SF-DeviL: Distributed Bluetooth Scatternet Formation Algorithm based on Device and Link Characteristics

Canan Pamuk and Ezhan Karaşan
Department of Electrical and Electronics Engineering
Bilkent University
Ankara, Turkey
{canan, ezhan}@ee.bilkent.edu.tr

Abstract

In recent years, Bluetooth has become very popular owing to the fact that it is a promising ad-hoc networking technology for short ranges. Although construction and operation of piconets is well defined in Bluetooth specifications, there is no unique standard for scatternet formation and operation.

*In this paper, we propose a distributed Bluetooth **S**catternet **F**ormation algorithm based on **Devi**ce and **L**ink characteristics (SF-DeviL). SF-Devil handles energy efficiency using class of devices and the received signal strength. SF-DeviL forms scatternets that are robust to position changes and battery depletions.*

1. Introduction

Bluetooth is a single chip radio solution that operates at 2.4 GHz ISM band by fast frequency hopping spread spectrum. It has a short-range radio link (10m-100m) capable of transmitting voice as well as data. The main advantages of Bluetooth are its robustness, dynamic configurability, low complexity, low energy consumption, low cost and universality.

Connection establishment procedure is defined in Bluetooth baseband specification [1] as a two step procedure: inquiry and page. During inquiry, senders discover and collect neighborhood information provided by receivers. During paging, senders connect to previously discovered receivers.

The smallest operation unit of Bluetooth is a *piconet*. A piconet consists of a master and up to seven slaves. Communication between nodes is provided by master polling. Members of a piconet use a particular frequency hop sequence that is calculated by using the Bluetooth device address (BD_ADDR) and clock of the master. This knowledge is exchanged by frequency hop synchronisation (FHS) packets, sent during the connection establishment procedure of Bluetooth.

Piconets can co-exist in time and space because each uses a different frequency hopping sequence. When there are more than eight active devices, several piconets are needed. *Scatternet*s are formed by combining these several piconets through nodes called *bridge*s. Bridges may be slave in one piconet and slave in the other piconet (S/S), master in one and slave in the other (M/S) or master in one piconet and slaves of two other piconets (M/S/S).

2. Scatternet Formation Problem

The problem of scatternet formation is the assignment of slave, master and bridge roles to devices. This assignment, affects the further operations of the scatternet such as routing and scheduling of bridge nodes, thereby affecting througput and power consumption.

Primary problems with scatternet formation can be summarised as following:

- Initially, devices have no knowledge about their surroundings. Thus, a centralized scatternet formation needs extensive messaging and is practically inefficient. A distributed approach should be used.
- Devices are mobile, so topology changes may take place frequently. The scatternet formation algorithm should be dynamic. It should take care of addition and deletion (failure) of nodes.
- Since Bluetooth modules are used mostly for mobile devices, power of these devices are supplied by batteries. So energy must be used efficiently. Role assignment should be done such that power is used as efficient as possible to increase the lifetime of the scatternet.
- Scatternet should be formed within a reasonable time.

3. Related Work

Salonidis at al. [2] present a distributed Bluetooth Topology Construction Protocol (BTCP), where an elected coordinator determines how a scatternet should be formed. If the coordinator fails, the formation protocol has to be restarted. BTCP's timeout value would affect the probability that a scatternet is formed. BTCP is not suitable for dynamic environments where devices can join

Proceedings of the Eighth IEEE International Symposium on Computers and Communication (ISCC'03)
1530-1346/03 $17.00 © 2003 IEEE

and leave after the scatternet is formed. Law et al. introduce a similar formation algorithm as BTCP that uses one phase and overcomes the timeout problem [3] and the scatternet is formed with certainty. This algorithm, LMS, has the objective of obtaining the minimum number of piconets to reduce interference. SF-DeviL algorithm aims reducing interference by connecting 'closer nodes', i.e., those nodes that have lower path loss between them, which also reduces the consumed power, when power control is used.BTCP and LMS assume all nodes to be within communication range with each other which may not hold due to limited range of Bluetooth.

[4], [5] and [6] are scatternet formation protocols for larger-scale Bluetooth networks, in which the devices can be out of range with one another. But a primary weakness of these algorithms is that, since the formed topologies are trees, the root node and upper layers of the tree are prone to be overloaded and may run out of power. By SF-DeviL, nodes are placed in the tree in an hierarchical order in terms of power and traffic generation rate. Battery capacity, batterylevel and traffic generation rate properties of devices are used to determine 'device grade' of each device which determines the position in the tree.

None of the scatternet formation algorithms considers energy efficiency that is especially important in the operation and lifetime of the formed scatternet. Energy efficient techniques in routing protocols for Bluetooth scatternets have been investigated and it is shown that a considerable gain in network life can be achieved by using distance based power control and battery level based master-slave switch [7]. SF-DeviL algorithm takes care of energy efficiency at scatternet formation procedure by using power control and connecting closer nodes.

4. Motivation for SF-DeviL Algorithm

4.1. Device Grade (DG)

Scatternets can be formed taking care of class of the device that the Bluetooth module is embedded onto. The battery capacity and traffic generation rate can be predicted using the class of the device information. Thus master, bridge and slave roles can be distributed intelligently. If a device having a high battery capacity and is likely to generate high traffic, is chosen to be a master or a bridge, the resulting scatternet will be more stable and power efficient.

For example, as shown in the scenaria in Figure 1, having a mobile as the master of several computers is not an intelligent choice, because a mobile phone has a low battery capacity compared a computer and generates less traffic. The mobile might run out of power which requires the scatternet to be reorganized. So assigning master role to a computer will result in a more robust scatternet.

For the formation of scatternet based on device characteristics, the Bluetooth module must know on which device it is embedded. Indeed the class of the device is known to the Bluetooth module. Also this knowledge is exchanged with neighboring devices during connection establishment procedure by the class of device/service field of the FHS packet. This is a 24-bit-field containing the knowledge of service class, major and minor device classes of the transmitting device [8].

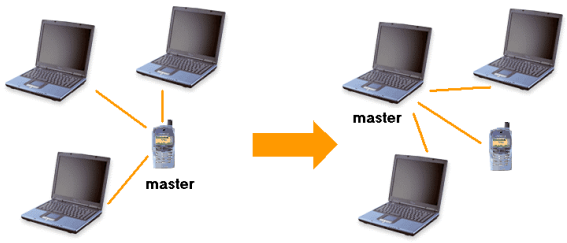


Figure 1. Piconet (scatternet) formation based on device characteristics.

SF-DeviL assigns a quantity called Device Grade (DG) to each class of device. DG is calculated using the class of the device and the battery level of the device as shown in (1). The first term contains the knowlegde of the power of the device and the second term, the traffic generation rate.

$$DG = w_p * DeviceBatteryGrade * BatteryLevel + w_t * TrafficGenerationGrade \quad (1)$$

$$(0 \le w_p , w_t \le 1)$$

In this expression, w_p and w_t represent the weights for power and traffic, respectively. DeviceBatteryGrade (DBG) indicates the power capacity of the battery of the device. BatteryLevel (BL) indicates the fraction of the battery that is available. TrafficGenerationGrade (TGG) is related to traffic generation rate of the device. The traffic can not be known prior to operation but class of the device determines the average possible traffic rate. For example a network access point will have a higher TGG than a computer.

DBG and TGG are predetermined according to the class of the device. Every Bluetooth device contains a table containing the DBG and TGG for different class of devices. Also BL is obtained from the device battery indicator. So each device calculates its DG upon initialization. Devices with larger and/or fuller batteries and higher traffic generation rate have larger DG.

SF-DeviL chooses a device with a large DG compared to other devices as a master or a bridge. SF-DeviL forms a scatternet having a spanning tree topology where the device with the largest DG is selected as the root and the devices with small DGs become slaves, i.e., leaf nodes of the tree. So the tree has a hierarchy, where nodes with larger DG fit in upper layers of the tree.

4.2. Received Signal Grade (RSSG)

Bluetooth modules have power control abilities [1]. After formation of the scatternet, each device reduces its transmit power to extend battery life and minimize interference. Using this property, if devices receiving strong signals from each other are connected, less power will be consumed in the scatternet for transmitting signals after power control is applied, thereby increasing the lifetime of the scatternet and reducing interference.

Consider the scenario in Figure 5, as an example. In the first topology all devices connect to the device with the largest DG, the desktop. This topology has some drawbacks. When one of the mobiles want to send something to the other mobile, the packets have to traverse all the way up to the desktop and then back to the other mobile. This requires the transmit power of the mobiles to be high so as to cover the desktop in their range. In the topology on the right, devices in proximity connect to each other. When this is done, the mobiles reduce their transmit power by power control. So possible interference and power consumption are reduced. In the second topology, there are two piconets in the scatternet where the laptop is an M/S bridge interconnecting two piconets.

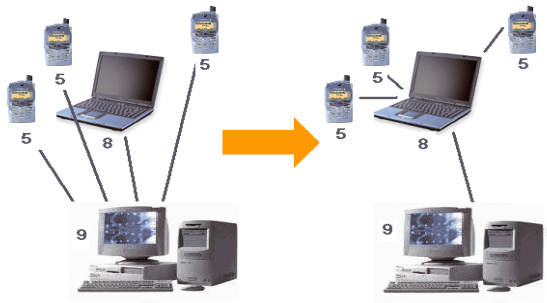


Figure 2. Scatternet formation based on link characteristics.

The Bluetooth module has the Received Signal Strength Indication (RSSI) that measures the received signal strength [1]. Each device assigns a Received Signal Strength Grade (RSSG) to each neighboring device based on the measured RSSI for that link. For example, when device i gets a packet from device j, it measures the received signal strength and has the knowledge of RSSG(j). SF-DeviL uses RSSG to construct the scatternet in such a way that closer nodes have direct connections. RSSG is quantized to very strong(VS), strong(S), medium(M) and weak(W) according to how strong the received signal is. The RSSG increases as the devices are closer to each other.

5. SF-DeviL Algorithm

All devices run a MAIN procedure upon initialization. Afterwards, route tables are constructed and as a last step unconnected tree roots page each other to form the scatternet. SF-DeviL is a distributed algorithm where each device X starts the MAIN procedure below:

```
MAIN:
1    Upon initialization, calculate DG(X)
2    do {
3    x = a random number
4    if  x is even
5        start inquirying
6    else
7        start inquiry scanning state
8    Alternate between inquiry, inquiry scan states
     until a device Y is discovered.
9    Add Y.DG(Y).RSSG(Y) to neighbor_list(X)
10   If  (Y==BestDevice(master of X,Y) )
11       Connect to Y
12       Do master/slave switch
13   while (discoveryTO not reached)
```

In line 1, X, upon initilization calculates its DG by using its class of device and its battery level indicator using (1). The BatteryCapacityGrade and TrafficGenerationGrade corresponding to the device's class are determined from a table that is written to the memory of the Bluetooth module.

By line 9, each device, forms a list of its discovered neighbors by adding the discovered devices to neighbor_list. The entry Y of neighbor_list(X) is bluetooth device address of Y, followed by DG and RSSG of Y. BestDevice procedure in line 10 is used to evaluate if the new discovered device Y, is a better master for X and described below.

Since the paging device becomes the master automatically, master/slave switching must be done in line 12 after each established connection [1]. This is done for assigning slave role to the leaf nodes, the M/S bridge role to the intermediate nodes, and finally master role to the root.

BestDevice(former_master, neighbor) is the procedure to find out which device is a "better" master for X.

```
BESTDEVICE(former_master, neighbor)
1  if(DG(neighbor) >DG(X))
2     //master of X must have larger DG than itself, so
3     //this neighbor has a change of being best.
4     if(former_master==NULL)
5        //X has no master yet
6        return neighbor
7     elseif (RSSG(neighbor)==VS)
8        //if this neighbor is very close, it is a better
9        //master for X  return neighbor
10    elseif ( [DG(neighbor)+RSSG(neighbor)] >
11       [DG(former_master)+RSSG(former_master)])
```

```
12        return neighbor
13    else
14        //this neighbor is not a better master than the
15        //former_master of X
16        return former_master
17 else
18     return former_master
```

If X does not discover a new neighbor for a specific timeout (DiscoveryTO), it stops procedure MAIN. After MAIN, X has either found a master and connected to it or it has declared itself as root.

Route tables are formed spreading from leaves to root. Each node sends its master a 'route table complete' packet, where all the descendants exist. Upon receiving from all of its slaves, a master sends itself a 'route table complete' packet and so on, until the roots form their route table.

Roots, compare their route table entries with neighbor_list entries. If there are some unconnected nodes, with larger or equal DG, each root starts paging these roots. Disconnected tree roots alternate between page and page scan states and each time connects to the best root entry using the procedure BestDevice with a difference in the first line: if DG(neighbor)>=DG(X). And the root with highest number of slaves becomes the master .

The resulting spanning tree has devices with the largest DG`s as root and bridges and the smallest DG devices are leaves.

6. Simulation Results

We developed a C++ based simulator using Bluetooth specifications [1]. Performance of SF-Devil and LMS are compared. Scatternet formation delay, number of piconets, network diameter (number of links in the longest path), average distance between nodes and the time where the first device battery depletes are used as performance metrics.

Nodes are randomly distributed in an area of 10x10m and are assigned random device classes where all of the devices are battery fed. All devices start procedure main at the same time and all have full batteries initially.

During MAIN no power control takes place since device discovery is not finished yet. After MAIN, power control is done to ensure that received power is held at the value -60dBm. We used the path loss model:

$$PL(d)=PL(d_0)+10.\gamma.\log(d/d_0)\ ,$$

where $PL(d_0=1m)$=-30dBm and γ=2.5. Due to power control, transmitter power is changed from 0dBm to –30dBm by steps of 2dB appropriate to [1].

Batterylevel of each device decreases inversely proportional to its BatteryCapacity. Power consumed per packet can be stated as:

$$P_{per\_packet}= P_{standby} + P_x\ ,$$

where P_x is $P_{transmit}$ for transmitters, $P_{receive}$ for receivers and P_{inter} for nodes that convey the packet from one node to the other. $P_{standby}$ is the power consumed in standby mode. The relation between these values is taken as:

$$P_{standby} = P_{receive} = P_{transmit}\ /316$$
$$P_{inter} = P_{receive} + P_{transmit}$$

Random traffic is generated, proportional to the TrafficGenerationGrade of each device. The devices are assigned full batteries initially. Assuming an average throughput of 300kbits/sec, the time at which the first device runs out of battery is investigated in Figure 3. This time is important since the scatternet formation algorithm needs to be restarted when a device fails.

Since devices with higher DG are assigned root and bridge roles and since closer nodes connect to each other, scatternets formed with SF-DeviL carry more traffic than LMS before the first battery depletion. The time of first depletion is more than LMS especially with large number of nodes. It is obvious that SF-DeviL, uses power more effectively than LMS.

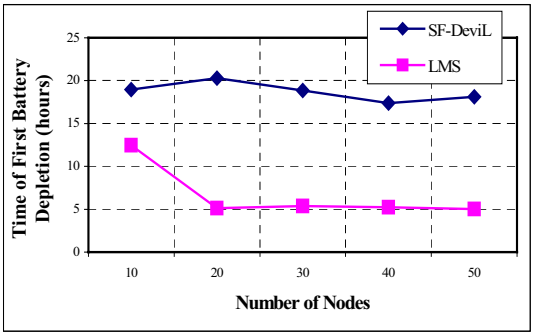


Figure 3. Time at which battery of first device depletes.

SF-DeviL forces each node to connect to the closer node with highest DG. For this reason, since area is kept constant, average distance between connected nodes decreases by increasing number of nodes. In Figure 4, the largest distance between nodes is about 3.4m and corresponds to a path loss of 43dBm. By LMS, distance between nodes is not a scatternet formation criteria. Thus the distance is about 5m on the average, which matches a path loss of about 47.5dBm. Average distance affects the transmitter power since power control is used, thereby affecting battery consumption.

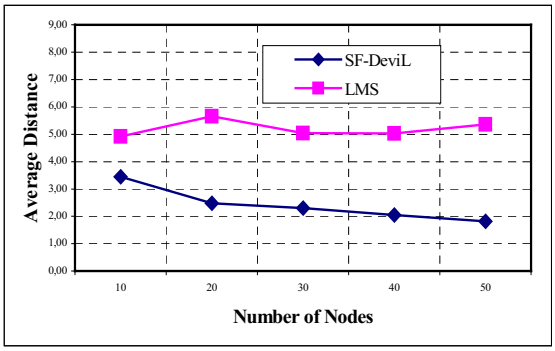


Figure 4. Average distance

Number of piconets is an important performance measure in scatternets. As the number of piconets increases, a decreasing trend in carried traffic is shown in [9]. By LMS, small number of piconets is aimed to reduce interference. By SF-DeviL, decreasing the number of piconets is a secondary aim (after power efficiency). Even so, it is seen that number of piconets in SF-DeviL is close to LMS as shown by Figure 5.

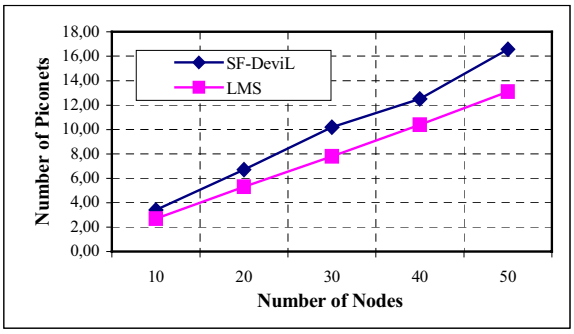
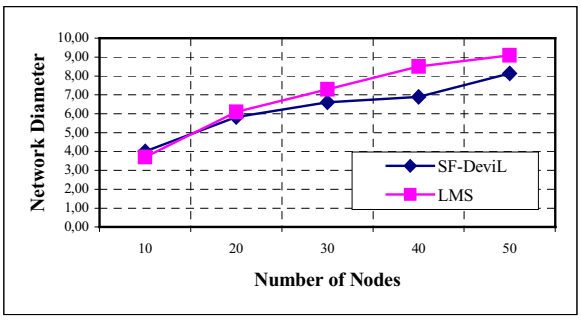


Figure 5. Number of piconets

Small network diameter means, small number of hops to the destination and therefore less power consumption and less traffic at intermediate nodes. SF-DeviL and LMS have nearly the same network diameter, by a small



superiority of SF-DeviL.

Figure 6. Network diameter

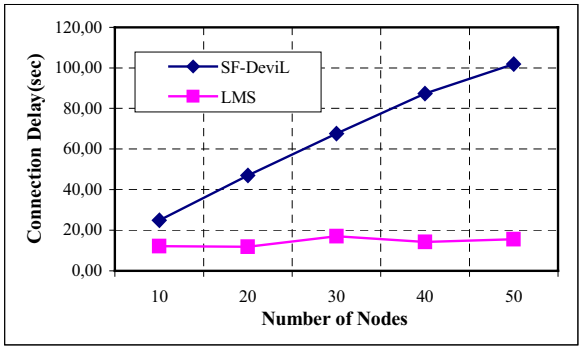


Figure 7. Scatternet formation delay

Finally, each scatternet formation algorithm has to have a tolerable formation delay in order to be applicable. The connection delay of SF-DeviL and LMS, are compared in Figure 7. Scatternet formation delay values on the graph for SF-DeviL are the values without timeout. DiscoveryTO=1min, added to these values give the actual formation delay. This is a large delay compared to LMS and also other existing scatternet formation algorithms. The delay merges from the MAIN procedure where each device tries to discover as many neighbors as it can before a timeout of 1min. Even so, the delay for SF-DeviL is raesonable when number of nodes in the scatternet is not very large and it increases linearly with number of nodes.

7. Conclusions

SF_DeviL algorithm forms Bluetooth scatternets based on device classes and energy efficiency. It can be adapted to current Bluetooth modules easily because it is compliant with the specifications [1]. It operates in a distributed fashion and results in a spanning tree where powerful devices are chosen as root or bridges.

Simulations prove that, using class and link characteristics during scatternet formation results in power efficiency during operation of the scatternet. The reduction of average distances shows the effect of link characteristics. SF-DeviL produces topologies with reasonable number of piconets, network diameters and connection delay.

The time complexity of scatternet formation using SF-DeviL depends on the duration of neighbor_list formation, which is proportional to number of nodes. But a reasonable duration for scatternet formation is found out from simulations. We also plan to incorporate recent work on reducing the connection establishment time in Bluetooth, which will make SF-DeviL algorithm operate faster [10-11].

SF-DeviL is also suitable to adapt to changes such as deletion and addition of nodes. By running procedure MAIN periodically it will be applicable in dynamic environments also. By this way, DG will be updated with decreasing batterylevels and devices that consume more

battery will move to the leaves. The lifetime of the scatternet is expected to increase this way.

We are also considering to handle mobility by adding a MobilityGrade term to equation (1). Because knowing the class of device, the prediction of how mobile it is can be done. This way, devices that are stable –besides power term and traffic generation rate- will be forced to become root or bridges. So in case of position changes, roots and bridges will be chosen out of stable devices.

8. References

[1] Bluetooth SIG, "Specification of the Bluetooth System", Version 1.1, http://www.bluetooth.com

[2] Theodoros Salonidis, Pravin Bhagwat, Leandros Tassiulas, Richard LaMaire. "Proximity Awareness and Ad Hoc Network Establishment in Bluetooth", Institute of Systems Research, University of Maryland Technical Report

[3] Ching Law, Amar K. Mehta, Kai-Yeung Siu, "A New Bluetooth Scatternet Formation Protocol", ACM Mobile Networks and Applications Journal, 2002

[4] Godfrey Tan, Allen Miu, John Guttag, Hari Balakrishnan, "Forming Scatternets from Bluetooth Personal Area Networks" MIT Technical Report, 2001

[5] Gergely V.Zaruba, Stefano Basagni, Imrich Chlamtac. "Bluetrees-Scatternet Formation to Enable Bluetooth Based Ad Hoc Networks" In Proceedings of IEEE International Conference on Communications, pages 273–277, 2001.

[6] Zhifang Wang, Robert J. Thomas, and Zygmunt Haas, "Bluenet – A New Scatternet Formation Scheme" In Proceedings of the 35th Annual Hawaii International Conference on System Sciences, January 2002.

[7] Balakrishna J. Prabhu and A. Chockalingam, "A Routing Protocol and Energy Efficient Techniques in Bluetooth Scatternets", IEEE ICC'2002, New York, 2002

[8] Bluetooth SIG, "Assigned numbers- Bluetooth baseband," http://www.bluetooth.org/assigned-numbers/baseband.htm [accessed at 25/11/02]

[9] Gy. Miklos, A.Racz, Z. Turanyi, A.Valko, P. Johansson, "Performance Aspects of Bluetooth Scatternet Formation", Proceedings of The First Annual Workshop on Mobile Ad-Hoc Networking and Computing,2000.

[10] Ryan Woodings, Derek Joos, Trevor Clifton, Charles D. Knutson, "Rapid Heterogeneous Connection Establishment: Accelerating Bluetooth Inquiry Using IrDA", ACM MobiCom, 2001

[11] Yelena Gelzayd, "An Alternate Connection Establishment Scheme in the Bluetooth System", M.S Thesis submitted to Polytechnic University, 2002.