Wireless Personal Communications (2006)
DOI: 10.1007/s11277-006-9014-0

A Credit-Based On-Demand QoS Routing Protocol Over Bluetooth WPANs*

YUH-SHYAN CHEN and KENG-SHAU LIU
Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan, 621, R.O.C.
E-mail: yschen@cs.ccu.edu.tw

Abstract. The quality-of-service (QoS) communication that supports mobile applications to guarantee bandwidth utilization is an important issue for Bluetooth wireless personal area networks (WPANs). In this paper, we address the problem of on-demand QoS routing with interpiconet scheduling in Bluetooth WPANs. A credit-based QoS (CQ) routing protocol is developed which considers different Bluetooth packet types, because different types of Bluetooth packets have different bandwidth utilization levels. This work improves the bandwidth utilization of Bluetooth scatternets by providing a new interpiconet scheduling scheme. This paper mainly proposes a centralized algorithm to improve the bandwidth utilization for the on-demand QoS routing protocol. The centralized algorithm incurs the scalability problem. To alleviate the scalability problem, a distributed algorithm is also investigated in this work. The performance analysis illustrates that our credit-based QoS routing protocol achieves enhanced performance compared to existing QoS routing protocols.

Keywords: Bluetooth, mobile computing, quality-of-service (QoS), routing protocol, wireless personal area network.

1. Introduction

Bluetooth is a low-power short-range wireless network technology that is designed to provide a cableless wireless communication environment for various kinds of personal communication system (PCS) devices, such as mobile phones, laptop computers, cordless headsets, and personal digital assistants (PDAs). On the basis of the IEEE 802.15.1 standard [1], Bluetooth devices establish wireless personal area networks (WPANs) that are similar to ad hoc network environments to provide wireless PCSs for supporting many valuable mobile applications. To build WPANs, Bluetooth devices can be used to provide useful services such as wireless Internet access or mobile multimedia applications in mobile ad hoc network (MANET) systems [2, 3]. Recently, wireless sensor networks (WSNETs) have been widely investigated, and these are constituted by a large number of low-power sensor nodes. According to limitations of low power and low costs of each sensor node, the characteristic of the Bluetooth is most closely matched to WSNETs, especially with regard to supporting multimedia services with high bandwidth requirements.

The design issues of a Bluetooth scatternet are still an open issue, because no detailed definition of scatternet formation is defined in the Bluetooth specifications [1]. Many scatternet

*This work was supported by the National Science Council of the Republic of China under grant nos. NSC-92-2213-E-194-022 and NSC-93-2213-E-194-028.

formation protocols have recently been proposed [4–7]. In addition, a number of researchers have addressed the issues of routing, scheduling, and QoS-extension routing in Bluetooth networks. First, on-demand routing approaches in Bluetooth scatternets were investigated [8–10]. Liu et al. [9] proposed an on-demand routing approach which combined scatternet formation techniques. With this on-demand protocol, it is not necessary to update and maintain the routing information in a routing table, but the Bluetooth core protocols need to be changed in order to offer on-demand routing capability. Bhagwat et al. [8] proposed an efficient method for route discovery and packet forwarding by encoding source route paths. Recently, Prabhu et al. [10] considered the power control capability in order to increase the network lifetime of Bluetooth scatternets.

In the Bluetooth specifications [1], round-robin (RR) scheduling is used, but has a time-slot wastage problem. Many scheduling results for Bluetooth networks have been investigated [11–13], and these attempted to ameliorate the time-slot wastage problem. First, an efficient pattern-matching polling (PMP) policy for data-link scheduling was proposed by Lin et al. [12]. To consider a piconet, the PMP policy calculates different combinations of Bluetooth packet types to search for better polling patterns. The system bandwidth is thus improved, especially for asymmetrical traffic. Yang et al. [13] proposed two scheduling policies, look-ahead (LA) and look-ahead round robin (LARR), to improve the conventional RR scheduling policy. Lin et al. [11] additionally proposed a power-saving scheduling scheme which utilizes the sniff mode. These protocols only provide piconet scheduling results.

Many researchers [14–17] have recently attempted to develop QoS-extension routing scheduling in Bluetooth scatternets. First, Cordeiro et al. [16] proposed dynamic slot assignment (DSA) and enhanced DSA (EDSA) schemes. A direct slave-to-slave communication model is presented in [16] to provide QoS requirements with better bandwidth utilization, shorter delays, lower overhead, and lower power utilization. Unfortunately, no interpiconet scheduling mechanism has been devised for when the source and destination nodes are located in distinct piconets. Baatz et al. [14, 15] indicated some scheduling design difficulties; for instance, a master cannot assume its slaves are always in the listening mode. They [14, 15] addressed the credit scheme and adaptive presence point density (APPD) scheme to provide a scatternet scheduling mechanism. Using the sniff mode, their scheduling mechanisms can be achieved without modification of the Bluetooth specifications [14, 15].

A QoS scheduling mechanism for scatternets was recently investigated by Kim et al. [17]. They presented a QoS-aware scheduling algorithm to resolve the contentious problem of bridge devices for Bluetooth scatternets, and investigated both centralized and distributed scheduling algorithms. The algorithms proposed by Kim et al. are only suitable for tree-structure scatternets [7]. The success rate of QoS-aware scheduling algorithms drops off for non-tree-structure scatternets. It also degrades the bandwidth utilization of each Bluetooth device.

In this paper, we address these on-demand quality-of-service routing and interpiconet scheduling problems. A credit-based QoS (CQ) routing protocol is developed which considers different Bluetooth packet types which have different bandwidth utilization levels [12]. This concept can improve the bandwidth utilization of Bluetooth scatternets. Interpiconet scheduling problems can be resolved by using our CQ approach. Observe that this paper mainly proposes a centralized algorithm to improve the bandwidth utilization for the on-demand QoS routing protocol. The centralized algorithm incurs the scalability problem. To alleviate the scalability problem, a distributed algorithm is also investigated in this work. The simulation results illustrate that our CQ routing algorithm performs better than Kim *et al.'s* approach [17].

The rest of this paper is organized as follows. Section 2 introduces some preliminary concepts. In Section 3, we develop the centralized QoS routing protocol, and the distributed QoS routing protocol is presented in Section 4. Section 5 discusses the experimental results. Finally, Section 6 concludes this paper.

2. Preliminary Concepts

In this section we discuss some background of this work. We give an overview of information on Bluetooth technology. Existing result presented by Kim et al. [17] which motivated our investigation is described.

2.1. ACL/SCO Link Properties of Bluetooth

According to Bluetooth specifications [1], each Bluetooth device [1] performs inquiry/inquiry scan and page/page scan processes to form a piconet. A scatternet is usually comprised of a set of piconets via relay (bridge) devices to extend the transmission range. After the inquiry/inquiry scan and page/page scan processes have been carried out, a Bluetooth device may enter into the *connection* state. In the connection state, two types of traffic links are used: the asynchronous connectionless (ACL) link and the synchronous connection-oriented (SCO) link. The ACL link provides data communication, while the SCO link supports circuit-switched connections for time-bounded information, such as audio transmissions. Observe that packets in SCO links are not retransmitted. This paper only discusses QoS problems for data transmission in ACL links.

In the following, we only investigate QoS issues in ACL links. The packet of an ACL link may have one, three, or five time slots as shown in Table 1 [1, 12]. The DM and DH packet types stand for data with medium and high rates, respectively. DM-type packets contain a forward error correction (FEC), but the DH-type packets do not contain the FEC. The bandwidth utilization is calculated by the amount of the data payload (bytes) for one packet type (DM1, DM3, DM5, DH1, DH3, or DH5) divided by the number of time slots used by that packet type. For instance, the bandwidth utilization of the DM5 packet is 44.8 bytes/slot ($224/5 = 44.8$), where the payload of the DM5 packet is 224 bytes and the number of time slots used is five. In addition, the bandwidth utilizations of the DM1, DM3, DM5, DH1, DH3, and DH5 packets are shown in Table 1. For instance, the bandwidth utilization of DM1 (17.0) $<$ that of DM3 (40.3) $<$ that of DM5 (44.8). Similar results for DH1, DH3, and DH5 are given in Table 1. This fact motivated us to develop an efficient QoS routing protocol in this work by taking different packet types with various bandwidth utilization levels into account.

Table 1. Bluetooth ACL data packets

Type	Used payload (bytes)	FEC (forward error correction)	Bandwidth utilization bytes/slot)
DM1	17	Yes	17.0
DM3	121	Yes	40.3
DM5	224	Yes	44.8
DH1	27	No	27.0
DH3	183	No	61.0
DH5	339	No	67.8

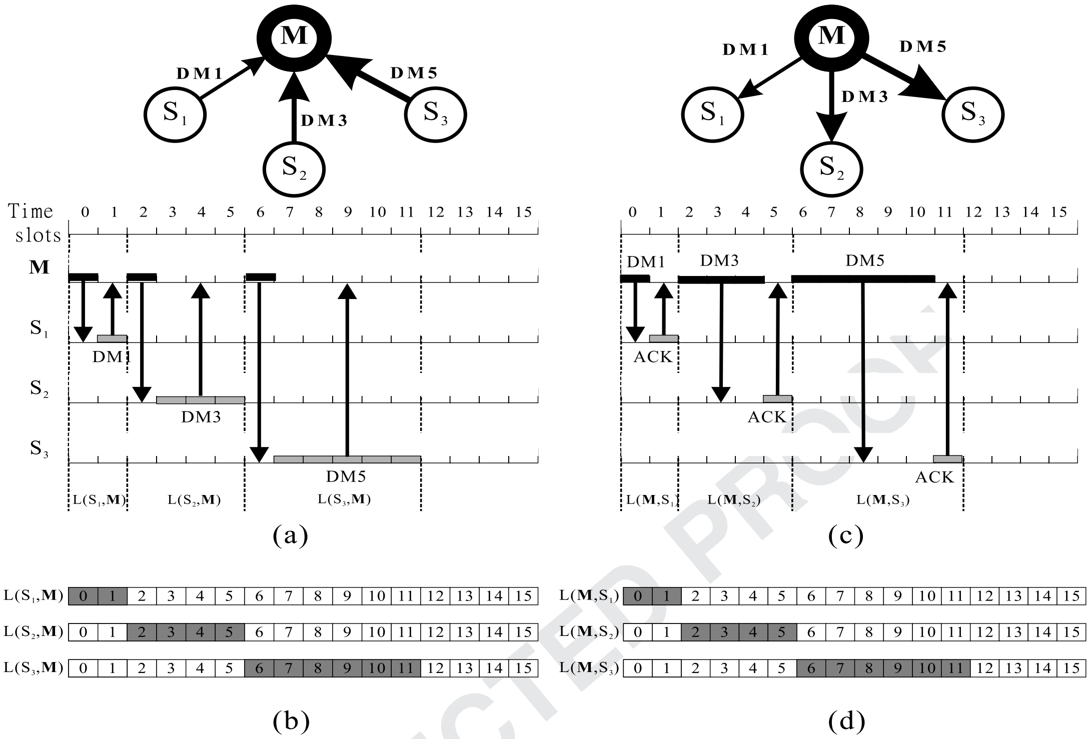


Figure 1. All conditions concerning communication between master and slaves.

In a piconet, a master transmits data to slaves in even-numbered time slots, and slaves always transmit data to the master in odd-numbered time slots. This can be achieved since the packet numbers of DM*x* and DH*x* are odd numbers, where *x* is 1, 3, or 5. One master and three slaves exist in a piconet, and the DM*x* packets are used for transmitting data as shown in Figure 1. For instance, the master polls the slaves, and the slaves transmit DM1, DM3, and DM5 packets back to the master as shown in Figure 1(a). The master directly transmits DM1, DM3, and DM5 packets to slaves as shown in Figure 1(c), and then the ACK. message is returned from the slave to the master.

A high-performance QoS routing protocol is designed if the protocol has better time slot utilization. To explain time-slot utilization, let $L(X, Y)$ or $\overleftrightarrow{XY}$ denote the traffic link between Bluetooth devices *X* and *Y*, where *X* and *Y* are the source and destination devices, respectively. Assume that there are δ time slots in a polling interval or cycle time. The black time slot denotes the busy time slot. Let $(\alpha_1, \alpha_2, \ldots, \alpha_l)$ denote the time slots in a polling interval of a Bluetooth device, where $l = 16$ or 32. As illustrated in Figure 1(b), a polling interval has 16 time slots. Time slots 1 and 2 are busy time slots for $L(S_1, M)$, time slots 2, 3, 4, and 5 are busy for $L(S_2, M)$, and time slots from 6 to 11 are busy for $L(S_3, M)$. This condition can be formally represented as DM1, DM3, and DM5 packets occupying time slots (α_1, α_2), $(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$, and $(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6)$, respectively. For instance as shown in Figure 1(b), $L(S_1, M)$, $L(S_2, M)$, and $L(S_3, M)$ occupy time slots (0,1), (2, 3, 4, 5), and (6, 7, 8, 9, 10, 11), respectively. Other examples are shown in Figure 1(d).

Consider link $L(S_1, M)$ as illustrated in Figure 1(a), master *M* polls slave S_1 in time slot “0”, and slave S_1 transmits packet DM1 to master *M* in time slot “1”. Transmission

DM1 uses even-numbered time slots. Other links, $L(S_2, M)$ and $L(S_3, M)$, for DM3 and DM5 using even-numbered time slots are also given in Figure 1(a). Similar results for links $L(M, S_1)$, $L(M, S_2)$, and $L(M, S_3)$ with DM1, DM3, and DM5 packets using even-numbered time slots are illustrated in Figure 1(c). To simplify our presentation, we only discuss the time-slot reservation scheme for even-numbered time slots to indicate the fact that each traffic pair adds the POLL or ACK, packets.

2.2. Basic Idea of the CQ Protocol

Lower bandwidth utilization of each Bluetooth device is the drawback of Kim et al.'s approach [17], because their approach mainly attempts to improve the success rate of searching for a QoS route by only utilizing DM1 packets. Using only DM1 packets leads to the problem of lower bandwidth utilization. For simplicity, we only describe the case of handling DM-type packets in our CQ protocol. Similar operations can be applied to DH-type packets. Furthermore, Kim et al.'s approach [17] is only suitable for tree-structure scatternets [7]. Lower success rates of searching for QoS routes are obtained with non-tree-structure scatternets. To improve the bandwidth utilization and success rate, a new QoS scheduling scheme was investigated which not only adopts DM1 packets but also uses DM3 and DM5 packets for the QoS scheduling. The basic idea of this protocol is to take advantage from the different types of Bluetooth packets that allow obtaining different bandwidth utilization levels, in contrast with previous works in which only DM1 packets are used, deriving in a lower bandwidth utilization.

Let $\{\alpha_1, \alpha_2, \ldots, \alpha_k\}$ denote a free time slot list of a Bluetooth device. For example, given sub-path (B,g,a) as shown in Figure 2(a), two DM1 packets at time slots (2, 3) and (14, 15) are used in link $\overleftrightarrow{Bg}$, and one DM1 packet at time slot (10, 11) is used in link $\overleftrightarrow{ga}$, where g is a bridge device and free time slot lists of *B, g*, and *a* are {0, 1, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13}, {0, 1, 4, 5, 6, 7, 8, 9, 12, 13}, and {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 14, 15}, respectively. Before describing our basic idea, we point out the result of Kim et al.'s scheme by giving an example. In the example, we find a QoS route (S, e, B, g, a, D) with the QoS requirement of transmitting 51 bytes. Figure 2(b) shows the initial bandwidth utilization of the Bluetooth scatternet. The divide-and-conquer approach in [17] repeatedly splits the *traffic-load* matrix into *traffic-load* sub-matrices until the sub-matrix contains only DM1 packets. This work guarantees the contention-free of the time slot reservation. Only DM1 packets are used to construct a QoS route. For example as shown in Figure 3, three DM1 packets, (2, 3), (8, 9), and (14, 15), are used in link $\overleftrightarrow{Se}$, three DM1 packets, (0, 1), (4, 5), and (10, 11), are used in link $\overleftrightarrow{eB}$, and three DM1 packets, (6, 7), (8, 9), and (12, 13) are used in link $\overleftrightarrow{Bg}$. But since it cannot find three DM1 packets in link $\overleftrightarrow{ga}$, therefore it failed to search for a QoS route.

To transmit 51 bytes, we can use three DM1 packets (six time slots) or one DM3 packet (four time slots) from Table 1. Our main idea is to use DM5, DM3, and DM1 packets to achieve the goal of using fewer free time slots. This work mainly improves the bandwidth utilization and increases the success rate when searching for a QoS route. Given the same scenario as in Figure 2, the QoS scheduling is failed as shown in Figure 3. Figure 4 gives a successful QoS scheduling result using our new scheduling scheme. In this example, a DM3 packet (0, 1, 2, 3) is used in link $\overleftrightarrow{Se}$, a DM3 packet (8, 9, 10, 11) is utilized in link $\overleftrightarrow{eB}$, three DM1 packets, (0, 1), (4, 5), and (12, 13), are used in link $\overleftrightarrow{Bg}$, and the DM3 packet (6, 7, 8, 9) is occupied in link $\overleftrightarrow{ga}$. The DM3 packet (6, 7, 8, 9) is finally used in link $\overleftrightarrow{aD}$. Therefore, a QoS route (S, e, B, g, a, D) with the QoS requirement of transmitting 51 bytes is successfully constructed.

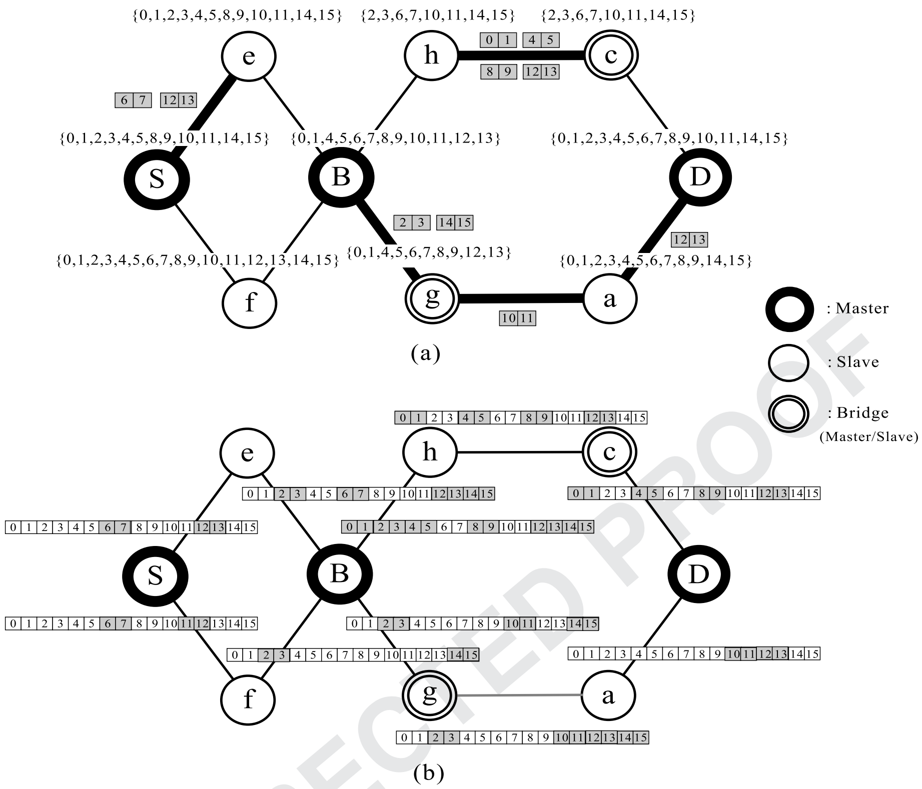


Figure 2. Bandwidth usage of a Bluetooth scatternet.

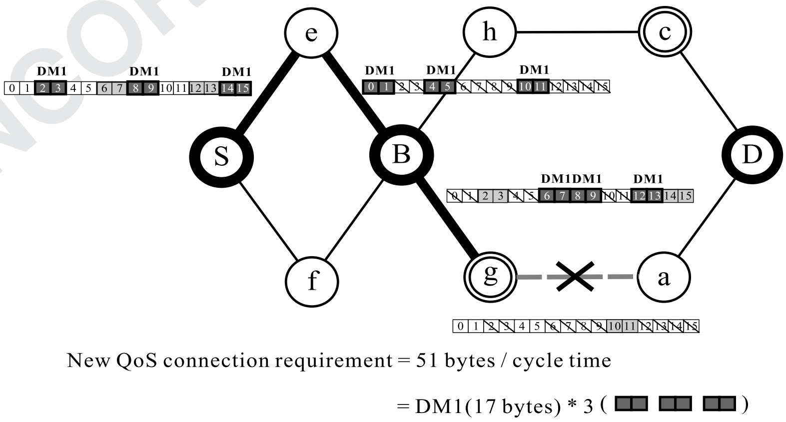


Figure 3. QoS scheduling result using Kim et al.'s algorithm.

It is mentioned in Table 1 that the bandwidth utilization of different packet types greatly differs. Packets with different types of support for QoS requests produce different bandwidth utilization levels. When a slave retains data to be transmitted, the slave must wait for a POLL packet sent from the master during a polling interval. If a scheduling method only uses DM1 packets for the QoS requirement, then the master wastes a great number of time slots when sending the POLL packet. This obviously causes the problem of low bandwidth

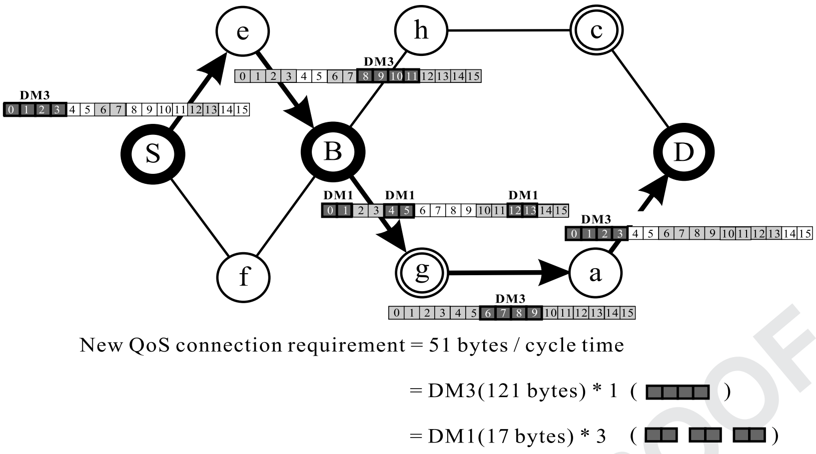


Figure 4. QoS scheduling result using our scheme.

utilization and degradation of the success rate. Developing a new scheduling scheme by adopting DM1, DM3, and DM5 packets is the key idea of our work. From Johansson et al.'s [18] investigation, the scheduling problem of finding the minimum assigned time slots in Bluetooth scatternets is an NP-complete problem. Some scheduling schemes were developed by Chen et al. [2, 3] for mobile ad hoc networks (MANETs). Chen et al. [2, 3] investigated the hidden-terminal problem between each mobile node when searching for a QoS route in MANETs. The key difference between developing QoS routes in MANETs and in Bluetooth scatternets is that a Bluetooth device cannot participate in more than one piconet at the same time. Efforts are made to develop a new, efficient QoS scheduling mechanism for Bluetooth scatternets. In the following, we propose centralized and distributed algorithms to develop efficient QoS scheduling in Bluetooth scatternets. The objectives of this work were not only to improve the bandwidth utilization but also increase the success rate of constructing a QoS route.

3. A Centralized On-Demand QoS Routing Protocol

To optimize the bandwidth utilization of a QoS route in a Bluetooth WPAN, a centralized QoS routing protocol is presented in this section. The main contribution of this work is to develop the centralized algorithm. The centralized algorithm is used to construct a QoS route which satisfies the QoS requirement, from a source node to a destination node, over a preformed Bluetooth scatternet [4–7]. A multi-hop scatternet is initially constructed by existing scatternet formation algorithms [4–7]. Each bridge device between two piconets in a scatternet maintains two different piconet clocks and corresponding hopping sequences to support the reasonable assumptions of the QoS time-slot reservations [4–7].

To improve the system performance, we developed a new and more-efficient QoS routing protocol by taking the factor of different-type packets with different bandwidth levels of utilization into account. To achieve high bandwidth utilization and high success rates of finding a QoS route, the centralized on-demand QoS routing protocol uses the development of free time-slot information collection and time-slot reservation phases. In the free time-slot information collection phase, many different paths, from a source node, with all free time-slot information of all links, are received at the destination node. In the time-slot reservation phase, both credit-based and optimal algorithms of time-slot reservation are given.

3.1. Phase I: Free Time-Slot Information Collection

A Bluetooth scatternet is assumed to initially be formed by existing formation protocols [4, 5, 6, 7]. The detailed collection of free time-slot information from source to destination nodes is performed. The source node initiates the QoS_REQuest, or BQ_REQ, packet and floods into Bluetooth scatternets until the BQ_REQ packets arrive at the destination node. Each BQ_REQ packet records all free time-slot information of links along a path from the source node to the destination node. The destination node receives information on many different paths, therefore a sub-graph with useful free time-slot information from the source node to the destination node is rebuilt at the destination node. Utilizing the sub-graph with useful free time-slot information allows us to develop near-optimal and optimal time-slot reservations.

In the following, we describe how to calculate free time slots between two adjacent nodes in Bluetooth scatternets. Let $\{\alpha_1, \alpha_2, \ldots, \alpha_k\}$ denote a free time-slot set for a Bluetooth device in a Bluetooth scatternet. For instance as shown in Figure 5(a), {0, 1, 2, 3, 4, 5, 8, 9, 10, 11, 14, 15} is the free time-slot set of node *S*. Given a pair of adjacent nodes, *A* and *B*, free time-slot sets of *A* and *B* are $\{\alpha_1, \alpha_2, \ldots, \alpha_{k_1}\}$ and $\{\beta_1, \beta_2, \ldots, \beta_{k_2}\}$ where $k_1 \neq k_2$. As mentioned in Section II, $\overleftrightarrow{AB}$ denotes the link between adjacent nodes *A* and *B*. An intersection function, $\cap(\{\alpha_1, \alpha_2, \ldots, \alpha_{k_1}\}, \{\beta_1, \beta_2, \ldots, \beta_{k_2}\}) = \{\gamma_1, \gamma_2, \ldots, \gamma_{k_3}\}$, is executed for link $\overleftrightarrow{AB}$ to calculate the shared free time slots of nodes *A* and *B*, where $\{\gamma_1, \gamma_2, \ldots, \gamma_{k_3}\} \in \{\alpha_1, \alpha_2, \ldots, \alpha_{k_1}\}, \{\beta_1, \beta_2, \ldots, \beta_{k_2}\}$, and $k_3 \leq \min(k_1, k_2)$. For example as illustrated in Figure 5(b), the free time-slot list of link $\overleftrightarrow{Bg}$ is {0, 1, 4, 5, 6, 7, 8, 9, 12, 13} = ∩({0, 1, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13}, {0, 1, 4, 5, 6, 7, 8, 9, 12, 13}). The BQ_REQ packet is defined as BQ_REQ (S_ADDR, D_ADDR, FT, PL, FTSL, BR, TTL), where the detailed definition is given in Table 2. The algorithm of free time-slot information collection is given below.

(A1) Source node *S* initiates and floods a BQ_REQ(S_ADDR = *S*, D_ADDR = *D*, FT = $\{\alpha_1, \alpha_2, \ldots, \alpha_{k_1}\}$, PL={}, FTSL={}, BR, TTL) packet into a Bluetooth scatternet, where *D* is the destination node, BR is the QoS requirement, and TTL is the time-to-live value.

(A2) If node *e* receives a BQ_REQ(S_ADDR= *S*, D_ADDR= *D*, current_FT, current_PL, current_FTSL, BR, current_TTL) packet from node e' in the Bluetooth scatternet, the current_TTL and D_ADDR fields are checked, and four cases are considered.

Table 2. Detailed definition of a BQ_REQ packet

Packet field	Field description
S_ADDR	Source node address
D_ADDR	Destination node address
FT	Free time slots of the current node
PL	List of node information that records the path from the source to the current traversed node
FTSL	Free time-slot list of links, each of which records the shared free time siots among the current traversed node and the last node recorded in the PL
BR	QoS requirement of the source host
TTL	Time to live: limitation of the hop-length in the search path

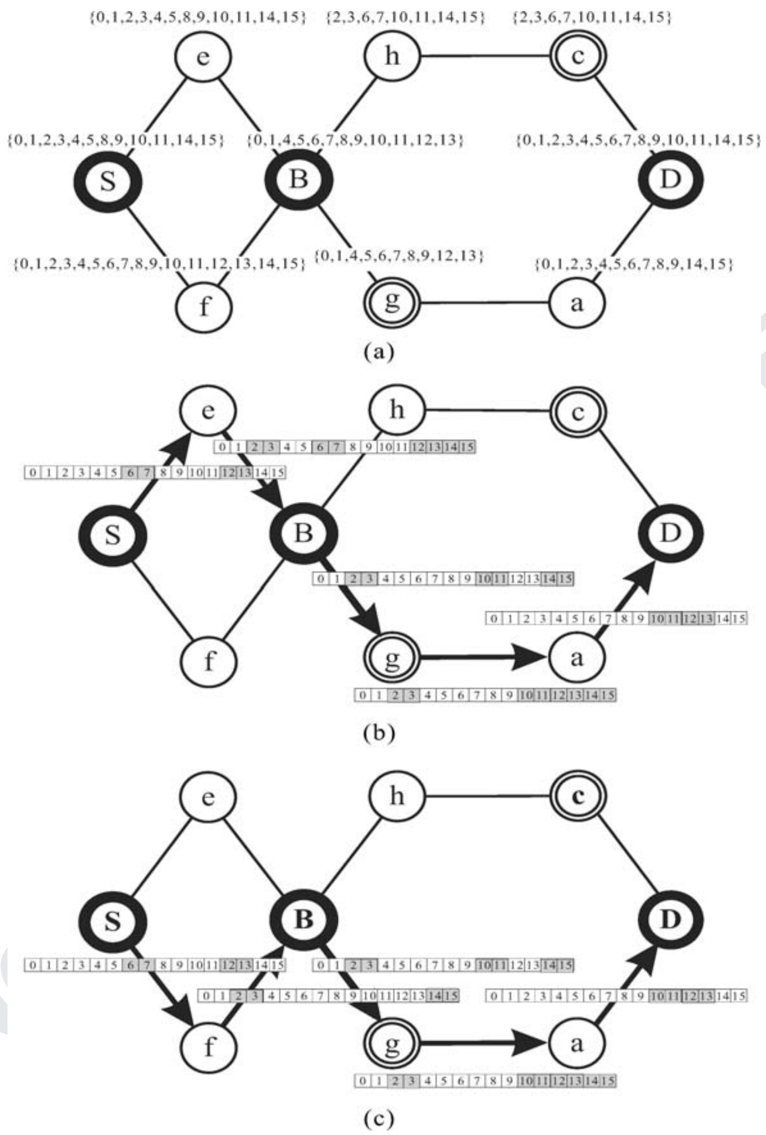


Figure 5. Free time slots of each node and the shared free time-slot list of each link.

(B1) If current_TTL is equal to zero and node e is not equal to D_ADDR of the BQ_REQ packet, then the current BQ_REQ packet is dropped.

(B2) If the shared free time slots $\{\gamma_1, \gamma_2, \ldots, \gamma_{k_3}\}$ of link $\overleftrightarrow{e'e}$ cannot satisfy the QoS requirement, BR, then the current BQ_REQ packet is dropped.

(B3) If node e is equal to D_ADDR, then go to step A3.

(B4) Node e appends e into the current_PL field and adds $\{\gamma_1, \gamma_2, \ldots, \gamma_{k_3}\}$ into the current_FTSL field, and decreases the value of the current_TTL. Node e floods BQ_REQ(S, D, itself_FT, {current_PL, e}, current_FTSL$\cup\{\gamma_1, \gamma_2, \ldots, \gamma_{k3}\}$, BR, current_TTL - 1) into the Bluetooth scatternet, where itself_FT is the free time-slot list of node e.

(A3) Destination node D waits for a period of time to receive many different BQ_REQ packets from the source node.

For example as shown in Figure 5(b)(c), BQ_REQ(S, D, $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 14, 15\}$, $\{S, e, B, g, a, D\}$, [$\{0, 1, 2, 3, 4, 5, 8, 9, 10, 11, 14, 15\}$, $\{0, 1, 4, 5, 8, 9, 10, 11\}$, $\{0, 1, 4, 5, 6, 7, 8, 9, 12, 13\}$, $\{0, 1, 4, 5, 6, 7, 8, 9\}$], 51, 0) and BQ_REQ($S$, D, $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 14, 15\}$, $\{S, f, B, g, a, D\}$, [$\{0, 1, 2, 3, 4, 5, 8, 9, 10, 11, 14, 15\}$, $\{0, 1, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13\}$, $\{0, 1, 4, 5, 6, 7, 8, 9, 12, 13\}$, $\{0, 1, 4, 5, 6, 7, 8, 9\}$], 51, 0) packets are received at destination node *D*.

3.2. Phase II: Time-Slot Reservation

To reserve time slots for constructing a QoS route with better and optimal bandwidth utilization, two centralized algorithms, (1) a credit-based algorithm and (2) an optimal algorithm, are presented as follows.

3.2.1. *Credit-Based Algorithm*

Packets comprised of different packet types used for the same QoS requirement result in different bandwidth utilization levels. To assign time slots to each link while considering different packet types, a better QoS route with high bandwidth utilization is presented. First, each link is assigned a different priority value. This value indicates the degree of influence with that link has with its neighboring links. A credit-based algorithm is developed based on the priority value of finding a QoS route with lower influence by neighboring links. The detailed description follows.

After collecting many BQ_REQ packets from a source node, all free time-slot information is obtained at the destination node. The destination node chooses one of them and performs the following operation. Without loss of generality, a path $(s_0, M_1, s_1, M_2, \ldots, M_i, s_i)$ is chosen where the source and destination nodes are s_0 and s_i. A *shared free-time slot* matrix, M_f, is used to indicate information of the shared free time slots of links $\overleftrightarrow{s_0M_i}$, $\overleftrightarrow{M_1s_1}$, $\overleftrightarrow{s_1M_2}$, ..., and $\overleftrightarrow{M_is_i}$ of route $(s_0, M_1, s_1, M_2, \ldots, M_is_i$. Each row of matrix M_f denotes the shared free time slots of links s_0M_1, M_1s_1, s_1M_2, ..., and M_is_i. For instance, consider route (S,e,B,g,a,D) as illustrated in Figure 5(b), matrix M_f is constructed of links $L(S, e)$, $L(e, B)$, $L(B, g)$, $L(g, a)$, and *L(a,D)* as illustrated in Figure 8(a). The first row of matrix M_f is $\{0, 1, 2, 3, 4, 5, 8, 9, 10, 11, 14, 15\}$ for *L(S,e)*. Let $F(X, Y)$ and $B(X, Y)$ respectively denote the free time slots and busy time slots of link $L(X, Y)$ or $\overleftrightarrow{XY}$. For example as shown in Figure 6, $F(e, B) = \{0, 1, 4, 5, 8, 9, 10, 11\}$ and $B(e, B) = \{2, 3, 6, 7, 12, 13, 14, 15\}$.

Given route $(s_0, M_1, s_1, M_2, \ldots, W, X, Y, Z \ldots, M_i, s_i)$, we consider three adjacent links, $\overleftrightarrow{WX}$, $\overleftrightarrow{XY}$, and $\overleftrightarrow{YZ}$ along the route. Number list $P(\delta)_{L(X,Y),L(Y,Z)}$ is constructed by $P(\delta_i)_{L(X,Y),L(Y,Z)}$ where $0 \le i \le$ polling_interval. Each $P(\delta_i)_{L(X,Y),L(Y,Z)}$ denotes a credit value of i-th time slots for links $L(X, Y)$ and $L(Y, Z)$ as follows.

$$P(\delta_i)_{L(X,Y),L(Y,Z)} = \begin{cases} 0, & \text{if } \delta_i \in B(X,Y), \\ 1, & \text{if } \delta_i \in F(X,Y) \cap F(Y,Z), \quad \text{where} \\ & \quad 0 \le i < \text{polling\_interval} \\ 2, & \text{otherwise;} \end{cases} \tag{1}$$

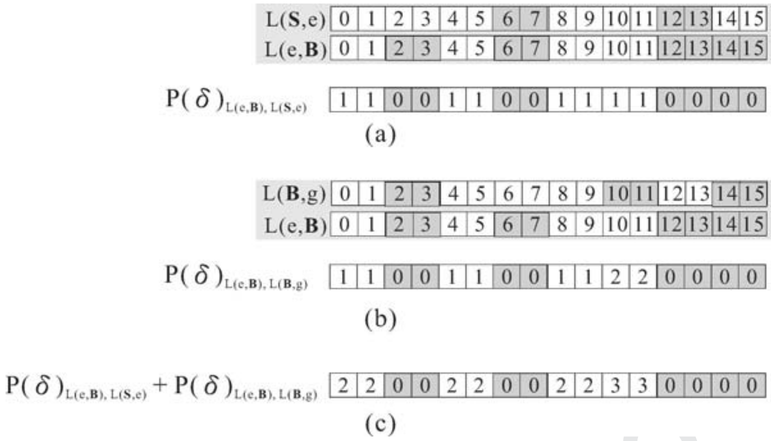


Figure 6. Example of the priority of the time-slot operation.

Similarly, number list $P(\delta)_{L(X,Y),L(W,X)}$ is used to denote the credit values of links $L(X, Y)$ and $L(W, X)$, where every $P(\delta_i)_{L(X,Y),L(W,X)}$ is defined below.

$$P(\delta_i)_{L(X,Y),L(W,X)} = \begin{cases} 0, & \text{if } \delta_i \in B(X, Y), \\ 1, & \text{if } \delta_i \in F(X, Y) \cap F(W, X), \quad \text{where} \\ & \quad 0 \leq i < \text{polling\_interval} \\ 2, & \text{otherwise;} \end{cases} \tag{2}$$

For instance as shown in Figure 6(a) and (b), $P(\delta)_{L(e,B),L(B,g)}$=1100110011220000 and $P(\delta)_{L(e,B),L(S,e)}$=1100110011110000. Each number in the number list of $P(\delta_i)_{L(X,Y),L(Y,Z)}$ and $P(\delta_i)_{L(X,Y),L(W,X)}$ ranges from 0 to 2, where '0' denotes that the i-th time slot of $\overleftrightarrow{XY}$ is busy, '1' denotes that the i-th time slots of $\overleftrightarrow{XA}$ and $\overleftrightarrow{WX}$ are free, and '2' denotes that the i-th time slot of $\overleftrightarrow{XA}$ is free but the i-th time slot of $\overleftrightarrow{WX}$ cannot be used. The value of '2' is the case of low influence of neighboring nodes. Therefore, the value of '2' has the highest priority. A similar rule can be applied to $P(\delta_i)_{L(X,Y),L(W,X)}$. Finally, a sum of the number list is calculated by $P(\delta)_{L(X,Y),L(Y,Z)} + P(\delta)_{L(X,Y),L(W,X)}$, where each number in the number list ranges from 0 to 4. For instance, $P(\delta)_{L(e,B),L(S,e)} + P(\delta)_{L(e,B)(B,g)}$=2200220022330000. The higher value in the sum number list is picked first because it has a lower influence capability.

Let p denote the permutation of any packet types to satisfy the QoS requirement. For example as shown in Figure 7, the QoS requirement is 224 bytes per cycle time, and four different packet types are produced, i.e., $p = 4$; (1) a DM5 packet, (2) two DM3 packets, (3) one DM3 packet and eight DM1 packets, and (4) 16 DM1 packets. For the other example shown in Figure 8(a), $p = 2$ for the QoS requirement of 51 bytes for $L(e, B)$ and one DM3 packet and three DM1 packets are produced.

With $P(\delta)_{L(e,B),L(S,e)} + P(\delta)_{L(e,B),L(B,g)}$, there are m conditions of time reservation if we only consider one kind of packet type to satisfy the QoS requirement. For the same example of $L(e, B)$ shown in Figure 8(a), if we consider one DM3 packet, it can be reserved for time slot (8, 9, 10, 11). Therefore, $m = 1$. If we consider three DM1 packets, they can be reserved on (0, 1), (4, 5), and (8, 9) or (0, 1), (4, 5) and (10, 11). Therefore, $m = 2$. Therefore, a computation time on $O(m \cdot p)$ is needed for the one-hop time reservation.

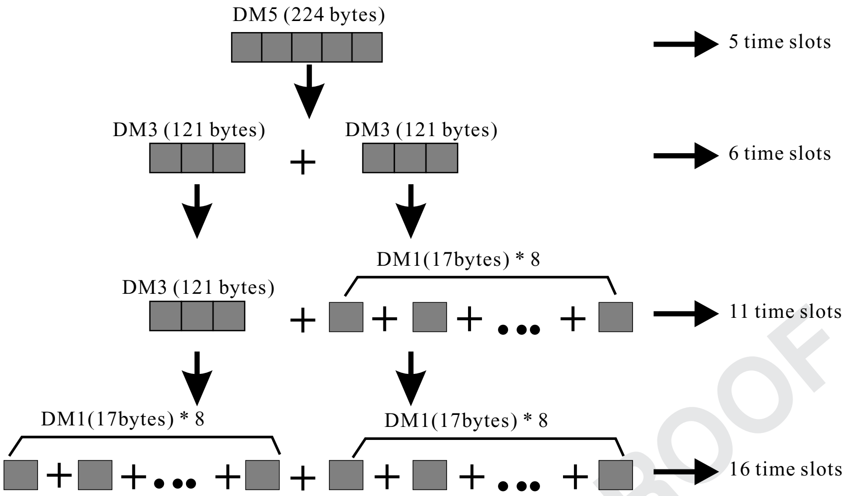


Figure 7. Permutation of each packet type.

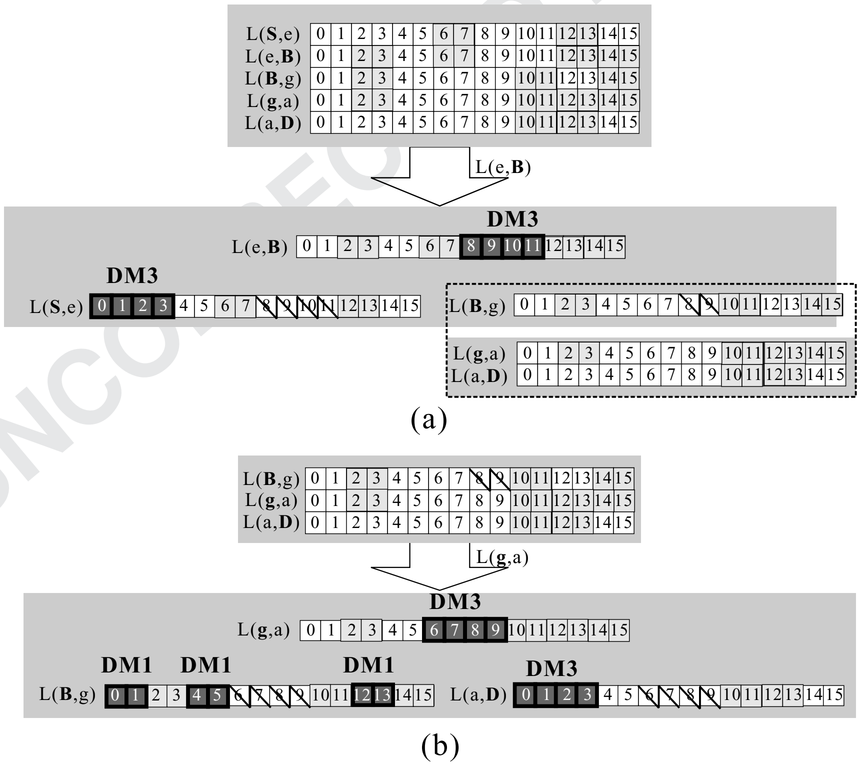


Figure 8. Time slots reserved step-by-step by the CCQ approach.

Given path $(s_0, M_1, s_1, M_2, \ldots, W, X, Y, Z, \ldots, M_i, s_i)$, if we have matrix M_f and information on the summed number lists $P_{(\delta)L(e,B),L(S,e)} + P(\delta)_{L(e,B),L(B,g)}$ then time-slot reservation is given as follows.

(C1) Link $L(X, Y)$ is selected from $(s_0, M_1, s_1, M_2, \ldots, W, X, Y, Z, \ldots, M_i, s_i)$ with a lower number of shared free time slots, such that route $(s_0, M_1, s_1, M_2, \ldots, W, X, Y, Z, \ldots,$

M_i, s_i) can be divided into two subpaths, ($s_0, M_1, s_1, M_2, \ldots, W, X$) and ($Y, Z, \ldots, M_i, s_i$), with two sub-matrices, M'_f and M''_f, where $M_f = M'_f + M''_f$. If there is more than one link with the same fewer number of free time slots, then we randomly select one link from them.

(C2) With the QoS requirement, we first try possible DM5 packets to satisfy the QoS requirement of link $L(X, Y)$. If these do not satisfy the QoS requirement, we continue to try possible DM3 packets to satisfy the QoS requirement. Then, if the QoS requirement is still not satisfied, we continue to try possible DM1 packets to satisfy it. All of the above operations depend on the priority of the summed number lists of $P(\delta)_{L(X,Y),L(Y,Z)} + P(\delta)_{L(X,Y),L(W,X)}$.

(D1) The time-slot reservation operations of steps C1 and C2 on sub-path ($s_0, M_1, s_1, M_2, \ldots, W, X$) are recursively performed with sub-matrix M'_f.

(D2) The time-slot reservation operations of steps C1 and C2 on sub-path ($Y, Z, \ldots, M_i, S_i$) are recursively performed with sub-matrix M''_f.

For instance as shown in Figure 8(a), a route (S, e, B, g, a, D) with M_f is split into two sub-paths, (S, e) and (B, g, a, D), with M'_f and M''_f. since $\overleftrightarrow{eB}$ has eight free time slots. The QoS requirement is 51 bytes per cycle time. As illustrated in Figure 8(a), DM3 is reserved to $L(e, B)$, and DM3 is recursively reserved to $L(S, e)$. Sub-matrix M''_f for (B, g, a, D) is split into (B, g) and (a, D) after DM3 is allocated to $L(g, a)$. Finally, three DM1 packets are reserved in $L(B, g)$ and one DM3 is allocated to $L(a, D)$.

After performing the time-slot reservation operation, the destination node replies with a REPly (RREP) packet from the destination node to the source node to reserve time slots with the QoS requirement and which releases all unrelated time slots in all other routes. The time complexity of the credit-based algorithm is given.

Lemma 1. *If n hops exist in a route from a source to the destination, then the time complexity of the credit-based algorithm is $O((m \cdot p)^n)$, where m is the number of all free time slots that can be used for a traffic pattern, and p is the permutation number of all traffic patterns.*

Proof: As there are n hops in a route, then our matrix is constructed from n rows. Therefore, the time complexity of the credit-based algorithm is $O((m \cdot p)^n)$.

□

3.2.2. *Optimal Algorithm*

To provide an optimal solution for constructing a QoS route, an optimal algorithm is presented as follows. Given route ($s_0, M_1, s_1, M_2, \ldots, W, X, Y, Z \ldots, M_i, s_i$) and matrix M_f, the optimal time slot reservation is given.

(E1) Every link $L(X, Y)$ is selected from ($s_0, M_1, s_1, M_2, \ldots, W, X, Y, Z \ldots, M_i, s_i$) by splitting ($s_0, M_1, s_1, M_2, \ldots, W, X, Y, Z \ldots, M_i, s_i$) with matrix M_f into the two sub-paths, ($s_0, M_1, s_1, M_2, \ldots, W, X$) and ($Y, Z, \ldots, M_i, s_i$), with the two sub-matrices, M'_f and M''_f, where $M_f = M'_f + M''_f$.

(E2) A credit-based scheme is applied to reserve time slots on link $L(X, Y)$. This work takes a computation time of $O(m \cdot p)$.

(E3) The time-slot reservation operations of steps E1 and E2 are recursively carried out on sub-path ($s_0, M_1, s_1, M_2, \ldots, W, X$) with sub-matrix M'_f until all links in sub-path ($s_0, M_1, s_1, M_2, \ldots, W, X$) have been selected for the time-slot reservation.

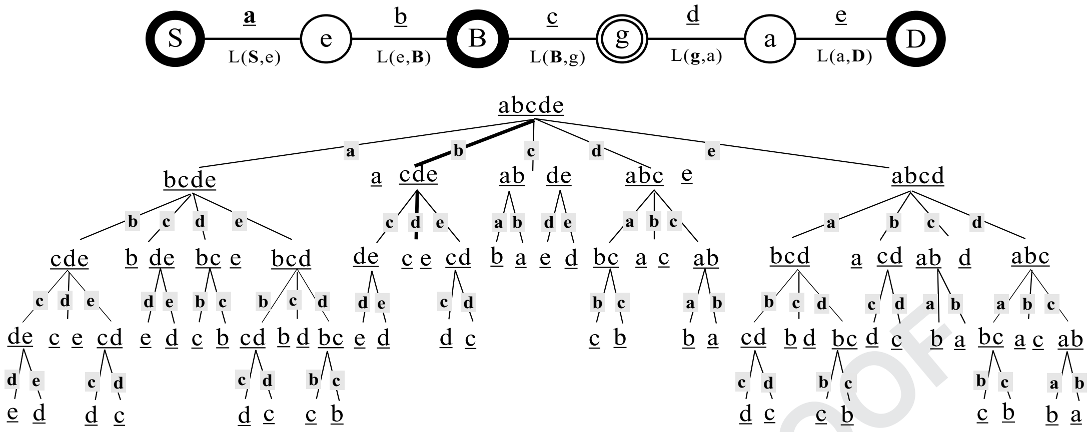


Figure 9. Examples of time-slot reservation in link $L(e, B)$.

(E4) The time slot reservation operations of steps E1 and E2 are recursively carried out on sub-path $(Y, Z \ldots, M_i, s_i)$ with sub-matrix M_f'' until all links in sub-path $(Y, Z, \ldots, M_i, s_i)$ have been selected for time-slot reservation.

All of the above recursive operations can be represented as a traversal tree as shown in Figure 9. This tree is named the time-slot reservation tree. For example as shown in Figure 9, to easily express the reservation operation, *a, b, c, d*, and *e* are used to indicate links $L(S, e)$, $L(e, B)$, $L(B, g)$, $L(g, a)$, and $L(a, d)$, respectively. The root of the tree is represented as <u>*abcde.*</u> After choosing links <u>*a*</u>, <u>*b*</u>, <u>*c*</u>, <u>*d*</u>, and *e* for the time-slot reservations, the children nodes of the root, <u>*bcde*</u>, <u>a cde</u>, <u>ab de</u>, <u>abc e</u>, and <u>*abcd*</u>, form the first level of the tree. The tree is constructed by recursively expanding all possible children nodes of every node on each level of the tree. Every path from the root to a leaf node produces a time-slot reservation pattern. For instance, the leftmost path, comprised of links <u>*a*, *b*, *c*</u>, and <u>*d*</u>, is a time-slot reservation pattern. The second one is comprised of links <u>*a*</u>, <u>*b*</u>, <u>*c*</u>, and *e*. For instance, detailed time-slot reservations for selecting links <u>*b*</u> and $d(L(e, B)$ and $L(g, a))$ are given in Figs. 10 and 11.

The total number of nodes of the tree is $n!$, where the hop number of the original route is n. Therefore, the time complexity of the optimal algorithm is $O(m \cdot p)^{n!})$.

Lemma 2. *If n hops exist in a route from a source to the destination, then the time complexity of the optimal algorithm is $O((m \cdot p)^{n!})$, where m is number of all free time slots that can be used for a traffic pattern and p is permutation number of all traffic patterns.*

However, the centralized QoS on-demand routing protocol suffers from a scalability problem in that all route paths and their time-slot information are stored in BQ_REQ packets. To reduce the scalability problem, a simple distributed QoS on-demand QoS routing protocol based on the centralized credit-based (CCQ) algorithm was developed in Section 4.

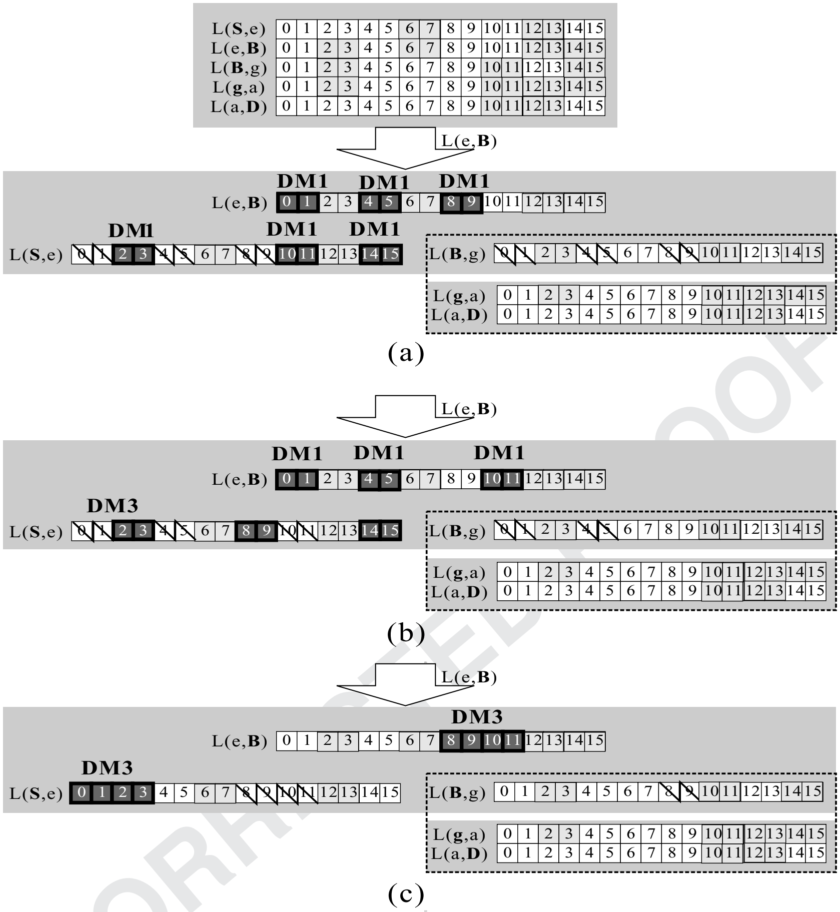


Figure 10. All conditions of time slot reservation of the optimal algorithm.

4. A Distributed On-Demand QoS Routing Protocol

This section presents a distributed credit-based QoS (DCQ) routing protocol. The DCQ protocol is directly modified from the CCQ protocol. In the DCQ protocol, a hop-by-hop distributed algorithm is designed by adopting the advantages of the credit-based algorithm.

The DCQ protocol floods a BQ_REQ packet every three hops and performs a time-slot reservation operation. The time-slot reservation packet (TSRP) is sent back through the preceding nodes to confirm the time-slot reservation. The detail definition of the time-slot reservation packet (TSRP) is given in Table 3. The above operation is repeatedly executed until arriving at the destination node; in this way, a QoS route is constructed.

In our DCQ algorithm, an extra field is appended, hop counter (HC), which limits the current BQ_REQ packet to three hops. The new BQ_REQ packet is redefined as BQ_REQ(S_ADDR, D_ADDR, FT, PL, FTSL, BR, HC, TTL). The TSRP is defined as TSRP(S_ADDR, D_ADDR, BQPL, TSRL, TTL). The DCQ algorithm is formally given as follows.

(F1) Source node S initiates and floods the BQ_REQ(S_ADDR $=S$, D_ADDR = D, FT $=\{\alpha_1, \alpha_2, \ldots, \alpha_{k_1}\}$, PL $=\{\}$, FTSL $=\{\}$, BR, HC $=1$, TTL) packet in a Bluetooth scatternet,

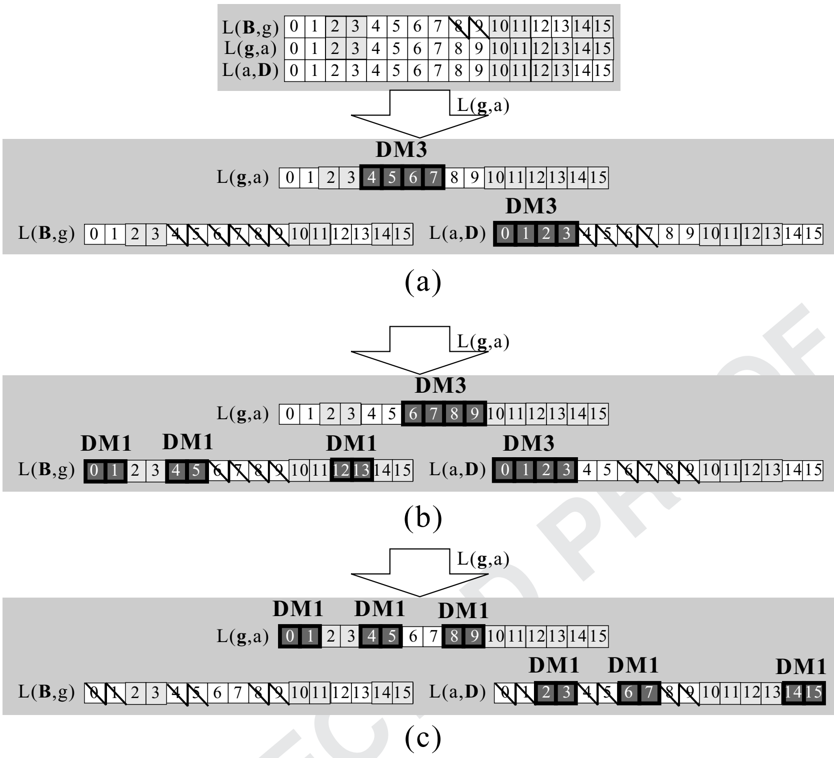


Figure 11. Examples of time-slot reservation in link $L(g, a)$.

where D is the destination node, and HC is the hop counter.

(F2) When node e receives a BQ_REQ(S_ADDR $= S$, D_ADDR $= D$, current_FT, current_PL, current_FTSL, BR, current_HC, current_TTL) packet from node e' in a Bluetooth scatternet, after determining the value for current_FTSL, BR, current_HC, and current_TTL, some situations are evaluated.

- (G1) If the TTL is equal to zero and node e is not equal to D_ADDR of the BQ_REQ packet, then the current BQ_REQ packet is dropped.
- (G2) If the shared free time slots $\{\gamma_1, \gamma_2, \ldots, \gamma_{k_3}\}$ of link $\overleftrightarrow{e'e}$ cannot satisfy the QoS requirement, BR, then the current BQ_REQ packet is dropped.
- (G3) If the HC is smaller than three, node e calculates the shared free time slots $\{\gamma_1, \gamma_2, \ldots,$

Table 3. Detail definition of a TSRP packet

Packet field	Field description
S_ADDR	Current node address
D_ADDR	Preceding third node address acquired from the PL of the BQ_REQ packet
BQPL	Path list with information of the preceding three hops acquired from the PL of the BQ_REQ packet
TSRL	All information of reserved time slots for the preceding three nodes
TTL	Time to live: hop-length within three hops

$\gamma_{k_3}\}$ of link $\overleftrightarrow{e'e}$, then adds the value to the field of FTSL. Moreover, the value of the HC is increased and the other fields of the BQ_REQ, such as the PL and TTL are updated. Finally, the BQ_REQ(S, D, itself_FT, $\{$current_PL, e$\}$, current_FTSL $\cup\{\gamma_1, \gamma_2, \ldots, \gamma_{k3}\}$, BR, current_HC+1, current_TTL-1) packet is reconstructed and forwarded to all neighboring nodes.

(G4) If the HC is equal to three, then node e performs steps F3 and F4 to execute the time-slot reservation.

(F3) Node e acquires all shared free time-slot lists from the FTSL in the BQ_REQ. Matrix M_f is constructed by three adjacent links: $L(W, X)$, $L(X, Y)$, and $L(Y, Z)$.

(H1) To select link $L(X, Y)$, matrix M_f is split into two submatrices, M'_f and M''_f, where $M_f = M'_f + M''_f$, M'_f contains only link $L(W, X)$, and M''_f contains only link $L(Y, Z)$.

(H2) The credit value of $P(\delta)_{L(X,Y),L(Y,Z)} + P(\delta)_{L(X,Y),L(W,X)}$ is calculated to perform the credit-based time-slot reservation.

(F4) After successful time-slot reservation, the following operations are executed.

(I1) Node e recalculates the shared free time slots $\{\gamma_1, \gamma_2, \ldots, \gamma_{k3}\}$ of link $\overleftrightarrow{e'e}$. Then, node e reconstructs the BQ_REQ(S_ADDR= S, D_ADDR= D, itself_FT, $\{$current_PL, e$\}$, $\{\gamma_1, \gamma_2, \ldots, \gamma_{k3}\}$, BR, HC= 2, current_TTL$-1$) packet and forwards it to the next hop until the destination node receives the BQ_REQ packet or the TTL is equal to zero.

(I2) Node e constructs a TSRP(S_ADDR, D_ADDR, BQPL, TSRL, TTL) which contains the result of the time-slot reservation for the preceding three nodes. Node e sends the TSRP back through the preceding three nodes to confirm the time-slot reservation.

For example as shown in Figure 12(a), node g receives the BQ_REQ packet and performs the time-slot reservation since the value of HC is 3. The distributed time-slot reservation for (S,e,B,g) is executed as illustrated in Figure 13(a). After that, node D receives the BQ_REQ packet from g as shown in Figure 12(b), and a distributed time-slot reservation for (B, g, a, D) is again executed as shown in Figure 13(b). The time complexity of the DCQ algorithm is given.

Lemma 3. *If n hops exist in a route from a source to the destination, then the time complexity of the DCQ algorithm is $O((m \cdot p)^n)$, where m is number of all free time slots that can be used for a traffic pattern and p is the permutation number of all traffic patterns.*

Proof: The local three-hop computation time is $O((m \cdot p)^2)$. The total number of the time slots reserved is about $\frac{n}{2}$. Therefore, the total computation time of the DCQ algorithm is $O((m \cdot p)^n)$. □

5. Performance Analysis

Our study mainly presents a new credit-based time-slot reservation protocol. To evaluate our credit-based protocol and Kim et al.'s protocol [17], we implemented them using the Network

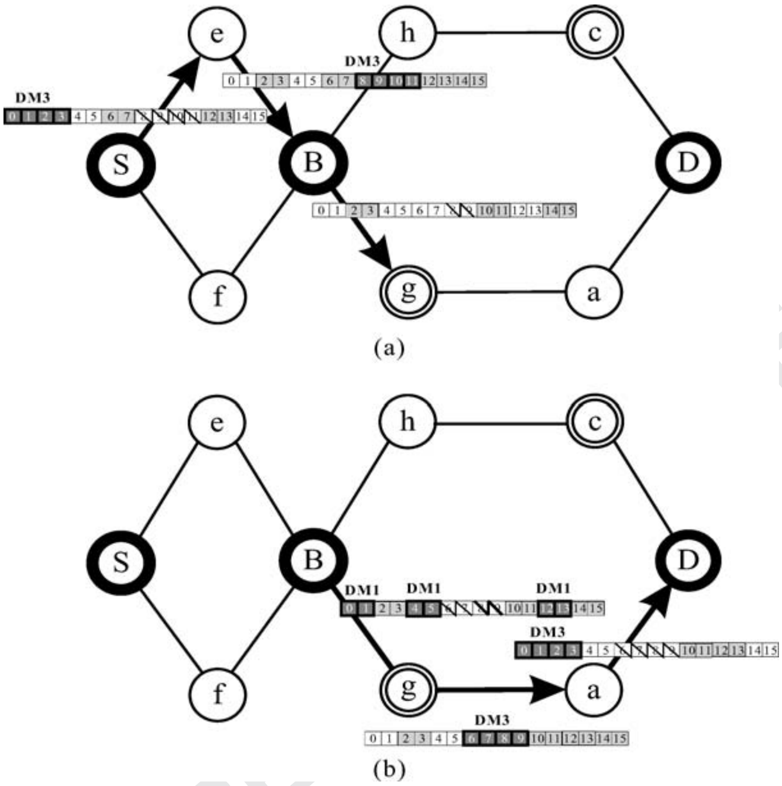


Figure 12. QoS route-discovery operation using the DCQ approach.

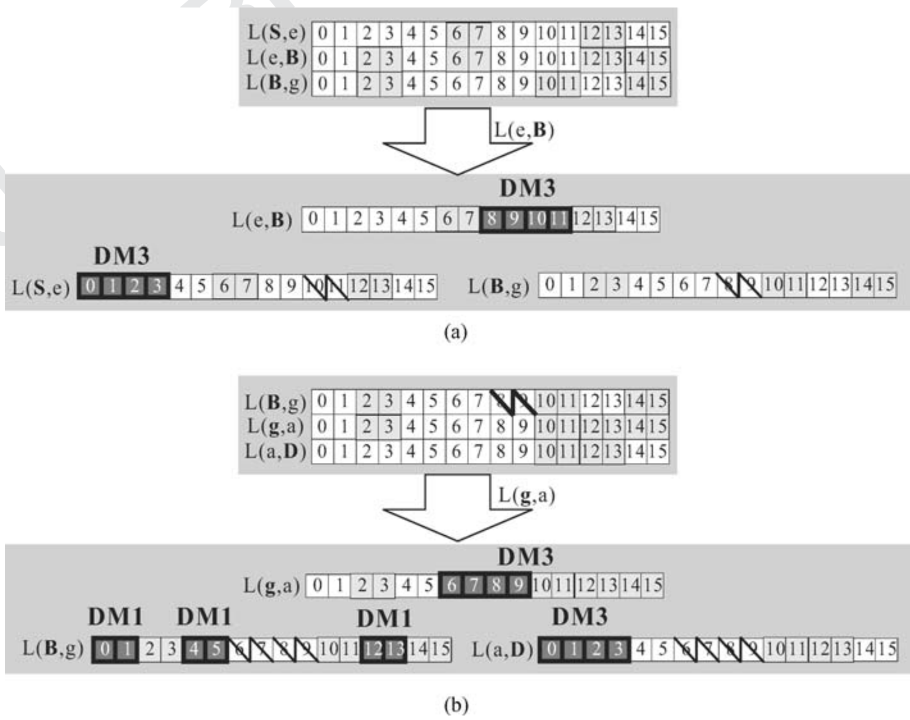


Figure 13. Time slots reserved step-by-step using the DCQ approach.

Table 4. Detailed simulation parameters

Parameter	Value
Number of Bluetooth devices	18
Network region	17×17 m^2
Radio propagation range	10 m
Mobility	No
Cycle time	16 time slots
Packet type	DM1 or DM3 or DM5
QoS requirements	*DM*1: *DM*3: *DM*5 = 1: 1: 1
	*DM*1: *DM*3: *DM*5 = 3: 1: 1
	*DM*1: *DM*3: *DM*5 = 1: 3: 1
	*DM*1: *DM*3: *DM*5 = 1: 1: 3

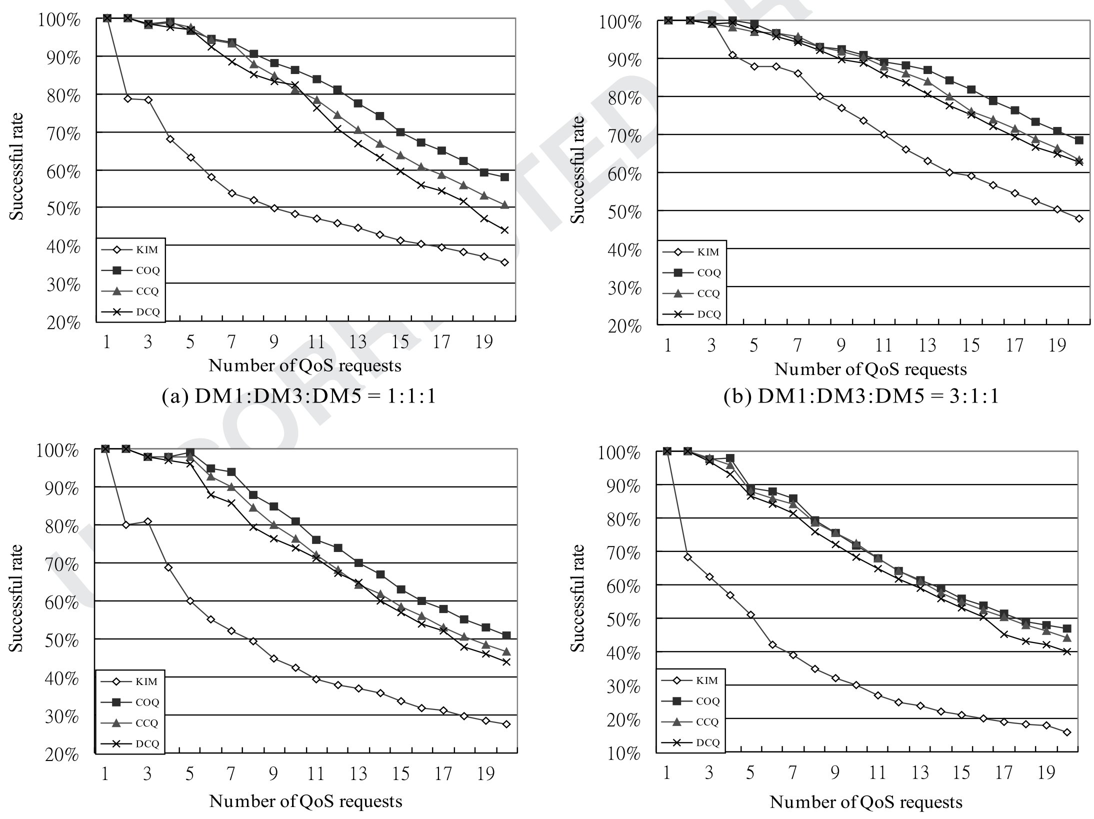


Figure 14. Success rate vs. the number of QoS requests.

Simulator (ns-2) [19] and BlueHoc [20].

In the following simulation, we used KIM, COQ, CCQ, and DCQ to denote Kim et al.'s algorithm [17], our centralized optimal QoS algorithm, our centralized credit-based QoS algorithm, and our distributed credit-based QoS algorithm, respectively. The system parameters are given in Table 4. For instance, four QoS requirement patterns of *DM*1: *DM*3: *DM*5 = 1:

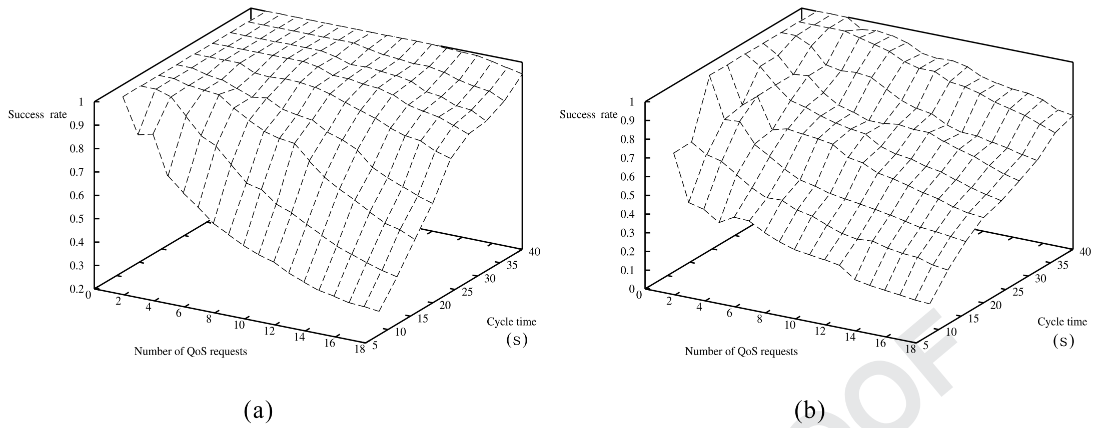


Figure 15. Success rates of (a) CCQ and (b) KIM vs. different cycle times.

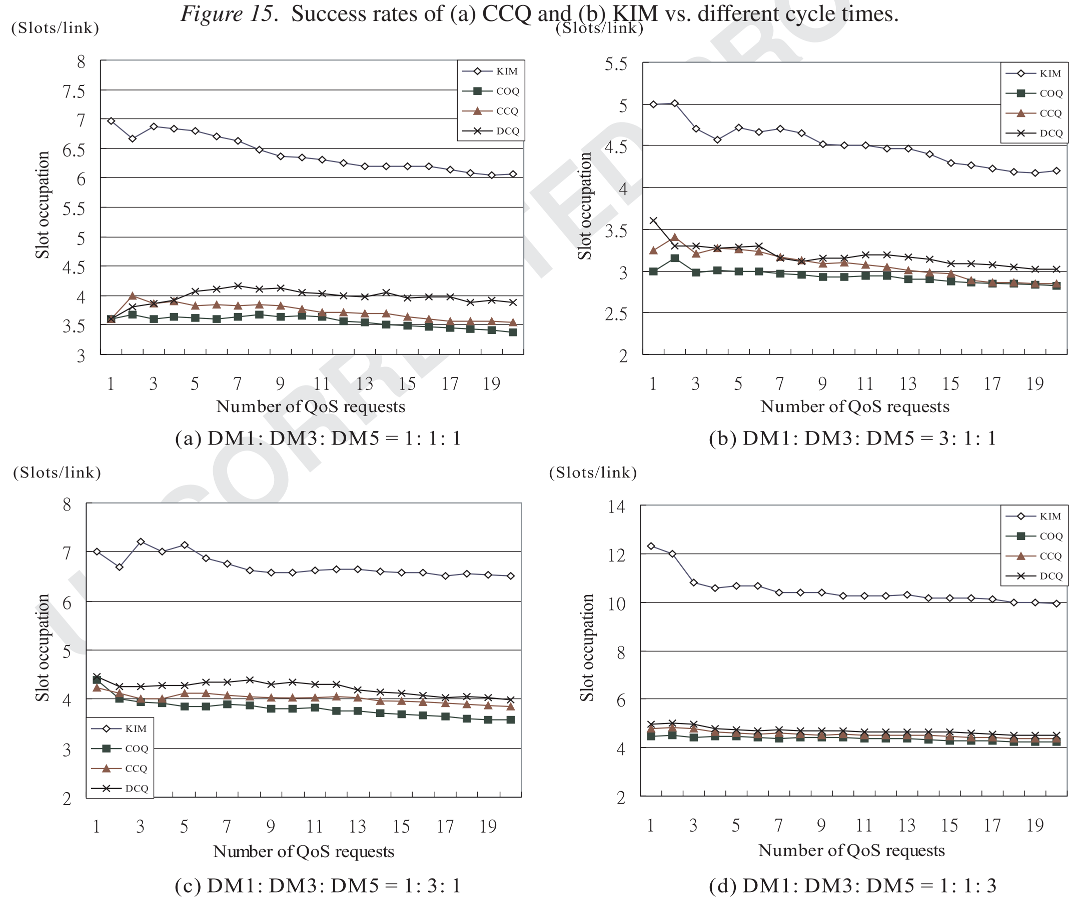


Figure 16. Slot occupation vs. the number of QoS requests.

1: 1, 3: 1: 1, 1: 3: 1, and 1: 1: 3 were used. The QoS requirement pattern used was *DM*1: *DM*3: *DM*5 = 1: 1: 3 to indicate that the probability of data transmission using packet *DM*5 is higher than those of *DM*1 and *DM*3, while *DM*1: *DM*3: *DM*5 = 1: 1: 1 indicates the same probabilities of data transmission of using the *DM*1, *DM*3, and *DM*5 packets. The performance metrics of the simulation are given below.

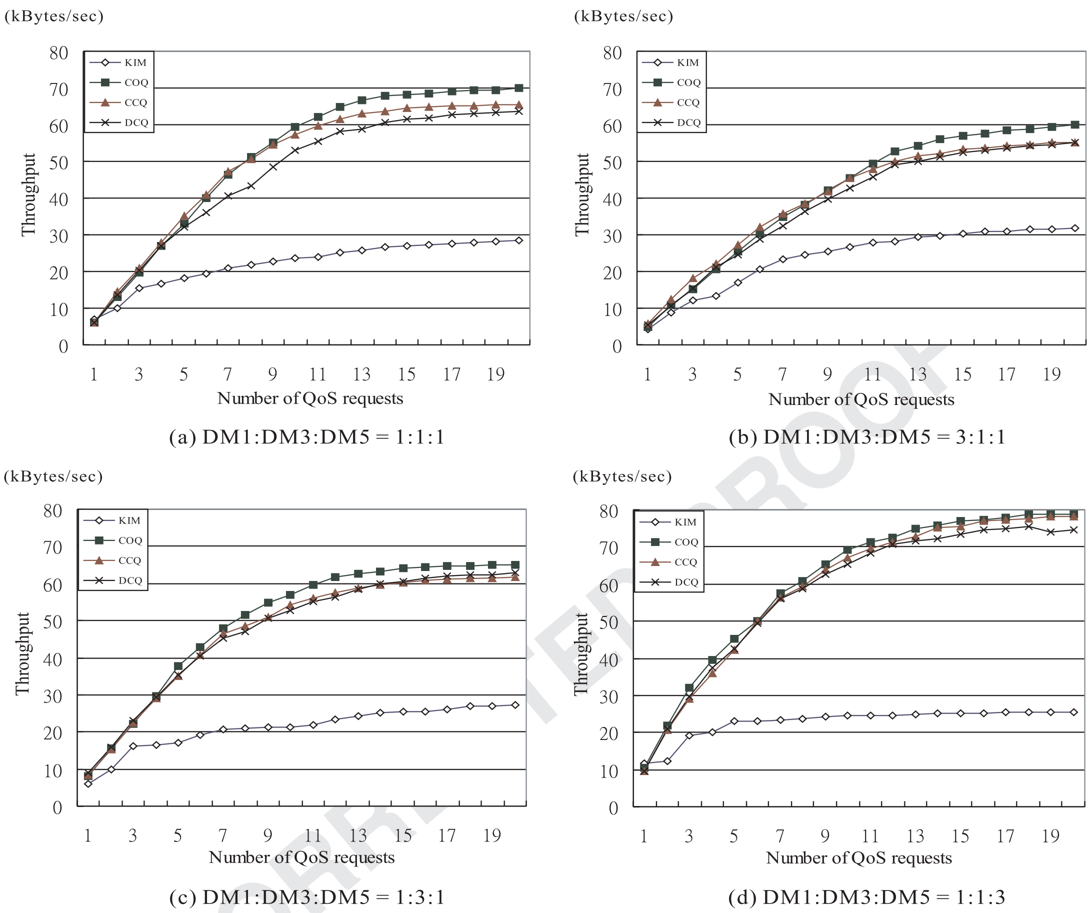


Figure 17. Throughput vs. the number of QoS requests.

- *Success rate*: the number of successful QoS route requests divided by the total number of QoS route requests.
- *Bandwidth efficiency*: the average number of data bytes which can be transmitted per time slot for a successful QoS route.
- *Slot occupation*: the average number of time slots occupied by a successful QoS route.
- *Throughput*: the number of data bytes received by all Bluetooth devices per unit time.

It is worth mentioning that an efficient on-demand QoS routing protocol over Bluetooth WPANs is achieved with a high success rate, a high bandwidth efficiency, a lower slot occupation, and high throughput. In the following, we illustrate our simulation results for success rate, bandwidth efficiency, slot occupation, and throughput from several aspects.

5.1. Performance of Success Rate

We first investigated the effect of various numbers of QoS requests. Figure 14 shows the success rate vs. number of QoS requests for four QoS requirement scenarios: $DM1$: $DM3$: $DM5 = 1$: 1: 1, 3: 1: 1, 1: 3: 1, and 1: 1: 3 as respectively illustrated in Figure 14(a), (b), (c), and (d). In general, COQ, CCQ, and DCQ had higher success rates than did KIM. This is because KIM

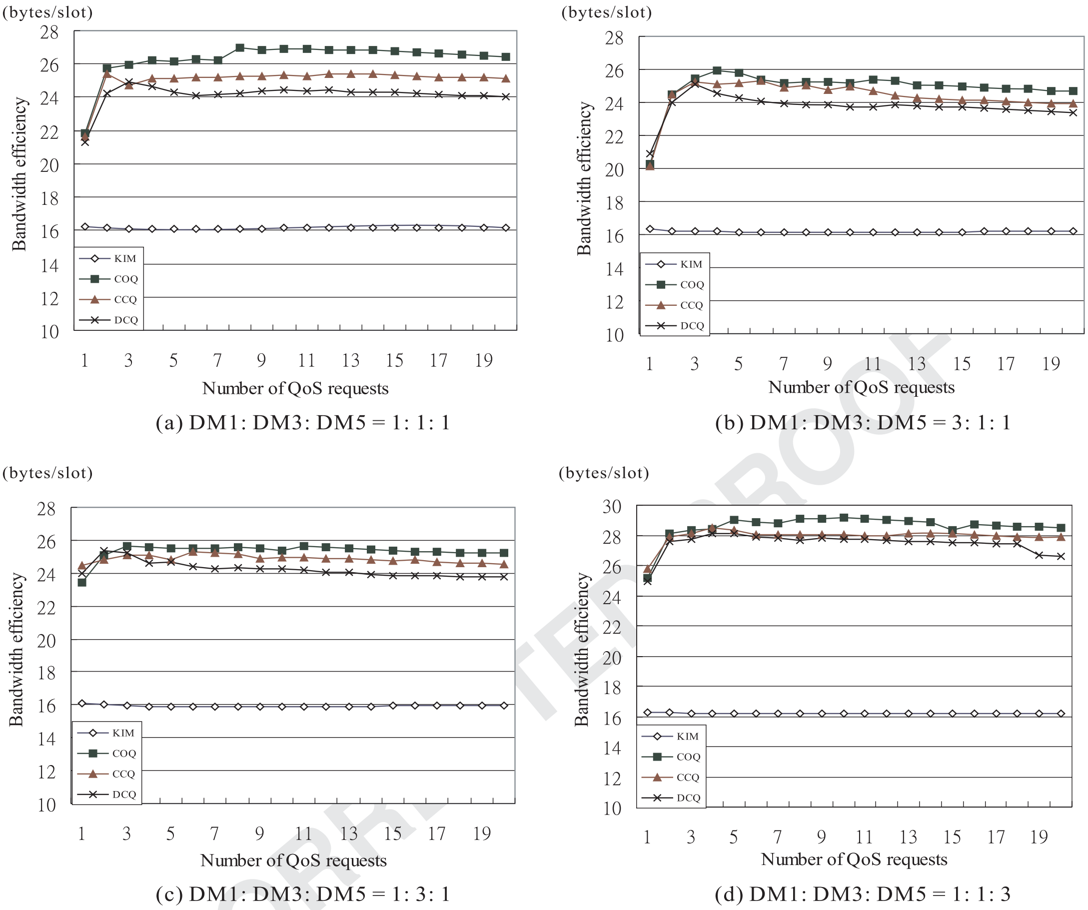


Figure 18. The bandwidth efficiency vs. the number of QoS requests.

wastes too many POLL time slots using DM1 packets. For instance, if a QoS requirement is 136 bytes/cycle time, KIM uses eight time slots (136/17 = 8 DM1) and eight POLL packets. To satisfy the QoS requirement, only DM1 packets are used in KIM. It is possible that KIM consumes most of the time slots for the POLL time slots in a cycle time, therefore the success rate quickly decreases. The low success rate for KIM seriously occurred in the case of *DM*1: *DM*3: *DM*5 = 1: 1: 3 as shown in Figure 14(d).

On the other hand, COQ, CCQ, and DCQ consider the QoS requirement using different packet types of DM5, DM3, and DM1; therefore, the utilization of POLL packets in a cycle time is lower than KIM. This leads to COQ, CCQ, and DCQ having better success rates than KIM. Generally speaking, the success rate of COQ > that of CCQ > that of DCQ > that of KIM as illustrated in Figure 14. This is because DCQ is a distributed algorithm, and COQ and CCQ are centralized algorithms. Figure 14 shows that the average success rate for KIM is about 66% of the average success rates of COQ, CCQ, and DCQ. Also with various numbers of QoS requests as shown in Figure 15, we studied the success rate with different values for cycle time. Figure 15 was obtained for the traffic pattern of *DM*1: *DM*3: *DM*5 = 1: 1: 1 for CCQ and KIM. It is obvious that our CCQ scheme had a higher success rate than KIM under different cycle times.

5.2. Performance of Slot Occupation

Figure 16 gives the simulation results of slot occupation vs. the number of QoS requests under four QoS requirement scenarios. Basically, the slot occupation of COQ < that of CCQ < that of DCQ < that of KIM for the four QoS requirement scenarios illustrated in Figure 16. The simulation results show that the average slot occupation of COQ, CCQ, and DCQ was about 57% that of KIM. It is interested that in the case of $DM1$: $DM3$: $DM5 =$ 1: 1: 3, COQ, CCQ, and DCQ had approximately the same slot occupation rates. This indicates that under the high QoS requirement scenario, we can use DCQ or CCQ protocol to have the good performance in slot occupation and by avoiding the high computation time if using COQ protocol. The performance of slot occuption of DCQ is better than that of CCQ.

5.3. Performance of Throughput

Figure 17 shows throughput vs. the number of QoS requests under four QoS requirement scenarios. The higher the success rate is, the higher the throughput will be. For instance, when the number of QoS requests was fewer than seven, the success rates of our schemes were greater than 90% as illustrated in Figure 14(a), then the throughput of our schemes gradually increased as shown Figure 17(a). But when the number of QoS requests increased, the success rate dropped as illustrated in Figure 14(a), throughput slowly increased as shown in Figure 17(a). Fig 17 shows that the throughput of COQ > that of CCQ > that of DCQ > that of KIM for the four QoS requirement scenarios. In addition, Figure 17(a), (c), and (d) show the lower performance of KIM when the number of QoS requests increases. Observe that the wastage of POLL time slots is more serious the higher the QoS requirement is. That is, the order of wastage of POLL time slots is $DM1$: $DM3$: $DM5$ = 3: 1: 1 < 1: 1: 1 < 1: 3: 1 < 1: 1: 3. Therefore, the number of successful connections for supporting the QoS requirement decreases for the case of $DM1$: $DM3$: $DM5 =$ 1: 1: 3. Therefore, KIM had the lowest throughput. This also explains why $DM1$: $DM3$: $DM5 =$ 3: 1: 1 has smooth curves for the success rate and throughput in Figures 14(b) and 17(b), respectively. Figure 17 shows that the average throughput of KIM was about 50% of those values for COQ, CCQ, and DCQ.

5.4. Performance of Bandwidth Efficiency

Figure 18 shows the bandwidth efficiency vs. number of QoS requests under four QoS requirement scenarios. Basically, Figure 18 illustrates that the bandwidth efficiency of COQ > that of CCQ > that of DCQ > that of KIM for the four QoS requirement scenarios. Based on the same reason that KIM only adopts DM1 packets and thus wastes POLL time slots, our scheme adopts DM1, DM3, and DM5 packets, and therefore the bandwidth efficiency can be improved as shown in Figure 18. Basically, the lower the QoS requirement is, the lower the bandwidth efficiency of our scheme will be. For instance, $DM1$: $DM3$: $DM5$ = 3: 1: 1 had the lowest bandwidth efficiency of our scheme (24 $\sim$ 26 bytes/slot). The higher the QoS requirement is, the better bandwidth efficiency of our scheme will be. For instance, $DM1$: $DM3$: $DM5 =$ 1: 1: 3 had the best bandwidth efficiency of our scheme (28 30 bytes/slot). Finally, Figure 18 shows that the average bandwidth efficiency of KIM was about 66% of those values for COQ, CCQ, and DCQ.

In summary, a new on-demand QoS routing protocol is achieved with a high success rate, high bandwidth efficiency, lower slot occupation, and high throughput, especially with high QoS data requirements.

6. Conclusions

In this paper, we address on-demand QoS routing and interpiconet scheduling problems. Different packet types have different bandwidth utilization levels. The basic idea of our developed protocol is to take advantage from the different types of Bluetooth packets that allow obtaining different bandwidth utilization levels, in contrast with previous works in which only DM1 packets are used, deriving in a lower bandwidth utilization. A centralized credit-based QoS routing protocol was mainly developed which considers different Bluetooth packet types. This work mainly improves the bandwidth utilization of Bluetooth scatternets. The interpiconet scheduling problem can also be resolved by our CCQ approach. However, the centralized algorithm incurs the scalability problem. To alleviate the scalability problem, a distributed algorithm is also investigated in this work. The performance of CCQ protocol is better than that of DCQ protocol. In addition, the simulation result illustrates that our CCQ and DCQ protocols had better performance compared to Kim et al.'s approach. The QoS scheduling may be designed in the L2CAP to possibly implement our scheduling scheme in actual Bluetooth devices.

Acknowledgment

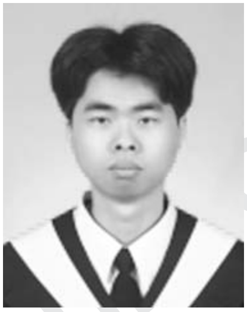
The authors appreciate the helpful comments and suggestions provided by the anonymous reviewers.

References

1. Bluetooth Special Interest Group, "The Bluetooth Specification Version 1.2", Technical report, available at http://www.bluetooth.com, November 2003.
2. Y.-S. Chen, Y.-C. Tseng, J.-P. Sheu, and P.-H. Kuo, "An On-Demand, Link-State, Multi-Path QoS Routing in a Wireless Mobile Ad-Hoc Network," *Computer Communications*, Vol. 27, pp. 27–40, January 2004.
3. Y.-S. Chen and Y.-T. Yu, "Spiral-Multi-Path QoS Routing Protocol in a Wireless Mobile Ad-Hoc Network," *IEICE Transactions on Communications*, Vol. E87-B, pp. 104–116, January 2004.
4. C.-T. Chang, C.-Y. Chang, and J.-P. Sheu, "BlueCube: Constructing a Hypercube Parallel Computing and Communication Environment over Bluetooth Radio System," In *Proceedings of the 2003 International Conference on Parallel Processing (ICPP)*, Taiwan, October 2003, pp. 447–454.
5. T.-Y. Lin, Y.-C. Tseng, K.-M. Chang, and C.-L. Tu, "A New BlueRing Scatternet Topology for Bluetooth with Its Formation, Routing, and Maintenance Protocols," *Wireless Communications and Mobile Computing*, Vol. 3(4), pp. 517–537, 2003.
6. C. Petrioli, S. Basagni, and M. Chlamtac, "Configuring BlueStars: Multihop Scatternet Formation for Bluetooth Networks," *IEEE Transactions on Computers*, Vol. 52, no. 6, pp. 779–790, June 2003.
7. M. T. Sun, C.-K. Chang, and T.-H. Lai, "A Self-Routing Topology for Bluetooth Scatternets," In *Proceedings of International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN)*, Makati City, Metro Manila, Philippines, May 2002, PP. 13–18.

8. P. Bhagwat and A. Segall, "A Routing Vector Method (RVM) for Routing in Bluetooth Scatternet," in *Proceedings of the Sixth IEEE International on Mobile Multimedia Communications (MOMUC)*, San Diego, CA, November 1999, pp. 375–379.
9. Y. Liu, M. J. Lee, and T. N. Saadawi, "A Bluetooth Scatternet-Route Structure for Multihop Ad Hoc Networks," *IEEE Journal on Selected Areas in Communications*, 21 pp. 229–239, February 2003.
10. B. J. Prabhu and A. Chockalingam, "A Routing Protocol and Energy Efficient Techniques in Bluetooth Scatternets," in *Proceedings of the IEEE International Conference on Communications (ICC)*, New York, May 2002, Vol. 5, pp. 3336–3340.
11. T.-Y. Lin and Y.-C. Tseng, "An Adaptive Sniff Scheduling Scheme for Power Saving in Bluetooth," *IEEE Wireless Communications*, Vol. 9, no. 6, pp. 92–103, December 2002.
12. T.-Y. Lin, Y.-C. Tseng, and Y.-T. Lu, "An Efficient Link Polling Policy by Pattern Matching for Bluetooth Piconets," *Computer Journal*, Vol. No. 2, pp. 169-178, March 2004.
13. D. Yang, G. Nair, B. Sivaramakrishnan, H. Jayakumar, and A. Sen, "Round Robin with Look Ahead: A New Scheduling Algorithm for Bluetooth," In *Proceedings of the International Conference on Parallel Processing Workshops (ICPPW)*, Vancouver, Canada, August 2002, pp. 45–50.
14. S. Baatz, M. Frank, C. Kuhl, P. Martini, and C. Scholz, "Adaptive Scatternet Support for Bluetooth using Sniffer Mode," in *Proceedings of the 26th Annual Conference on Local Computer Networks (LCN)*, Tampa, Florida, November 2001, pp. 112–120
15. S. Baatz, M. Frank, C. Kuhl, P. Martini, and C. Scholz, "Bluetooth Scatternets: An Enhanced Adaptive Scheduling Scheme," in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Society (INFOCOM)*, New York, June 2002, Vol. 2, pp. 782–790.
16. C. Cordeiro, S. Abhyanker, and D. P. Agrawal, "Design and Implementation of QoS-driven Dynamic Slot Assignment and Piconet Partitioning Algorithms over Bluetooth WPANs," In *Proceedings of the Twenty-third Annual Joint Conference of the IEEE Computer and Communications Society (INFOCOM)*, Hong Kong, March 2004.
17. Y. M. Kim, T.-H. Lai, and A. Arora, "A QoS-Aware Scheduling Algorithm for Bluetooth Scatternets," In *Proceedings of the International Conference on Parallel Processing*, Kaohsiung, Taiwan, October 2003, pp. 455–462.
18. N. Johansson, U. Korner, and L. Tassiulas, "A Distributed Scheduling Algorithm for a Bluetooth Scatternet," in *Proceedings of the Seventeenth International Teletraffic Congress (ITC)*, Salvador da Bahia, Brazil, September 2001, pp.
19. VINT Project, "Network Simulator version 2 (ns2)," Technical report, available at http://www.isi.edu/nsnam/ns, June 2001.
20. IBM research, "BlueHoc, IBM Bluetooth Simulator," Technical report, available at http://www124.ibm.com/developerworks/opensource/bluehoc/, February 2001.

Yuh-Shyan Chen received the B.S. degree in computer science from Tamkang University, Taiwan, Republic of China, in June 1988 and the M.S. and Ph.D. degrees in Computer Science and Information Engineering from the National Central University, Taiwan, Republic of China, in June 1991 and January 1996, respectively. He joined the faculty of Department of Computer Science and Information Engineering at Chung-Hua University, Taiwan, Republic of China, as an associate professor in February 1996. He joined the Department of Statistic, National Taipei University in August 2000, and joined the Department of Computer Science and Information Engineering, National Chung Cheng University in August 2002. Dr. Chen served as *Co-Editors-in-Chief* of International Journal of Ad Hoc and Ubiquitous Computing (*IJAHUC*), *Editorial Board Member* of Telecommunication System Journal, International Journal of Internet Protocol Technology (*IJIPT*) and The Journal of Information, Technology and Society (*JITAS*). He also served as *Guest Editor* of Telecommunication Systems, special issue on "*Wireless Sensor Networks*" (2004), and *Guest Editor* of Journal of Internet Technology, special issue on "*Wireless Internet Applications and Systems*" (2002) and special issue on "*Wireless Ad Hoc Network and Sensor Networks*" (2004). He was a Vice Co-Chair, Wireless IP Symposium of WirelressCOM2005, USA (2005) and a Workshop Co-Chair of the 2001 Mobile Computing Workshop, Taiwan. Dr. Chen also served as IASTED Technical Committee on Telecommunications for 2002–2005, WSEAS International Scientific Committee Member (from 2004), Program Committee Member of IEEE ICPP'2003, IEEE ICDCS'2004, IEEE ICPADS'2001, ICCCN'2001–2005, MSN'2005, IASTED CCN'2002–2005, IASTED CSA'2004–2005, IASTED NCS'2005, and MSEAT'2003–2005. His paper wins the 2001 IEEE 15th ICOIN-15 *Best Paper Award.* Dr. Chen was a recipient of the 2005 Young Scholar Research Award given by National Chung Cheng University to four young faculty members, 2005. His recent research topics include mobile ad-hoc network, wireless sensor network, mobile learning system, and 4G system. Dr. Chen is a member of the IEEE Computer Society, IEICE Society, and Phi Tau Phi Society.



Keng-Shau Liu received the M.S. degree in Computer Science and Information Engineering from National Chung Cheng University, Taiwan, Republic of China, in July 2004. His research includes wireless LAN, Bluetooth, and mobile learning.