Degree-Constrained Multihop Scatternet Formation for Bluetooth Networks

Chiara Petrioli
Dipartimento di Informatica
Università degli Studi di Roma, "La Sapienza"
E-mail: petrioli@dsi.uniroma1.it

Stefano Basagni
Dept. of Elec. and Computer Engineering
Northeastern University
E-mail: basagni@ece.neu.edu

***Abstract*—In this paper we describe *BlueMesh*, a new protocol for the establishment of *scatternets*, i.e., multihop ad hoc networks of Bluetooth devices. BlueMesh defines rules for device discovery, piconet formation and piconet interconnection so to achieve the following desirable properties. *a*) BlueMesh generates connected scatternets without requiring the Bluetooth devices to be all in each other transmission range. *b*) The BlueMesh scatternet topology is a mesh with multiple paths between any pair of nodes. *c*) BlueMesh piconets are made up of no more than 7 slaves. Simulation results in networks with 200 nodes show that BlueMesh is effective in quickly generating a connected scatternet in which each node, on average, does not assume more than 2.3 roles. Moreover, the length of routes between any two nodes in the network, is comparable to that of the shortest paths between the nodes.**

I. INTRODUCTION

Among the new technologies for wireless communication in the unlicensed ISM band (2.4GHz), the *Bluetooth* (BT) technology [1] is emerging as one of the most promising enabling technologies for ad hoc networks.

When two BT devices (or BT nodes, as we will also refer to them in the paper) 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, no more than 7 of which can be active (i.e., actively communicating with the master) at the same time. Mechanisms of "parking" and "unparking" of slaves are provided to a master for managing a piconet with more than 7 slaves.

All active devices in a piconet share the same channel (i.e., a frequency hopping sequence) which is derived from the unique ID and Bluetooth clock of the master. Communication to and from a slave device is always performed through its master.

A BT device can timeshare among different piconets. In particular, a device can be the master of one piconet and a slave in other piconets, or it can be a slave in multiple piconets. Devices with multiple roles act as *gateways* to adjacent piconets, thus creating a multihop ad hoc network called a *scatternet*.

Although describing methods for device discovery and for the participation of a node to multiple piconets, the BT specification does not indicate any method for scatternet formation. Out of the solutions proposed in the literature so far for scatternet formation, those described in [2], [3], and [4] assume the radio vicinity of all devices ("single hop" topologies), which is not always the case in realistic scenarios. The solution proposed in [2] works only for networks of at most 36 nodes. The topology of the scatternet generated by the protocols presented in [3] and [4] is a tree, which limits efficiency and robustness. The more general case of multihop topology is considered in [5]. The generated scatternet is yet again a tree. Moreover, the proposed protocol relies on a designated device to start the scatternet formation process, which renders the solution dependent on the proper functioning of a single node. To the best of the authors' knowledge, the only solutions for scatternet formation in multihop topologies that produce topologies different from a tree are those presented in [6], [7] and [8]. In [6] the generated scatternet is made up of piconets that may have more than 7 slaves. This may result in performance degradation, as slaves need to be parked and unparked in order for them to communicate with their master. The problem of reducing the total number of slaves to less than 7 is solved in [7] by first applying degree reduction techniques to the network topology graph, and then executing the actual scatternet formation protocol on a topology in which no node has more than 7 neighbors. These techniques require each node to be equipped with additional hardware that provides to the node its current (geographic) location (e.g., a GPS receiver). Beyond being potentially expensive, this solution is not feasible when such extra hardware is not available. No results on the performance of the protocols described in [6] and [7] are provided. The scatteret formation scheme proposed in [8], BlueNet, produces a scatternet whose piconets have a bounded number of slaves, but it is unable to always guarantee the connectivity of the resulting mesh.

In this paper we present and evaluate the performance of a new scatternet formation protocol for multihop Bluetooth networks that overcomes the limitation of the solutions listed above. Beyond being fully distributed, our protocol does not require nodes to be in the transmission range of each other and generates a scatternet whose topology is a connected mesh rather than a tree. No piconet in the generated scatternet has more than 7 slaves, thus avoiding the overhead due to the need to park and unpark slaves, and no extra hardware is required,

0-7803-7632-3/02/$17.00 ©2002 IEEE

thus making the protocol a viable solution for any network of Bluetooth devices. Given that the topology of the generated scatternet is a mesh, we name the protocol *BlueMesh*.

The idea behind BlueMesh is to generate a connected scatternet by selecting some masters among the network nodes, and allowing each master to select at most 7 slaves. The selection of the Bluetooth masters is driven by the suitability of a node to be "best fit" for serving as a master. This is realized by equipping each node with a *weight*, i.e., a real number ≥ 0, that each node can dynamically compute locally, depending on some parameters of interest to the prevailing network application (e.g., battery power, mobility, other node resources, etc.). The selection of the slaves is performed in such a way that if a master has more than 7 neighbors, it chooses 7 slaves among them so that via them it can reach all the others. In order to perform this selection a master needs to know its two-hop neighbors. Once masters and slaves are selected, i.e., piconets are formed throughout the network, gateways are chosen so that there is an inter-piconet route between all masters that are at most three hops away (i.e., all adjacent piconets are interconnected). This condition ensures the connectivity of the BlueMesh scatternet.

In order to realize the scatternet formation as described, BlueMesh proceeds in two phases:

1. The first phase, *topology discovery*, concerns the discovery of each node one and two-hop neighboring devices. This phase is executed at the device start of operation.

2. The phase of *scatternet formation* takes care of piconet formation and their interconnection to form a scatternet. This phase starts as soon as the previous phase is over and proceeds in successive iterations through which connectivity is achieved progressively.

Through the use of simulations we have demonstrated that BlueMesh is effective in quickly producing a scatternet which is a connected mesh. We observe that independently of the number $n \leq 200$ of network nodes the average number of iterations needed to complete the second phase of the protocol never exceeds 4.2. Moreover, BlueMesh scatternets have the desirable properties that nodes have no more than 2.3 roles (on average), and that its routes are not significantly longer than the shortest routes between any two nodes in the *geographic network topology*, i.e., in the topology of the network we would obtain considering a link between any two nodes that are neighbors.

The rest of the paper is organized as follows. Section II and III describe the two phases of BlueMesh. Simulation results are shown in section IV. Conclusions are finally drawn in Section V.

II. TOPOLOGY DISCOVERY

The problem of one-hop neighbors discovery in Bluetooth has been dealt with extensively in [2] (for "single hop" networks, i.e., networks in which all devices are in each other transmission range) and [6], [9] (for multihop networks). The BT inquiry and paging procedures are used to set up two-node temporary piconets through which two neighboring devices exchange identity, weight and synchronization information needed in the following phases of the scatternet formation protocol. This information exchange allows a symmetric knowledge of one node's neighbors, in the sense that if a node u discovers a neighbor v, node v discovers u as well.

After having formed a local list of all its neighbors, a node exchanges this list with its neighbors, thus achieving two-hop neighbors knowledge. The detailed description and the correctness of this phase of the protocol (i.e., the termination of the neighbor list exchange so that each node knows all its two-hop neighbors) can be found in [10].

III. SCATTERNET FORMATION

In this section we describe the scatternet formation phase of the BlueMesh protocol. Aim of this phase is to divide the BT devices into piconets and to interconnect them through gateways to form a connected scatternet. Scatternet connectivity is guaranteed by establishing an inter-piconet route between any two masters that are at most three hops away [11, Theorem 1].

This phase of BlueMesh proceeds in successive iterations. Each iteration is executed by the network nodes that have not yet exited the execution of the protocol at some previous iteration. Let us call $G_i = (V_i, E_i)$ the network topology graph at iteration i, $i \geq 1$. G_1 is simply the geographic network topology. Each of the G_i, $i > 1$,is the subgraph of G_{i-1} that spans the nodes of V_{i-1} that did not exit the execution of BlueMesh in one of the previous iterations. In each iteration i, piconets are formed from the nodes in the topology graph G_i. Then, masters of neighboring piconets (i.e., at most three hops away) are interconnected. Piconet interconnection is achieved either via a common slave (called a *gateway slave*: This is the case of masters that are one or two hops away and have selected a common slave in the current iteration) or via two *intermediate gateways* (case of masters of neighboring piconets that are three hops away or that have not selected common slaves in this iteration). Gateway slaves are selected in the current iteration so that the corresponding piconets are joined. Masters then proceeds to select the itermediate slaves, which are the nodes that proceed onto the next iteration. All masters, slaves that have not been selected as gateways, and the gateway slaves exit the execution of BlueMesh at this time. BlueMesh terminates when all nodes have exited the execution of the protocol.

We now describe in detail the operations performed by a node v which is executing BlueMesh at the generic iteration i, $i \geq 1$. (In describing this phase, we assume that BT devices are scattered in the plane and that they know their ID, their weight and the ID and the weights of all their two-hop neighbors. See Section II).

Each iteration of the protocol is performed locally at each node v and it is made up of two parts: *Role selection* (for piconet formation) and *Gateway selection*.

First part: Role selection.

Role selection is executed by every node at the very beginning of each iteration i (in the case of the first iteration role selection is performed as soon as the topology discovery phase is over). Based on its weight and the weight of its one-hop neighbors, a node determines whether it is an *init node* in G_i, i.e., a node whose weight is bigger than all its neighbor's weight. Only init nodes go to page mode. All the other nodes go to page scan mode.[1] An init node v executes the procedure MASTER(v) whereas every non-init node u executes the procedure NONINIT(u). In all procedures, $N(v)$ denotes the set of all v's one-hop neighbors, $S(v)$ is the set of at most 7 slaves selected by each master v and $C(v)$ denotes the set of v's bigger nodes that are slaves (denoted in the following as bigger slaves) and smaller neighbors.

```
MASTER($v$)
  $myRole \leftarrow master$
  PAGEMODE
  COMPUTES($v$)
  for each $u$ in $S(v)$
     do PAGE($u, v, master, true,$ NIL)
  for each $u$ in $C(v) \setminus S(v)$
     do PAGE($u, v, master, false,$ NIL)
  EXIT
```

Each master v choses as slaves those neighbors that "cover" all the other neighbors in the sense that if a neighbor u is not selected as v's slave, then at least one of u's neighbors has been selected by v. Such a coverage is always possible by selecting at most 5 slaves [5].

Slave selection is performed at each master as follows.

```
COMPUTES($v$)
  $S(v) \leftarrow \emptyset$
  $U \leftarrow C(v)$
  while $U \neq \emptyset$
     do $x \leftarrow$ bigger in $U(v)$
        $S(v) \leftarrow S(v) \cup \{x\}$
        $U \leftarrow U \setminus N(x)$
  $S(v) \leftarrow S(v) \cup$ GET($7 - |S(v)|, C(v) \setminus S(v)\}$)
```

The function GET(m, W) returns a set of m nodes from the set W (for instance, the m smaller ones, or randomly chosen, or the bigger ones, etc.). It is clear from the operation of the above procedure that $|S(v)| \leq 7$. (We notice that $S(v)$ is nothing but a *dominating set* in the subgraph that spans v's two-hop neighbors.) [2]

Once an init node executing *Master(v)* has selected the neighbors in $S(v)$, it starts paging all its neighbors u in $C(v)$ to tell them whether it has selected them or not. The parameters of the page are the destination node u, the initiator of the page, v, the initiator role r (either master or slave), a Boolean j which is *true* if v selected u for its piconet and, finally, the list of masters M that have selected v (if v is a master M is the empty list). Once all the paging is done, node v exits the execution of the role selection part of this iteration.

Every non-init node v executes the following procedure NONINIT(v).

```
NONINIT($v$)
  PAGESCANMODE
  for each bigger neighbor $w$
     do WAIT PAGE($v, w, r, j,$ NIL)
        if $r = master$ and $j = true$
           then $myRole \leftarrow slave$
                JOIN($w$)
                $M \leftarrow M \cup \{w\}$
  if $myRole = slave$
     then PAGEMODE
          for each $w$ in $C(v)$
             do PAGE($w, v, slave, false,$ NIL)
          PAGESCANMODE
          for each $w$ in $C(v)$
             do WAIT PAGE($v, w, r, j,$ NIL) from smaller $w$
                WAIT PAGE($v, w, slave, false, M$) from bigger $w$
          PAGEMODE
          for each $w$ in $C(v)$
             do PAGE($w, v, slave, false, M$)
          PAGESCANMODE
          for each smaller slave $w$
             do WAIT PAGE($v, w, slave, false, M$)
          EXIT
     else MASTER($v$)
```

Node v starts by waiting for all "bigger neighbors" w to page it and communicate their role. (A bigger neighbor is a neighbor with a bigger weight.) If node w is a master and it has selected node v, then v joins w'spicon et. Once all bigger neighbors have paged, node v knows whether it has already joined the piconet of some bigger masters or not. In the first case, node v is the slave of the bigger masters that selected it. In the latter case node v is going to be a master itself. In any case, node v goes to page mode and starts paging smaller neighbors that may need the information about v's role to decide their own. It will also page all the bigger slaves, since they need to know about v's role for interconnecting piconets later in the iteration. (Bigger masters already exited the execution of this part of the iteration.) The rest of the procedure depends on whether node v has decided to be a master or to be a slave. In the former case, v acts exactly as if it was an init node. If node v is a slave, after having informed its smaller neighbors and bigger slave neighbors (i.e., nodes in $C(v)$)of its decision, it goes back to page scan mode. It then waits for smaller neighbors to communicate the decision they make on their role, and for bigger slaves to inform it about their complete list of masters. Once this exchange has been completed, node v knows the role of all its neighbors, and which (if any) of its masters have invited it to join their piconet. In order for the role selection to be completed, v's list of masters has to be distributed to all its slave neighbors, and v has to become aware of their list of masters. Therefore, v goes to page mode and communicates its list of masters to all its neighbors in $C(v)$, and finally it switches again to page scan mode waiting for its smaller slaves to communicate their list of masters. When all these operations have been completed, each node has a knowledge of its neighbors ID, role, and in case they are slaves, of all the piconets they belong to. Node v then exits the role selection part of this iteration and moves to the gateway selection part.

---

[1] Given the definition of weight and given the total ordering of the set of all nodes' weights, there is always at least an init node.

[2] Note that the protocol can be easily modified to generate a scatternet whose piconets do not have more than a generic number $k \geq$ of slaves.

Second part: Gateway selection.
When the role selection part of an iteration is over, the nodes start the gateway selection phase of the iteration. In this part all slaves communicate to their master(s) information about the roles of their neighbors, their neighbors' list of masters, and whether some of their neighboring masters selected them as slaves. Based on this information each master decides which slaves to select as gateways and to which piconet in order to obtain a connected scatternet. If a pair of masters have selected common slaves, they choose the bigger one among them as gateway slave. This is the preferred way to interconnect adjacent piconets. Whenever no gateway slave can be selected to interconnect neighboring piconets, intermediate gateways are selected, again based on their weight. Upon completion of these operations, the gateway slaves, together with the masters and the non-gateway slaves, exit the execution of the BlueMesh protocol. The intermediate gateways proceed to the following iteration to form the extra piconets needed to interconnect them, hence providing connectivity between their piconets.

Lack of space prevents us to detail here the proof of correctness of the BlueMesh protocol. Termination and correctness (expected output) of each iteration can be found in [10] along with the proof that the resulting scatternet is connected.

IV. SIMULATION RESULTS

We have simulated BlueMesh to demonstrate its effectiveness in generating a connected scatternet. We used a simulator of BT-based ad hoc networks with 200 nodes, implemented in C++.[3] The Power Class 3 BT nodes (i.e., those nodes with a maximum transmission radius of 10 meters) are randomly and uniformly scattered in a geographic area which is a square of side L. For each number of nodes $n = 10, 20, 30, \ldots, 200$, the side L has been chosen so to obtain connected networks. The resulting geographic network topology graph is a *unit graph* in which two nodes are in each other transmission range if and only if their distance in the plane (Euclidean distance) is ≤ 10. The measures investigated concern the average number of iterations needed to complete the protocol, i.e., to obtain a connected scatternet, and the set of metrics identified in the literature as measures of the "quality" of a scatternet (e.g., [3]). These metrics are a) the average number of piconets in the scatternet, b) the average number of roles (master or slave) assigned to each node, c) the average number of slaves per piconet, and d) the average length of the routes in the scatternet, and its comparison to the average length of the routes in the geographic network topology. All experiments have been performed on a number of topologies that allow us to achieve a confidence level of 95% and a precision within 5%.

The average number of iterations required to complete the scatternet formation is depicted in Fig. 1. We observe that BlueMesh scales well: as the number of nodes increases linearly from 10 to 200, the average number of iterations increases logarithmically (from 1.2 to 4.1).

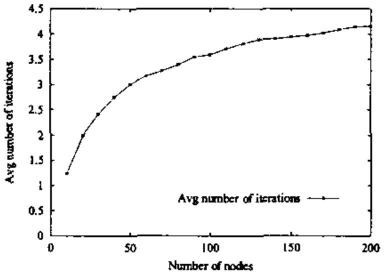

Fig. 1. Average number of iterations.

Fig. 2 shows that the average number of piconets in the scatternet generated by BlueMesh grows linearly with the number of nodes in the network. The portion of network nodes which are masters (i.e., the number of piconets) varies from 29% in networks with $n = 10$ nodes to 41.5% when $n = 200$. A significative part of the generated piconets (up to 45% when $n = 200$) are generated after the first iteration, namely, they are needed for piconet interconnection. This mainly depends on the BT technology that requires that, whenever two piconets are interconnected through intermediate slaves, an extra piconet be created in which one of the two slaves is the master, and the other is the slave.

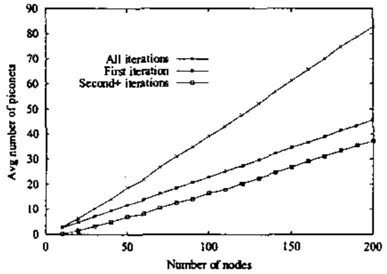

Fig. 2. Average number of piconets in different iterations.

One of the characteristics of the scatternets generated by BlueMesh is that each of its piconet has no more than 7 slaves. As shown in Fig. 3, on average, this limit is seldom approached, given that the average number of slaves per piconet is always between 3 and 4.5, independently of the increasing number of nodes.

A critical performance measure for Bluetooth Scatternets is the (average) number of roles assigned to each node. A high number of roles per node translates into reduced throughput performance since nodes can be active only in one piconet at a time. In Fig. 4 we show the average number of roles assumed

---

[3] Currently, our study is limited to network-layer details, thus the BT stack is not implemented.

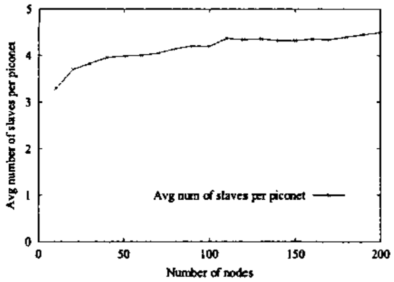

Fig. 3. Average number of slaves per piconet.

by BT devices in a BlueMesh scatternet. The average number of roles per node slightly increases with n to take into account the increased number of nodes in the piconets generated after the first iteration. However, the average number of roles assumed by each node is always very low, and never exceeds 2.3.

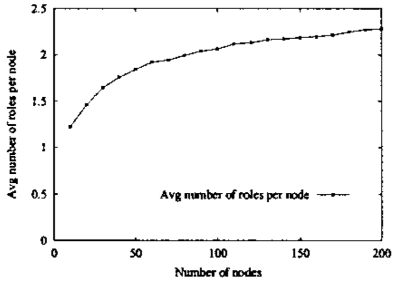

Fig. 4. Average number of roles for each node.

Fig. 5 shows the increase of the average length of the shortest paths between pairs of BT devices in the scatternet, with respect to the average length of the shortest paths between BT nodes in the geographic network topology. Increased route lengths are unavoidable due to the need for all communications in a BT network to pass through the master (two neighboring slaves that belong to the same piconet cannot communicate directly), and to the choice of BlueMesh to always interconnect masters that are neighbors through one gateway slave or through a pair of neighboring slaves. However, the increase in the average route length is very limited, ranging from a 28% increase in networks with $n = 200$ nodes to a 60% increase (from 1.47 to 2.35) when $n = 10$.

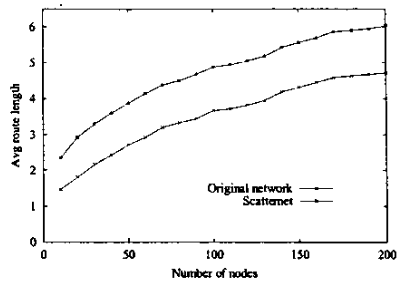

Fig. 5. Routes in the original topology and in the scatternet.

V. CONCLUSIONS

In this paper we have presented BlueMesh, a new protocol for the establishment of a Bluetooth-based multihop ad hoc network. Considering all aspects of the BT technology, BlueMesh describes how to perform topology discovery, piconet formation and piconet interconnection so that, starting from a network of devices geographically connected, a connected scatternet is always generated and each of its piconet does not have more than 7 slaves. Through the use of extensive simulations we have demonstrated that BlueMesh is effective in quickly producing a scatternet, which is a connected mesh. In particular we have observed that the BlueMesh scatternet has the desirable properties that nodes have no more than 2.3 roles (on average), and that its routes are not significantly longer than the shortest routes between any two nodes in the geographic network topology.

REFERENCES

[1] http://www.bluetooth.com, *Specification of the Bluetooth System, Volume 1, Core*, Version 1.1, February 22 2001.

[2] T. Salonidis, P. Bhagwat, L. Tassiulas, and R. LaMaire, "Distributed topology construction of Bluetooth personal area networks," in *Proceedings of the IEEE Infocom 2001*, Anchorage, AK, April 22–26 2001, pp. 1577–1586.

[3] C. Law, A. K. Mehta, and K.-Y. Siu, "Performance of a new Bluetooth scatternet formation protocol," in *Proceedings of the 2001 ACM International Symposium on Mobile Ad Hoc Networking & Computing, MobiHoc 2001*, Long Beach, CA, 4–5 October 2001, pp. 183–192.

[4] G. Tan, "Interconnecting Bluetooth-like personal area networks," in *1st Annual Student Oxygen Workshop*, Gloucester, MA, July 16 2001.

[5] G. Záruba, S. Basagni, and I. Chlamtac, "Bluetrees—Scatternet formation to enable Bluetooth-based personal area networks," in *Proceedings of the IEEE International Conference on Communications, ICC 2001*, Helsinki, Finland, June 11–14 2001.

[6] S. Basagni and C. Petrioli, "A scatternet formation protocol for ad hoc networks of Bluetooth devices," in *Proceedings of the IEEE Semiannual Vehicular Technology Conference, VTC Spring 2002*, Birmingham, AL, May 6–9 2002.

[7] I. Stojmenovic, "Dominating set based Bluetooth scatternet formation with localized maintenance," in *Proceedings of the Workshop on Advances in Parallel and Distributed Computational Models*, Fort Lauderdale, FL, April 2002.

[8] Z. Wang, R. J. Thomas, and Z. Haas, "Bluenet—A new scatternet formation scheme," in *Proceedings of the 35th Hawaii International Conference on System Science (HICSS-35)*, Big Island, Hawaii, January 7–10 2002.

[9] S. Basagni, R. Bruno, and C. Petrioli, "Performance evaluation of a new scatternet formation protocol for multi-hop Bluetooth networks," in *Proceedings of WPMC 2002*, Honolulu, Hawaii, October 27–30 2002.

[10] C. Petrioli, S. Basagni, and I. Chlamtac, "Bluemesh: Degree-constrained multihop scatternet formation for Bluetooth networks," *ACM/Kluwer MONET*, 2002, in press.

[11] I. Chlamtac and A. Faragó, "A new approach to the design and analysis of peer-to-peer mobile networks," *Wireless Networks*, vol. 5, no. 3, pp. 149–156, May 1999.

[12] C. Petrioli, S. Basagni, and I. Chlamtac, "Configuring bluestars: Multihop scatternet formation for Bluetooth networks," *IEEE Transactions on Computers, special issue on Wireless Internet (Y.-B. Lin and Y.-C. Tseng, eds.)*, 2002, in press.