A New Scatternet Formation Protocol for Bluetooth Networks

Hong-Yi Chang and Chang Wu Yu
Department of Computer Science and Information Engineering
Chung-Hua University
cdr@seed.net.tw cwyu@chu.edu.tw

Abstract

Bluetooth is a new low cost and short-range radio technology. Mobile devices equipped with bluetooth chipset use FH (Frequency Hopping) method in a frequency band of 2.4 GHz to construct a piconet with other chipsets. Different piconets can communicate through relay devices. To date, many researchers have proposed different scatternet formation protocols. However, most of them consider how to fast construct a scatternet with minimal piconets or how to maintain the desired topology.

In the paper, we propose a new scatternet formation protocol for Bluetooth networks. We aim to construct a scatternet with longer live time. Moreover, our protocol can maintain the desired scatternet formation even when the topology of the network changes. Simulation results show that our protocol outperforms BTCP in many respects including construction time and network live time.

Keyword: Bluetooth, Scatternet, Piconet, Formation, Construction.

1. Introduction

Bluetooth is a wireless communication technology that have 10-meter transmission distance, low cost, FHSS high resist interference. Mobile devices equipped with Bluetooth chipset use FH Frequency Hopping method in 2.4 GHz ISM (Industrial Scientific Medical) frequency band to construct a piconet with other chipsets. Bluetooth is likely to become another promising platform for ad hoc networks, which have many applications including emergency search-and-rescue operations, meetings or conventions in which persons wish to quickly share information, data acquisition operations in inhospitable terrain, and automated battlefield [1].

A Bluetooth scatternet is composed of piconets. Each piconet has a master and at most seven slaves. The master controls the whole piconet operation, and devices in the same piconet can communicate to each other through the master. If devices are in different piconets, they can communicate to each other through multiple relay devices. In recent years, many researchers have proposed different scatternet formation protocols including BCTP and bluetree [7]. However, most of them consider how to fast construct a scatternet quickly [1, 2] or how to maintain their desired topology [5, 6, 11] without considering the following facts.

Fact (1): Different roles consume different power.

Master devices and relay devices play important roles in a scatternet. First, master and relay devices usually consume much battery than slave devices do. If master devices run out of power or move away, its slaves might be reassigned to other piconets. When the topology of the scatternet is rebuilt and reorganized frequently because of improper master/slave assignments, the power consumption of the scatternet rises dramatically and the live time of the network will be shortened.

Fact (2): Constructing and maintaining a desired scatternet take too long time still.

Bluetooth standard defines the asymmetric link method [1]. Before two devices are linked, each device should be assigned with either master or slave role. When master device do inquiry and slave device do inquiry scan at the same time, both link together (See Figure 1). Some other research provides symmetric method [4, 9]. Every Bluetooth device takes inquiry and inquiry scan in turns [4]. But two Bluetooth devices should spend more than 659.375 ms to link up [4]. If the set up time is less than 659.375ms, these devices may fail to link together. It will take longer time to construct and then maintain a desired scatternet formation.

Fact (3): Constructing traditional network topologies for Bluetooth networks is questionable.

To date, many scatternet formation schemes with different topologies including tree [5, 7], star [4, 9] and ring [10, 11] have been proposed for Bluetooth. They maybe work well in wired networks. However, one may raise the question: Are they also suitable for Bluetooth networks? Obviously, the problem is hard to answer and needs thorough investigation.

Based on above facts, a new scatternet formation protocol is designed to select master and relay devices properly so that the resulting scatternet can live longer. The scatternet formation can be constructed quickly and maintained by our protocol efficiently when the topology of the network changes frequently. Simulation results show that our protocol outperforms BTCP in many respects including construction time and network live time.

The rest of the paper is organized as follows. In Section 2, some necessary backgrounds are introduced. We survey previous results related to scatternet formation protocols in Section 3. In Section 4, we propose the new scatternet formation protocol in detail. We also combine ns2 and bluehoc to simulate our scatternet formation protocol. And the simulation results are compared with BCTP in Section 5. Finally, Section 6 concludes the paper.

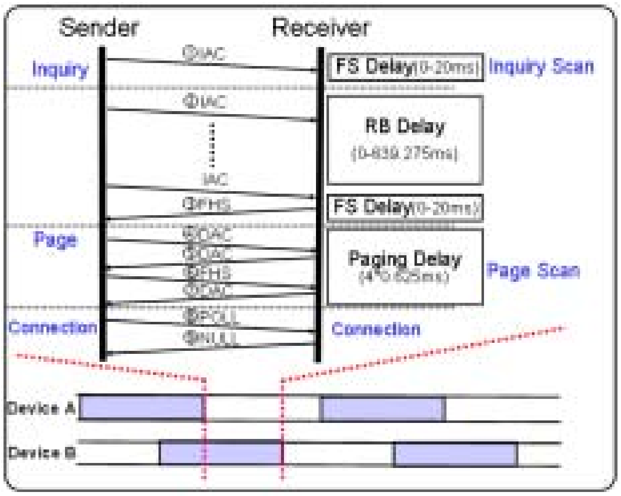

Figure 1: Bluetooth link formation procedure

2. Background

The specifications for Bluetooth network define an asymmetric protocol to establish connection. It must fix the role that was always acted beforehand. The master role run inquiry/ page procedure and the slave role run inquiry scan/page scan procedure. (Figure1)

2.1 Asymmetric protocol to establish connection:

According to the Bluetooth Baseband specification the Bluetooth asymmetric protocol for link formation starts by the sender starting in the INQUIRY state and the receiver in the INQUIRY SCAN state. Given two units, one operating as a sender and the other as a receiver, the term Frequency Synchronization delay [1] (or FS delay) refers to the time until the sender transmits Inquiry Access Code (IAC) packets [1] at the frequency the receiver is currently listening on. When the receiver detects an IAC packet from the sender, we say *HIT* occurring (The sender *HIT* the receiver and the receiver is *HIT* by the sender.) There is a FS delay until hit. After the first hit, the unit in Inquiry Scan (receiver) backs off for an amount of time that is uniformly distributed between 0ms (0 slot) and 639.375ms (1023 slots). The time while the receiver backs off are called the Random Backoff delay (or RB delay). When the receiver unit wakes up, it starts listening again at the hop it was listening to before backing off. After a second FS delay, a second *HIT* happens. Then the receiver sends back to the sender its Frequency Hopping Sequence (FHS) packet [1]. Upon the sender receives the FHS, the paging procedure happens instantaneously and the connection is established. The paging procedure delay is negligible since it immediately follows the inquiry procedure with the time cost of 4 slots (1 slot = 625 μs). Thus the [2] approximates the link formation delay *R* using the following equation:

$$R = 2FS+RB \text{ [1]} \qquad (1)$$

Where *FS* and *RB* are uniform random variables in $[0,T_{coverage}]$ and $[0,r_{max}]$ respectively [2]. From (1), the link formation delay is dominated by $2T_{coverage}+r_{max}=2*16+1023=1055$slots (659.375ms) for the 32-hop system and 1039slots (649.375ms) for the 16-hop system [2]. And the expected link formation delay is approximate 528slots (330ms).

2.2 Symmetric protocol to establish connection:

The way of the establishment of Asymmetric in section 2.1, We discover that the very big problem exists easily, Because in the real network environment, we can't define role beforehand, Therefore, the scholar executes Inquiry and Inquiry by using the Random alternation [4], It produces a pair of Inquiry/Inquiry Scan (Figure 1), Device A and Device B's produce connect. But the probability is not high through Random method to match, because match time must greater than 2FS+RB=659.375ms.

3. Related Work

In this section, we survey previous scatternet formation schemes. In [4], BTCP assigns a variable VOTES to each node. Initially, variable is set to 1. Each node alternates between INQUIRY and INQUIRY SCAN states. When a node is setting in INQUIRY state discoveries, the other one is in complementary state, and then a link is established. The two nodes can compare their VOTES variables. The loser sends all information about nodes it has learned to the winner, and then, enters PAGE SCAN state to wait for further connection. The winner increases its VOTES by the VOTES of loser, and then continues state alternation. At the end, a leader will be elected, and it will control the topology formation. In order to minimize the number of piconets, the topology should satisfy the following properties:

(a) The resulting topology is connected
(b) Two masters share one bridge
(c) A bridge connect two piconets
(d) The maximum number of slaves per piconets is seven
(e) The resulting scatternet should consist of the minimum number of piconets.

So the leader computes the number of master's p (see [4]).

$$p = \left\lceil \frac{17-\sqrt{289-8N}}{2} \right\rceil \quad 1 \le N \le 36$$

The leader assigns p masters and $\frac{p(p-1)}{2}$ bridges. Every master x will receive slave list (x) and bridge list (x) from the leader. Then the masters establish links according to their slave list (x) and bridge list (x). After all links are established, the topology is constructed.

The above algorithm is not fully distributed, and it must elect coordinator first, then through final coordinator centralize divide group and assign role. From experiments we observed, if whole network bigger than 36 nodes, and whole system performance will become very bad, since the overall message complexity is too big.

In [2], main author improved [4] symmetric method problem. In [4], although improved asymmetric link method defect, random method used, but link failed easily, because not enough time or use the same role in the same time (like both are master), so make performance lower. This paper proposes to use inquiry scan procedure first then inquiry procedure to increase link probability.

The randomized algorithm suggest in [3] uses the following assumptions:

Assumption-1: For a network of N nodes and maximum cluster size S, the ideal number of masters is

$$k = \left\lceil \frac{N}{S+1} \right\rceil.$$

Assumption-2: If there are enough nodes in INQUIRY SCAN state, a node in INQUIRY state can collect at least S responses during CLUSTER_TIME_OUT period.

Assumption-3: A node in INQUIRY state can collect at least 2k responses during SUPER_MASTER_TIME_OUT period.

Every node conducts $T = \frac{\log(1-\frac{k}{N})}{\log(1-p)}$ rounds of Bernoulli trials with parameter p to determine its role randomly. The nodes that succeed at least once become Master-designates, and the others become Slave-designates. The actual number of Master-designates X would be between 3k/4 and 2k with probability $1-2e^{-k/4}$.

In[5], this paper propose a distributed algorithm of construct Scatternet, the major contribution provide detail algorithm describe, like how to merge two piconet etc., and prove the time complex degree is *O* (log *n*) and message degree is *O* (*n*) when use algorithm to construct scatternet.

In [6], a topology formation algorithm, called Tree Scatternet Formation (TSF), which forms scatternet in tree structure. In this paper construct scatternet to tree. They think construct a tree to help data routing and scheduling. They also provide program simulation and mathematics to prove. Prove tree structure is good. Besides, from experiment we can find node number bigger than 50, TSF represent is still good, but BTCP is very bad [4]. Besides, they use GIAC packet, the loop problem is solved by the design.

In [7], authors proposed an algorithm that the scatternet is formed with tree, in this paper is more like analyze, not emphasize how to form the algorithm.

In [8], they provide two algorithms, first algorithm construct on Dijkstra algorithm, purpose to reach average short path. Another algorithm is constructing on Ford-Fulkerson algorithm, purpose to make maximum transmission flow.

In [9], authors provides one distributed algorithm to build up scatternet to star structure. The major method is every node in the network will give one weight when they start. And then every node exchanges their weight to each other. If one node weight is more than the neighbor, then it will change role to master, and form piconet with neighbor node. And these neighbor join piconet will change role to slave. After it, two masters between piconets can by middle node with 2 Hop or 3 Hop to communicate; these nodes have chance become bridge. The defect is: author is not use experiment simulation and mathematics proves the algorithm performance and accuracy.

In [10], authors provide two algorithms to build up ring structure. First is Node_ID Algorithm: this method is set every network device have one number, by this number can let every device in the network link to chain; the number is bigger than chain head then link this new device from head, number is smaller than chain head then link this new device from tail. Wait for a while, the head and tail can link to ring structure, another method is Head_SeekScan like first method, the most different is it only can link new device from head. Feeling the two methods is not considering very detail.

In [11], authors propose a ring structure to form the scatternet and develop its routing protocol.

4. A New Scatternet Formation Protocol for Bluetooth Networks

4.1 An Outline of the Protocol

We propose a new scatternet formation protocol. This protocol consists of device discovery algorithm, scatternet construction algorithm and scatternet recovery algorithm. In the device discovery algorithm, each device discovers its neighbor devices alternating inquiry state and inquiry scan state, and exchange its weight values, scatternet construction algorithm contains two phase: piconet formation and bridge selection. In the piconet formation phase, a device built a piconet based on the weight values acquired by the device discovery algorithm, then elected appropriate master, define neighbor device that smaller weight to be slave and run Page scan process, wait master paging to join piconet. In bridge selection phase, elected appropriate bridge connects them into large network, call scatternet. After that, each device can start data communication with other devices. Assuming device mobility, scatternet connectivity id maintained by using the scatternet recovery algorithm. The last scatternet recovery algorithms consider master and bridge power to deplete, move to other place, etc. Reasons in scatternet and it not fit it, how to repair scatternet topology. The details of the protocol are described below.

4.2 Device Discovery Algorithm

First, in whole Bluetooth network environment, the role of every device is not defined. According to their power characteristic, we generate a weight value for each node. The other is through running Inquiry/Inquiry Scan by switching weight value by each other, but for avoid generate collisions when start at the same time, so every device are all into the process of Inquiry Scan and wait *Tinq* total 16 Time Slot = 20ms. If in *Tinq* time not received Inquiry packet from other device and it switch to Inquiry process to send IAC (Inquiry Access Code) packet, and wait 1024 Time Slot = 639.625ms to collect all the answer from the device around by FHS (Frequency Hopping Sequence) packet. So this algorithm total spend *Td* is*Tinq*+639.625ms or 2**Tinq*+639.625ms.

Device Discovery Algorithm:

```
DeviceDiscovery( )
Set Td of the device discovery algorithm;
Enter Inquiry Scan mode;
While (Td>0){
if (mode<> 'Inquiry') {
compute the Tinq of the next procedure;
execute inquiryScan(min(Td , Tinq ));
if receives IAC packet return;
switch to 'Inquiry' mode; }
else{
compute the Tinq of the next procedure;
execute inquiry(min(Td , Tinq));
switch to 'Inquiry Scan' mode; }}
Exchange weight between two device ;
```

4.3 Scatternet Construction Algorithms

Scatternet construction algorithm divides two phases. First phase is Piconet Formation Phase, second is Bridge Selection Phase. Detail step is described in below:

4.3.1 Piconet Formation Phase

To prevent competition by all devices trying to construct piconets simultaneously, each device sets a piconet formation start timer with value *Tp* randomly determined from 0 to *Tp*.

Based on the information gathered in the previous phase, namely, the ID, the weight, and synchronization information of the discovered neighbors, each device performs the protocol locally. The rule followed by each device is: device *u* decides whether it is going to be a master or a slave depending on the decision made by the neighbors with bigger weight. Each device compares the neighbor lists, which collect by it. If the weight value is bigger in the neighbor list, define role to master and run Page process. If it's not the bigger, it runs page scan process. Then, device which defined Master role paging device that in neighbor list and create piconet. And masters use its bluetooth device address as the piconet's piconet id (PID), and send information to slave that in piconet.

Piconet Formation Algorithm：

```
PiconetFormations( )
Set Tp of the Piconet Formation algorithm;
While (Tp>0){
if (for each neighbor u: myWeight > uWeight){
myRole ='master';
go to page mode;
Page all smaller neighbors into piconet;
Set uRole='Slave'}
Else go to page scan mode;
Master send PID to all Slave;}
```

4.3.2 Bridge Selection Phase

After Piconet Formation Phase, enter "Bridge Selection Phase", elect appropriate bridge connects piconet that construct in pervious phase into scatternet. We will discus in three situations below:

1. Two piconets have more than two device defined bridge roles:

In the Piconet Formation Phase end, if Slave have Paged by more than two Master, that mean this Slave can be defined Bridge and receive more than one PID, and this device will reply to the master that higher weight value fit bridge role. Than, bigger weight value of Master base on PID and weight value define suitable bridge, and throught Bridge device with other Master in Piconet to switch PID value. In bluetooth network, one Piconet only have 7 Slave (include Bridge). So because of avoid redundancy to waste piconet member numbers, appoint not suitable bridge of slave. Break it connects with other Piconet is defined Bridge device into Connection status to terminate any process.

In Figure 2, between Master of weight value 10 and Master of weight value 9 have two devices of weight value 3 and 8 as slave role at the same time. If we can pass through two devices to communicate between Piconets, these two devices can define bridge role. But consider Piconet in bluetooth network environment, it limit one Master can serve seven salve, so between two piconet. If have more than one bridge device, it will redundancy the member numbers in Piconet. So we propose the algorithm limit between two piconets. Only one has a bridge to communicate. So like the example in figure 2, we choose the bigger weight value device (like weight value is 8), no role use bridge device (like weight value 3) and it cancel two connections which master's weight is smaller in Piconet, two master devices of Piconet still record it information to backup bridge, in the future if bridge leave or power consumption, it will take from backup list bridge and reach usability and fault tolerance. Beside it will switch piconet number (PID) by each other, so in the future it can construct bridge by the piconet number (PID) list which maintain by each other.

2. When it have not paged device situation:

In piconet formation phase, the device don't have bigger weight in neighbor list will enter Page Scan process, but during time *Tp*, the slave not be paged into piconet. So, the device define role to master and change state, "enter page" state, and page device in the neighbor list, paging device that in neighbor list to join piconet, in bridge selection phase, elected appropriate bridge connects them into scatternet.

For example, the device that weight value 5, may wait time *Tp* and not be paged to piconet as slave, and switch role to master and generate one piconet number(PID) and through bridge selection phase to define device as bridge which Weight value7, construct Scatternet. See figure 2.

Bridge Selection Algorithm：

```
BridgeSelection ()
if (role=slave){
if (Connected() = false){
to establish piconet between the two slave;}
if (connect to two master){
instruct master that has bigger weight
selected bridge;
the master selected has bigger weight slave
set role = 'bridge';
keep the other's slave to backup list;
notice break connect for smaller weight master;
exchange PID between two piconet;}}
else{
if(state='page scan')}
set role='master';
page neighbors into piconet;}}

Connected()
If (Check Between Salve's PID lis the sane){
return true;}
else{
return false;}
```

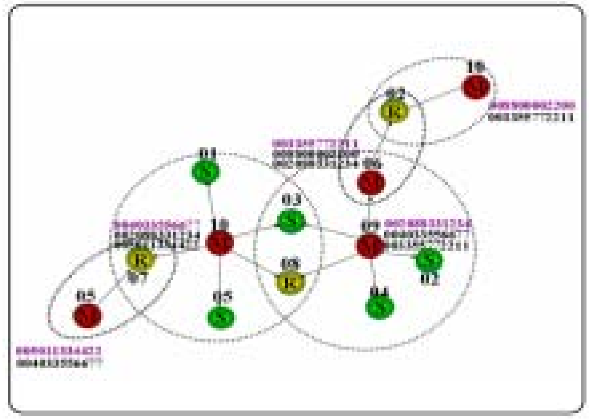

Figure 2: Scatternet construction case

3. Check scatternet is connectted or not connected:

Each slave of piconet will check the different Piconet slave with itself in the transmission range. If the slave which belong different piconet weight bigger than, and not contain PID which not the same with itself. Than switch role to master and run page procedure to formed new piconet and connect them into scatternet.

Check haven't it separate and not connect with piconet of scatternet, we can through piconet (PID) to determine this piconet have ever connect near piconet through any slave. If not, it will create new piconet to connect piconet and switch PID number of piconet for each other.

4.4 Scatternet Recovery Algorithms

If the scatternet recovery algorithm is applied, each device sets a recovery start timer with the value *Tr* when it starts the scatternet recovery algorithm. When the timer is expired, a device starts the scatternet recovery algorithm. Frist we must determine these devices are to leave, we will to discuss with three kind of device in piconet for Bluetooth recovery method. Simulation conditions are shown below.

(1) A slave may get to be out of communication range of master because of mobility.

A slave, which hasn't received poll packets from master for a polling time, assumes that it no longer belongs to the piconet, and change state into page scan state, Besides the master polling slave not responding, than to notify near piconet's master and page this slave join piconet. Use this method can to avoid run inquiry/nquiry scan procedure redundancy, improving recovery time. If it to exceed time *Tr* can not be paged by any master, than to execute scatternet formation algorithm again.

(2) A bridge may get to be out of communication range of master because of mobility.

If master want communication with another piconet's master, but to detect can not communication with another by bridge. So, the master can use backup bridge list create by bridge selection algorithm to elect bigger weight as new bridge. Use this method can to reduce cost that recreate bridge. Bridge will detect link that communication with master any time. If link break and change role to slave, then notify master recreate bridge.

(3) A master may get to be out of communication range of master because of mobility.

When master polling all slave of piconet not responding, Assumes that it no longer belongs to piconet. Than, reset device and execute scatternet formation algorithm again.

5. Simulation Results and Analysis

Bluehoc [12] developed at IBM is a well-known Bluetooth extension module for ns simulation tool.

In the following subsections, we assume the radio range of the Bluetooth devices is 10 meter in our simulation case.

We show the tradeoff between *Td* and resulting topology, and justify the chosen of *Td* proposed in our paper. Then we compare the setup delay of our algorithm and BTCP in [4].

5.1 Discovery Timeout (*Td*)

We assume the Bluetooth devices are distributed in a room with space of 10x10 square meters; we show the result under different *Td* in Figure 3. We can see the tradeoff between *Td* and connected ratio. However, large *Td* increases the formation delay.

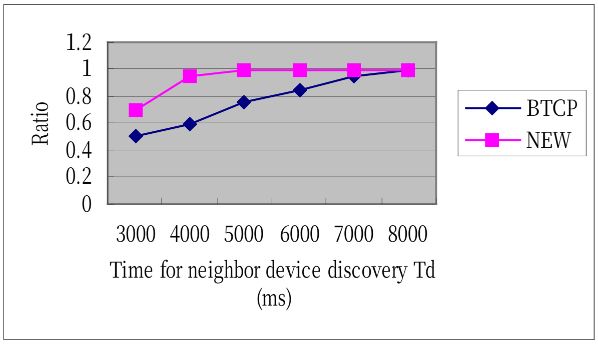

Figure 3-a: Number of devices = 20

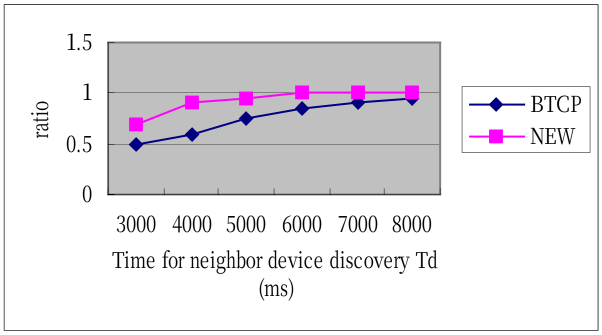

Figure 3-b: Number of devices = 30

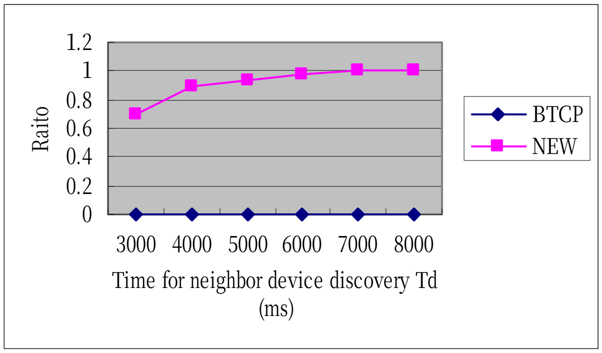

Figure 3-c: Number of devices = 40

Figure 3: show the ratio of the number of detected neighbor devices to the number of all neighbor devices.

Moreover, in Figure 3 we can see that the connected ratio up to 95%~100% when *Td* equals 5200ms. By comparing the four charts in Figure 3, we see that the connected ratio does not depend on the number of devices, but depends on the value of *Td.* In Figure 3-c, we can see BTCP can not be used more than 36 device number.

5.2 Setup Efficiency

In this subsection, we compare our algorithm with BTCP proposed in [4]. The setup delay is shown in Figure 3. We consider the scenarios that devices spread in a room of space 10×10 square meters. The BTCP is proposed only for devices that are within radio range of each other. We choose *Td* equal to 5200ms. In Figure 3, the BTCP do not describe how a topology should be constructed when devices are more than 36.

BTCP has a restriction that all Bluetooth devices are within radio range of each other. Our algorithm aims to release the restriction. In BTCP, a device stops neighbor discovery when it is beaten. In our algorithm, a device changes its state alternation sequence when it is beaten, and stops neighbor discovery when it cannot find neighbors with different state alternation sequences. Thus in our algorithm, a device cannot stop neighbor discovery when it is beaten. We can see it as the penalty to release the restriction.

Evidently, our algorithm is as efficient as BTCP. We analyze the reason as follows. The setup delay of BTCP is dominated by the time used in leader election.

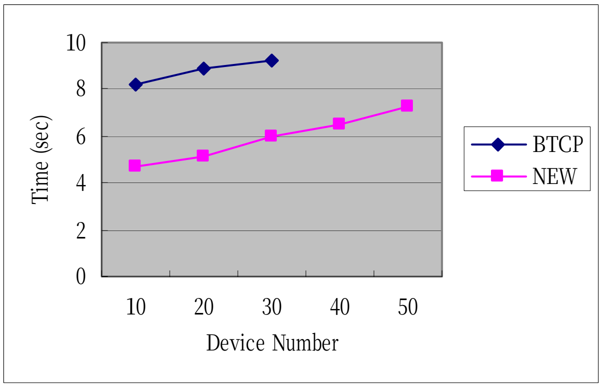

Figure 4: Scatternet formation time for NEW and BTCP

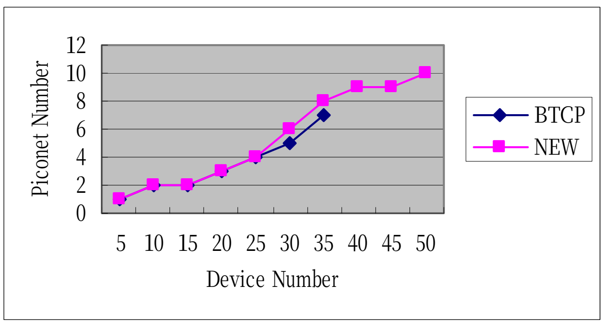

Figure 5: Number of Piconet for NEW and BTCP

6. Conclusion and future work

In the paper, a new scatternet formation protocol is designed to select master and relay devices properly so that the resulting scatternet can live longer. The scatternet formation can be constructed quickly and maintained by our protocol efficiently when the topology of the network changes frequently. Simulation results show that our protocol outperforms BTCP in many respects including construction time and network live time.

References

[1] The Bluetooth Interest group," http://www.bluetooth.com."

[2] Han-Chiuan Luo, Hsiao-Kuang Wu, and Gen-Huey Chen, "A Fast Bluetooth Symmetric Link Formation Protocol-SF/CAI (Scan First / Collision Avoidance Inquire), " *Wireless Conference on European*, 2002, pp. 50-57.

[3] Alok Aggarwal, Manika Kapoor, Lakshmi Ramachandran, and Abhinanda Sarkar, "Clustering algorithms for wireless ad hoc networks," *Proceedings of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 2000, Aug., pp. 54-63.

[4] Theodoros Salonidis, Pravin Bhagwat, Leandros Tassiulas, and Richard LaMaire, "Distributed Topology Construction of Bluetooth Personal Area Networks," *INFOCOM*, 2001, vol. 3, pp. 1577 –1586.

[5] Ching Law and Kai-Yeung Siu, "A Bluetooth scatternet formation algorithm," *GLOBECOM*, 2001, vol. 5, pp. 2864-2869.

[6] Godfrey Tan, Allen Miu, John Guttag, and Hari Balakrishnan,"Forming Scatternets from Bluetooth Personal Area Networks," *MIT Technical Report*, MIT-LCS-TR-826, October 2001.

[7] G.V. Zaruba, S. Basagni, and I. Chlamtac, "Bluetrees - scatternet formation to enable Bluetooth-based ad hoc networks," *IEEE International Conference on Communications (ICC)*, 2001, pp. 273-277.

[8] Z. Wang, R.J. Thomas, and Z.J. Haas, "Bluenet - a New Scatternet Formation Scheme," *35th Hawaii International Conference on System Science* (HICSS-35), 2002, pp. 61-70.

[9] S. Basagni and C. Petrioli, "A Scatternet Formation Protocol for Ad Hoc Networks of Bluetooth Devices," *IEEE Vehicular Technology Conference*, 2002, vol. 1, pp 424-428.

[10] C.C. Foo and K.C. Chua, "BlueRings-Bluetooth Scatternets with Ring Structures," *IASTED International Conference on Wireless and Optical Communication* (WOC), 2002, pp. 50-57.

[11] T.-Y. Lin, Y.-C. Tseng, and K.-M. Chang, "Formation, Routing, and Maintenance Protocols for the BlueRing Scatternet of Bluetooths," *Hawaii Int'l Conf. on System Sciences (HICSS)*, 2003, pp. 50-57.

[12] http://www-24.ibm/developerworks/projects/bluehoc