Computer Networks 52 (2008) 2764–2778

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

TMCP: Two-layer multicast communication protocol for Bluetooth radio networks

Chao-Tsun Chang [a], Chih-Yung Chang [b,*], Sheng-Wen Chang [b]

[a] Department of Information Management, Hsiuping Institute of Technology, Dali, Taichung, Taiwan, ROC
[b] Department of Computer Science and Information Engineering, Tamkang University, Tamsui, Taipei, Taiwan, ROC

ARTICLE INFO

Article history:
Received 5 October 2006
Received in revised form 9 May 2008
Accepted 16 June 2008
Available online 21 June 2008

Responsible Editor: Dr. E. Chong

Keywords:
Bluetooth
Multicast
Procotol
Role switching
Routing

ABSTRACT

Bluetooth is a low power, low cost, and short-range wireless technology developed for Personal Area Networks (PANs). A Bluetooth multicast group is a set of Bluetooth devices that desire for periodically receiving the multicast messages from the same source. For reducing the propagation delay and saving the bandwidth and energy consumptions, a multicast tree which connects all multicast members serves for the delivery of multicast messages. However, a given connected scatternet topology may not be appropriate for constructing an efficient multicast tree and hence causes power consumption and end-to-end delay. This paper develops a two-layer multicast communication protocol (TMCP) using role switching techniques for constructing an efficient multicast tree. The proposed TMCP collects as many as possible the members into the same piconet, reduces the length of multicast paths and assigns each member with a proper role. The constructed multicast tree has several features including as few as possible the non-member devices, the smallest tree level and the minimal propagation delay. Experiment results show that the TMCP offers efficient multicast service with low power consumption and small delay.

1. Introduction

Bluetooth [1] is a low cost, low power, and short-range wireless communication technology. In a Bluetooth network, piconet is a basic networking element which consists of a master device and ups to seven slave devices. A scatternet is a wireless network comprising several connected piconets.

To avoid the co-channel interference, the Radio Frequency (RF) module hops over 79 channels in a speed of 1600 times per second. The design of short packet and fast hopping mechanisms increases the communication reliability of two Bluetooth devices. According to master's 48-bit Bluetooth address, a unique hopping sequence could be derived for each piconet. In a piconet, all devices synchronize with the master's clock and apply the master's hopping sequence as their channel hopping policy. Different piconets adopt different hopping sequences and therefore allow many piconets with parallel communications using different channels.

When a master intends to connect with a slave, the master initially stays in the inquiry state and obtains the information about slave's Bluetooth address and clock. Then the master changes to a page state and switches to the channel derived from slave's Bluetooth address which is obtained in the inquiry state. In the page state, the master sends slave a frequency hop synchronization (FHS) packet which contains a 3-bit active member address (or AM_ADDR in short) and its own Bluetooth address and clock information. The master's Bluetooth address and clock information help slave to derive the hopping sequence whereas the AM_ADDR is used for slave's identification. Then the communication link between master and slave is constructed. By using the same hopping sequence,

---

* Corresponding author. Tel.: +886 2 26215656x2749; fax: +886 2 26209749.

E-mail addresses: cctas@email.hit.edu.tw (C.-T. Chang), cychang@cs.tku.edu.tw, cychang@mail.tku.edu.tw (C.-Y. Chang), swchang@wireless.cs.tku.edu.tw (S.-W. Chang).

1389-1286/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2008.06.008

the master and slaves of a piconet may communicate at the same channel in a polling-based policy.

A bridge is a Bluetooth device that participates in more than one piconet. The bridge switches among the participating piconets and relays messages from one piconet to another. The bridge stays in active mode in exactly one piconet and stays in power saving mode (sniff or hold modes) in the others participating piconets at the same time. Since a bridge can participate more than one piconet and only play the master role in exactly one piconet, the role of a bridge could be either slave/slave (or S/S in short) or master/slave (or M/S in short). An S/S bridge alternatively plays the slave role in each participating piconets whereas an M/S bridge alternatively plays a master role in exactly one piconet and plays the slave role in the others. Fig. 1 displays a scatternet where an S/S bridge connects piconets P_a and P_b and an M/S bridge connects piconets P_b and P_c. A bridge can deliver messages from one piconet to another, allowing message transmission across piconets.

In recent year, there are more and more devices embedded with the Bluetooth chips for providing short-range wireless communication among them. These Bluetooth devices in a specific area (for example, a bus or an airport lobby) will construct a scatternet. A multicast service center might provide a group of Bluetooth devices with multicast services such as video program, audio services, information or short-message sharing, or other multimedia services. In these scenarios, an efficient multicast protocol is required to construct an efficient multicast tree for reducing the data traffic load and saving the energy consumptions.

In a Bluetooth scatternet, the multicast service provides data transmission from source device to several predefined member devices which may belong to different piconets. A multicast protocol is required to construct an efficient multicast tree from source to all members. In previous research, a routing vector method (RVM) routing protocol [2] has been developed for establishing a route from single source to single destination in a Bluetooth scatternet. However, if the multicast tree is established by repeatedly applying the RVM routing protocol to establish a path from the source to each member, the tree will be inefficient due to creating a huge amount of control packets and redundant traffic. Moreover, the constructed tree contains too many non-member nodes since the shared path is not considered.

In literature, multicast service has been widely discussed in wireless ad hoc networks. Based on those discussions, an efficient multicast tree has at least three features: Firstly, the multicast tree contains as few as possible the non-members nodes, reducing the propagation delay and the power and bandwidth consumptions. Secondly, the height of a multicast tree is minimized for reducing the propagation delay from source to all members. Lastly, the multicast tree contains as many as possible the shared links for reducing the bandwidth consumption. A number of multicast protocols [3–9] have been proposed for wireless ad hoc networks, based on 802.11 radio system. Previous studies [3–5] proposed shared tree scheme to reduce the bandwidth consumption. However, the radio characteristics of Bluetooth network and 802.11 WLAN are quite different. Each Bluetooth device in a piconet plays a role of master, slave or bridge. The role constraint makes two slaves cannot directly communicate with each other, even though they are within the radio transmission range. Furthermore, two devices that belong to different piconets cannot directly communicate with each other, even though they have no constraint in role playing. This is because that the different piconets adopt difference hopping sequences and use different channels at the same time. Therefore, applying the existing multicast protocols developed for 802.11 ad hoc networks to Bluetooth scatternet cannot create an efficient multicast tree. Even the recent work [21] proposes a multicast approach for quickly sending a large file in piconets by applying the strategies of small and parallel piconets, the proposed protocol which constructs the scatternet for supporting the multicast service increases the impact to the other services and is not available to provide the multicast service for a given scatternet.

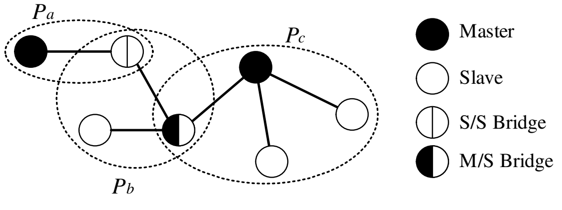


Fig. 1. A scatternet consists of three piconets. An S/S bridge connects piconets P_a and P_b and an M/S bridge connects piconets P_b and P_c.

This paper proposes a Two-layer Multicast Communication Protocol (TMCP) that applies the role switching techniques to construct an efficient multicast tree. TMCP is a single-source multicast protocol that takes the Bluetooth characteristics into account. We observed that collecting more members into a piconet can take more advantages from radio broadcasting nature on executing the multicasting. Furthermore, it can also reduce the tree height and the number of non-member nodes participating in the constructed tree. The proposed protocol maintains the given scatternet as the first layer and constructs an overlay multicast tree on the second layer so that the multicast service can be performed on the second layer and has the least impact on the first layer's service. In the second layer, TMCP reconstructs the tree topology by trying to collect members into the piconets. During the linkages establishing, TMCP uses the role switching operations to collect members as many as possible in a piconet and therefore the reconstructed tree topology is good for applying the broadcasting operations. As a result, the constructed tree keeps the three abovementioned features of an efficient multicast tree. The multicast tree constructed on the second layer has least impact on the communication activities originally performed on the first layer and therefore multiple communication activities can work well in the scatternet. Simulation study shows that TMCP is cost effective when the transmission load associated with multicasting is heavy and the number of members is small.

The rest of this paper is organized as follows: Section 2 introduces the problem formation and preliminary. Section 3 firstly presents the basic concepts of multicast tree construction then details the proposed multicast protocol. Section 4 depicts the experimental results and studies the performance improvements. Conclusions are finally described in Section 5.

2. Problem formation and preliminary

This section introduces first the considered network environment and multicast problem. Then the difference between 802.11 and Bluetooth radio networks and the basic role switching operations adopted in the proposed protocol are presented.

2.1. Network environment and problem formation

A Bluetooth multicast group $G = \{s, m_1, m_2, \ldots, m_k\}$ is a set of Bluetooth devices that comprises of a source s and k multicast members. The source device provides multicast service and intends to periodically broadcast the multicast messages to all members via a multicast tree. Given a connected scatternet, we investigate how to establish an efficient multicast tree over the scatternet so that the multicast messages can be delivered from source to all members along the constructed multicast tree.

2.2. Challenges of developing an efficient multicast protocol

This sub-section addresses the challenges in developing an efficient multicast protocol for Bluetooth radio networks.

2.2.1. The hopping sequence constrains the broadcasting benefit

There were comprehensive studies of multicast protocols proposed in previous researches [3–9]. Most of existing multicast protocols are developed based on 802.11-based ad hoc network environment. However, there are many differences between Bluetooth and 802.11 radio networks. Applying existing multicast protocols to Bluetooth networks cannot construct an efficient multicast tree. Take Fig. 2(a) and (b) as examples. Let devices *a* and *b* be within the radio transmission range. In 802.11 ad hoc networks, devices *a* and *b* can directly communicate with each other. However, they cannot directly communicate with each other in Bluetooth networks due to the role and hopping sequence constraints.

As shown in Fig. 2(a), devices *a* and *b* play the slave role in the piconet P_c. As claimed in the Bluetooth specification [1], slave to slave communication must be achieved by packet forwarding from source slave to master, then from master to destination slave. That is, in a piconet, a slave cannot directly communicate with other slave. The role constraint may raise the inefficient communication in a multicast service. Fig. 2(b) depicts another case that devices *a* and *b* cannot directly communicate with each other due to the constraint of hopping sequence. Since devices *a* and *b* belong to different piconets P_c and P_d, respectively, they apply different hopping sequence and therefore stay at different channels at the same time.

Since different piconets adopt different hopping sequence, the tree construction request message transmitted by a master, say m_1, using broadcasting operation can only be received by those slaves belonging to the same piconet with master m_1 and cannot be received by all member nodes within the master's communication range. Some slave nodes may stay within the communication range of m_1 but they and m_1 belong to different piconets and therefore stay in different channels, resulting that they are unable to receive the request message for tree construction. This radio characteristic of Bluetooth is different quite a bit from that of 802.11 and therefore the multicast protocols previously developed for the ad hoc networks cannot create an efficient multicast tree for a given Bluetooth scatternet. How to use broadcasting operation to deliver the request message of tree construction to all nodes in the communication range so that they have opportunities to connect with the best parent in a given scatternet will be one of the challenges in constructing an efficient multicast tree.

2.2.2. Difficulty in constructing a tree in a distributed manner

Given a set of multicast members, it is difficult and time consuming to construct a tree that contains all members. Consider the tree construction in the way that all member devices randomly enter inquiry or inquiry scan state, aiming to quickly connect with each other. One challenge in this way is that the inquiry and inquiry scan operations are the most time consumption operations [21,22] and they takes 10.24 s to establish a link in the worst case [1]. Another difficulty is to construct a loop-free tree structure without global information. Nodes playing the leaf roles in the tree are required to establish exactly single link with the other device while nodes playing the intermediate nodes are required to establish at least two links. All nodes construct the tree in a distributed manner. However, it is difficult to guarantee the constructed tree is loop-free since there is no central point to make decision the number

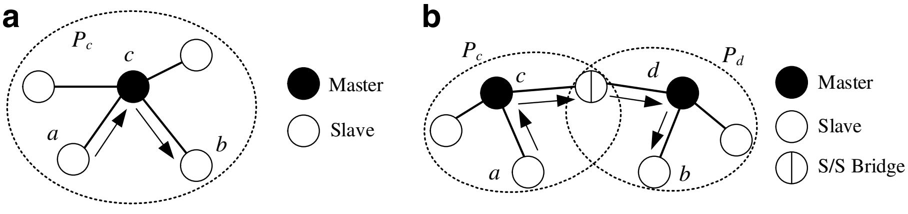


Fig. 2. The difference between 802.11 and Bluetooth radio networks. (a) In Bluetooth piconet, even though device *a* is located in device *b*'s signal coverage range, they cannot directly communicate with each other due to the role constraints. (b) In Bluetooth scatternet, even though device *a* is located in device *b*'s signal coverage range, they cannot directly communicate with each other due to their difference in hopping sequences.

of links and the destination to which each node should establish.

2.2.3. Existing routing protocols developed for Bluetooth scatternet cannot help to construct an efficient multicast tree

In the literature, a class of scatternet formation protocols [12–20] was proposed for constructing a connected scatternet. Given a connected scatternet, a routing protocol, named RVM [2], was proposed for creating a route from single source to single destination. The path construction of RVM mainly applies flooding-based mechanism which discovers a route by flooding a route search packet over the scatternet. In RVM, the source node which intends to establish a communication path should first send a request packet to its master, and the master will then broadcast the packet to all its slaves. Upon receiving the request packet, those slaves that interconnect more than one master will relay the packet to the neighboring masters until the master of the destination device is found. By considering all possible paths linking the source and destination, the source host can ascertain the shortest communication path. By repeatedly applying the RVM protocol to construct a route from the same source to each member can obtain a multicast tree. However, the constructed tree is not efficient since it may introduce too many non-member nodes participating the tree.

2.2.4. Difficulties in collecting member nodes to the same piconet

A multicast group is usually predefined and a multicast message usually contains a multicast ID to indicate that the message intend to be sent to which member group. In providing the multicast service, the source node or forwarding node usually broadcast the multicast message to their neighbors. On receiving the multicast message, each neighboring node checks the multicast ID and picks up the message if it belongs to the multicast group. Then the forwarding nodes in the tree will broadcast the received packet to their neighbors while the non-forwarding nodes drop the packet. However, in a Bluetooth scatternet, different piconets adopt different channel hopping sequence and therefore stay at different channels at the same time. When a master node broadcasts a multicast message, only its slave can receive the message. To maximize the benefit of a broadcasting operation, multicast members should be collected in the same piconet as many as possible.

However, some challenges are encountered in collecting group members into the same piconet. Since a Bluetooth device randomly searches for and connects with other devices by using inquiry/inquiry scan and page/page scan operations in the scatternet construction phase, an uncontrolled scatternet topology will be obtained and hence the group members may fall in different piconets. Collecting the member devices from different piconets will face two difficulties. First, the member device should change its linkage from the old piconet to the new one. The link reconnection by executing the inquiry/inquiry scan and page/page scan is time-consuming. How to utilize the existing linkages in the given scatternet to speedup the link reconnection process will be the most concerned issue to construct an efficient multicast tree. A member device that plays a slave role requires to know the new master information including the 48-bit Bluetooth and clock offset so that the new link can be rapidly established with the master without executing the time consuming inquiry and inquiry scan operations. For a member device that plays a master role, how the master device serves the original slaves after it reconnects to a new master is another challenge. Secondly, each piconet can only have seven slaves at most. How to reconnect the member devices as many as possible into the same piconet in a distributed manner will be another critical issue in constructing an efficient multicast tree.

2.2.5. Improper role-playing constrains the efficiency of message delivery

The multicast tree may be consisted of many piconets since the number of slaves in a piconet cannot be greater than seven. A bridge is response to forward the multicast messages from one piconet to another. A S/S bridge plays the slave role in all participating piconets, whereas an M/S bridge plays the master role in one piconet and the slave role in other participating piconets [10]. Consider a S/S bridge device that intends to broadcast the received multicast packet in its piconet. Unfortunately, the broadcasting operation can only be performed by a master device since only the master device connects to all slave devices in a piconet. Furthermore, the S/S bridge will increase the guard time which is a required time interval switching from one piconet to another and synchronizing with the switched piconet. Thus, improper role-playing would constrain the execution of broadcasting operation and increases propagation delay and power consumption. Since the device roles have been determined in the given scatternet, how to reassign the proper role for each node in the multicast tree to improve the efficiency of message delivery is another major challenge of developing the multicasting protocol.

2.3. Role switch operations

In Bluetooth networks, the efficiency of a multicast tree highly depends on the role of device and the tree topology. However, the given Bluetooth scatternet is formed by each device randomly searching and connecting with other devices to construct a link, using the inquiry/inquiry scan and the page/page scan operations. Thus the given scatternet topology is uncontrolled. Since Bluetooth devices establish their links at random, the multicast members may belong to different piconets and play improper roles for multicast service. For a given scatternet and a set of multicast members, this paper aims at constructing an efficient single-source multicast tree containing all multicast members in the second layer other than the given scatternet topology. The developed protocol dynamically applies the role switching operations to reduce the height of multicast tree as well as the numbers of piconets and bridges. In this sub-section, the fundamental role switching procedures which will be used in constructing the multicast tree are introduced.

Role switching enables two devices to exchange their roles very rapidly, rather than establishing a new link by executing the time-consuming inquiry and inquiry scanning processes. In the process of constructing a multicast tree, three basic role switching operations are used as basic functions. These functions are used in establishing a new link in the second layer for reassigning the role of each tree node to a better one. In addition, the role switching operations are also used in collecting geographically neighboring members into the same piconet. Three role switching functions [11,12,25] are introduced.

2.3.1. Piconet combining

As depicted in Fig. 3(a), devices *b* and *a* play the master and slave roles in piconet P_2, respectively. In case that device *b* intends to play a slave role, it sends a role switching request to exchange its role with device *a*. The role switching operation combines two piconets P_1 and P_2 into a single piconet P_1, as shown in Fig. 3(b). The role switching operation also eliminates the bridge role of device *a*. This type of role switching reduces the number of bridges and the number of piconets and is referred as *piconet combining.*

2.3.2. Piconet splitting

A role switching operation also can split a piconet into two piconets. As depicted in Fig. 4(a), in piconet P_1, slave device *b* intends to create a new piconet P_2 and plays the master role in the new piconet P_2. Device *b* accordingly initiates a role switching request to device *a*. As depicted in Fig. 4(b), device *b* creates a new piconet P_2 and plays a master role in P_2. Device *a* then alternatively participates in two piconets P_1 and P_2 and plays master and slave roles, respectively. This kind of role switching increases the number of piconets and bridge devices and is referred as *piconet splitting.*

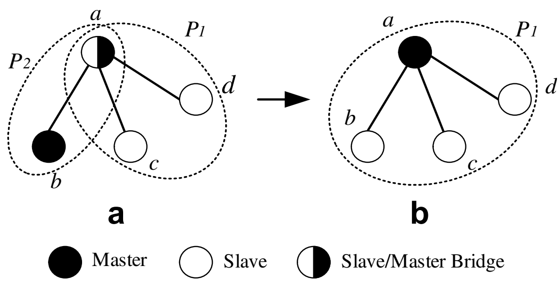


Fig. 3. (a) Topology before executing *Piconet Combining* operation. (b) Topology after executing *Piconet combining* switching operation.

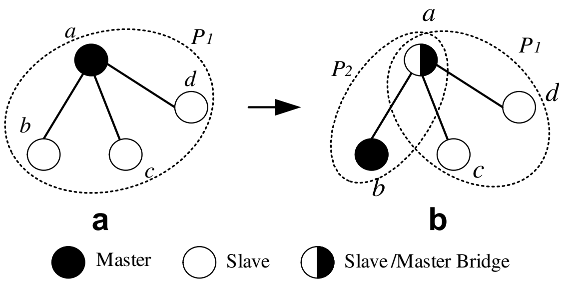


Fig. 4. (a) Topology before executing *Piconet splitting* operation. (b) Topology after executing *Piconet splitting* operation.

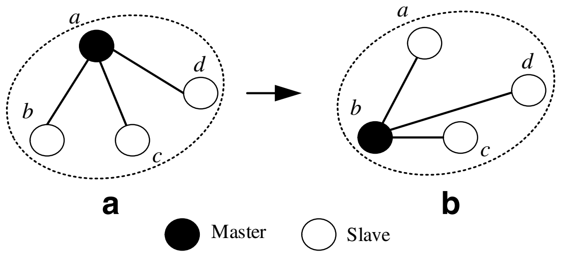


Fig. 5. (a) Topology before executing *Piconet TakeOver* operation. (b) Topology after executing *Piconet Take Over* operation.

2.3.3. Piconet takeover

Role switching operation also allows a slave to take over the resources from its master. For example, as shown in Fig. 5(a), device *b* initiates a role switching request: Piconet takeover. On receiving the request, device *a* asks the other slaves including *c* and *d* to enter the page scan state and transfer their BD_ADDR and clock information to device *b*. Device *b* thus enters the page state and rapidly connects with devices *c* and *d*. As shown in Fig. 5(b), device *b* takes over all slaves from device *a*, playing a master role in the piconet. This type of role switching operation enables a slave to take over all slaves belonging to its master.

3. The multicast protocol for constructing an efficient multicast tree

This section presents the proposed multicast protocol for constructing an efficient multicast tree over the given scatternet. Firstly, the basic concepts and examples related to the proposed multicast protocol are described. Then, the details of the proposed multicast protocol are presented.

3.1. Basic concepts of multicast tree construction

The proposed protocol presents a novel two-layer topology for constructing an efficient multicast tree. The given scatternet is treated as the first layer and the temporal multicast tree will be constructed as the second layer of network topology over the scatternet. The following introduces basic concepts of the proposed multicast tree construction protocol.

To construct an efficient multicast tree over a Bluetooth scatternet, the characteristics of role and hopping sequence in Bluetooth network should be taken into account in the design of multicast protocol. Fig. 6 depicts the ideal multicast tree in a Bluetooth scatternet. The root of the multicast tree is the source. Each piconet contains exactly seven member slaves so that the tree height is minimized. All internal nodes in the tree participate two piconets, one is the upstream piconet and the other is the downstream piconet. These internal nodes play the slave/master role, using slave role to receive multicast message in the upstream piconet and then broadcasting the received multicast message to its children in the downstream piconet.

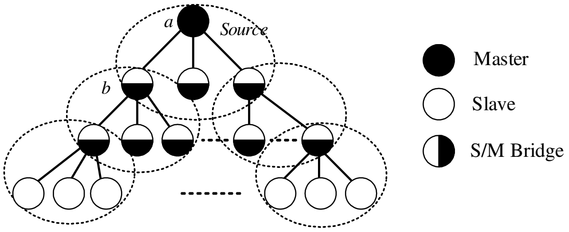


Fig. 6. An ideal multicast tree in Bluetooth scatternet.

In particular, each piconet consists of multicast members where the parent plays a master role and all children play the slave role so that parent can use broadcasting operation to deliver the multicast message to its children. For instance, source a broadcasts the multicast message to its children. On receiving the message, each child, say *b*, enters sniff mode in the upstream piconet and actively switches to the downstream piconet. Then device *b* plays the master role in the downstream piconet and uses broadcasting operation to forward the multicast message to all children in the downstream piconet. After that, master *b* enters sniff mode in the downstream piconet and then switches back to the upstream piconet, playing slave role and waiting for receiving the next multicast message from master *a*. Since each piconet of the multicast tree consists of multicast members, the broadcasting operation executed by master can get maximal advantage by delivering the multicast message to a maximal number of multicast members. In addition, since there is no non-member node in the tree, the tree height can be minimized, reducing the end-to-end propagation delay. Furthermore, the transmission delay due to bridge switching from one piconet to another can be minimized. As the bridge receives multicast message in the upstream piconet, it can actively switch to the downstream piconet, play the master role, and immediately use the broadcasting operation to rapidly forward the received message to all members in the downstream piconet.

Figs. 7 and 8 compare the difference of multicast trees constructed by the flooding-based and the proposed mechanisms. Given a connected scatternet shown in Fig. 7(a), the flooding mechanism which has been widely used in 802.11-based ad hoc networks broadcasts a request from source node for constructing a multicast tree. On receiving the request message, each bridge device switches to another piconet and forwards the message to the other devices in that piconet. Each master broadcasts the request message as it received the message from one bridge and discards the messages it has broadcasted previously. As the multicast member receives the request message, it replies with a reply packet to source by traveling the path the request message passing through. The multicast tree constructed by applying the flooding-based mechanism is shown in Fig. 7(b). The given scatternet and the constructed multicast tree are treated as the first and the second layers of network topology, respectively. The

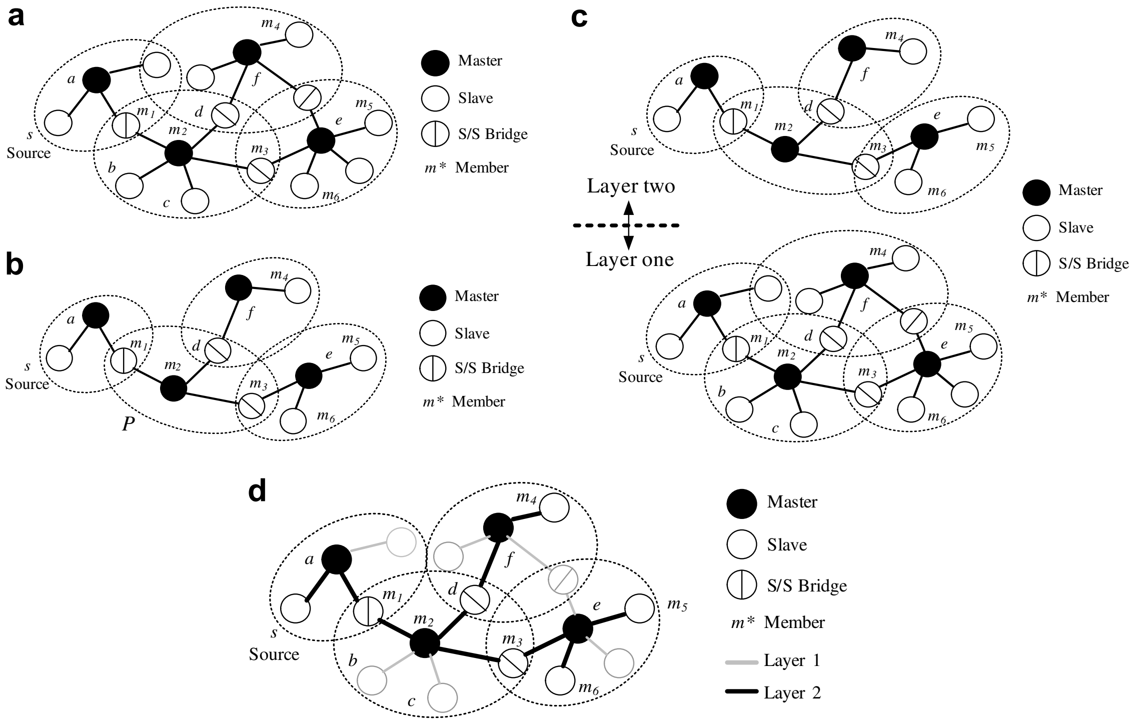


Fig. 7. An example of multicast tree construction by applying the flooding-based mechanism. (a) An example of connected scatternet for constructing an efficient multicast tree. (b) An inefficient multicast tree constructed by applying the flooding-based mechanism on the scatternet shown in (a). (c) The relationship between the constructed multicast tree in (b) and original scatternet in (a). (d) The view of combination of the constructed multicast tree (second layer) and the original scatternet (first layer).

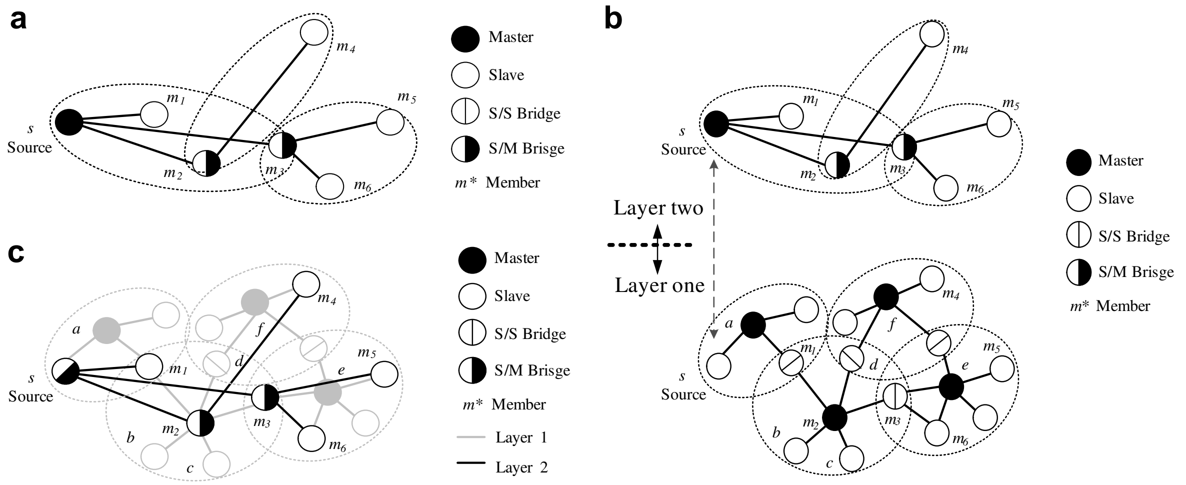


Fig. 8. An example of efficient multicast tree constructed by applying the TMCP mechanism. (a) An efficient multicast tree constructed by applying the role switching operations on the scatternet shown in Fig. 7(b). (b) The relationship between the constructed multicast tree and the original scatternet shown in Fig. 8(a). (c) The view of combination of the constructed multicast tree and original scatternet in Fig. 8(a).

relationship between the two layers is shown in Fig. 7(c). The view of the combination of the constructed multicast tree and the original scatternet is presented in Fig. 7(d) where the constructed multicast tree has been highlighted.

The multicast tree constructed by applying flooding-based mechanism exhibits several disadvantages. First, the constructed tree includes a lot of non-member devices. As shown in Fig. 7(b), there are 11 devices participated the multicast tree where only 7 devices are multicast members. Four non-member devices in the multicast tree not only consume their power and bandwidth but also increase the tree height, increasing the end-to-end delay of multicast service. Furthermore, as shown in Fig. 7(b), since there are three non-member devices in piconet P, when device m_2 broadcasts the multicast message in its piconet, the non-member devices consume power for idle listening the message. Secondly, the efficiency of the constructed multicast tree highly relies on the original topology of the given scatternet. As shown in Fig. 7(b), no new link is established for the multicast tree. That is, the constructed multicast tree is a sub-graph of the given scatternet. In case that the multicast members are distributed in several different piconets, the multicast tree constructed by applying the flooding-based mechanism is inefficient since few members obtain multicast message from a broadcasting operation and the large tree height increases the propagation delay.

To improve the tree topology as showed in Fig. 7(b), TMCP applies the role switching operations to rapidly change the improper role of the multicast members and reduce the tree height. When each multicast member receives the request message, it tries to establish a new connection from itself to the previous member along the path the request message passed through. Therefore, the efficient multicast tree can be reconstructed on the second layer as shown in Fig. 8(a). Fig. 8(b) shows the original scatternet and the constructed multicast tree in the first and the second layers. The construction of multicast tree in the second layer cause some nodes play additionally roles in the second layer. For example, source node s plays a slave node in the original scatternet but plays a new master role in the second layer. Hence, node s becomes a bridge node in the resultant scatternet as shown in Fig. 8(c). The figure depicts the resultant scatternet where the constructed multicast tree has been highlighted.

In comparing Figs. 7(b) and 8(a), the constructed multicast tree by applying TMCP significantly reduces 67% average length from source to each member and thus largely reduces the transmission delays from source to destinations. Furthermore, in the multicast tree shown in Fig. 8(a), the number of forwarding nodes is reduced from six to two, saving the bandwidth and energy consumptions. In addition, the role of the bridge device in the multicast tree shown in Fig. 8(a) is changed from S/S to S/M. Since the master controls the scheduling in a piconet, on receiving a multicast message, the S/M bridge can immediately switch to the downlink piconet, change its role from slave to master, and actively broadcasts the multicast message to all its slave members. The details of the proposed multicast protocol are described in the next section.

3.2. The TMCP protocol

The TMCP mainly consists of two phases: phase I constructs a multicast tree by flooding the route search packets over the given scatternet and assigns a proper role to each tree node. Phase II uses role switching operations to reorganize the topology of the constructed multicast tree and tries to collect member nodes into the same piconet. The following describes the details of phases I and II.

3.2.1. Multicast tree construction phase

Initially, the source device creates and broadcasts the route search packet which contains the information including its BD_ADDR and the clock offset to help rapidly establish tree links from itself to several child nodes. Here we assume the source node is s and the BD_ADDR and clock_offset are respectively denoted by s.BD_ADDR and s.clock_offset. Then, the source device enters page scan state and waits for establishing new downstream links with members that receive the route search packet in the downstream piconets.

On receiving the route search packet, each non-member device and member device executes different operations. If the packet has been received before, it will be discarded. The non-member device simply forwards the search packet to its neighboring nodes while a multicast member should deal with the up-link and down-link connections as described in below. Firstly, the multicast member, say m, extracts s.BD_ADDR and s.clock_offset from the packet sent by source s. Then device m replaces the packet with its own information, m.BD_ADDR and m.clock_offset, and then rebroadcasts the packet to its neighbors. After that, the member device stay in page scan and page states in turn. In the page state, device m tries to use the received s.BD_ADDR and s.clock_offset information to construct a new upstream link with the source node s which has stayed in page scan state. Alternatively, in the page scan state, the member device m waits for constructing a downstream link with those members who received the route search packet sent by itself.

All the other members that receive the route search packet should execute the same operations as device m does. A specific duration is set for each member device to construct the upstream link and downstream links. When the duration is time out, phase I is completed. Consequently, the multicast tree is constructed. However, the constraints including distance between members and the maximal number of slaves (equals to seven) in a piconet may cause the failure in establishing an upstream link. The path which is discovered by search packet between members will be used to delivery the multicast message instead of the failure link. If an upstream link of member h cannot be established, the member h will send a reply packet back to the upstream member m and the path traversed from h to m would be recorded in the reply packet. On receiving the reply packet, the member m includes the path from m to h as part of multicast tree for delivering multicast messages from itself to member h.

Take Fig. 7(a) as an example. Assume the multicast group is $\{s, m_1, m_2, m_3, m_4, m_5, m_6\}$ where device s is the source device. In this phase, source device s creates the search packet containing its BD_ADDR and clock offset and forwards it to device a. Then, source device s enters page scan state and waits for other members in downstream piconets to establish a downstream link. On receiving the search packet, device a simply rebroadcasts the packet to all slaves in its piconet since it is not a multicast member. When the device m_1 receives the search packet, it records s.BD_ADDR and s.clock_offset and replaces those values in the packet with m_1.BD_ADDR and m_1.clock_offset because that it is a multicast member. Then, device m_1 forwards the packet to device m_2 and sets the slave roles in sniff mode for T time slots while it enters page state and tries to establish a link with device m_1 using the information s.BD_ADDR and s.clock_offset. As an upstream link (s, m_1) is established, device m_1 enters page scan state and waits for establishing the downstream links. The other links (m_1, m_2), (m_2, m_3), (m_2, m_4), (m_3, m_5) and (m_3, m_6) will be established in this phase and their device roles may be changed during the construction of the upstream and downstream links. After executing the phase I, the constructed multicast tree and the relationship between the two layers are shown in Fig. 9(a) and (b), respectively.

In phase I of TMCP, there are three tasks including packet *RSP* transmission, packet *RRP* reply as well as downstream and upstream links establishment. To avoid the situation that there are double links connecting two Bluetooth devices, when node x executes *Establish_uplink*(x, y), node x will check if the old link (x, y) was existed. Node x will break the old link (x, y) before establishing the upstream link if the old link (x, y) was existed and node x serves as a master associate with that link. The formal algorithm of the multicast tree construction is illustrated as follows:

Algorithm. The multicast tree construction (phase I)

```
[For the source device $s$]
  Broadcast a route search packet RSP($s$.BD_ADDR, $s$.clock_offset) to all its neighboring nodes;
  Execute Establish_downlinks($s$);
[For the other devices $x$]
  Wait and receive a route search packet RSP($u$.BD_ADDR, $u$.clock_offset);
  If the packet has been received before, it will be discarded;
  If Non-member($x$) Broadcast the RSP($u$.BD_ADDR, $u$.clock_offset);
  else {
    Store $u$.BD_ADDR and $u$.clock_offset;
    Broadcast the RSP($x$.BD_ADDR, $x$.clock_offset);
    Execute Establish_uplink($x$, $u$);
    Execute Establish_downlinks($x$);
  }
}
Function Establish_uplink($a$, $b$){
  If link ($a$, $b$) exists // avoid double-link establishment
    If role($a$) is master
      Break link ($a$, $b$);
    else return;
  Enter page state;
  Set timeout = $T$ time slots for connecting the upstream link ($a$,$b$);
  Set link_flag = false;
  While (not timeout) and (not link_flag){
    If the upstream link ($a$,$b$) is established
      link_flag = true;
    If (link_flag) and (role($a$) is master)
      Switch the role of $a$ to slave;
  }
  If (not link_flag) Send a route reply packet RRP(multicast_path($a$,$b$)) to device $b$;
}
Function Establish_downlinks($k$){
  Enter page scan state;
  Set $T$ time slots for connecting the down-links;
  While (not timeout) {
```

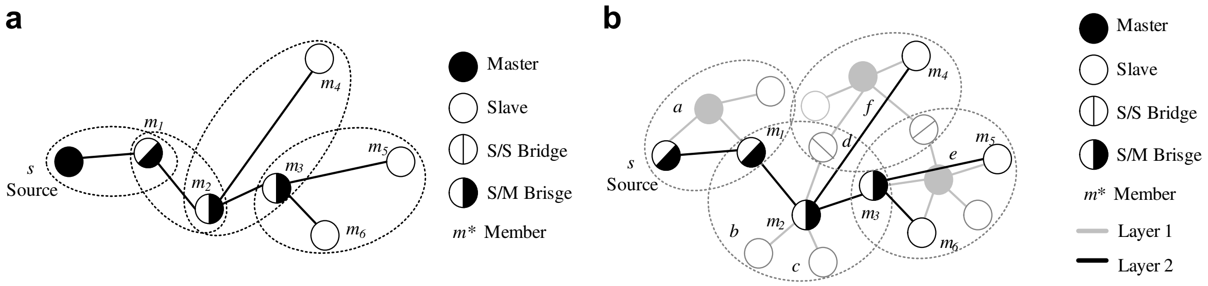


Fig. 9. An example of multicast tree constructed after executing the phase I of TMCP in Fig. 7(b). (a) A multicast tree constructed by applying the phase I of TMCP on scatternet of Fig. 7(a). (b) The view of combination of the constructed multicast tree and the original scatternet shown in Fig. 7(a).

```
    Enter page scan state and try to establish a downstream link $(k,d)$;
    If (the downstream link $(k,d)$ is established) and (role($k$) is slave)
        Switch the role of $k$ to master;
    If a route reply packet $RRP(multicast\_path(a,b))$ is received
        Store the $multicast\_path(a,b)$;
  }
}
```

3.2.2. Multicast tree reorganization phase

After executing phase I, a multicast tree has been constructed in the second layer of the given scatternet. Though all members try to directly connect to the upstream members in the first phase, however, members with different tree levels still belong to different piconets. In this phase, the role switching operations are applied to reorganize the structure of the multicast tree accordingly so that members can be collected into the same piconet as many as possible, exploiting the advantages of broadcasting operation. All masters in the tree apply the *Member-Switch* operations to invite the multicast members in the downstream piconets to join their piconets, reducing the tree height and the number of piconets in the tree. The following first introduces the *Member-Switch* procedure and then presents the algorithm of the multicast tree reorganization phase.

The *Member-Switch* operation is an advanced role switching operation adopted in TMCP. Let m_x and m_y be the masters of two neighboring piconets and m_s be a slave connecting to m_y. The operation of *Members_Switch*(m_x, m_y, m_s) allows that slave m_s connects with m_x and disconnects with m_y when the link (m_s, m_y) does not exist in the original topology. Take Fig. 10(a) as an example. Devices a and e are masters of P_1 and P_2, respectively. After master e initiates the *Members_Switch*(e,a,b) operation, device b switches from piconet P_1 to P_2, hence it connects with master e and disconnects with master a. To achieve this, master e sends a request packet to device b then enters the page scan state for establishing a new link with device b. On receiving the packet sent from e, master a calculates the clock offset between devices e and b and then forwards e.BD_ADDR and the resultant clock offset between devices e and b to device b. According to the received information, device b enters page state and uses e.BD_ADDR and e.clock_offset to derive the hopping sequence of device e, trying to connect with master e. As the link (e,b) is established, device b disconnects with master a. The resultant topology after executing *Members_Switch*(e,a,b) is shown in Fig. 10(b). This operation can reduce the route length and the number of piconets and thus saves the power and bandwidth consumptions and reduces the end-to-end delay.

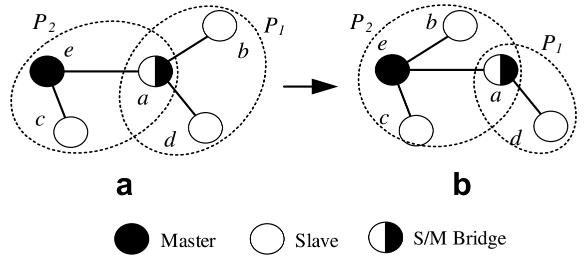


Fig. 10. (a) Topology before executing *Members_Switch*(f,a,b). (b) Topology after executing *Members_Switch*(f,a,b).

The operations of phase II are described in below. The root node will firstly broadcast a query packet to its children. On receiving the query packet, each child, say x, replies to its parent with a query reply packet which contains the BD_ADDR and clock information of its child members and asks them to connect with the root by applying *Members_Switch* operation. In case that any new member, say y, connects with its grandparent, the root will try to connect with node y's child member by doing similar operations as it has done for node x. As far as the root completes the member collection operations, all its children will execute the member collection operations as root has done. As a result, members in the downstream piconets will be invited to join the upstream piconet for reducing the tree height and exploiting the broadcasting advantages.

Following the previous results shown in Fig. 9, we assume that devices s, m_1, m_2, and m_3 are within the transmission range and the distances from device s to m_4, m_5 and m_6 are larger than the communication range. After completing the phase I, the source device starts executing phase II. Firstly, source device switches its role to the master in the second layer and broadcasts a query packet to its slave m_1, inviting the downstream member devices which belong to the other piconet to join its piconet. On receiving the query packet, the device m_1 replies to source s with a query reply packet which contains information of all members in its piconet. Here, the query reply packet contains the set $\{m_2\}$. According to the message replied by device m_1, source s executes *Members_Switch*(s,m_1,m_2) operation which establishes the link (s,m_2) and breaks the link (m_1,m_2).

Since source s has a new child m_2, it again sends a query packet to device m_2 and intends to invite more members in the downstream piconets of m_2 to join its piconet. On receiving the query packet, the device m_2 does the similar operation as device m_1 does, replying to source s with a query reply packet which contains members m_3 and m_4 in its piconet. Then source s executes *Members_Switch*(s,m_2,m_3) and *Members_Switch*(s,m_2,m_4) operations. Since the distance between devices s and m_4 is larger than communication range, the link (s,m_4) cannot be established.

The device s then sends a query packet to the new neighboring device m_3 and intends to construct more links with members in the m_3's downstream piconets. However, the distances from s to m_5 and m_6 are larger than the communication range, resulting the execution of *Members_Switch*(s,m_3,m_5) and *Members_Switch*(s,m_3,m_6) operations failure.

The source s then broadcasts to its slaves m_1, m_2 and m_3 a member collecting packet which request them to set up more links with members in the downstream piconets. On receiving the packet, devices m_1, m_2 and m_3 switch their roles to the master in layer two and broadcast query packets to their slaves in their piconets. Since devices m_4, m_5 and m_6 play pure slaves, all of them reply the null messages to the sender devices m_1, m_2 and m_3. As devices m_1, m_2 and m_3 receive the null messages, the process of phase II is completed. After executing the tree reorganization phase, the resultant multicast tree topology is shown in Fig. 8(a).

The algorithm of the multicast tree reorganization is illustrated as follows:

Algorithm. The multicast tree reorganization (phase II)

```
[For the source device s]
  Execute Collecting_members(s);
  If Exist(multicast_path(s, d))
    Send a member collecting packet MCP(s) along the multicast_path(s, d) to device d;
  Broadcast a member collecting packet MCP(s);
[For the other devices x]
  Switch the device role to slave in layer two;
  Wait and receive a query packet QP(u);
  If Layer2_Role(x) = 'Slave' {
    Reply a query reply packet QRP(x, u, NULL);
    Return;
  }
  If Layer2_Role(x) = 'Master/Slave' {
    Set the set of neighboring slaves in layer two be N;
    Reply a query reply packet QRP(x, u, N);
  }
  Wait and receive a member collecting packet MCP(u);
  Execute Collecting_members(x);
  If Exist(multicast_path(x, d))
    Send a member collecting packet MCP(x) along the multicast_path(x, d) to device d;
  Broadcast a member collecting packet MCP(x);
}
FunctionCollecting_members(a) {
  Switch the device role to master in layer two;
  Broadcast a query packet QP(a);
  Set the set of neighboring slaves to be N;
  While ($N \neq \Phi$) {
    Wait and receive a query reply packet QRP(d, a, T);
    If (T is NULL) $N = N - \{d\}$;
    else {
      For (each $k \in T$) Execute Members_Switch(a, d, k);
      If a link (a,k) is established {
        Broadcast a query packet QP(a);
        $N = N + \{k\}$;
        If Layer2_Master_links(a) = 7 Return;
      }
      else $N = N - \{k\}$;
    }
  }
}
```

In the next section, the performance studies of TMCP are investigated.

4. Performance study

Given a connected scatternet constructed by the formation protocol [13], the flooding mechanism and the proposed TMCP are compared. In previous research, the RVM routing protocol [2] was developed for establishing a route from single source to single destination in a Bluetooth scatternet. The path construction of RVM mainly applies flooding-based mechanism which discovers a route by flooding a route search packet over the scatternet. Therefore, the RVM is a flooding-based routing protocol. This section investigates the performance study using the GloMoSim simulator [23]. In our experimental environment, we simulate all overheads raised by executing the TMCP and flooding-based algorithms. These overheads include the packet transmissions, power consumption and time required for scatternet construction, tree construction, piconet switching and role switching operations.

The environment is set as follows: The size of experimental region is set at 10 × 10, 20 × 20, or 30 × 30 units, while the communication range of a Bluetooth device is set at a constant 10 units. In the scatternet, the number of Bluetooth devices is set at 80 whereas the number of multicast members varies ranging from 25 to 55, and their locations are randomly determined. The packet type DH1 is considered in the traffic flow. The inter-piconet synchronization and scheduling are important issues which will affect the performance results. In the simulation, the inter-piconet scheduling algorithm proposed in [24] is adopted to synchronize the packet transmission between neighboring piconets. Each performance result was obtained from the average results of 100 experiments. Performance measures considered herein include the ratio of forwarding nodes, the power consumption of multicast tree construction and multicasting, average height of multicast tree, average degree of internal tree nodes, construction delay of multicast tree, guard time between piconets and propagation delay of multicast services.

One of the most important features of an efficient multicast tree is that it contains few or even no forwarding nodes which are not multicast members but participate in the multicast tree. Forwarding nodes not only consume the power and bandwidth resources but also increase the transmission delay. Fig. 11 compares the TMCP and flooding mechanisms in terms of the ratio of forwarding nodes, which is measured by the ratio of the number of non-member nodes that participate in the multicast tree to the number of all nodes in the multicast tree. Different sized regions, including 10 × 10, 20 × 20, and 30 × 30 units, are investigated to evaluate the ratio of forwarding node. In general, the ratio of forwarding nodes decreases with the average number of members in a piconet which decreases with the total number of members in a fixed region with constant devices, as shown in Fig. 11. This is because that more members in a transmission range result smaller opportunities for non-member nodes to participate the tree. In case of the region with size 10 × 10, the multicast tree constructed by applying TMCP contains no forwarding node.

Fig. 12 presents two experiment results: The first one depicts the difference in power consumptions for executing phases I and II of TMCP under different region sizes. In comparison, the energy consumption in phase II is approximately a half of that in phase I. We notice that the energy consumption in phase I is equal to that required for the path construction in previous work RVM [2] since RVM [2] applies the flooding-based mechanism to construct a routing path. This implies that TMCP will additionally consume a half energy consumption than RVM in order to reduce the tree height, propagation delay and energy consumption for message multicasting. However, TMCP works well when the traffic arrival rate is high and the number of members in the scatternet is small since the constructed multicast tree will efficiently delivers the data traffic to those members. Hence, we claim that Fig. 12 is a good result for TMCP in terms of energy consumption when the transmission load associated with multicasting is heavy and the number of members is small. In the later paragraph, Fig. 19 will reflect this argument. The other observation from Fig. 12 is that the power consumption of multicast tree construction in TMCP increases with the number of members because that TMCP establishes some extra links between member devices. In the phase II of TMCP, the number of *Members_Switch* operations executed for reducing the tree height increases with the number of members within the communication range, and hence increases the power consumption in tree reorganization.

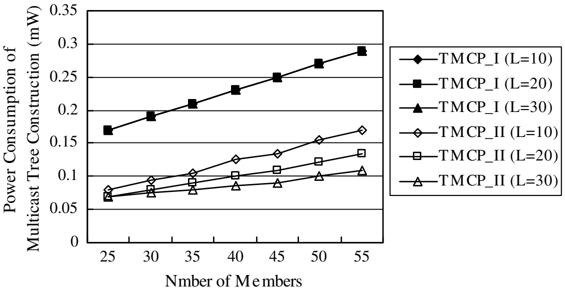


Fig. 12. The power consumptions for executing phases I and II of TMCP. The region size is set at $L \times L$ units, where $L = 10$, 20 or 30.

Fig. 13 compares the flooding and TMCP mechanisms in terms of the average length of multicast tree. In general, the length of multicast tree increases with the number of piconets and the number of members in TMCP. As shown in Fig. 13, multicast tree constructed by applying TMCP has a smaller length in all cases of different region sizes. Since the multicast tree constructed by TMCP collects as many as possible the member devices into the same piconet, TMCP has a smaller tree length in average. Fig. 14 compares the average down-link number of internal nodes in the constructed multicast tree. In average, TMCP has a higher degree in all cases of region sizes. The average down-link number of internal nodes decreases with the region size since the members are scattered in a big range. An interesting result is that the average down-link number of internal nodes decreases with the number of members in case of regions size 10 × 10. The reason is analyzed in below. Since a piconet can only contain at most seven slaves, the increase in number of members in a small re-

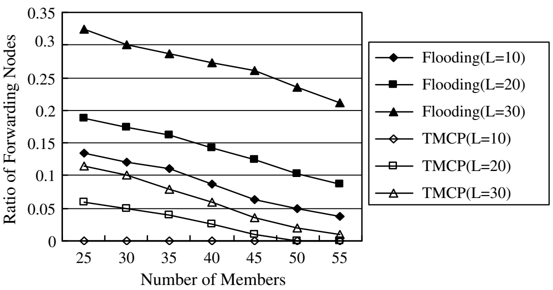


Fig. 11. Comparisons of TMCP and flooding mechanisms in terms of the number of forwarding nodes in the constructed multicast tree. The region size is set at $L \times L$ units, where $L = 10$, 20 or 30.

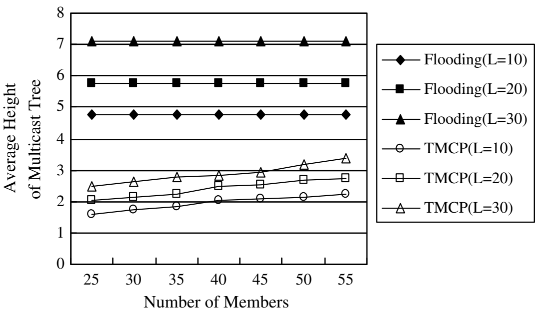


Fig. 13. Comparison of the TMCP and flooding mechanisms in terms of the tree height. The region size is set at $L \times L$ units, where $L = 10$, 20 or 30.

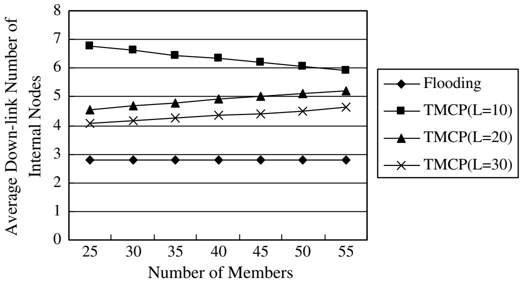


Fig. 14. Comparison of the TMCP and flooding mechanisms in terms of the down-link number of internal nodes. The region size is set at $L \times L$ units, where $L = 10$, 20 or 30.

gion can enlarge the tree height. However, the phase II of TMCP is difficult to balance the down-link number of internal node between sibling sub-trees and piconets closer to the leaf are more difficult to fully collect seven members. Therefore, when the tree height is enlarged, the average down-link number of internal nodes decreases.

Although TMCP consumes more energy than flooding scheme in the tree construction phase, however, TMCP reduces the tree height and further saves the energy consumption in data transmission phase. Fig. 15 shows the comparison of TMCP and flooding mechanisms in terms of power consumption which are consumed in tree construction, tree organization, and data transmission under various transmission loads. As shown in Fig. 15, TMCP results lower energy consumption than flooding scheme when the transmission loads are set at 0.1K, 5K and 10K bytes. Consequently, it is cost effective to construct the multicast tree by applying TMCP when the transmission load associated with multicasting service is heavy or when the multicast service involves few members.

In TMCP, the upstream and downstream link constructions in phase I are the major source of construction delay. Fig. 16 presents the construction delay of phase I and phase II of TMCP. In a multicast tree, bridge nodes are responsible to transmit multicast packets from in turn among the participating piconets. However, bridge switching, from one piconet to another, raises the guard time overhead which drops the Bluetooth performance. Fig. 16 measures the TMCP and flooding mechanisms in terms of the guard time overhead involved for bridge switching among the participating parents. The TMCP assigns every bridge node with a slave role in the parent piconet and with a master role in the child piconet. In executing the multicast service, master of each piconet may broadcast the multicast packet to all bridges. On receiving the multicast packet, each bridge node then switches from the parent piconet to the child piconet, plays the master role, and then immediately broadcasts the received multicast packet to all members in its piconet. However, applying the flooding mechanism, a bridge might play a slave role in the downstream piconet. When a bridge intends to broadcast the multicast packets to all members in the downstream piconet, it should wait for polling from the master of the downstream piconet. Consequently, TMCP has a smaller guard time overhead than flooding scheme as shown in Fig. 17.

Fig. 18 compares the flooding and TMCP schemes in terms of the tree construction time. The proposed TMCP scheme applies role switching and member collection operations, thus takes more time for tree construction. However, it reduces the tree height and therefore saves end-to-end delay, bandwidth consumption and power consumption. Fig. 19 shows the comparison of TMCP and flooding schemes in terms of transmission delay by considering transmission load at 0.1K, 5K and 10K bytes when the region size is set at 20 × 20 units. The traffic arrival rate and the number of members in the multicast group would be the two major factors that impact the performance of a multicast tree. To construct an efficient multicast tree, the TMCP creates some control overheads and delays. How-

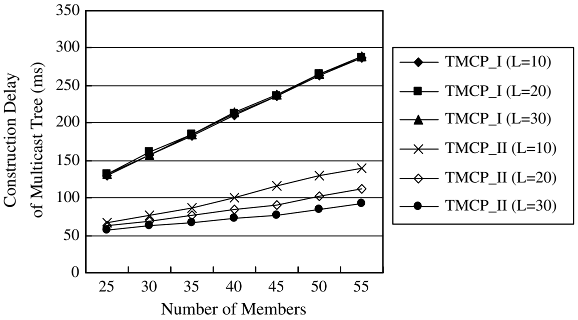


Fig. 16. Tree construction delay of phases I and II of TMCP. The region size is set at $L \times L$ units, where $L = 10$, 20 or 30.

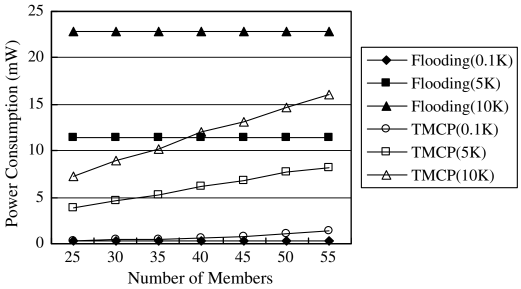


Fig. 15. Comparison of TMCP and flooding mechanisms in terms of power consumption under various transmission loads. The region size is set at 20 × 20 units.

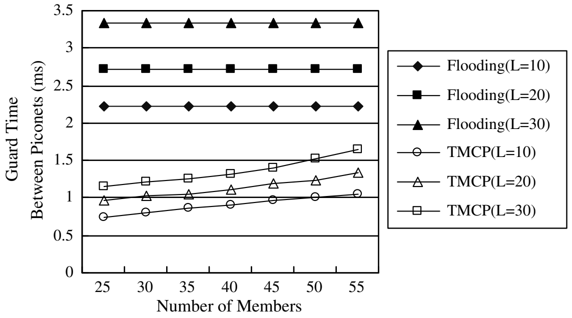


Fig. 17. Comparison of TMCP and flooding mechanisms in terms of the guard time. The region size is set at $L \times L$ units, where $L = 10$, 20 or 30.

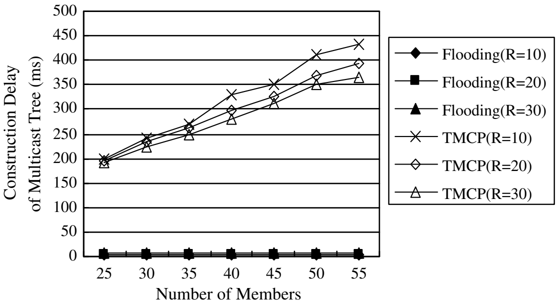


Fig. 18. Comparison of TMCP and flooding schemes in terms of tree construction. The region size is set at $L \times L$ units, where $L = 10$, 20 or 30.

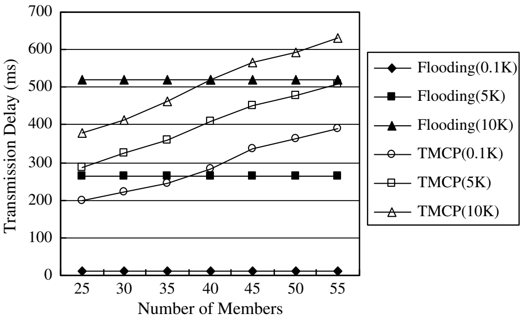


Fig. 19. Comparison of TMCP and flooding schemes in terms of the propagation delay of multicast service under various transmission loads. The region size is set at 20 × 20 units.

ever, the constructed tree saves a large amount of data traffic. On the other hand, the flooding mechanism saves the extra control overhead but creates a large amount of data traffics since the constructed tree contains a lot of non-member Bluetooth nodes. In case that there are a large number of members in the scatternet, most nodes in the constructed tree by applying the flooding mechanism would be receivers. Therefore, the traffics are not redundant traffics. However, if the number of members is small, the constructed tree by applying flooding mechanism is not efficient since it contains too many non-member nodes. This will cause that a large amount redundant traffics which are not needed by the non-member nodes in the tree. Therefore, the flooding mechanism will have a poor performance when the traffic arrival rate is high but the number of members in the scatternet is small. On the contrary, TMCP works well when the traffic arrival rate is high and the number of members in the scatternet is small since the constructed multicast tree will efficiently delivers the data traffic to those members. TMCP is cost effective in reducing the average tree height, especially when the transmission load associated with multicasting is heavy (10K bytes) and the number of members is small (25) as shown in Fig. 19. Briefly, the multicast tree constructed by applying TMCP has good properties including small tree height, few forwarding nodes, and proper role assignment for each node in the tree.

Fig. 20 investigates the end-to-end delay of the QoS and non-QoS multicast services provided based on the multicast tree constructed by TMCP and flooding mechanisms. In the experiment, each piconet is given a basic traffic which grows at a rate of 2K bytes/s throughout the experiment time period. The basic traffic is a non-QoS service and is denoted by "Piconet(non-QoS)". In addition to this basic traffic load, a multicast service with a traffic load of 20K bytes/s is also initiated for comparing with the basic traffic in their end-to-end delay. Three types of multicast services are compared with the basic traffic. The first type of multicast service, denoted by "Flooding(non-QoS)", is provided based on the multicast tree that was constructed by applying the flooding mechanism and which did not ask for a QoS requirement. The second and third types of multicast services are provided based on the multicast tree that was constructed by applying the TMCP. Notations "TMCP(QoS)" and "TMCP(non-QoS)", respectively, denote the two types of multicast services with and without QoS requirements.

Let the maximal delay requested by the QoS multicast service be one second and let the DH5 packet be used to deal with the traffic in symmetric transmission mode. In each experiment, only one multicast service is able to coexist with the basic traffic generated in each piconet. In general, the transmission delays of the non-QoS services, including Flooding(non-QoS) and TMCP(non-QoS), increased along with the basic traffics. When a given traffic becomes larger than 40K bytes/s, the traffic in each piconet exceeds the supported maximal bandwidth. Since the

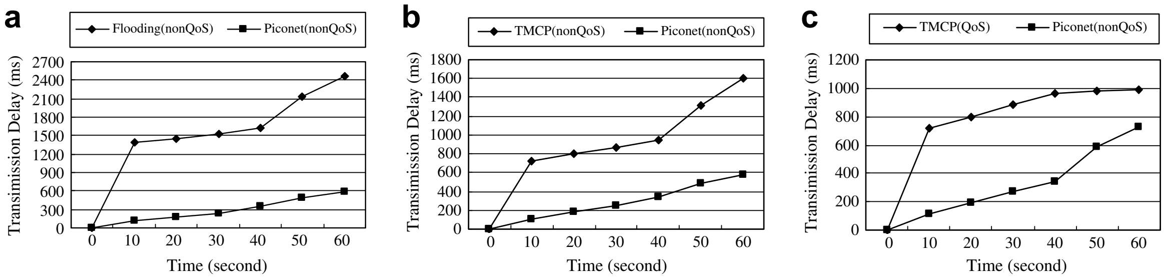


Fig. 20. Comparison of Flooding(non-QoS), TMCP(non-QoS) and TMCP(QoS) schemes in terms of transmission delay. The number of members and region size are set at 45 and 20 × 20 units, respectively.

Flooding(non-QoS) and TMCP(non-QoS) fairly share available time slots with the Piconet(non-QoS), their end-to-end delays increase significantly, as shown in Figs. 20(a) and (b). TMCP is proposed to reconstruct a good scatternet topology to reduce the tree height of a multicast tree. A small-height multicast tree can also help reduce the delay time for multicast services under inter-piconet scheduling. In order to guarantee the end-to-end delay of a multicast service, each node of the constructed multicast tree should calculate the reserved time slots according to the sub-tree height to control the guaranteed delay and request for the reserved time slots from the master in the same piconet. As a result, TMCP(QoS) increases along with the given traffic, but its transmission delay comes close to one second even though the total traffic in each piconet exceeds the maximal bandwidth a piconet can support. To guarantee the QoS requirement of TMCP(QoS), the inter-piconet scheduling algorithm prior allocates the available time slots to TMCP(QoS). Therefore, the transmission delay of Piconet(non-QoS) significantly increases after the time unit 40 because the TMCP(QoS) would then occupy the reserved time slots. The experiment of Fig. 20 indicates that the multicast tree constructed by TMCP can reduce the end-to-end delay and inter-piconet scheduling, which guarantees that the QoS requirement is essential for providing a QoS multicast service.

5. Conclusions

This paper purposes TMCP, a two-layer multicast tree construction protocol, which constructs an efficient tree for multicast service in Bluetooth networks. Given a connected scatternet, TMCP treats the original scatternet as the first layer and constructs an efficient multicast tree on the second layer. Therefore, the other communication services executed on the original scatternet are not affected. Applying the role switching operations, TMCP collects as many as possible the multicast members into the same piconet so that the tree height can be reduced. As a result, the multicast service can take full advantages from the broadcasting operations. Simulation results show that the proposed TMCP outperforms flooding scheme in terms of the energy and bandwidth consumptions and the propagation delay.

References

[1] Bluetooth specification 1.1b.

[2] B. Pravin, S. Adrian, A routing vector method (RVM) for routing in Bluetooth scatternets, in: Proceeding of the 6th IEEE International Workshop on Mobile Multimedia Communications (MoMuC), November 1999, pp. 375–379.

[3] S.K. Das, B.S. Manoj, C.S.R. Murthy, A dynamic core based multicast routing protocol for ad hoc wireless networks, in: Proceeding of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), June 2002, pp. 24–35.

[4] C.R. Dow, J.H. Lin, K.T. Chen, S.C. Chen, S.F. Hwang, A scalable and reliable multicast routing protocol for ad-hoc wireless networks, Wireless Personal Communications 34 (4) (2005) 335–357.

[5] C.C. Chiang, M. Gerla, L. Zhang, Adaptive shared tree multicast in mobile wireless networks, in: Proceeding of the IEEE Global Telecommunications Conference (GLOBECOM), November 1998, pp. 1817–1822.

[6] S.J. Lee, W. Su, M. Gerla, On-demand multicast routing protocol in multihop wireless mobile networks, Mobile Networks and Applications 7 (6) (2002) 441–453.

[7] L. Layuan, L. Chunlin, A distributed multicast routing protocol with QoS constraints, in: Proceeding of the IEEE 10th International Conference on Networks (ICON), August 2002, pp. 37–42.

[8] S. Lee, W. Su, J. Hsu, M. Gerla, R. Bagrodia, A performance comparison study of ad hoc wireless multicast protocols, in: Proceeding of the 19th Conference of the IEEE Communications Society (INFOCOM), March 2000, pp. 565–574.

[9] J.G. Jetcheva, D.B. Johnson, Adaptive demand-driven multicast routing in multi hop wireless ad hoc networks, in: Proceeding of the 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), October 2001, pp. 33–44.

[10] D. Famolari, P. Agrawal, Architecture and performance of an embedded IP Bluetooth personal area network, in: Proceeding of the 2000 IEEE International Conference on Personal Wireless Communications (ICPWC), November 2000, pp. 75–79.

[11] C.Y. Chang, G.J. Yu, C.F. Lin, T.T. Wu, Relay reduction and route construction for scatternet over Bluetooth radio systems, in: Proceeding of the IEEE 16th International Conference on Information Networking (ICOIN), 2 January 2002, pp. 5B2.1–5B2.10.

[12] C.Y. Chang, K.P. Shih, S.C. Lee, C.H. Tseng, Adaptive role switching protocols for improving scatternet performance in Bluetooth radio networks, in: Proceeding of the 15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), September 2004, pp. 375–379.

[13] L. Ramachandran, M. Kapoor, A. Sarkar, A. Aggarwal, Clustering algorithms for wireless ad hoc networks, in: Proceeding of the 4th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM), August 2000, pp. 54–63.

[14] Z. Wang, T.J. Robert, H. Zygmunt, Bluenet – a new scatternet formation scheme, in: Proceeding of the 35th Hawaii International Conference on System Science (HICSS), January 2002, pp. 61–69.

[15] G.V. Zaruba, S. Basagni, I. Chlamtac, Bluetrees – scatternet formation to enable Bluetooth-based ad hoc networks, in: Proceeding of the IEEE International Conference on Communications (ICC), 2001, pp. 273–277.

[16] C. Law, A.K. Mehta, K.Y. Siu, Performance of a new Bluetooth scatternet formation protocol, in: Proceeding of the ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), October 2001, pp. 183–192.

[17] G. Roch, R. Jacob, S. Saswati, V. Evangelos, Forming connected topologies in Bluetooth ad hoc networks, in: Proceeding of the 18th International Teletraffic Congress (ITC), August 2003, pp. 1011–1020.

[18] Y. Kim, K.H. Jang, K.H. Tchah, Efficient peer-to-peer communications in Bluetooth, IEICE Transactions on Communications E86-B (10) (2003) 3023–3029.

[19] L. Ching, K.Y. Siu, A Bluetooth scatternet formation algorithm, in: Proceeding of the IEEE Global Telecommunications Conference (GLOBECOM), November 2001, pp. 2864–2869.

[20] K.E. Persson, D. Manivannan, Distributed self-healing Bluetooth scatternet formation, in: Proceeding of the International Conference on Wireless Networks (ICWN), June 2004, pp. 325–334.

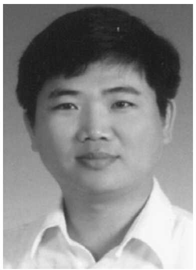
[21] L. Farkas, B. Bakos, P. Spànyi, A practical approach to multicasting in Bluetooth piconets, in: Proceeding of the IEEE Wireless Communications and Networking Conference (WCNC), March 2006, pp. 1360–1366.

[22] S. Baatz, M. Frank, P. Martini, C. Scholz, A worst-case model for co-channel interference in the Bluetooth wireless system, in: Proceeding of the 28th IEEE International Conference on Local Computer Networks, October 2002, pp. 346–355.

[23] X. Zeng, R. Bagrodia, M. Gerla, GloMoSim: a library for parallel simulation of large-scale wireless networks, in: Proceeding of the Twelfth Workshop on Parallel and Distributed Simulation, May 1998, pp. 154–161.

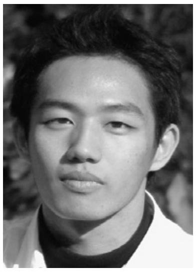
[24] J.P. Sheu, K.P. Shih, S.C. Tu, C.H. Cheng, A traffic-aware scheduling for Bluetooth scatternets, IEEE Transactions on Mobile Computing 5 (7) (2006) 872–883.

[25] J. Bray, C.F. Sturman, Bluetooth Connect Without Cables, Prentice-Hall, America, 2001.



Chao-Tsun Chang was born in Taipei, Taiwan. He received the Ph.D. degree in Computer Science and Information Engineering from National Central University, Taiwan, in 2006. He joined the faculty of the Department of Department of Information Management, Hsiuping Institute of Technology, Taiwan, as an Assistant Professor in 2006. In the recent 10 years, he has directed 11 researching projects including three national NSC projects and 8 information system developments. He is a member of the IEEE Computer Society and Communication Society. His current research interests include wireless sensor networks, Bluetooth radio networks, Ad Hoc wireless networks, and mobile computing.

Chih-Yung Chang received the Ph.D. degree in Computer Science and Information Engineering from National Central University, Taiwan, in 1995. He joined the faculty of the Department of Computer and Information Science at Aletheia University, Taiwan, as an Assistant Professor in 1997. He was the Chair of the Department of Computer and Information Science, Aletheia University, from August 2000 to July 2002. He is currently an Associate Professor of Department of Computer Science and Information Engineering at Tamkang University, Taiwan. He served as an Associate Guest Editor of *Journal of Internet Technology* (JIT, 2004), *Journal of Mobile Multimedia* (JMM, 2005), and a member of Editorial Board of *Tamsui Oxford Journal of Mathematical Sciences* (2001–2005). He was an Area Chair of *IEEE AINA*'2005, Vice Chair of *IEEE WisCom*'2005 and *EUC*'2005, Track Chair (Learning Technology in Education Track) of *IEEE ITRE*'2005, Program Co-Chair of *MNSAT*'2005 and *UbiLearn*' 2006, Workshop Co-Chair of *INA*'2005, *MSEAT*'2003, *MSEAT*'2004, and Publication Chair of *MSEAT*'2005 and *SCORM*'2006. He is a member of the IEEE Computer Society, Communication Society and IEICE society. His current research interests include wireless sensor networks, mobile learning, Bluetooth radio networks, Ad Hoc wireless networks, and mobile computing.



Sheng-Wen Chang received the B.S. degree in Computer Science and Information Engineering from Tamkang University, Taiwan, in 2004. Since 2005, he was working toward his Ph.D. degree and currently he is a Ph.D. candidate in Department of Computer Science and Information Engineering at Tamkang University. He is a student member of the IEEE Computer Society, Communication Society and IEICE society. He has won lots of scholarships in Taiwan and participated in many projects of Bluetooth radio networks and wireless sensor networks. His research interests include Bluetooth radio networks, wireless sensor networks, WiMAX, Ad Hoc wireless networks and mobile computing.