final spanning tree from the various sub-trees. These operations include joining trees at nodes common to them (merging), removing a child from its parent's tree if the tree grows too large (pruning) and reversing the parent child relation for two given nodes (root inversion). Because efficient algorithms for incrementally obtaining an optimal tree (according to our performance criteria) are not complex and easily programmed, we believe that using a tree representation and associated operations facilitates the process of scatternet formation.

The proposed algorithms are completely distributed in nature. We neither assume that all the devices in the "room" are in pair-wise communication range with each other nor that all the nodes are simultaneously active (turned on) at the start of an election process. In other words we assume that the nodes are asynchronous and have limited amounts of power available for communication.

An election process is a series of pair-wise elections between active nodes that are within communication range. An election process is conducted until a subset of all such possible elections have completed. The "election rules" are those that dictate when election processes start and stop and also which nodes can participate in future elections. The order in which a Bluetooth device (node) participates in an election is modeled as random. Also, any node can randomly turn on while an election process is in progress. We also identify special nodes called coordinators that are responsible for (and perhaps computationally better equipped to) coordinate the formation of the final scatternet. Finally we refer to bridge nodes as those that are in range of different sets of communicating nodes during successive election processes.

We compare two sets of election rules in this paper. In both algorithms, a node that wins an election contest gains the votes of the losing node. The node with more votes will win a pair-wise election contest unless that node has greater than 6 immediate children (slaves in its piconet). Recall that the Bluetooth standard restricts piconet membership to seven (plus master) thus allowing a node (master)

seven children (slaves) and one parent in the topological spanning tree. This restriction causes the final scatternet thus obtained to have a higher fan-out per node (more children per parent) and a low network diameter (a spanning tree with less depth), which are both desirable characteristics. By allowing the maximum number of nodes in a piconet, we ensure that the routes between nodes of the final scatternet have low hop-counts.

4.1 Election Algorithm 1:

The first algorithm relies on multiple election processes to form the final scatternet. Each election process passes on some 'partial' proximity information to the coordinators to be used to merge different piconets (sub-trees) before the next election process. A node that loses an election is not allowed to compete in any further elections (is removed from the election list) of the current election process and its vote tally is added to the winner's vote tally. In terms of the tree structure representation, the loser node becomes the child of the winner: the loser's sub-tree is "hung" on the winner's. The election process continues until a time-out occurs. Subsequent election processes are initiated by the coordinators after a brief period when all of its nodes are released. For each election process completed, the coordinator uses simple tree operations like merging, pruning and root inversion to obtain the optimal scatternet. Trees obtained after successive election processes are merged using bridge nodes.

4.2 Election Algorithm 2:

In the second algorithm, a node is allowed to continue to participate in the election process regardless of whether it won its previous election. In this case, there is only one election process and it clearly needs to be longer in terms of the number of individual election contests (duration) than an election process of election algorithm 1. Here, a losing node can win only if it competes with another losing node or with a node that can not accommodate anymore slaves. Once the election process is over, the nodes of the resulting spanning tree are grouped into piconets that are bridged together, as required.

4.3 Routing:

In both algorithms, some node proximity information is gained that may not be used for the formation of the final spanning tree (scatternet spine). However, this information is stored in part at the coordinator nodes and in full at the ultimate leader of the scatternet and can therefore be used by the master nodes for routing of data packets among the nodes. This kind of information prevents the breakdown of communication between two nodes if an intermediate node moves away or fails, as long as there is an additional known path between them. Also, the traffic can be

routed via paths that are not the shortest in terms of distance, to prevent a node that is in range with many nodes from becoming a bottleneck. In this manner we can use additional proximity information gained during the election process to prevent bottlenecks and enhance robustness (as seen in [7]) while at the same time not allowing routing decisions to affect the formation of a reasonably good scatternet (as seen in [8]).

4.4 Hierarchical Topological Management:

Besides being clearly distributed and allowing for node asynchronicity, our approaches can scale well in the event that a number of nodes join (or leave) the Bluetooth population. Scalability is achieved when the scatternet is hierarchically arranged with a global coordinator for the whole network spanning a small group (say ≤ 10) of “regional” coordinators, each of which in turn span a group of local coordinators, each of which finally span a small group of piconets. A possible drawback of this scheme is that the scatternet thus obtained may not be optimal in terms of connectivity, but we believe that this drawback is more than offset by the fact that the hitherto described tree operations are not computationally intensive and can be handled by the low power Bluetooth devices.

5 Simulation Model

To compare the performance of both algorithms by simulation, we described the devices with data structures having fields for vote tally, election trees in which they are involved, etc. We generated a randomly ordered list (election list) with pair-wise entries of nodes such that each pair is in communication range and thus eligible to contest an election amongst itself. This election list is dynamic in size: it grows when a node wakes up and shrinks when a node is removed from the election contest. Nodes are picked at random from this list to participate in an election contest.

6 Performance Criteria

We compare the two election algorithms in terms of time and computational complexity and attempt to determine the suitability of one approach over the other as a function of the number of nodes present in the room. Performance measures of the scatternets obtained for these algorithms include: the number of elections required to find the largest scatternet, the size of the largest scatternet formed in a given amount of time (measured as the total number of pair-wise election contests) and the degree of connectivity (more parent-child relations and more children per parent) which can be measured in terms of the network diameter.

7 Results (observed and expected)

We will report results of a more comprehensive simulation study in [13]. We compare our two proposed algorithms with those previously/jointly proposed using the aforementioned criteria. These criteria include the average scatternet diameter and the diversity of routes available for data routing. The following figure depicts 39 Bluetooth randomly positioned devices in a room that is 60m by 60m. Devices are assumed in communication range if they are within 15m of each other. In the following figure, the circle radii correspond to the device range (15m).

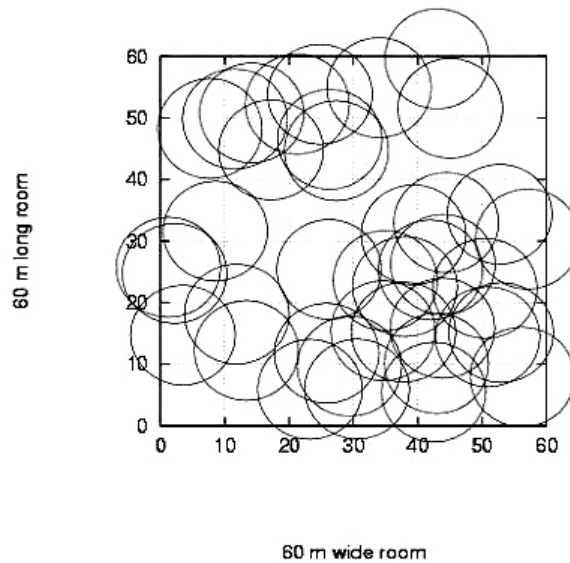
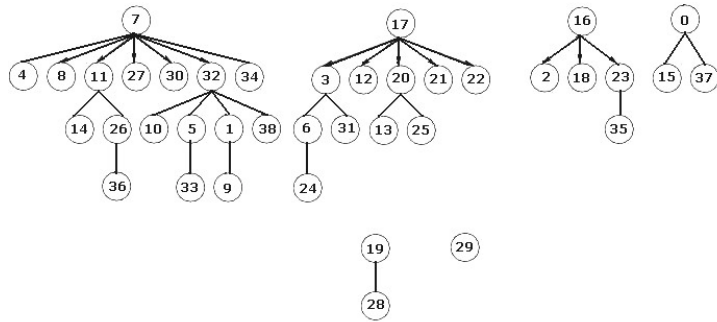
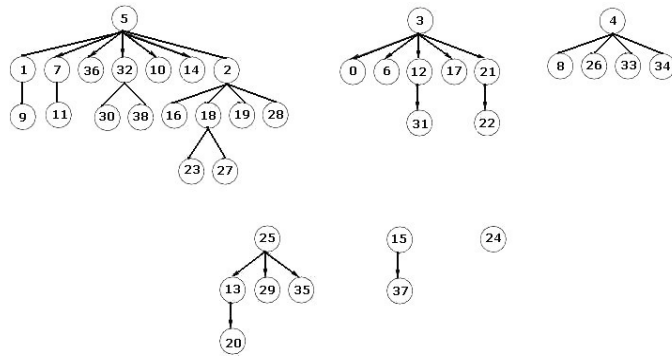


Figure 1: One Sample Bluetooth “room”.

The results reported below are for this room. The following sequence of figures depicts the formation of a scatternet according to Algorithm 1, i.e., multiple election processes, followed by Algorithm 2 (single election process).

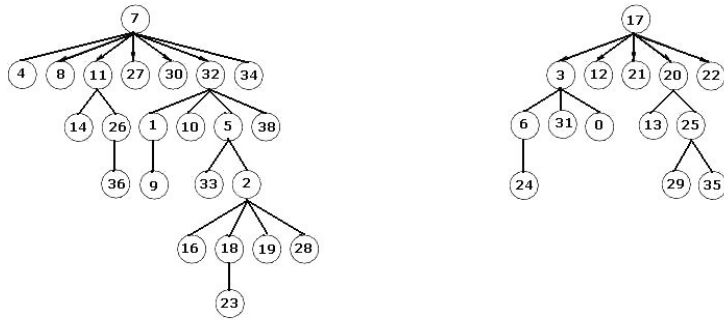


The coordinators and their trees at the end of the first election process



The roots and their trees at the end of the second election process

Figure 2: Tree partitions resulting from two successive and independent election processes.



The trees at the end of phase one

Figure 3: After coordinators prune the trees they have received and add them to their scatternet.

Whereas for the algorithm 2, we observe that the average pair-wise election contests required for obtaining the final scatternet for the rooms under consideration was 146.1.

We also varied the maximum number of allowable slaves in a piconet from 5 to 8. We found, predictably, that the diameter of the network grew significantly with decreasing numbers of slaves per piconet from 12.7 (8 slaves per piconet) to 24 (5 slaves per piconet) for algorithm 1. Similar results we obtained for algorithm 2. These results are illustrated below:

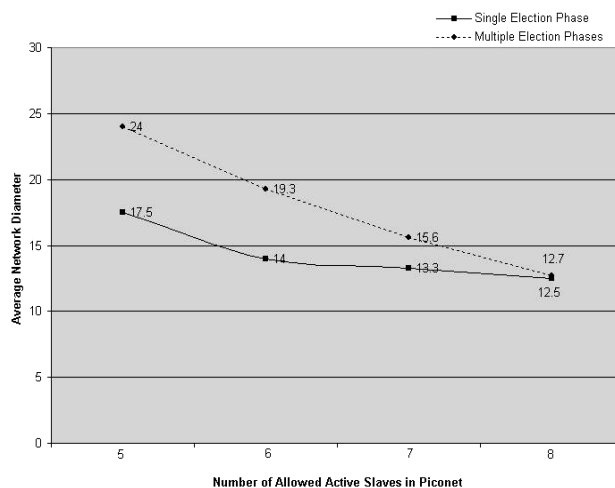


Figure 6: Average network diameter versus Number of slaves in the piconet for algorithms 1 and 2.

We observe that in the arrangement conforming to the Bluetooth standard, where 7 slaves are allowed in a piconet, algorithms 1 and 2 resulted in a spanning trees with mean depth of 15.6 hops and 13.3 hops respectively (+/- 1 with 95% confidence).

We varied the number of nodes in a room and recorded the number of elections required to form the final scatternet using algorithms 1 and 2 for these devices. As expected the number of total pair-wise election contests required to form the final scatternet rises sharply with the increase in the number of Bluetooth devices present in the room for both the algorithms. Preliminary results show that algorithm 1 (multiple election phase) continues to take lesser number of total election contests to form the scatternet than algorithm 2 (single election phase), although this difference decreases as the number of nodes in the room is decreased. Detailed results on this aspect will appear in [13].

Finally, we record the average network diameter of final scatternets and number of pair-wise election contests required for these scatternets as a function of the number of nodes in the room. The active devices are chosen such that there is no partitioning of the network. Clearly, the network diameter depends upon the actual positioning of the devices in the room and may vary substantially for the given number of devices too. We present below a typical example to illustrate the differences between the two algorithms. The following figure depicts the change in the network diameter obtained using the two algorithms, when the number of active devices in a given room is varied between 5 and 40 and number of maximum allowable slaves in a piconet is 7. We can observe for this example that Algorithm 2 (single election phase) consistently results in scatternets of slightly lower diameter than those obtained using algorithm 1 (multiple election phases).

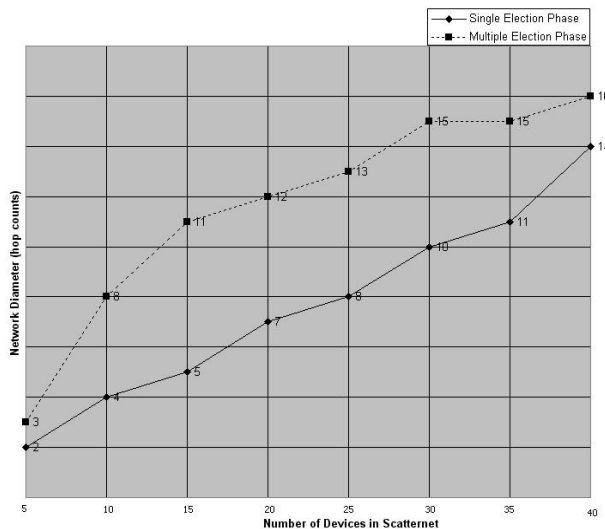


Figure 7: An example of the variance of network diameter with the number of devices in the scatternet.

8 Conclusion

In this paper, we presented a tree-based approach for Bluetooth scatternet formation. We also proposed and compared two algorithms based on such an approach. We believe that for power limited Bluetooth devices, such methods, which are not computationally intensive, are highly desirable. Both the algorithms are distributed and scalable and can be used to forge a scatternet of Bluetooth devices that are not necessarily in pair-wise communication range with each other.

We conclude that both the two algorithms presented here are similar in performance, with one resulting in savings in time (and power) and the other in a more optimal scatternet in terms of the average network diameter and route diversity.

9 References

1. V. Bhatnagar, A. Choudhary and G. Kesidis, "Bluetooth Scatternet Formation Using Proximity Information of an Election Protocol." CSE Department Technical Report, The Pennsylvania State University, PA, August 2001.
2. T. Salonidis, P. Bhagwat, L. Tassiulas, and R. LaMaire. "Distributed topology construction of Bluetooth personal area networks." In Proc. IEEE INFOCOM '01, Anchorage, Alaska, April 2001.
3. R. Savikumar, B. Das and V. Bharghavan, "The Clade Vertebrata: Spines and Routing in Ad Hoc Networks." IEEE Symposium on Computers and Communication (ISCC) '98, Athens, June 1998.
4. www.bluetooth.com
5. B. Miller and C. Bisdikian. "Bluetooth Revealed: The Insider's Guide to an Open Specification for Global Wireless Communication." Prentice-Hall, Englewood Cliffs, NJ, 2000.
6. Rajeev Shorey, Abhishek Das, Ashu Razdan, Abhishek Ghose, Huzur Saran. "Enhancing Performance of Asynchronous Data Traffic over the Bluetooth Wireless Ad-hoc Network." IEEE INFOCOM '01, Anchorage, April 2001.
7. Z. Wang, R. J. Thomas, Z. Haas. "Bluenet – A New Scatternet Formation Scheme." Proceedings of the 35th Hawaii International Conference on System Sciences - 2002.
8. Gergely V Zaruba, Stefano Basagni and Imrich Chlamtac. "Bluetrees – Scatternet Formation to Enable Bluetooth-Based Ad Hoc Networks". IEEE International Conference on Communications (ICC) 2001, Helsinki, June 2001.
9. Ching Law and Kai-yeung Siu. "Bluetooth Scatternet Formation Algorithm." Proceedings of the IEEE Symposium on Ad Hoc Wireless Networks. San Antonio, Texas, USA, November 2001.
10. C. Chevally, R.E. Van Dyck and T.A. Hall. "Self-Organizing Protocols for wireless sensor networks". Conference on Information Sciences and Systems, Princeton University, March 2002.
11. Jeremy Elson and Deborah Estrin. "Random, Ephemeral Transaction Identifiers in Dynamic Sensor Networks". Proceedings of 21st International Conference on Distributed Computing Systems. 2001.
12. L. Subramanian and R.H. Katz. An Architecture for Building Self-Configurable Systems". Proceedings of IEEE MobiHoc, 2000.
13. Vivek Bhatnagar. Masters Thesis: "Bluetooth Scatternet Formation". The Pennsylvania State University. Fall 2002.