

# A Bluetooth Scatternet-Route Structure for Multihop *Ad Hoc* Networks

Yong Liu, *Student Member, IEEE*, Myung Jong Lee, *Senior Member, IEEE*, and  
Tarek N. Saadawi, *Senior Member, IEEE*

**Abstract**—Bluetooth scatternets, integrating polling, and frequency hopping spread-spectrum in their medium access control protocol, provide a contention-free environment for Bluetooth devices to access the medium and communicate over multihop links. Currently, most available scatternet formation protocols tend to interconnect all Bluetooth devices at the initial network startup stage and maintain all Bluetooth links thereafter. Instead of this “big scatternet” approach, we propose a scatternet-route structure to combine the scatternet formation with on-demand routing, thus eliminating unnecessary link and route maintenances. To the best of our knowledge, this is the first effort to address on-demand scatternet formation with every detail. We introduce an extended ID (EID) connectionless broadcast scheme, which, compared with original Bluetooth broadcast mechanism, achieves very much shortened route discovery delay. We also propose to synchronize the piconets along each scatternet route to remove piconet switch overhead and obtain even better channel utilization. Furthermore, we present a route-based scatternet scheduling scheme to enable fair and efficient packet transmissions over scatternet routes. Network performance analysis and simulations show that scatternet routes can provide multihop wireless channels with high network utilization and extremely stable throughput, being especially useful in the transmission of large batches of packets and real time data in wireless environment.

**Index Terms**—*Ad hoc* networks, medium access control, on-demand routing, scatternet formation, scatternet scheduling.

## I. INTRODUCTION

CURRENTLY, IEEE 802.11, the standard for wireless local area network (LAN), becomes the most widely used medium access control (MAC) protocol in the study of multihop *ad hoc* networks, however, Xu and Saadawi [1] pointed out recently that because of the existence of *sensing range* (interfering range), which is typically larger than the usually considered *transmission range*, IEEE 802.11 cannot solve the hidden node and exposed node problems in multihop wireless networks. These problems may eventually lead to serious TCP instability and unfairness. In essence, the problems caused by IEEE 802.11 originate from its nature of medium contention.

Manuscript received May 1, 2002; revised October 22, 2002. This work was supported in part by the Communications and Networks Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

Y. Liu is with the Department of Electrical Engineering, City University of New York, Brooklyn, NY 11220 USA. (e-mail: yliu@ee.cuny.cuny.edu).

M. J. Lee and T. N. Saadawi are with the Department of Electrical Engineering, City University of New York, New York, NY 10031 USA (e-mail: lee@ccny.cuny.edu; saadawi@ccny.cuny.edu).

Digital Object Identifier 10.1109/JSAC.2002.807338

To avoid simultaneous usages of common channels, strict medium access control and channel management mechanisms are required especially for multihop packet transmissions. The newly developed Bluetooth technology [2] provides a good example to control medium access in contention free manners: inside each piconet, polling, and time-division duplex (TDD) are utilized for sequential medium access; different piconets employ different frequency hopping code-division multiple-access (FH-CDMA) channels to prevent mutual interferences. Bluetooth specification defines a scatternet structure to interconnect overlaying piconets as a multihop *ad hoc* network. Within a scatternet, two Bluetooth devices with their distance larger than their transmission range, but less than their sensing range use different time slots or stay in different piconets without interfering with each other, therefore, the hidden node and exposed node problems can be completely eliminated.

As the detailed scatternet formation and scheduling mechanisms were left open by the Bluetooth specification, a few efforts have been made to form fully connected, topology optimized scatternet from originally isolated Bluetooth devices [3]–[7]. All these algorithms put scatternet formation as an isolated action in data link layer without considering real traffic conditions and traffic requests. Any point-to-point links created at initial scatternet formation moment or added later on are kept active or maintained periodically to assure the connectivity of the whole scatternet at the data link level, despite most links may be fed no traffic most of the time. This unnecessary link maintenance wastes plenty of power—one of the scarce resources to wireless devices. Some researchers [8], [9] noticed this problem and suggested the integration of scatternet link formation with on-demand routing, but no thorough analysis and implementation details are available yet. There are also several studies concentrating on the design of scatternet scheduling [10]–[13], which coordinates the intrapiconet and interpiconet timing to keep the scatternet running fairly and efficiently. Most of these designs belong to a family named “rendezvous point” algorithm [9].

To make the scatternet structure more suitable to serve in mobile *ad hoc* networks, we propose to build scatternets only along the multihop routes with traffic demands. This route-type scatternet can be simply named as scatternet route. As the scatternet routes survive along with the on-going traffic flows, no unnecessary Bluetooth link maintenance is needed. The scatternet route is designed to maintain a special master-slave alternate structure so that every two direct adjacent devices in the route can be interconnected via a Bluetooth link. To remove the piconet switch

overhead imposed on the bridge devices, we suggest aligning the time slots of all piconets along each scatternet route. The formation of the scatternet route is similar to the common on-demand routing protocols. We introduce an extended ID (EID) connectionless broadcast mechanism [14] to expedite the route discovery and construction. Finally, in order to keep the scatternet route running at a stable and optimal state, we design a route-based scatternet-scheduling algorithm.

The rest of the paper is organized as follows: first, a short overview of Bluetooth technology is given; and then, the scatternet-route structure and its formation details are described; after that, the performance of the scatternet route is analyzed; and finally, we show our simulation results and draw conclusions.

## II. OVERVIEW OF BLUETOOTH TECHNOLOGY

Bluetooth technology is developed and promoted by the Bluetooth special interest group (SIG) as a global solution for short-range wireless communication operating in the unlicensed 2.4 GHz ISM (industrial, scientific, medical) band. Bluetooth radio employs a fast (1600 hops/s), FH-CDMA technique. A set of 79 1-MHz carriers have been defined at  $(2,402 + k)$  MHz,  $k = 0, 1, \dots, 78$ . Pseudorandom hopping sequence, with a nominal hop dwell time of 625  $\mu$ s, is derived from Bluetooth device address, unique to each device.

As a connection-oriented wireless communication technology, Bluetooth requires presetting of communication channel between two “conversation” devices. This is done by the formation of piconet. In general, the “conversation” initiator names itself as master and creates a piconet with a pseudorandom hopping sequence determined by its Bluetooth device address. The phase of this hopping sequence is derived from its native clock. The master has to detect its neighbors and invites them into its piconet. The participant neighbors serve as slaves of the piconet and follow the hopping sequence and timing of their master. The piconet setup, i.e., point-to-point link establishment between Bluetooth devices, is a two-step procedure: inquiry and paging.

*Inquiry Process:* The inquiry process is used to discover the existence of neighboring devices and to collect their information (like address, clock, etc.). In the inquiry stage, the master device keeps probing at different hop channels and in between listens for response packets. The 32 frequencies of the inquiry hopping sequences are divided into two 16-hop parts, named *A* train and *B* train. A single train must be repeated for at least 256 times before a new train is used. The neighbors of the master scan the inquiry channel periodically. The inquiry scan window should be a little longer than the length of a single train (16 time slots). So if, for example, the *A* train begins firstly and the scan frequency of one neighbor falls into the *A* train, it can “catch” the master within one inquiry scan window; however, if one neighbor scans with a frequency belonging to the *B* train, it has to wait until the *A* train cycle is over and the *B* train cycle begins. When the neighbor device catches the master for the first time, it enters a random backoff (RB) state to prevent several neighbors responding at the same time. According to the Bluetooth specification, the master device needs at least three train

switches, or has to stay at the INQUIRY state for 10.24 s (each train period lasts 2.56 s) to collect all possible responses from its neighbors in an error-free environment.

*Paging Process:* With the address and clock information of its neighbors, the master can contact the desired neighbor via the paging process, which is very similar to the inquiry process. Since the page hopping sequence is determined by the address and clock of the neighbor being paged and this information is already obtained by the master in the inquiry process, the master can estimate the current scan frequency of the neighbor and put it in the middle of the *A* train. This enables the neighbor to detect the invitation from the master within one page scan window and helps the master reduce the paging delay to a value less than one page scan interval (e.g., 1.28 s for mode R0).

After the setup of synchronization and physical channel via inquiry and paging processes, the control is moved from the Baseband to the link manager protocol (LMP) layer for link and security establishment. The logical link control and adaptation protocol (L2CAP) and its above layers can then startup for data transmissions.

As different piconets use different FH-CDMA channels, to enable the intercommunications between different piconets, Bluetooth allows one device to participate in two and more overlaying piconets by applying time multiplexing. Multipiconet devices serve as bridges to interconnect isolated piconets and extend the Bluetooth network from the simple piconet structure to a complex multihop *ad hoc* network—scatternet.

Within a piconet, the medium access is managed by a polling mechanism controlled at the master device. A TDD is used to realize the full duplex communication between each master-slave pair. The multipiconet device introduced in the scatternet structure has to obey the master-slave polling mechanism in each of its piconets. Before it switches from one piconet to another, it is suggested to enter one of the three power-saving modes to pause its activity in the old piconet. Switch overhead (up to one time frame, 1.25 ms) will present whenever the new piconet and the old piconet do not have their time slots synchronized. To achieve efficient network performance, it is extremely important to establish the polling schedules for the master devices, as well as the switching schedules for the bridge devices. Unfortunately, neither of these is specified in Bluetooth protocols.

## III. SCATTERNET-ROUTE STRUCTURE AND ITS ON-DEMAND FORMATION

Bluetooth, unlike most other wireless technologies with connectionless data link layer, requires explicit link formation and channel maintenance between communication devices. To conserve power, the physical channel and data link are only created and supported when two Bluetooth devices really desire to “talk” with each other. If no traffic request presents, Bluetooth devices always go to a low power “STANDBY” state without bothering links and connections with each other. This power saving principle, however, was ignored by most scatternet formation algorithms, which tended to interconnect all Bluetooth devices as a big scatternet at the initial network startup stage and maintain the full connectivity of the network at the data link level.

Instead of the big scatternet approach, we propose a scatternet-route structure to enable the dynamic establishment of Bluetooth links only along the traffic routes. The differences between our scatternet-route structure and the big scatternet structure lie in three aspects:

**Traffic Dependency:** The big scatternet is built independent of the traffic conditions of the network. Once established, all Bluetooth links in the big scatternet are maintained at all times to ensure the connectivity of the whole network, whether there is on-going traffic or not. The formation of a scatternet route is, however, initiated by the traffic request at one source device, which expects to contact another device (the destination) multiple hops away. A route is then built to relay packets from the source to the destination. As soon as their “conversation” is over, the scatternet route is released immediately to allow involved devices to take a rest at the low power state. In short, scatternet routes live on traffic flows.

**Network Coverage:** The big scatternet includes all the Bluetooth devices within its reach. Each device attached to the scatternet has to maintain at least one Bluetooth link with its neighbors to support the connectivity of the whole scatternet. In our scheme, one scatternet route contains only the devices along the traffic path from the source to the destination. All the other devices not involved in the on-going traffic stay alone in the low-power state.

**Combination With Routing:** In the big scatternet approach, the scatternet formation, which is implemented in the data link layer, is separated from the routing functions. An additional scatternet routing protocol is required to discover and set up multihop routes. The end-to-end route structure and the on-demand formation mechanism of the scatternet route suggest the integration of the scatternet formation with the scatternet routing. On the one hand, the source needs the help from the routing process to determine the devices to be included in the scatternet route. On the other hand, the routing process needs the scatternet structure information to complete the setting of routing table in each relay device. The combination of these two functions allows them to join their control packets and get rid of redundant control traffic. This is especially beneficial to power constraint wireless devices.

To sum up, the differences between the big scatternet approach and the scatternet-route approach are similar to those between table-driven routing protocols like destination-sequenced distance-vector (DSDV) [20] and on-demand routing protocols like *ad hoc* on-demand distance vector (AODV) [21]. The former maintains network-wide link or route information for fast route setup, but incurs substantial signaling traffic and power consumption. The latter suffers long route setup delays, however is more power efficient in mobile *ad hoc* environment. The readers are referred to [18] for more about this tradeoff. We believe on-demand approach is favored to power-limited wireless devices, like Bluetooth devices.

#### A. Structure Design

According to the Bluetooth specification, each Bluetooth device is required to establish master-slave relationship with its immediate neighbors before their communications. To interconnect several devices as a multihop scatternet route, this master-slave relationship has to be assured between every two

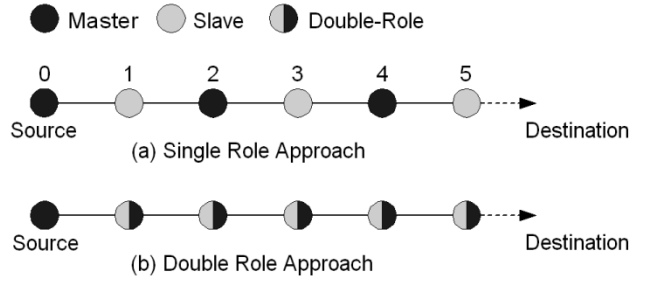


Fig. 1. Scatternet route structure.

direct adjacent devices, in other words, the master and slave roles have to be alternately placed along the scatternet route. Fig. 1 shows two possible structures of the scatternet route. In the single role approach, each “master” device creates a piconet to support its upstream<sup>1</sup> and downstream “slaves,” while each “slave” device serves as slave/slave bridge to connect its upstream and downstream piconets. It is easy to see that the single role scatternet route consists of minimum number of piconets. As all piconets in the Bluetooth network share the same set of 79 channels, fewest piconets in each scatternet route can keep the overall network interferences in a low level. Kalia *et al.* [15] also found out that the scatternet having only slave/slave bridges achieves higher system throughput and lower delay than other scatternet structures. Unfortunately, the single role requirement and the strict master-slave alternate order make this approach vulnerable to dynamic route changes and updates. The double role structure, with all relay devices having both the master and slave roles, is much more robust to the network variations. To obtain favorable network performance, as well as good flexibility, we propose to assign single roles to most devices along the scatternet routes; however, in critical areas suffering structure mismatch, we prefer to use double-role devices, i.e., master/slave bridges, to prevent the small-scale structure mismatch from spreading over the whole scatternet route.

Fig. 2 shows our scheme to utilize the double-role devices to fix the structure mismatch problems. For example, in Fig. 2(a), an old scatternet route (...1–2–3–4–5–6...) updates itself to a shorter one (...1–2–7–8–6...). All devices in the old route were set as single roles—master or slave/slave—for optimal network performance. In the updating, two new devices (7, 8) ask for the enrollment to the scatternet route to replace the old devices 3–5. As device 7 and 8 locate between two master devices, if we still enforce the single role principle here, all devices before 7 or after 8 have to change their roles to maintain the master-slave alternate structure in the new route. To localize the structure adjustment, we name device 7 as a double-role bridge and assign slave/slave role to device 8, so that device 7 can create its own piconet to connect device 8 and at the same time join the piconet of device 2 as slave. All the other devices, including device 2 and device 6, need not change their roles. In Fig. 2(b), structure mismatch happens when one scatternet route tries to bypass a broken link. Similarly, setting double-role at one of the new devices helps control the structure adjustment within a

<sup>1</sup>We use “upstream” for the immediate neighbor in the direction toward source and “downstream” for the one in the direction toward destination.

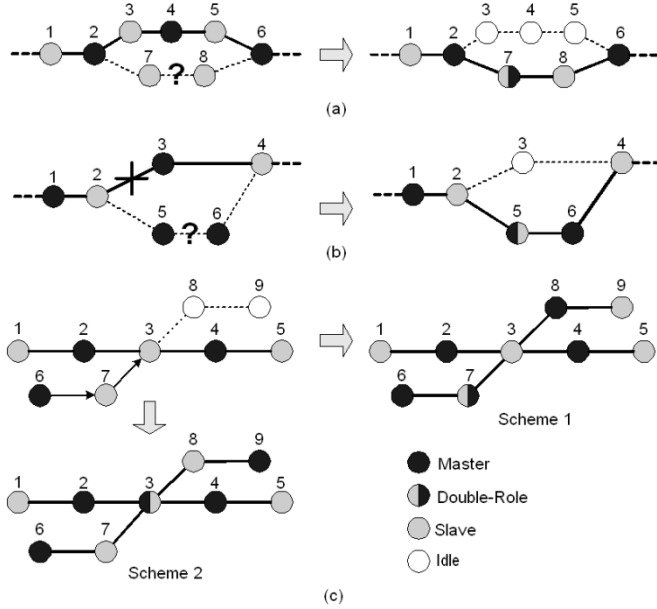


Fig. 2. Remedy of structure mismatch.

minimum range. In Fig. 2(c), a new scatternet route is created to traverse an existing route but causes structure mismatch at the crossover device. Two schemes are illustrated to define roles in the new route while leaving the structure of the old route untouched. Scheme 1 names the crossover device (device 3) again as slave in the new route, thus making it serve in four piconets! Since different piconets are not synchronized with each other, the crossover device, when switching from one piconet to another, has to take extra time to resynchronize with the new piconet. The more piconets one device serves in, the bigger switch overhead it suffers. In scheme 2, the situation is improved since the crossover device acts as master in the new route and only jumps among three piconets.

In fact, not only the crossover device, every bridge device, either slave/slave or master/slave, undergoes extra charges imposed by piconet switches. As shown in Fig. 3(a), the switch overhead at slave 1 wastes it one time frame (each time frame consists of two time slots) in each packet transfer cycle, so device 1 has to spend at least three time frames to relay a single packet. The master devices, although immune to the switch overhead, are slowed down by the slave/slave bridges. If all piconets along the scatternet route synchronize their time slots, like that in Fig. 3(b), the packet transfer cycle in each device can be reduced to two time frames, which means great increase in both channel utilization and network throughput. The piconet synchronization can also decrease the possibility of channel interferences, thus, further improving the channel capacity [16]. The scatternet-route synchronization can be implemented in the route formation stage. We will discuss it with more details in the following section.

### B. On-Demand Formation of Scatternet Route

Like most of the on-demand routing protocols, whenever there is a traffic request in a source device and no available route information can be used, a route-discovery-packet (RDP) is flooded to the whole network to find the destination. Upon receiving the first RDP, the destination sends a route-reply-packet

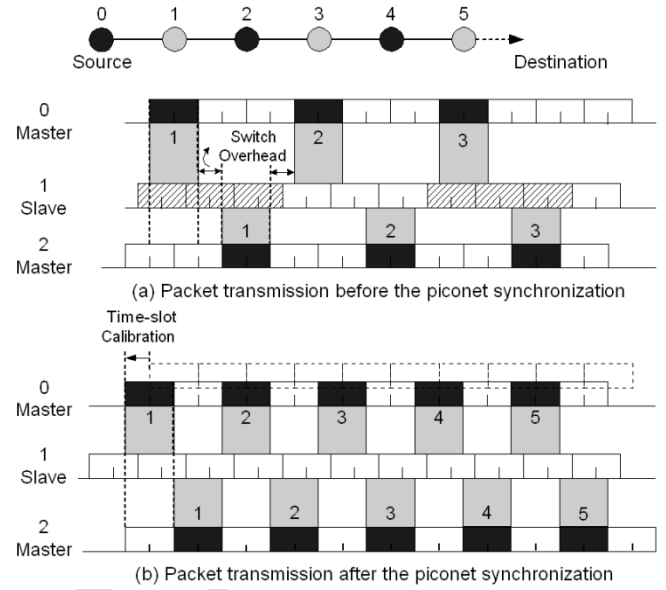


Fig. 3. Scatternet-route synchronization.

(RRP) back to the source along the new discovered route. Together with the feedback of the RRP, point-to-point Bluetooth links are created to connect the members of the new route and at the same time the routing tables of these devices are filled in with new discovered route information. When the RRP arrives at the source device, the scatternet route is also ready for the first data packet from the source.

Since Bluetooth asks for channel establishment, i.e., piconet setup, between two devices even if they attempt to exchange a single packet, this connection-oriented communication mechanism brings Bluetooth devices new challenges in packet flooding and multihop link establishment—the key components of the scatternet-route formation procedure.

1) *Flooding-Based Route Discovery*: Let us first look at the packet-flooding problem, or more straightforward, how to broadcast a single packet from one Bluetooth device in an efficient way? We investigate three possible broadcast schemes.

*L2CAP Broadcast*: This is the conventional Bluetooth broadcast mechanism involving the functions up to the L2CAP layer. In order to broadcast a packet out, the source device firstly initiates an inquiry process (with a delay of 10.24 s) to discover its neighbors; and then it pages all these neighbors one-by-one (with average page delay of  $1.28/2 = 0.64$  s per neighbor) to establish point-to-point links in the LMP level; after that, the RDP can be broadcast in the L2CAP layer by setting the active member address of the packet as 0; and lastly, the neighbors are detached, respectively, to rebroadcast the RDP.

We can make a simple calculation to estimate the route discovery delay under L2CAP broadcast mechanism. In the network shown in Fig. 4, a RDP sent from the source 0 has to travel at least seven hops before it reaches the destination 21. Suppose the RDP goes along the middle straight route, which is also the shortest path between the source and the destination. In most cases, each device in the straight route needs to send the RDP only to its three neighbors in the direction toward destination since those neighbors in the reverse direction have already

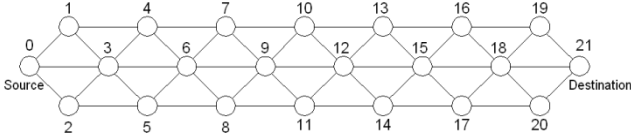


Fig. 4. Multihop Bluetooth network.

been taken care of by its upstream neighbors. If we count only the inquiry and paging delays and omit the small delays in other steps, the propagation of the RDP through the first six hops costs  $6 \times (10.24 + 0.64 \times 3) = 72.96$  s. In the last hop, when device 18 finds the destination 21 in the inquiry process, it contacts device 21 directly without bothering other neighbors, so the delay in the last hop is around  $10.24 + 0.64 = 10.88$  s. All together, the seven-hop route discovery needs at least:  $72.96 + 10.88 = 83.84$  s.

In practice, the route discovery delay is often longer than this value because of the “disturbances” from the neighbors outside the shortest route. For example, if both device 3 and device 1 discover device 4, but device 1 contacts device 4 in an earlier time, device 3 may get no response when paging device 4. This unsuccessful paging process produces a delay (paging time-out, 2.56 s) much larger than the normal paging delay (0.64 s), thus adding extra time expense.

In L2CAP broadcast scheme, only when all the neighbors are ready in the piconet, does the source device release the RDP. Therefore, the neighbors enter the piconet earlier have to wait a long period for the source to invite other neighbors. Since the source merely has a single packet to send, it is preferable to let each neighbor get the RDP right after it enters the piconet, so that the neighbor could continue the rebroadcast without additional waiting. Although this new approach introduces overhead for the individual transfer of the RDP to every neighbor, as the transmission of one single packet takes only one time frame (1.25 ms), the overhead is very small. Furthermore, we no longer bother the L2CAP layer since the RDP can be defined as a LMP control packet. This new approach can be simply named as LMP broadcast scheme.

**LMP Broadcast:** In this scheme, after the source finishes the inquiry process, it pages its neighbors one by one and transfers the RDP via a LMP procedure right after each paging. When receiving the RDP correctly, the neighbor is released instantly to spread the packet.

Look at the same network of Fig. 4, if every device in the middle straight route is lucky enough to have its downstream device in the same route as its first neighbor to page, such as device 0 pages device 3 first and device 3 pages device 6 first, the seven-hop route discovery needs only  $(10.24 + 0.64) \times 7 = 76.16$  s. However, if device 4 happens to be the first neighbor of device 3, device 6 has to wait its turn to get the RDP, which delays the flooding of the RDP from device 6. The unsuccessful paging may also happen here to introduce extra delay to the route discovery.

Although the LMP broadcast scheme simplifies the Bluetooth broadcast operation in some sort, it does not touch the primary factors leading to the big broadcast delay—the inquiry and page processes. To further reduce this delay, we have to go down to the Baseband layer for new broadcast mechanism.

**EID broadcast:** In fact, the inquiry process itself contains a very good connectionless broadcast mechanism: the master device sends out inquiry message continually in a particular inquiry channel to probe its vicinage, while its neighbors scan the same inquiry channel periodically to catch the possible inquiry message. Since the master changes its frequency train every 2.56 s and each neighbor scans the inquiry channel in a maximum period of 2.56 s, the inquiry message can arrive at all neighbors within 5.12 s. Therefore, if we put the route discovery information into the inquiry message, the average RDP broadcast delay can be reduced to  $5.12/2 = 2.56$  s, a huge decrease!

The inquiry message is originally designed as a small ID packet with the size of 68 bits. In each transmission time slot (TX), two inquiry packets are sent out with an interval of  $244.5 \mu\text{s}$ , which is used for frequency switch. Since most of modern frequency synthesizers can switch to a new frequency within  $100 \mu\text{s}$ , Phillips proposes to extend the size of the inquiry ID packet, so that more useful information can be attached to the inquiry message and broadcasted to the neighbors immediately [14]. This is shown in Fig. 5. If we leave  $100 \mu\text{s}$  for the frequency switch, 144 more bits could be added to the inquiry message.

The RDP should include such items as source Bluetooth device address (BD\_ADDR, 48 bits), destination BD\_ADDR (48 bits), upstream BD\_ADDR (48 bits), upstream clock (26 bits), packet sequence number ( $> 4$  bits), hop limit ( $> 4$  bits), EID type ( $> 4$  bits), etc. Altogether, a space of at least 182 bits is needed. Since one EID packet cannot provide such a big space, we have to employ two EID packets (as type 1 and type 2 in Fig. 6) to separate the load.

Unlike the original inquiry process, when the source device broadcasts the RDP embedded inquiry message, it does not require the instant acknowledgment from its neighbors. So the response time slot (RX) can also be used to broadcast the route discovery information. In our design, the EID type 1 is sent out by the source at the transmission time slot (TX), while the EID type 2 is sent out at the original response time slot (RX). When one of the neighbors scans the inquiry channel and catches the type 1 packet, it will wait for the type 2 packet at the corresponding “inquiry response frequency” one time slot after it receives the first packet. This whole process is illustrated exactly in Fig. 6. After the neighbor receives both these packets correctly, it changes to INQUIRY state to rebroadcast the RDP in the same way.

We indicated previously that the average broadcast delay in this EID approach is 2.56 s per hop. Seven-hop route discovery, thus, needs  $2.56 \times 7 = 17.92$  s, which is still much larger than the route discovery delay in common channel networks. Our study reveals that this large delay is ascribed to the current Bluetooth inquiry mechanism. Bluetooth allows devices to scan one out of 32 inquiry frequencies with the scan intervals up to 2.56 s, however, the inquirer each time probes only half of these 32 frequencies (i.e.,  $A$  or  $B$ ). As the inquirer does not know the scan intervals of its neighbors, it keeps probing one frequency train for at least 2.56 s before it changes to the other one. The neighbors “wake up” within this 2.56 s, but with the scan frequencies not falling into the current probing train, cannot acquire the inquiry packet and have to wait one or more scan intervals for the appearance of the proper train. To prevent the happening of this

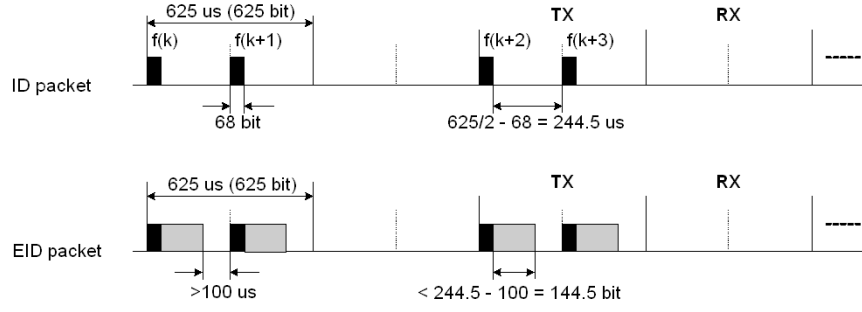


Fig. 5. EID packet in the inquiry process.

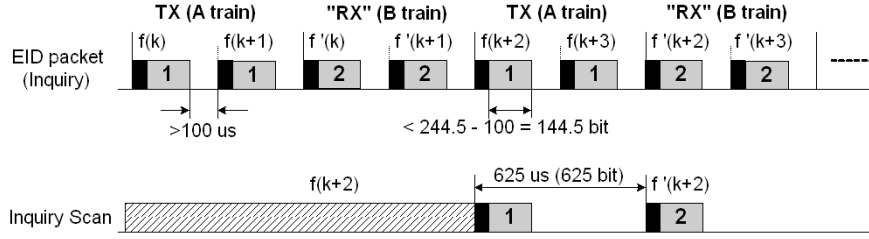


Fig. 6. Baseband EID broadcast scheme.

scan failure, we further propose to use the disengaged “inquiry frequencies” to replace the “inquiry response frequencies” in the broadcasting of the type 2 EID packets. As shown in Fig. 6, if A train is employed to broadcast the type 1 packets in the TX time slots, B train is used in the RX time slots to send out the type 2 packets, vice versa. In this way, the inquirer is able to probe the whole 32 inquiry frequencies within the period of one inquiry scan window. In other words, once the EID packets are available in the inquiry channel, the neighbors can catch them as soon as they “wake up.”

Now, the broadcast delay and the route discovery delay depend on the inquiry scan intervals (ISIs) of the Bluetooth devices. For example, if all Bluetooth devices use the maximum ISI of 2.56 s, the average broadcast delay is  $2.56/2 = 1.28$  s; seven-hop route discovery requires  $1.28 \times 7 = 8.96$  s, which is only 1/2 of the half-channel probing scheme (17.92 s). However, if the devices scan the inquiry channel more frequently, say with an ISI of 0.64 s, the seven-hop route discovery needs only  $7 \times 0.64/2 = 2.24$  s. The ISI of one Bluetooth device is determined by its load in the data channel and its power-saving mode. The setting of this parameter is in fact a tradeoff among route setup delay, channel utilization, and power consumption.

With proper packet broadcast mechanism, the source, as well as all the relay devices, is able to send the RDP out quickly and efficiently. Before the rebroadcast of the new received RDP, the relay device needs to check its “freshness” and creates an item in the routing table to record new route request.

2) *Backward Formation of Scatternet Route*: Upon receiving the first RDP, the destination responds immediately with a RRP, which starts up the scatternet formation process along the newly discovered route. The transfer of the RRP is much easier than that of the RDP. Each device in the new route simply pages its upstream neighbor, creates a physical channel in between and delivers the packet.

The RRP can be used by the relay device to transmit some of its parameters, such as its master/slave role, its address and

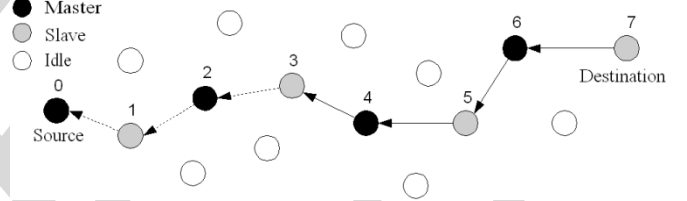


Fig. 7. Backward scatternet-route formation.

clock, etc., to its upstream device. With this information, the upstream device can attach itself to the new forming scatternet, completes its route setting and synchronizes its time slots. The upstream device then updates the RRP and transfers it to its own upstream device to guide this new device to the expanding scatternet and so on.

Consider the three key operations in each relay device when it is activated by the RRP.

*Scatternet formation*: As we are building a scatternet with alternate master-slave structure, the role of current device could be decided by its downstream neighbor, which is already a member of the expanding scatternet. For example, in the network showing in Fig. 7, device 4 has settled down in the scatternet as a “master.” It indicates this in the RRP transferred to device 3, so that device 3 can quickly find out its own role as “slave.” Device 3 simply joins the piconet headed by device 4 to extend the scatternet one more hop. Similarly, device 2 can decide its role as “master” when informed by the RRP from device 3. In order to join the scatternet, it creates a new piconet to connect device 3. Thus, hop-by-hop, all the devices along the route are linked up into the scatternet.

*Route setting*: In the route discovery process, each relay device adds a new item to its routing table to record the route request, but when the downstream information is not available then, the entry in the route table is left uncompleted. Now, since the downstream device leading to the destination is settled and

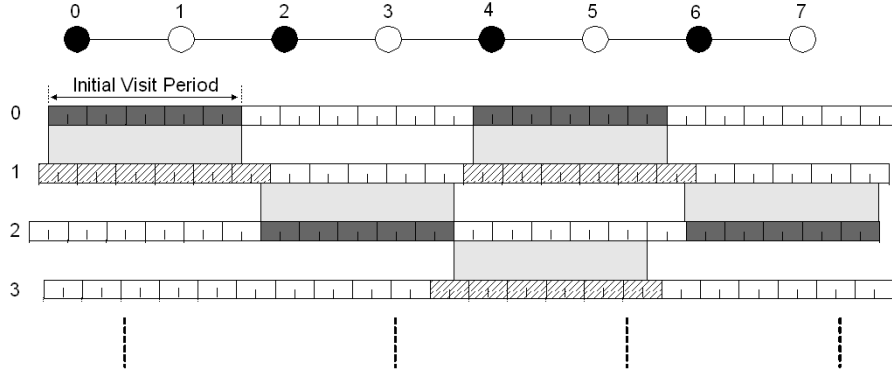


Fig. 8. Single route scheduling.

its identity information could be obtained from the RRP, the current device is able to complete the route entry. To make life easier, the routing table will also include the active member addresses or piconet addresses of the upstream and downstream neighbors.

*Route-wide synchronization:* The route wide synchronization can be achieved along with the propagation of the RRP too. Since the scatternet formation begins from the destination and the piconet covering the destination device is the first piconet of the scatternet route, it is natural to use the timing of this piconet as the standard. All the other piconets should calibrate the beginning of their time slots with this standard timing. We still use the network of Fig. 7 to illustrate this procedure. Here, the piconet headed by device 6 holds the standard timing. Device 5, when invited to this piconet, synchronizes with the standard time slot via common page process. Device 5 calculates the difference between the standard time slot and its native time slot and with the reference of this difference, device 5 can easily follow the standard time slot by adjusting the start point of its native time slot. When device 5 pages device 4 and transfers the RRP, it uses the standard time slot instead of its native time slot, so that device 4 can also synchronize with the standard time slot and get the timing difference. Device 4, as a “master” device, will use the standard timing in all its operations inside this route. Thus, the piconet headed by device 4 is synchronized with the piconet headed by device 6. This same procedure propagates to other piconets along the route. So finally, the timing of the whole scatternet route is unified.

We are not inclined to synchronize the clocks of different routes as it makes the protocol too complicated. If two or more routes have several devices in common, the shared devices must record the clock offsets (over their native clocks) of all the passing routes, so that they can quickly switch from one route to another by changing the clock offset. The route switch is very similar to the piconet switch and of course, it generates switch overhead as well.

When the source device receives the RRP and finishes the last-hop setup, the scatternet route is ready for use. However, to keep the scatternet route running properly and efficiently, we still get some work to do.

### C. Scatternet Scheduling

The special structure of the scatternet route and its on-demand formation mechanism demand a new route-based scheduling al-

gorithm. All relay devices along the scatternet route should coordinate their timing and regularize their behaviors to keep the scatternet route running efficiently and smoothly. The crossover device, serving in two or more scatternet routes, should also consider the fairness issue when scheduling route switch points and service periods.

Let us first examine the single route scheduling, or intraroute scheduling. As one scatternet route is created on-demand for a traffic request and survives along with the traffic flows, each relay device should frequently check its two intraroute neighbors for possible packets to transfer. To make it simple, we ask each device to visit its neighbors periodically. The initial timing of the relay devices is coordinated in the scatternet-route startup stage to prevent time mismatch between any master-slave pair. To be more specific, we briefly illustrate the scatternet-route startup process following the diagram of Fig. 8. Before the source of the scatternet route sends out the first packet, all relay devices along the route wait for the “activation” from their upstream neighbor. Source device 0 defines the initial visit period (IVP) and sends packets to device 1 first. After an IVP, device 1 is released to contact device 2, which is waiting right there for device 1. Device 1 spends the same IVP to transfer all packets from the source to device 2. When device 2 activates device 3 for continuous transfer, device 1 switches back to the source, which is always ready for new transfer or receiving. And then device 1 switches again to device 2, which just finishes the communication with device 3 and turns back to device 1. In this way, each device perfectly matches its timing with its neighbors and pushes packets quickly to the destination. The source can adjust the visit period (VP) dynamically for different traffic type: for sporadic and unpredictable transmission and receiving, the VP is set as a very small value, so that the relay devices can quickly detect the bursty packets and deliver them as soon as possible; for the transmission of large batch of packets, the VP can be enlarged to reduce the switch frequency of the bridge devices. This is beneficial to the unsynchronized scatternet route since the one-time switch overhead is amortized over more packets transmitted in one VP. Larger VP also enables the Bluetooth devices to use big size packets, such as DH3 and DH5, thereby achieves even more improvement in channel utilization.

The key of the multiroute scheduling is to determine the timing of the crossover device. Although each passing route synchronizes its own piconets, different scatternet routes do not align their time slots. This adds route switch overhead at the

TABLE I  
SCATTERNET-ROUTE THROUGHPUT

$S_{yn}$	Packet Type	$n$	Throughput (kb/s)	Network Utilization
1	DH1	1	57.6	5.76%
1	DH1	3	74.1	7.41%
1	DH1	5	78.5	7.85%
1	DH3	1	234.2	23.4%
1	DH5	1	310	31%
0	DH1	1	86.4	8.64%
0	DH1	3	86.4	8.64%
0	DH1	5	86.4	8.64%
0	DH3	1	292.8	29.28%
0	DH5	1	361.6	36.16%

crossover device. So it is preferred to decrease the route switch frequency and let the crossover device stay in one route for a pretty long period before switching it away. To treat different scatternet routes in a fair way, the crossover device assigns a “credit account” [13] to each route to track its capacity usage. When the route currently in service has its credit lowered to a minimum threshold, it is laid aside by the crossover device to give place to the next route with the highest credit. It is also possible to account other parameters, such as traffic priority or traffic condition, etc., in the calculation of route credits to make the route selection meet different quality-of-service (QoS) requirements. The neighbors of the crossover device check this “route switch” frequently for possible access opportunities and accumulate packets during the vacation periods. Once catching the crossover device, they make fully use of the service time to empty their storage and take over new packets. The big size Bluetooth packets, DH3 and DH5, can be used in the communications between the crossover device and its neighbors for higher efficiency in the bottleneck area.

#### IV. PERFORMANCE OF SCATTERNET-ROUTE

Let us follow the diagrams of Figs. 3 and 8 to derive the throughput expression of scatternet routes. Suppose one relay device each time collects  $n$  packets from its upstream neighbor and then switches to the downstream neighbor to transfer them. If the transmission of a single packet with the size of  $l$  costs  $m$  time frames, the total time for the relay of  $n$  packets is:  $(2 \times n \times m + s_{yn}) \times t_f$ . The  $s_{yn}$  here is the synchronization factor: for the unsynchronized scatternet route, it represents the piconet switch overhead and has the value of one (one round switches take exactly one time frame); for synchronized scatternet route, as no switch overhead exists, the  $s_{yn}$  is zero. Each time frame has the length of  $t_f$ . Now, we can have the expression of the scatternet-route throughput as

$$\frac{n \times l}{(2 \times n \times m + s_{yn}) \times t_f}.$$

Table I shows the scatternet-route throughputs and channel utilization (Bluetooth Channel Capacity = 1 Mb/s) for some typical cases. From the performance differences, we can conclude the following.

- Route wide synchronization does achieve considerable throughput improvement.

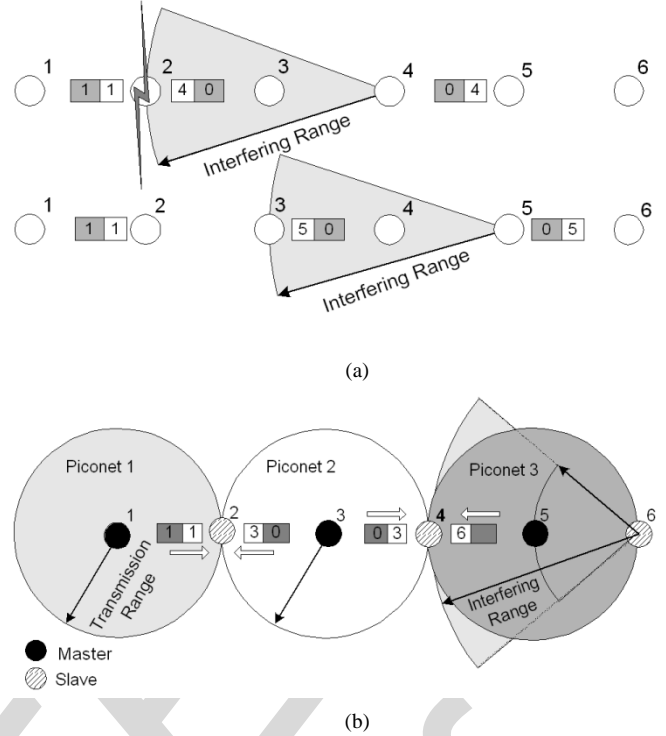


Fig. 9. Multihop packet transmission. (a) Common channel based MAC. (b) Multichannel based MAC.

- Collecting more packets in one visit period, i.e., increasing  $n$ , is favorable for the unsynchronized scatternet routes with significant throughput enhancement. However, it does not offer any help to the synchronized scatternet route.
- Larger Bluetooth packets, DH3 and DH5, raise the network utilization up to 36.16%, which is great improvement! These large packets are very useful to support asymmetric traffic such as the transmission of large file in uni-direction.

Compared with the contention based MAC like IEEE 802.11, the contention-free, multichannel MAC mechanism adopted by the scatternet route helps it achieve high network utilization and stable route throughput.

Fig. 9 gives an example to illustrate the improvement of network utilization when the string-topology network changes its MAC protocol from the common channel based scheme to the multichannel approach. It is assumed that the signal from each device can interfere its two-hop away neighbors, i.e., the interfering range of each device is twice as large as its transmission range. In both cases, device 1 keeps transmitting packets along the string route to device 6. If all devices share the same channel, as shown in Fig. 9(a), device 2, 3 and 4, when relaying the packets from device 1, keeps jamming the medium around device 2. Therefore, device 1, after sending out the no. 0 packet, cannot continue the transmission until device 4 relays the packet 0. So the channel utilization can never exceed 25%. In the scatternet route case of Fig. 9(b), since different piconets use different channels, the packets from device 3–4 cannot interfere the transmission between device 1 and device 2. So the network utilization can reach up to 50%. In practice, the overhead of the packet headers and the capacity consumed by the acknowledgment (ACK) packets, which are not counted

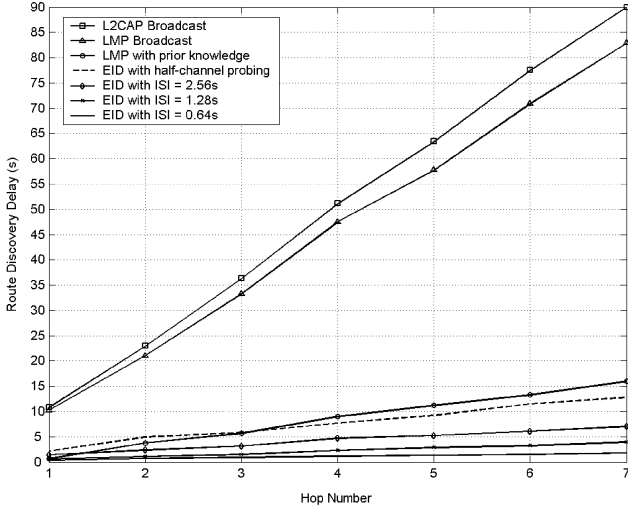


Fig. 10. Route discovery delay.

in above derivations, depress the network utilizations in both cases.

Under IEEE 802.11, each device has no idea of the possible disturbances from the two-hop away neighbors. This leads to frequent packet collisions and retransmissions, wasting the network capacity and degrading the channel utilization greatly. In [17], the performance of a seven-hop network using IEEE 802.11 MAC is investigated. The network utilization is shown to be less than 9%. The scatternet route, with the high utilization up to 36.16% (and still a 28.8 kb/s channel in return direction), is obviously superior to the multihop networks running IEEE 802.11.

As pointed out in [1], frequent, unpredictable packet collisions make transmission control protocol (TCP) running unstably over IEEE 802.11 based *ad hoc* networks. In the scatternet route, since the behaviors of all devices are well coordinated to prevent packet collisions, the TCP “slow start” algorithm can make the source device quickly detect the available network capacity and keep the whole scatternet route staying at the optimal running state. Without the detainments caused by packet collisions and retransmissions, data packets can be transferred from the source to the destination quickly and smoothly.

## V. SIMULATION RESULTS

We implement our scatternet formation protocol based on the extension and improvement of Bluehoc—a Bluetooth simulation model from IBM [19]. A Bluetooth network with 22 devices is simulated. The network topology is shown in Fig. 4. Traffic request is generated from the source—device 0 and the destination is set as device 3, 6, 9, ..., 21, respectively, to emulate the route discovery and formation over multiple hops. Data transmissions over one string topology scatternet route and two crossed scatternet routes are also simulated to investigate their network performances.

Fig. 10 shows the route discovery delays under the three broadcast schemes. There are two particular cases: one is the LMP broadcast with prior knowledge, i.e., each Bluetooth device already has the record of the address and clock information of all its neighbors (through former activities). In this case,

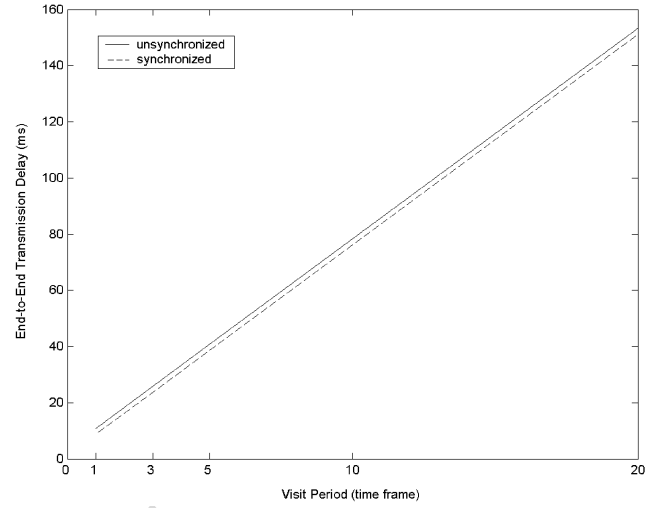


Fig. 11. End-to-end packet transmission delay.

the time consuming neighbor discovery process is omitted, so the route discovery delay is decreased hugely. However, because of the connection-oriented broadcast mechanism and possible page failures (as mentioned in Section III-B), the route discovery delay climbs quickly with the increase of the hop number. Another case is the EID with half channel probing, i.e., the EID scheme without replacing the “inquiry response frequencies” with “inquiry frequencies”. In this case, even if the neighbors continuously scan the inquiry channel, because of the possible scan failures, the broadcast delay cannot be reduced enough to be practical. When we upgrade the EID to enable the fully probing of the inquiry channel within one inquiry scan window, the route discovery delay is shortened remarkably with the decrease of the inquiry scan interval (ISI). For instance, with current Bluetooth specification of ISI = 2.56 s, the seven-hop route discovery delay is 7 s. If the inquiry scan interval is set as 0.64 s, the seven-hop route discovery delay can be as small as 1.8 seconds! There is again a tradeoff between the inquiry scan interval and power consumption. In one extreme, by shortening inquiry scan interval, we can reduce inquiry process time arbitrarily small. What this means is that slaves need to scan the channel more frequently, thereby consuming more power during inquiry process.

As we discussed, because of the “disturbances” from the devices outside the shortest route, the actual route discovery delays by using the L2CAP (around 90 s for seven hops) and LMP (around 83 s for seven hops) schemes are longer than our derivation results (83.84 s and 76.16 s, respectively). However, in the cases of the EID scheme, the actual route discovery delays (e.g., 7.03 s for ISI = 2.56 s) is shorter than what we derived (8.96 s) since the marginal devices may “wake up” earlier than the middle-route devices, thus providing express paths for the RDP.

The great improvement achieved by the EID scheme over the L2CAP and LMP schemes demonstrates that connectionless broadcast mechanisms are the proper ways to spread short messages in Bluetooth networks.

When scatternet routes are formed, data packets can be delivered along them in a very short time. Fig. 11 reveals the end-to-end packet transmission delays over a seven-hop

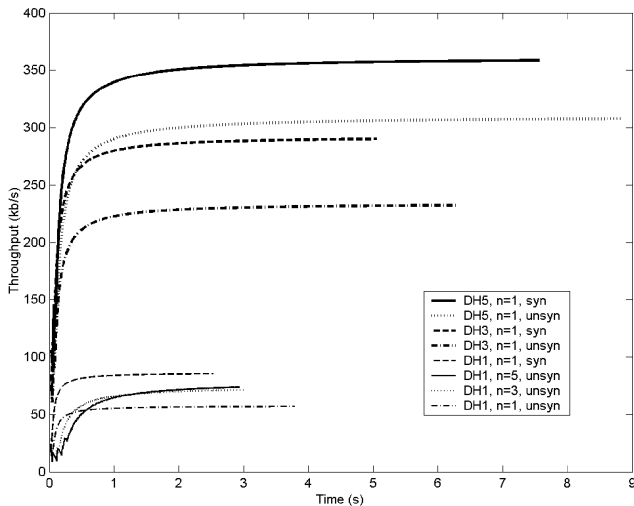


Fig. 12. Throughput of single scatternet route.

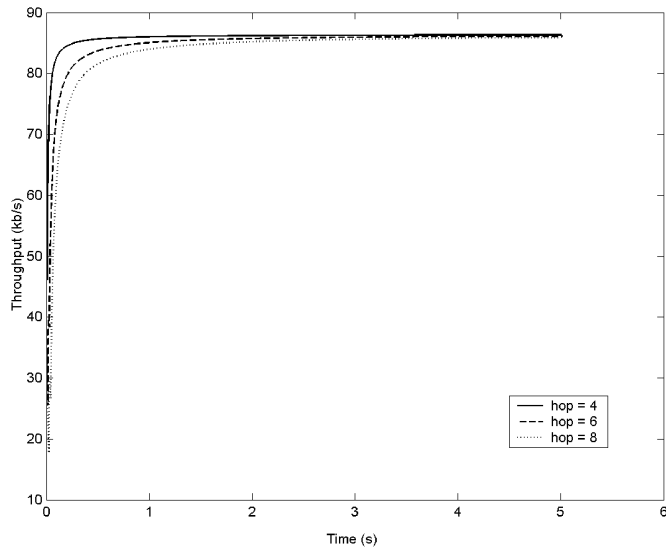


Fig. 13. Throughput versus hop.

scatternet route (DH1 packet). The increase of the visit period lengthens the packet buffer time in each relay device and eventually leads to large packet transmission delays. For unsynchronized scatternet routes, as bigger visit periods are better for channel utilization, there is a tradeoff between the transmission delay and the channel efficiency. For synchronized scatternet routes, bigger visit periods cannot improve the channel utilization any more; on the contrary, they cause bigger packet transmission delays. So the visit period should be set as small as possible. We can also see that the synchronized scatternet route achieves better performance than the unsynchronized one because of its elimination of piconet switch delays.

To probe the throughputs of the scatternet routes, we add TCP module to the source and destination devices of each scatternet route and make the sources keep sending packets to the destinations. Fig. 12 displays single route throughput in eight situations. We can see, in all situations, the scatternet-route throughput rises quickly to reach the stable level and stays in the optimal state thereafter. The final scatternet-route throughputs match well with our analysis results in Table I. Fig. 13

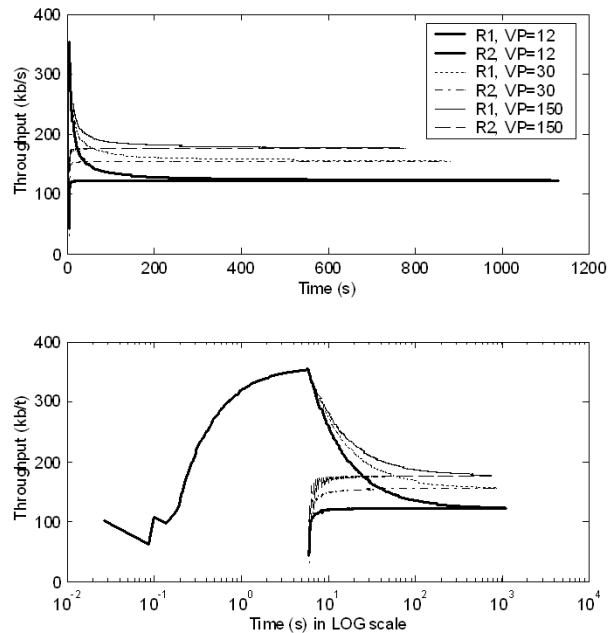


Fig. 14. Throughput of two crossed scatternet routes.

demonstrates that the throughput of the scatternet route is independent of the route length. This is quite favorable since under IEEE 802.11 MAC, the network throughput drops rapidly with the increase of the network length [17]. Fig. 14 displays the throughput transformations in two crossed scatternet routes. We use DH5 packet type and synchronize the piconets of both routes. Route 1 starts up first at the 0 second and quickly reaches its stable state. At the sixth second, route 2 begins its traffic. As both routes share the capacity of the crossover device, the throughput of route 1 is brought down and the throughput of route 2 climbs. We compare the throughputs of three cases with different route VPs. As expected, when the VP is increased, the route switch overhead is shared by more packets transmitted within one visit; the network utilization and throughput are increased as well. When the crossover device uses a very large route VP, both of these routes reach a throughput just a little less than the half of the single route throughput. Also, the TCP unfairness problem showing up in IEEE 802.11 based networks [1] completely disappear as shown in Fig. 14, where we can see the final throughputs of the two routes reach to a same value in all three cases.

## VI. CONCLUSION

In this paper, we propose a scatternet-route structure in favor of multihop data transmissions in wireless *ad hoc* networks. The scatternet route is designed to have a special master-slave alternate structure to enable the devices along the route connect together via Bluetooth links. We describe the on-demand formation procedure of the scatternet route, as well as a route-based scatternet-scheduling algorithm. Aiming at the long route discovery delay arising from the connection-oriented Bluetooth broadcast mechanism, we elaborately design an EID connectionless broadcast scheme to utilize the Bluetooth inquiry channel to broadcast the route discovery messages. Most importantly, it is shown that our EID scheme can provide an

acceptable route setup delay for multihop *ad hoc* environment. To remove the piconet switch overhead suffered by the bridge devices inside the scatternet route, we further propose to align the time slot of all piconets along each scatternet route. The synchronized scatternet route is shown to reach higher network throughput and undergo shorter data transmission delays. The performance analysis and simulation results demonstrate that our scatternet-route structure can achieve high network utilization and stable route throughput. These features make it particularly suitable to transfer large files and real time data through multihop wireless links. Continuous efforts will be made to break through the limits of Bluetooth protocols and extend the Bluetooth MAC to a more general one that is particularly suitable to mobile *ad hoc* networks.<sup>2</sup>

## REFERENCES

- [1] S. Xu and T. Saadawi, "Does the IEEE 802.11 MAC protocol work well in multihop wireless *ad hoc* networks?," *IEEE Commun. Mag.*, vol. 39, pp. 130–137, June 2001.
- [2] Specification of the Bluetooth System [Online]. Available: <http://www.bluetooth.com/developer/specification/specification.asp>
- [3] T. Salonidis, P. Bhagwat, L. Tassiulas, and R. LaMaire, "Distributed topology construction of Bluetooth personal area networks," in *Proc. IEEE INFOCOM'2001*, Anchorage, AK, Apr. 2001, pp. 1577–1586.
- [4] L. Ramachandran, M. Kapoor, A. Sarkar, and A. Aggarwal, "Clustering algorithms for wireless *ad hoc* networks," in *Proc. 4th Int. Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, Boston, MA, Aug. 2000, pp. 54–63.
- [5] C. Law, A. K. Mehta, and K. Siu, "Performance of a new Bluetooth scatternet formation protocol," presented at the ACM MobiHoc'2001, Long Beach, CA, Oct. 2001.
- [6] G. V. Zaruba, S. Basagni, and I. Chlamtac, "Bluetrees—Scatternet formation to enable Bluetooth-based *ad hoc* networks," in *Proc. IEEE ICC'2001*, Amsterdam, The Netherlands, July 2001, pp. 273–277.
- [7] G. Tan, A. Miu, J. Guttig, and H. Balakrishnan, "Forming Scatternets From Bluetooth Personal Area Networks," MIT, LOCATION, MIT Tech. Rep., MIT-LCS-TR-826, 2001.
- [8] B. Raman, P. Bhagwat, and S. Seshan, "Arguments for cross-layer optimizations in Bluetooth scatternets," in *Proc. Symp. Applications and the Internet*, 2001, pp. 176–184.
- [9] P. Johansson, M. Kazantzidis, R. Kapoor, and M. Gerla, "Bluetooth: An enabler for personal area networking," *IEEE Network*, vol. 15, pp. 28–37, Sept./Oct. 2001.
- [10] N. Johansson, U. Korner, and L. Tassiulas, "A distributed scheduling algorithm for a Bluetooth scatternet," presented at the 17th Int. Teletraffic Congress, Salvador da Bahia, Brazil, Sept. 2001.
- [11] A. Racz, G. Miklos, F. Kubinsky, and A. Valko, "A pseudo random coordinated scheduling algorithm for Bluetooth scatternets," in *Proc. ACM MobiHoc'2001*, Long Beach, CA, Oct. 2001, pp. 193–203.
- [12] N. Johansson, F. Alriksson, and U. Jonsson, "JUMP mode—A dynamic window-based scheduling framework for Bluetooth scatternets," in *Proc. ACM MobiHoc'2001*, Long Beach, CA, Oct. 2001, pp. 204–211.
- [13] S. Baatz, M. Frank, C. Kuhl, P. Martini, and C. Scholz, "Adaptive scatternet support for Bluetooth using sniff mode," in *Proc. 26th Annual Conf. Local Computer Networks*, Tampa, FL, Nov. 2001, pp. 112–120.
- [14] B. Zhen and Y. Kim, "Location management support of IP over Bluetooth," presented at the 3Gwireless'2002, San Francisco, CA, May 2002.
- [15] M. Kalia, S. Garg, and R. Shorey, "Scatternet structure and inter-Piconet communication in the Bluetooth system," presented at the IEEE National Conf. Communications 2000, New Delhi, India, 2000.
- [16] S. Souissi and E. F. Mehofer, "Performance evaluation of a Bluetooth network in the presence of adjacent and co-channel interference," *IEEE ETS Broadband, Wireless Internet Access*, pp. XXX–XXX, Apr. 2000.
- [17] S. Xu and T. Saadawi, "Revealing and solving the TCP instability problem in 802.11 based multihop mobile *ad hoc* networks," in *Proc. IEEE VTC'2001*, vol. 1, 2001, pp. 257–261.
- [18] E. M. Royer and C. K. Toh, "A review of current routing protocols for *ad hoc* mobile wireless networks," *IEEE Personal Commun.*, vol. 6, no. 2, pp. 46–55, Apr. 1999.
- [19] BlueHoc: Bluetooth Performance Evaluation Tool [Online]. Available: <http://oss.software.ibm.com/bluehoc/>
- [20] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proc. ACM SIGCOMM'94*, London, U.K., Sept. 1994, pp. 234–244.
- [21] C. E. Perkins, E. M. Belding-Royer, and S. Das, "Ad hoc on demand distance vector (AODV) routing," IETF, LOCATION, IETF Internet Draft, 2002.



**Yong Liu** (S'02) received the B.S. and M.E. degrees from the Department of Automation, University of Science and Technology of China, Hefei, Anhui, China, in 1996 and 1999, respectively, the M.S.E.E. degree from Polytechnic University, New York, NY, in 2001 and is currently working towards the Ph.D. degree in electrical engineering at City University of New York, City College, New York.

His research interests include: mobile and wireless communication and networks, personal communication technologies, and sensor networks.



**Myung Jong Lee** (S'87–M'90–SM'98) received the B.S. degree from Seoul National University, Seoul, Korea and the M.S. and Ph.D. degrees in electrical engineering from Columbia University, New York, NY, in 1986 and 1990, respectively.

He joined the Department of Electrical Engineering, City College and Graduate Center, City University of New York (CUNY), New York, in 1990, where he is currently an Associate Professor in the Department of Electrical Engineering. His recent researches focus on various aspects of wireless *ad hoc* networks, sensor networks and personal area networks including Bluetooth. He has published over 50 refereed journals and conference papers.

Dr. Lee received CUNY's Excellence Performance Award in 1999. He served many IEEE conferences as Program Committee Member and Session Chair.



**Tarek N. Saadawi** (S'78–M'80–SM'84) received the B.Sc. and M.Sc. degree from Cairo University, Cairo, Egypt and the Ph.D. degree from the University of Maryland, College Park (all in electrical engineering).

Since 1980, he has been with the Electrical Engineering Department, The City University of New York, City College, New York. His current research interests are telecommunications network, high-speed networks, multimedia networks, *ad hoc* networks, and packet radio networks. He has

published extensively in the area of telecommunications networks. He is a coauthor of *Fundamentals of Telecommunication Networks* (New York: Wiley, 1994). He is also the lead author of Egypt Telecommunications Infrastructure Master Plan covering the fiber network, IP/ATM, DSL, and the wireless local loop.

Dr. Saadawi is a Former Chairman of IEEE Computer Society of New York City (1986–87). He has received IEEE Region 1 Award in 1987 and the Nippon Telegraph and Telephone (NTT) of America for research on Broadband Telecommunication Networks.

<sup>2</sup>The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government.