

Bluetooth scatternet formation of wireless devices based on maximal independent sets

Yassine Daadaa, Ivan Stojmenovic and Nejib Zaguia
School of Information Technology and Engineering (SITE)
University of Ottawa
800 King Edward Avenue
Ottawa, Ontario, Canada, K1N 6N5
{ydaadaa,ivan,zaguia}@site.uottawa.ca

Abstract

Bluetooth standard allows the creation of piconets, with one node serving as its master and up to seven nodes serving as slaves. Additional slaves must be parked, with significant overhead involved for parking and unparking them. Although the standard allows for the creation of a collection of connected piconets, called scatternet, it does not give any particular protocol for it. Given a set of Bluetooth nodes which are positioned so that their unit disk graph is connected, the Bluetooth scatternet formation (BSF) problem is to select piconets, and master and slave roles in each piconet, so that the obtained scatternet is connected, has some desirable properties and good performance with respect to some metrics. This article describes a BSF protocol based on maximal independent sets. Simulations show its advantage over the best competing protocol.

1. Introduction

Bluetooth and Wi-Fi are two widely adopted technologies for wireless communication between users/devices. They are competing and complementary at the same time, since Bluetooth is more suitable for short communications and provides direct communication between any two nodes, while Wi-Fi requires access point and generally is applied on somewhat longer distances. When two Bluetooth devices come into each other communication range (i.e., they become neighbors), in order to set up a communication link, one of them assumes the role of *master* of the communication and the other becomes its *slave*. This simple "single-hop" network is called a *piconet*, and may include many slaves. Two nodes in a *scatternet* (collection of piconets) can communicate by finding a route between them, where each hop is a master-slave pair of

nodes from the same piconet. A node may serve as the master in at most one piconet, and as a slave in unlimited number of other piconets. In any piconet, no more than 7 slaves can be active (i.e., actively communicating with the master) at the same time. Costly mechanisms for *parking* and *unparking* of slaves are provided to a master for managing a piconet with more than 7 slaves. While a master node is serving as slave in an other piconet, its own piconet (if it has master role in any of them) will be idle.

A closely related problem is *neighbour discovery* (or *device discovery*), that is, how two nodes find each other and establish communication. Almost all proposed solutions assume that Bluetooth technology is used for both neighbour discovery and data communication in the created scatternets. Most BSF protocols use the following device discovery scheme which is described in [7]. The device discovery is performed by each node randomly entering into an *inquiry* or an *inquiry scan mode* (with equal probabilities), and randomly selecting the time length for being in the mode repeatedly until a timeout expires. Inquiry nodes select a repeated pattern of 32 frequencies (out of a total of nearly hundred available frequencies) and send signal on selected frequency in given spot. Inquiry scan nodes also select a frequency at random in each spot and listen to the transmission at the selected frequency. The discovery (and establishment of master-slave relationship) occurs when both sender and receiver nodes are at the same frequency. The timeout for overall device discovery protocol (for finding sufficient number of neighbors) is experimentally determined to have the best value of about 8 seconds. After discovery phase, data communication phase is directed by master nodes, which decide timing and the series of frequencies to be used for frequency hopping based communication between devices.

The main criteria used to evaluate BSF protocols will be on how they achieved the main mission, of providing a connected and a degree limited scatternet topology. Obviously, there is tiny probability that two nodes will never find each other, therefore connectivity cannot be guaranteed even for a network consisting of two nearby nodes. Thus the assumption is natural, and connectivity is judged subject to established neighbours knowledge. We assume that all nodes discovered all their neighbours in the initialization phase. We assume the unit disk graph model of wireless communication, where two nodes can communicate with each other if and only if the distance between them is at most R , where R is transmission radius which is equal for all nodes. We consider *multi-hop scenarios*, where some nodes are not within transmission range of each other, but are connected via other nodes.

The connections between adjacent piconets are done through either a common slave, (called a *gateway slave*), or through a pair of neighboring slaves, (called *intermediate gateways*). In the later case, and since each hop is a master-slave pair of nodes from the same piconet, one of the two intermediate gateways must become the master of a new piconet that includes the other intermediate gateway as slave.

A formation algorithm should not depend on a central component. We only consider *localized protocols*, where each node makes formation decisions solely based on the information from its neighbours (possibly also 2-hop neighbours). Protocols can be further evaluated according to their message complexity, time complexity and memory requirements.

Figure 1 illustrates a scatternet with 11 piconets. Master nodes are in darker colors. Node 2 is slave of nodes 19, 12 and 5, but is master of its own piconet with nodes 1 and 13 as slaves. The bridges participate in piconets on time division basis. It is therefore anticipated that scatternet formation protocols should have (at least) the following goals in mind:

- minimizing the number of piconets, and therefore the number of master nodes;
- minimizing the number of slave roles for each master node;
- minimizing the number of master-slave bridges.

Section 2 presents a literature review. Section 3 describes our new MIS (maximal independent set) based BSF protocol. Section 4 presents experimental data on several evaluation criteria. Conclusions and references complete this article.

2. Related work

A comprehensive survey of BSF protocols is given in [6]. We will present here only one protocol, which was declared as the best in [6] based on a number of criteria.

Petrioli and Basagni [5, 1] described a scheme (called BlueMesh) that guarantees connectivity and limits the number of slaves in each piconet. Their neat scheme does not require position information, but instead the local information is extended to two-hop information, with a two round device discovery phase for obtaining necessary information. That is, in the second round each pair of nodes exchange information on their one hop neighbors, thus getting to know all neighbors which are two-hop away. BlueMesh is a modified clustering process, where selection of slaves is performed in such a way that, if a master has more than seven neighbors, it chooses up to seven slaves among them so that it can reach all the others via the chosen ones. Such coverage is always possible with up to five slaves [2]. The scatternet formation proceeds in iterations. Nodes that participate in a given iteration perform the modified clustering process. Initially all nodes are undecided. In each iteration, init-nodes (nodes having the largest weight among their immediate undecided neighbors) create piconets, by choosing at most seven neighbors as slaves, and deleting the remaining edges. The iteration stops when all nodes are decided. All created masters, together with the slaves that are not selected for links with slaves from other piconets, withdraw from the next iteration.

Simulations by the authors show that the created scatternets have a low average number of roles per node (about two), with an average path length increase between 20% and 80%. The number of iterations grows slowly with the number of nodes (it is about 4.5 for networks with 200 nodes). The method may show weaknesses on some other metrics, especially about the worst-case number of slave roles a node can assume. For instance, in case of dense networks (e.g., complete graph), the second largest node in a neighborhood may end up serving as slave to all the masters in the same neighborhood. Nevertheless, among all methods that do not use position information, the method [5, 1] appears to be currently the best available method for multi-hop networks.

The BlueMesh algorithm [5, 1] is illustrated in Figure 1. In the first iteration, clustering scheme will select nodes 19, 18, 15, 14, 11, 5 as master nodes with slaves indicated by arrows. Masters also select node 13, 12, 10, 7, 2, 1 as intermediate gateway, and these nodes move into the second iteration. Node 13 becomes master of the piconet formed by 13 and 2 to connect piconets 15 and 19 and piconets 15

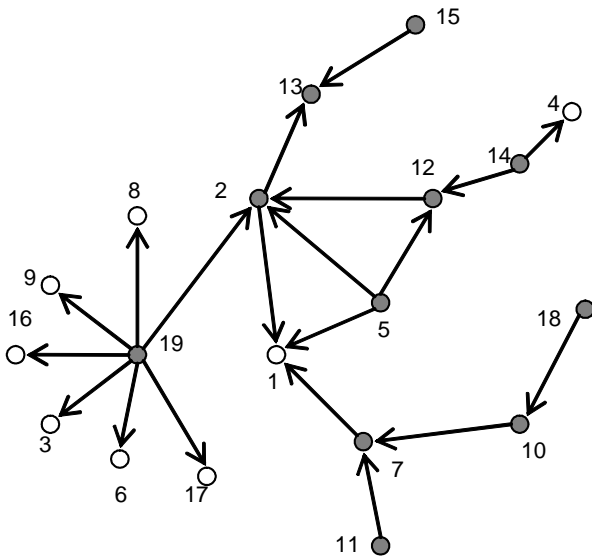


Figure 1. A scatternet consisting of piconets with masters 19, 18, 15, 14, 11, 5, 13, 12, 10, 7 and 2

and 5. Node 12 becomes master of the piconet formed by 12 and 2 to connect piconet 19 and 14. Node 10 becomes master of piconet formed by 10 and 7 to connect piconets 18 and 11 and node 7 become master of the piconet formed by 7 and 1 to connect piconets 5 and 11. By the end of the second iteration nodes 1 and 2 act as intermediate gateways to piconet 7 and 13. Node 2 is then elected as master of this piconet. The algorithm terminates with a connected scatternet.

3. MIS based BSF

We now describe a new BSF protocol. It attempts to simplify the BlueMesh procedure. The protocol essentially interprets the slave selection as the maximal independent set problem, and reduces the process to two iterations, after the initial discovery process. In the discovery process, two-hop neighbors are learned, as in BlueMesh; this is required to run MIS based slave selection.

Our procedure is based on applying certain key(s) for decisions. The procedure will be expressed in terms of a general key. Specific options available for selecting keys are: *node ID*, *-ID*, *(degree, ID)*, *(-degree, ID)*, *(slave degree, ID)*, *(slave degree, degree, ID)*. The *Degree* of a node is the number of its 1-hop neighbors. The *Slave degree* is the number of slaves selected by MIS procedure explained below (the application of this key may require additional

round of information exchange). When key is a record composed of an ordered list of subkeys, the comparison is made using lexicographic ordering, i.e. by the first of them; if equal then the comparison is made by the second keys; if also equal then the third keys is used. Keys can also be based on link to neighbors. For example, the number of common neighbors of the two nodes *A* and *B* can be used as the key. This may give priority to select further neighbors, thus creating shorter paths.

The maximal independent set $MIS(X)$ of a set of nodes *X* is a set of nodes *Y* from *X* such that no two nodes from *Y* are connected (*Y* is an independent set), and *Y* is not a proper subset of another set with the same property (maximal). After learning two-hop neighbors, each node selects the MIS of its neighbouring nodes as the set of its potential slaves. Each node *A* has $key_1(A)$, known to all its neighbouring nodes. Let *Z* be a set of neighbours of a given node *S*. To find $MIS(S)$, the node *S* chooses a node *A* from *Z* with minimal $key_1(A)$. Node *A* is declared a slave of *S*, and is eliminated from *Z*, together with all *A*'s neighbours. This is repeated until *Z* becomes empty. If the number of selected slaves is less than seven then, as in BlueMesh, additional slaves can be selected up to the limit, if desired. However, we did not consider this option in our examples and our simulation. Nodes do not send messages about their potential piconets (so this does not count as a iteration) unless it is required for applying key in the next iteration of the algorithm.

Our proposed MIS based scheme preserves all created piconets if it is possible and meaningful. Two neighbors *A* and *B* could select each other as slaves, and thus a resolution to keep only one relationship, based on an arbitrary key_2 , is needed. Let $key_2(A) < key_2(B)$, then *A* deletes *B* from the list of its slaves, while *B* preserves *A* as its slave. Thus node with higher key_2 remains master in a symmetric relationship. If, after such application, certain node remains without slaves, it does not need to keep its piconet.

The MIS based BSF algorithm is illustrated in Figure 2. The node ID is used as key_1 and also as key_2 . In the first iteration, each node creates its own piconet by selecting its Maximal independent set, after this iteration the graph is a connected scatternet where each node has its own piconet. In the second iteration each node decide, based on a clustering based elimination/ confirmation scheme, whether its piconet is needed for the scatternet connectivity. Accordingly, nodes 19, 14, 13, 12, 10, 9, 7, 3, 10, 2 and then 1 decide to keep their piconets. However the other nodes decide to delete their piconet. For example, node 19 keeps his piconet because of piconet 14. Node 17

decides to delete its piconet because of piconet 19. Node 16 decides to delete its piconet because of piconet 19. Node 15 decides to delete its piconet because of piconet 13 etc. by the end of this iteration the obtained scatternet is connected.

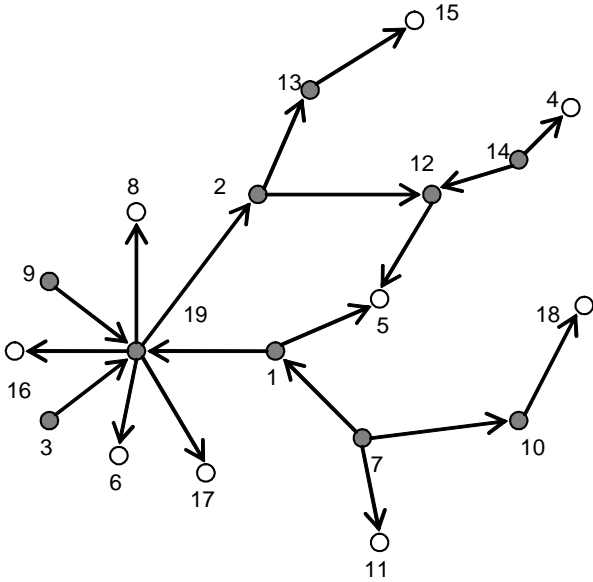


Figure 2. Illustration of the MIS based BSF

We will now prove that this variant preserves scatternet connectivity.

Theorem 1 *For any choices of key_1 and key_2 the scatternet remains connected, if the initial graph is connected and arbitrarily defined.*

Proof:

We will prove that the constructed scatternet contains a minimal spanning tree (MST) as its subgraph. Let the 'length' of each edge be 1. To distinguish edges, we make them different and uniquely sortable by defining $key(B, A) = key(A, B) = (key_1(A), key_1(B))$ where $key_1(A) < key_1(B)$. Note that neighbors B of node A are considered as slaves in ascending order of $key(A, B)$. We then follow Kruskal's MST construction algorithm, where edges are considered for inclusion in MST in ascending order, and included if and only if their addition does not create a cycle. Let (K_2, K_3) , $key_1(K_2) < key_1(K_3)$ be a pair of nodes that did not select each other as slaves. We show then that this edge is not in MST. Since K_3 did not select K_2 , there exist common neighbor K_1 , selected by K_3 as slave, with $key_1(K_1) < key_1(K_2)$ (node can be eliminated only by neighbor with lower key). Then $key(K_1, K_3) < key(K_2, K_3)$, and $key(K_2, K_1) < key(K_2, K_3)$. Therefore edges (K_1, K_2) and (K_1, K_3) have lower keys than

edge (K_2, K_3) and were already considered previously for MST inclusion. Thus pairs (K_1, K_2) and (K_1, K_3) are already connected within MST (either by being in MST or by presence of cycles connecting them). But then the pair (K_2, K_3) is already connected in MST, and is not included in MST. It follows that all MST edges are selected for master-slave relations, and the scatternet remains then connected. Regardless of key_2 , the elimination of master slave pair AB when pair BA already exists, and elimination of piconets left without any slave, obviously does not create partitions. ■

While the number of slaves of each master is limited, and the scatternet is connected, the number of slaves for each node is not limited, and in some cases, e.g. complete graph, one node can be selected as slave to all other nodes. This is the same problem shared with the BlueMesh. The main advantage of the new algorithm is to reduce the number of iterations to two and therefore obtain a faster BSF.

4. Experiments

In this section, we present our experimental results that compare designed algorithms in terms of various characteristics. The comparison is made only with BlueMesh with the choice of the ID as the key, and we did not include other existing schemes. This choice is based on the fact that among all methods that do not use position information, BlueMesh has big advantages (No more than 7 slaves per piconet, an average of 2.3 roles per node, route lengths not significantly longer than shortest path in the network) and appears to be currently the best available method for multi-hop networks [6]. In the experimental results presented here, we choose a total of $n = 30$ wireless nodes which are distributed randomly in a square area. Each node is specified by random X and Y coordinate values. We experimented with two values for the average number of neighbors d . Each node on average has d neighbors, $d = 10$. This can be controlled by sorting all $n(n - 1)/2$ edges, and selecting $nd/2$ -th as the transmission radius. Nodes are numbers from 1 to 30 as they are generated, which serves as node ID in BSF decisions. For the MIS based BSF, the *node ID* is used as key_1 and (*slave degree, ID*) as key_2 .

All nodes can be divided into several categories, according to the type and number of roles taken in the process. Thus a given node can be:

- (1) master only, denoted by M ;
- (2) slave only, denoted by S , possibly to a few piconets, this can be further divided as S_p , where p is the number of piconets where this slave node serves;
- (3) master of one piconet and slave in other piconets, denoted by MS or in general MS_p , where p is the number

of piconets in which this node serves as slave.

There are various metrics that can be used for evaluating scatternets. Persson, Manivannan and Singhal [4] listed the following criteria for constructing scatternets: complete scatternet connectivity, maximized aggregate bandwidth, minimized average routing path length, maximized average node availability, minimized bridge switching overhead, communication group clustering, self-healing, multi-hop node participation, and on-demand scatternet formation. They concentrate on traffic independent measures when evaluating the performance of scatternets. Hodge and Whitaker [3] listed the following such measurements: number of piconets, average number of slaves per piconet, average number of roles per device, average number of bridges per piconet, average number of bridges between piconets, number of master-slave bridges, average shortest-path length, bottleneck and average path latency. In our experiments, we focused on the following parameters: number of iterations, number of piconets, average number of slaves per piconet, average number of roles per device, average shortest-path length. Table 1 displays the results of such experiments. Although the number of piconets with MIS is higher than Bluemesh we noticed a substantial gain on the average shortest path length and the number of slaves per piconet.

	BlueMesh	MIS
Average Number of Slaves per Piconet	3	1.7
Average Shortest Path Length	3.8	3.4
Average Number of Iterations	2.6	2
Average Number of Roles per Device	1.6	1.8
Number of Piconets	12	19

Table 1. Comparison of the BSF based on Maximal Independent Sets and Bluemesh

5. Conclusions

The algorithm can be improved by reducing number of piconets, without sacrificing connectivity. Node *A* can decide to cancel its piconet if it observes that the scatternet remain connected locally after such decision is made. Local

knowledge consists of the awareness of two hop neighbors, and existing piconets announced from neighbors. Note that a node may decide to keep its piconet because of local needs, although in reality the connection may exist through the rest of network. Piconet cancellation procedure may be applied before breaking mutual slave decisions of neighboring nodes, so that the proper relation is selected based on the need to perhaps cancel one piconet completely rather than predetermine the outcome and therefore perhaps keep piconet with a sole 'wrong' slave.

To apply piconet cancellation procedure, we need to define an order of making decision among nodes. This can be decided by a key_3 , and can mimic a clustering style approach. Initially all nodes are undecided. The decision is made by a node that has the highest key_3 among all its original undecided one hop neighbors, and is communicated to all neighbors. This in turn allows some other nodes to make their decisions.

Note that there are other options to consider about the order of making decisions, which depend on what nodes to wait for before making decision, whether to first announce original decision and then to confirm/deny it. For instance, [6] suggested first announcing original decisions and then considering only key_3 of selected slaves and masters of neighboring intended piconets for finalizing decisions.

There is a number of Bluetooth scatternet formation protocols already proposed in the literature. It can be observed that very few of them satisfy most of the desirable characteristics. There are relatively few actual implementations and comparisons. Device discovery appears to be the most time consuming operation. Forming scatternets is still a formidable task, because of the device discovery and the extra complexity imposed by the Bluetooth technology on the implementation of the distributed algorithms.

We anticipate that more Bluetooth scatternet formation schemes will be developed in the near future, and that some modifications to Bluetooth specifications could be made to find solutions which satisfy a number of desirable properties and make it suitable for commercial applications in the multi-hop scenarios. An interesting and a major open problem in the area is to design of a BSF algorithms that will guarantee connectivity and degree limitation (for both of the master and the slave roles) and without using the position information.

References

- [1] S. B. C. Petrioli and I. Chlamtac. Degree-constrained multi-hop scatternet formation for bluetooth networks. *Mobile Net-*

- works and Applications*, 9:33–473, 2004.
- [2] S. B. G.V. Zaruba and I. Chlamtac. Bluetrees - scatternet formation to enable bluetooth based ad hoc networks. In *Proc. IEEE ICC, 2001*.
 - [3] L. Hodge and R. Whitaker. What are the characteristics of optimal bluetooth scatternets. In *Mobiquitous, Boston*.
 - [4] D. M. K. E. Persson and M. Singhal. Bluetooth scatternets: criteria, models and classification. *Ad Hoc Networks*, pages 33–473, 2004.
 - [5] C. Petrioli and S. Basagni. Degree-constrained multihop scatternet formation for bluetooth networks. In *Proc. IEEE GLOBECOM 2002, Taipei, Taiwan*.
 - [6] I. Stojmenovic and N. Zaguia. Bluetooth scatternet formation in ad hoc wireless networks. In *Performance Modeling and Analysis of Bluetooth Networks: Network Formation, Polling, Scheduling, and Traffic Control (J. Mistic and V. Mistic)*, pages 147–171. Auerbach Publications (Taylor and Francis Group), 2006.
 - [7] L. T. T. Salonidis, P. Bhagwat and R. LaMaire. Distributed topology construction of bluetooth personal area networks. In *Proc. IEEE INFOCOM, 2001*.