Invited Paper

DOI : 10.3745/JIPS.2009.5.4.167

Journal of Information Processing Systems, Vol.5, No.4, December 2009 **167**

Black Bridge: A Scatternet Formation Algorithm for Solving a New Emerging Problem

Minyi Guo*, Yanqin Yang*,, Gongwei Zhang*, Feilong Tang* and Yao Shen***

Abstract: Nowadays, it has become common to equip a device with Bluetooth. As such devices become pervasive in the world; much work has been done on forming them into a network, however, almost all the Bluetooth Scatternet Formation Algorithms assume devices are homogeneous. Even the exceptional algorithms barely mentioned a little about the different characteristics of devices like computational abilities, traffic loads for special nodes like bridge nodes or super nodes, which are usually the bottleneck in the scatternet. In this paper, we treat the devices differently not only based on the hardware characteristics, but also considering other conditions like different classes, different groups and so on. We use a two-phase Scatternet Formation Algorithm here: in the first phase, construct scatternets for a specified kind of devices; in the second phase, connect these scatternets by using least other kinds of devices as bridge nodes. Finally, we give some applications to show the benefit of classification.

Keywords: *Bluetooth, Statternet Formation, Bluetooth Communication Protocol*

1. Introduction

Bluetooth, the new technology named after the 10th Century Danish King Harold Bluetooth, is an always on, low-power, and short-range radio technology, aiming at simplifying communications among Internet devices.

Bluetooth uses low-cost transceiver microchips in each device. It supports the universal short-range wireless capabilities using the 2.4 GHz band which is available globally for unlicensed low-power uses. And Bluetooth divides the band into 79 channels (each 1MHZ wide) and changes channels up to 1600 times per second. The communication range supported by Bluetooth is power-class-dependent: class 1 supports maximum 100 meters, using 100mW; class 2 supports maximum 10 meters, using 10mW; class 3 supports maximum 1meter, using 1mW. In most cases, the effective range of class 2 is used. So in our paper, we mainly refer to pure class 2 network. We say two devices are in communication range, if the distance between these two devices is less than 10 meters.

Bluetooth is used in many products, such as mobile phones, printers, headsets, keyboards, game consoles. Though the original goal of Bluetooth is just to replace cable, it is convenient to use Bluetooth to construct a PAN (Personal Area Network), which provides a "last meter" solution for application personalization. Another important usage of Bluetooth is constructing an ad hoc wireless network. Comparing with other wireless networking like Wi-Fi, Bluetooth devices are cheaper and easier to connect. If two Bluetooth-based devices want to connect to each other, they just need to be in communication range, but they must play different roles: one act as master, the other acts as slave. A master only can connect up to seven slaves. The network formed by a master and its slaves is called piconet. Two piconets can form a bigger network by sharing a device, which appears in both piconets. The network formed by several piconets is called scatternet. There has been much research on scatternet formation, and this technology is useful in future.

Generally speaking, we can partition Bluetooth- based devices into several kinds, according to different aspects of the devices. However, in order to formulate the problem and simplify experiments, we only classify the devices into two kinds in this paper, and more kinds' classification will be researched in future. We first construct scatternets for devices of the same kind without depending on other kind devices. Then we merge these scatternets by using other kind devices as bridge nodes. Our goal is to make these bridge nodes as few as possible. The biggest difference between our algorithm and pervious scatternet formation algorithms is: we treat devices differently, but they treat devices equally or almost equally. At last, we propose some applications to show the advantage of classification.

The remaining of this paper is structured as follow: In

Manuscript received November 30, 2009; accepted December 7, 2009.
Corresponding Author: Minyi Guo
* Department of Computer Science and Engineering, Shanghai Jiao Tong Univ., Shanghai, China {guo-my,tang-fl,shen_yao}l@cs.sjtu.edu.cn, {yang-yq,zhang-gw}@sjtu.edu.cn
** Department of Computer Science and Technology, East China Normal University, Shanghai, China (yang-yq@sjtu.edu.cn)

section 2, we talk about related works; in section 3, we describe Bluetooth specification simply and formulate the problem; in section 4, we explain the algorithm and analyze some parameters; in section 5, we give some applications. At last, we conclude the paper and point out the future work.

2. Related work

Many Bluetooth Scatternet Formation Algorithms have been proposed prior to this work. R.M Whitaker et.al and I. Stojmenovic et.al give a very detail survey on those, and classify these algorithms into different categories according to different criteria, like guarantee connectivity, degree limitation, and topologies and so on[3,10,11,13,14].

One of the main categories is about topologies. There are many topologies used in Bluetooth scatternet formation: tree, ring, mesh, 1-Factors [14]. G.V. Zaruba et.al (Bluetree) construct the Bluetooth scatternet into a tree structure [15]. The tree was grown from a root, and the number of roles of each node in the tree can assume are limited to two. Obviously, the root of the tree leans to be bottleneck. S. Sunkavai et.al propose a mesh topology scatternet formation algorithm [4]. This kind of scatternet has short delays for new node acceptance. C.C. Foo et.al propose the use of ring structure for scatternet [16]. The advantages of this structure appear on many aspects: its added reliability, ease of packet routing and low scatternet scheduling overheads. As for 1-factor, people may be not familiar. 1-factor is a special matching in a Graph G. A matching in a Graph G is a subset of edges in G where no two are incident. If the matching has half edges of G when there are even edges in G, we call this matching a perfect matching. If the edges (n) are odd, and the matching has (n-1)/2 edges, we call this matching a near-perfect matching. A perfect or near perfect matching is known as 1-factor. S. Baatz et.al use 1-factor structure to ease the problem of inter-piconet scheduling [17].

Another important category is about centralized formation or distributed formation. The strongpoint of centralized formation is that it can gain the knowledge of the graph and then find the best performance for the Scatternet. On the other side, due to it has to know the visibility graph, its use is limited. M. Ajmone-marsan et.al use a min-max formation to find the optimal topology that provides full connectivity [9]. H. Sreenivas et.al use genetic algorithm to construct the Scatternet with the goal to minimize the number of created piconet [8]. Distributed formation can provide low algorithm complexity and energy cost. T. Salonidis et.al first build the Bluetooth Scatternet using distributed logic [7,12]. L. Barrière et.al introduce a distributed formation solution based on projective geometry for single-hop network, where each device is in the communication range of any other devices. The node in the final Scatternet has limited connected degree [2].

3. Preliminaries

3.1 Bluetooth architecture

Below is a simplified Bluetooth architecture [6]:

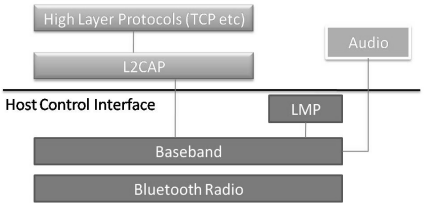

Fig. 1. Bluetooth Architecture

The radio layer defines the requirement of the Bluetooth transceiver devices operating in the 2.4 GHZ ISM band [1]. It is the lowest layer in the Bluetooth architecture, and uses pseudo-random hopping sequences with a fast hopping rate of 1600 hops per second.

The baseband layer is similar to the physical layer in general network architecture. The baseband manages many things like physical channels, links, error correction, hop selection and so on. Here we will mostly concentrate on link. The set up of connection between two devices need two steps: inquiry procedure and page procedure. The Bluetooth device uses inquiry procedure to know the address of the other device. Then the device performs the page procedure to connect to the other one using the returned address.

LMP (Link Manage Protocol) takes care of link configuration and authentication. The authentication uses a link key. If two devices do not have common link key, it can be created from a PIN. Besides, the link key can be changed temporarily, thought this change can only be valid for the session. Another important use of LMP is its support for name request to another device. The name consists of a maximum of 248 bytes according to the UTF-8 standard. If you want to close the connection between two devices, just use LMP to detach them.

HCI (Host Control Interface) is the interface to LMP and baseband, and it can access to hardware status and control registers.

All protocols above HCI are software based. L2CAP (Logic Link Control and Adaptation Protocol) is the lowest

layer of such protocols. The connection primitives it provides are: set up, configure and disconnect.

For other and more detail information, please refer to [1, 6].

3.2 Problem formulation

Given a set of Bluetooth devices, we partition them into two classes according to a specified criterion like CPU Speed or affiliated Group. Our goal is to create Scatternets for a specified class, and we should make the number of Scatternets and the number of bridge nodes minimum. The reason we want to decrease the number of Scatternets is to make connectivity. If you can merge two Scatternets into one, as a result, any device in both Scatternets can communicate with more devices. The advantage of decreasing bridge nodes will be told in the "Application" section.

There are mainly three cases for the problem:

- Case I:

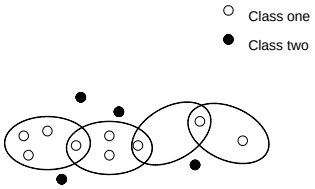

Fig. 2. Totally connectivity case

In this case, any two nodes of class one can connect to each other by using multi-hop, without a node of class two as a bridge node.

- Case II:

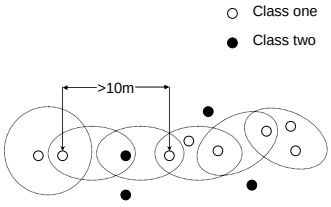

Fig. 3. Partly connectivity case

In this case, devices of class one can form only one Scatternet, but it should use devices of class two as bridge nodes. Our goal is to make these nodes least.

- Case III:

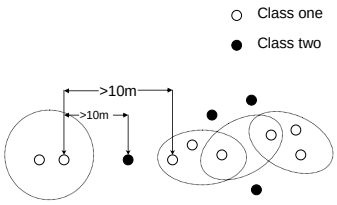

Fig. 4. Non connectivity case

In this case, the number of formed Scatternets of class one is more than one. So we should first make this number least. Then we should make the number of bridge nodes least.

4. Scatternet formation algorithm

We use a two-phase based algorithm to achieve our goal. In the first phase, we use an existing algorithm to construct Scatternets for devices of the same kind. In the second phase, we connect these Scatternets by using devices of other kind as bridge nodes. Then we can make the number of Scatternets least. Meanwhile, we should use as few bridge nodes as possible to connect those Scatternets.

In order to narrate clearly, we name one kind of devices as white devices, and call the other kind as black devices. We will build Scatternets for white devices and use black devices as bridge nodes.

4.1 Phase I

In this phase, we will construct scatternets for white devices, without containing black devices.

Much work has been done on this topic, any algorithm can use in phase I theoretically. Because we will use the concept of super node in second phase, the algorithm had better be centralized. If you use other algorithms, we may need to use election algorithm to elect a super node out of the master nodes in the same Scatternet.

When we choose an algorithm for phase I, we take the good properties listed below into account [5]:

- Small number of piconets: Since all piconets share the same set of 79 channels, there will be more collisions when there are more piconets.
- Small maximum degree of the devices: The degree of a device is the number of piconets to which the device belongs. Since piconets communicate through shared nodes, if a node belongs to many piconets, it can be the bottleneck of inter-piconet communication
- Network diameter: This is the maximum number of hops between any pair of devices

After choose an appropriate algorithm. We should modify the connection function. We only connect two devices of the same kind. This can be implemented in many ways: one way is to hash Group ID or CPU Speed or other criterion into PIN, then use this PIN to create the link key, only the same kind device can have the same key; another way is to get name of another device using LMP, then extract

information from the name for judging kinds, and disconnect the device if it is a different kind.

4.2 Phase II

After Phase I, we get several scatternets only formed by white devices:

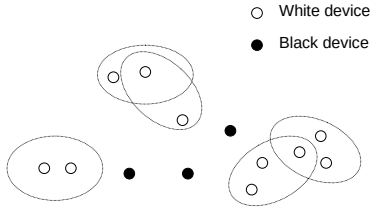

Fig. 5. Result of Phase I

We do not show all black devices in the picture.

What we should do in phase II is to connect these scatternets by using black devices as bridge nodes, and our goal is to use the fewest black devices.

At first, all the super nodes in each Scatternet perform the following Main procedure:

```
Main()
1. For each white device u in the scatternet
2.     Call Send() for node u
3. EndFor
```

When the super node executes this procedure, it asks all the nodes including itself in the scatternet to send messages to probe for other scatternets.

The white device in the scatternet performs the following Send procedure.

```
Send()
1. Construct a message
2. Send this message to neighbor black devices
```

The white device firstly creates a message, and then only sends this message to all the black devices it can reach in single hop. The black devices will forward the message. At last, the message will reach other scatternet.

The message we send here has a scatternet ID and a hop count. The scatternet ID represents where the message comes from. In order to make the scatternet ID unique, we can use the physical address of super node as its Scatternet ID. The hop count increase 1 every time when it is transmitted to next node. And the destination scatternet will know how many hops the message takes from the source scatternet. Besides the message also has to record the black devices path on which it is routing. So the destination scatternet knows how to transfer the data to the source scatternet in the shortest path.

Each white device only prepares to receive messages from other scatternets and it will not forward the messages, excluding submitting them to super node. Below is the Receive procedure for white devices:

```
Receive()
1. When the white device receive a message m
2. If the message comes from the scatternet
3.     throw away the message
4. Else
5.     send the message to the super node //don't
       increase hop count here
6. EndIf
```

The super node in the scatternet will analyze the messages to get the best path which makes bridge nodes least. This is the Analyze Procedure:

```
Analyze()
1. If message m is from a new scatternet
2.     record the scatternet ID and the hop count
       and the black devices path
3. Else
4.     If the new hop count is less than the old
       hop count
5.         update the hop count and the black
           devices path, and other path information
6.     Else If the new hop count is equal to old
       one
7.         Record the new path
8.     EndIf
9. EndIf
```

By performing the sending and receiving procedure, the scatternets will get the fewest black devices as bridge nodes. We take an example as follow:

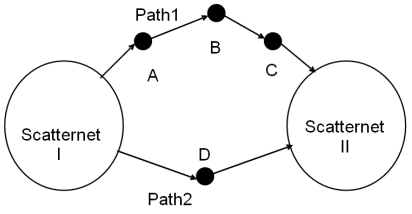

Fig. 6. An example to describe scatternet's sending and receiving procedure

Two messages send by Scatternet I will finally arrive at Scatternet II. If the message passing through Path1 arrives at Scatternet II first, Scatternet II records (Scatternet I, 4,

[A,B,C]), which means the message comes from Scatternet I, uses 4 hops, and the routing path is black devices A,B,C (Line 2 in Analyze procedure is performed). Then the other message passing through Path2 arrives at Scatternet II, Scatternet II updates (Scatternet I, 4, [A, B, C]) to (Scatternet I, 2, [D]) (Line 4-5 Anaylyze procedure is performed). It means it only needs one black device to connect Scatternet I and Scatternet II. In another case, the message passing through Path2 arrives at Scatternet II first; Scatternet II records (Scatternet I, 2, [D]). And when message passing through Path1 arrives, Scatternet II does not do anything, because the new hop count is more than old hop count. So no matter which case occurs, we finally get the shortest path.

When there are more scatternets, things will be a little complicated.

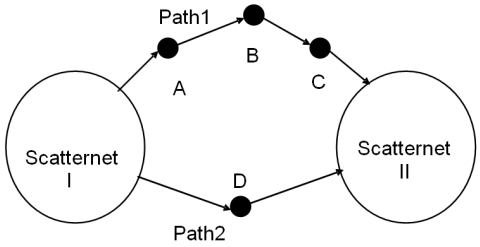

Fig. 7. A more complicated case

Here path1 and path2 both use least bridge nodes. Because the shortest path between Scatternet III and Scatternet II share bridge nodes with Path2, we choose Path2 here. When the scatternet receive no more messages, it seems that the formation has completed. Then the super node will delete the redundant paths (Line 7 in Analyze Procedure) by comparing the number of share nodes in each path. Because we emphasize the least number of bridge nodes here, we do not take it into account that a bridge node may be bottleneck. We will do it in future work.

The black devices are just used for forwarding messages.

Now we start to describe black devices' sending procedure:

```
Send()
1.  If there are messages in the message queue
2.      For each node w in its communication range
3.          send message to w
4.      EndFor
5.  EndIf
```

Of course, the black devices can refuse to receive a message. But if the black devices agree to receive the message, they perform the following Receive procedure:

```
Receive()
1.  When a message comes
2.  check the scatternet ID
3.  If have stored scatternet ID and hop count is less
      // There has been another message coming
      from the same scatternet in a shorter path
4.        throw away the message
5.  Else
6.        store scatternet ID and hop count
7.        Increase the hop count and add the de-
      vice itself to the path
8.        add the message the queue
9.  EndIf
```

Line 3 also has another important use: it can avoid messages run in a circle.

4.3 Analysis

Message plays an important role in our algorithm. Message complexity analysis is the first thing we should do.

Before analyzing, we assume the following parameters:

- The total number of devices is n
- The total number of black devices is m

So we have (n-m) white devices. According to our algorithm, every white device will create messages and send them to neighbor black devices. There are at most m black devices in a white device's communication range. So white devices send at most (n-m)m messages. For every black device, it is responsible for forwarding messages, so the messages it sends depends on its receiving messages. These messages all originate from white devices. So under worst case, a black device will receive (n-m)m messages, and sends them to its neighbors devices. Each black device sends (n-m)mn messages at most. All black devices sends at most $(n-m)^2mn$ messages. All sending messages are $(n-m)m+(n-m)^2mn=O(n^4)$.

This is the message complexity for worst case. Then we will do an experiment to analyze the message complexity.

We set the Area to $50x50(m^2)$ in our experiment, and distribute n nodes to this Area in random position. We increase n from 10 to 80 by a step of 10.

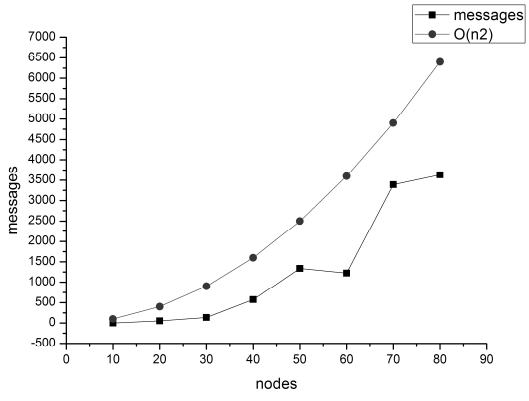

Fig. 8. Messages used

From the graph, we can see that, in fact, messages used are less than n^2, where n is the number of the node. This is because, there are often limited black nodes in a white device's neighbor, so white devices usually send $c_1(n-m)$, where c_1 is the average number of black devices in a white device's neighbor. This is also true for the black devices: there are usually a few devices in its neighborhood. So black devices send $c_1(n-m)c_2m$ messages, where c_2 is the average number of devices in a black device's neighborhood. This means the message complexity is $O(n^2)$ for most cases. In fact, this also can explain the line slope. As there are more nodes in the area, there are more neighbor devices for a device. This means c_1 and c_2 are larger, so the messages will increase more quickly as n grows. A device may be isolated from other devices. This will decrease the sending message. As a result, there are some exceptions when n grows.

The first goal in our algorithm is to make the number of scatternets least:

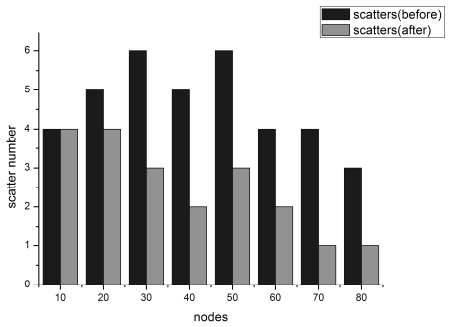

Fig. 9. Scatternet number compare

From the graph, we can see we actually reduce the number of scatternets. When the total number of nodes is 10, because they are sparse in the area, we cannot decrease the scatternets even using black nodes as bridges.

Now we can turn to bridge nodes:

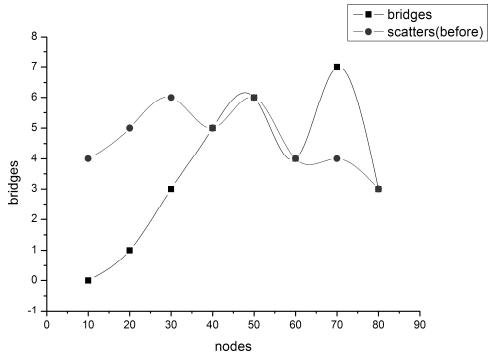

Fig. 10. Relation between scatternet number and bridge node number

As the number of scatternets increases, the bridge node number often increases, but not always. This is because if two scatternets are near, they need less bridge nodes to connect them.

5. Applications

Now we take some examples to show the advantage of the classification.

Scenario 1: Two (or more) groups of students observe different kinds of flowers in a garden. Each student has a Bluetooth device, like PDA or mobile phone. If a student finds its target flower, he or she may take the picture and sends it to other member of the same group. Members in other groups have no interesting to this picture.

The problem comes. We only need to broadcast the file to members in the same group. Assume a white device want to share a picture. The best case is that we only send the packages to other white devices. It is not practical. If you want to send packages between two white devices, you should connect them, and you may use black devices as bridge nodes. Now these black devices will receive packages which they do not care about. Because receiving message costs time and energy, it is better to involve less black devices.

At first, let us compare the number of involving black nodes among our algorithm (classify), bluetree, bluetree (optimize):

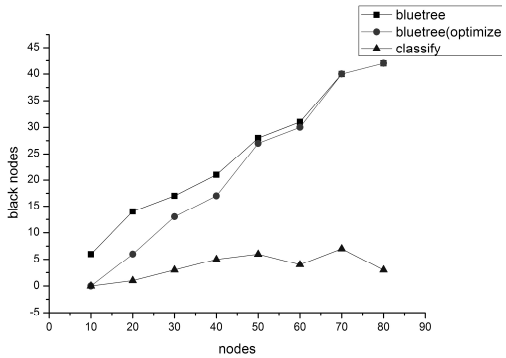

Fig. 11. Number of involving black nodes

We can see that our algorithm use the least black nodes, due to its one of the goals of our algorithm.

Suppose the size of the picture is 2M. According to Bluetooth specification [6], the payload of a packet is 0-2745 bits. If the payload contains day field, it should also contains a payload header. So the max size a package can contain is nearly 339 bytes. In order to totally transfer the picture, we should use 6187 packages:

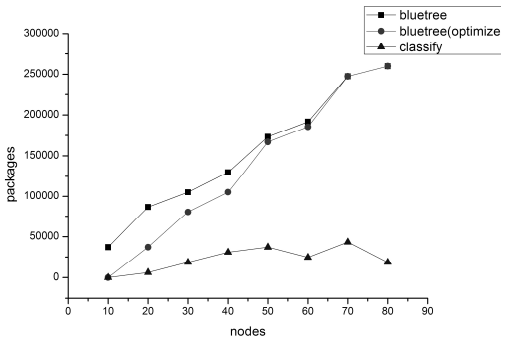

Fig. 12. Number of packages received by black nodes

Watching y-axis carefully, we can discover that if we do not control the number of involving black nodes, they will receive many packages. The type of package we choose will use 5 time slots, where each time slot is 625 μs. A black node will waste 3.125 ms to receive a package it does not care about. As these packages grow more, black nodes will waste more time and more energy to handle them. If there are tens of thousands of such packages, you can imagine how much time and energy they waste. So the number of involving black nodes is as few as possible.

Scenario 2: White device has transfer rate of 100k/s, and black device has transfer rate of 20k/s. Now we want to transfer files between white devices:

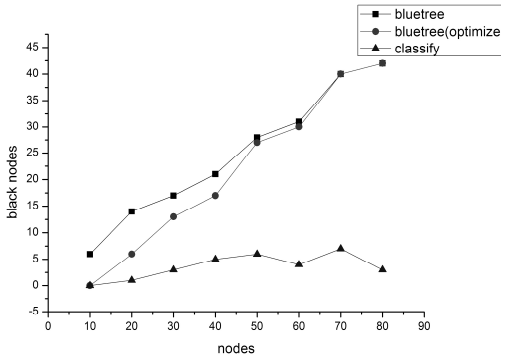

Fig. 13. Average transfer rate among white devices

If two white devices have to use a black node as bridge, the transfer rate will decrease. So the less the black nodes involving, the higher the average transfer rate is. Due to classification, we have a better average transfer rate.

6. Conclusion and Future work

In this paper, we first formulate a new emerging problem in scatternet formation: the need to classify devices. And then propose an algorithm to solve this problem. We use two phase in our algorithm: phase I to construct scatternets only for white devices, phase II merge these scatternets into one by using black devices as bridge. We use the fewest black devices in our algorithm.

There are some constrains in our algorithm, like not considering mobility and fault-tolerance. In future, we will do research on removing these constrains. Besides we will also try to improve our algorithm to reduce message complexity in future.

References

[1] Bluetooth tutorial, http://www.palowireless.com/infotooth/tutorial.asp

[2] L. Barrière, L. Narayanan, J. Opatrny," Dynamic construction of Bluetooth scatternets of fixed degree and low diameter," Proceedings of the fourteenth annual ACM-SIAM symposium on discrete algorithms.

[3] S. Jung, U. Lee, A. Chang, D.K. Cho, M. Gerla, "BlueTorrent: Cooperative Content Sharing for Bluetooth Users," Proceedings of the Fifth IEEE International Conference Pervasive Computing and Communications (PerCom'07) pp.47-56.

[4] S. Sunkavai and B. Rarnalmurthy, "MTSF: A Fast Mesh Scatternet Formation Algorithm for Bluetooth Networks," in Proceedings of IEEE Global Telecommunications Conference, Vol.6, pp.3594-3598, Nov., 2004.

[5] C. Law, K.Y. Siu, "A Bluetooth scatternet formation algorithm," Global Telecommunications Conference, 2001.

[6] Bluetooth Special Interest Group, http://www.bluetooth.com

[7] J. Haartsen, W. Allen, and J. Inouye, "Bluetooth: Vision, Goals, and Architecture", Mobile Computing and Communications Review, Volume 1, Number 2.

[8] H. Sreenivas and H.A. li, "An Evolutionary Bluetooth Scatternet Formation Protocol", Proceedings of the 37 Hawaii Int. Conf. on System Sciences, Jan., 2004.

[9] M. Ajmone-marsan, C.F. Chiasserini, A. Nucci, g. Carello, and L. de Giovanni, "Optimizing the Topology of Bluetooth Wireless Personal Area Networks", Proc. INFOCOM, 2002.

[10] J. Yun, J. Kim, Y.S. Kim, J. Ma, "A three-phase ad hoc network formation protocol for Bluetooth systems", Proc. 5th Int. Symp. Wireless Personal Multimedia Communications WPMC, Hawaii, Oct., 2002.

[11] C. Zhang, V. Wong and V.C.M. Leung, "TPSF+: A new two-phase scatternet formation algorithm for Bluetooth ad hoc networks", IEEE Globecom'04, Dallas, TX, Nov., 2004.

[12] T. Salonidis, P. Bhagwat, L. Tassiulas, R. LaMaire, "Distributed topology construction of Bluetooth per-

sonal area networks", Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM, 2001.

[13] R. M. Whitaker, L. Hodge, I. Chlamtac, "Bluetooth Scatternet Formation: A Survey", Ad Hoc Networks, Volume 3, Issue 4, July 2005, Pages 403-450.

[14] I. Stojmenovic and N. Zaguia, "Bluetooth scatternet formation in ad hoc wireless networks", Performance Modeling and Analysis of Bluetooth Networks: Network Formation, Polling, Scheduling, and Traffic Control, Auerbach Publications (Taylor and Francis Group) (2006), pp.147–171.

[15] G.V. Zaruba, S. Basagni, I. Chlamtac, "Bluetrees scatternet formation to enable Bluetooth-based ad hoc networks", ICC 2001.

[16] C. C. Foo and K. C. Chua, "BlueRings - Bluetooth Scatternets with Ring Structures," in Proceedings of IASTED International Conference on Wireless and Optical Communication (WOC), Banff, Alberta, Canada, Jul., 2002.

[17] S. Baatz, C. Bieschke, M. Frank, P. Martini, C. Scholz, C.Khul, "Building efficient Bluetooth scatternet topologies from 1-factors", Proceedings of the IASTED International Conference on Wireless and Optical Communications, WOC 2002, Banff, Alberta, Canada.

Minyi Guo
He received Ph.D. in Computer Science from the University of Tsukuba, Japan. Before 2008, he had been a Research Scientist of NEC Corp.,Japan, and professor of University of Aizu, Japan. He is now Head and Distinguished Professor of Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China. He has published more than 250 papers in international journals and conferences. His research interests include parallel and distributed processing, parallelizing compilers, pervasive computing, embedded software optimization, molecular computing, and software engineering. He is a senior member of IEEE, a member of the ACM, IPSJ and IEICE.

Yanqin Yang
She is a Ph.D. student at department of Computer Science and Engineering of Shanghai Jiao-tong University, Shanghai, China and she is also a lecturer at East China Normal University. From November 2007 to July 2008, she was a research assistant in Hong Kong Polytechnic University. She received the M.S. degree in Computer Science at Harbin Institute of Technology in 2002. Her research interests include embedded system and compiler optimization.

Gongwei Zhang
He is a master student in department of Computer Science and Engineering of Shanghai Jiao-tong University, Shanghai, China. From September 2007 to April 2008, he was a research assistant in University of Aizu, Japan. He received the B.S. degree in Mathematics at Shanghai Jiao-tong University in 2006. His research interests include embedded system and compiler optimization.

Feilong Tang
He received his Ph.D degree in Computer Science and Technology from Shanghai Jiao Tong University (SJTU), China in 2005. From September 2007 to October 2008, Dr.Tang was a visiting researcher in the University of Aizu, Japan. Currently he works with the Department of Computer Science and Engineering at SJTU, China. He is also a Postdoctoral Research Fellow of The Japan Society for the Promotion of Science (JSPS), Japan. His research interests include wireless sensor networks, grid and pervasive computing, distributed transaction processing and reliability computing.

Yao Shen
He received the M.S. degree and the Ph.D. degree in Computer Science from Shanghai Jiao Tong University, Shanghai, China in 2004 and 2007. Currently he is an Assistant Professor of the Department of Computer Science and Engineering, Shanghai Jiao Tong University. His research interests include topology control and power management in wireless networks and human- computer-interaction in pervasive computing.