A time-slot leasing-based QoS routing protocol over Bluetooth WPANs

Yuh-Shyan Chen and Tsung-Hung Lin*

Department of Computer Science and Information Engineering,
National Chung Cheng University, Chiayi 621, Taiwan
Fax: +886 5 272 0859 E-mail: yschen@cs.ccu.edu.tw
E-mail: dust@cs.ccu.edu.tw
*Corresponding author

Abstract: In this paper, we investigate a new efficient QoS routing protocol based on the time-slot leasing mechanism over Bluetooth WPANs. The "transmission holding" problem is incurred because the master node is the communication bottleneck for each slave-master-slave communication in Bluetooth. To alleviate this problem, a time-slot leasing scheme is adaptively incorporated into our scheme to provide a completely new QoS routing protocol. This QoS routing protocol can additionally offer extra slave-to-slave QoS communication capability to effectively reduce the workload of the master node. Finally, simulation results demonstrate that this protocol can significantly improve the success ratio, delay time, throughput, and bandwidth utilisation.

Keywords: Bluetooth; time-slot leasing (TSL); quality of service (QoS); wireless personal area network (WPAN); wireless communication.

Reference to this paper should be made as follows: Chen, Y-S. and Lin, T-H. (2005) ‘A time-slot leasing-based QoS routing protocol over Bluetooth WPANs’, *Int. J. Ad Hoc and Ubiquitous Computing*, Vol. 1, No. 3, pp.167–183.

Biographical notes: Dr. Yuh-Shyan Chen is an Associate Professor of Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi, Taiwan, ROC. He received his PhD Degree in Computer Science and Information Engineering from the National Central University, Taiwan, ROC, in 1996. His research interests include wireless communication and mobile computing, mobile ad-hoc network, wireless sensor network, and mobile learning system. He served as Co-Editors-in-Chief of *International Journal of Ad Hoc and Ubiquitous Computing* and Associate Editor of *Telecommunication System Journal*. His paper receipts the Best Paper Award of *IEEE 15th International Conference on Information Networking (ICOIN-15)*, Japan, 2001, and published papers in many major wireless communication and mobile computing conferences and journals.

Tsung-Hung Lin received his BS Degree in Computer Science from Tamkang University, Taiwan, ROC, in June 1988 and his MS Degree in Computer Science and Information Engineering from National Chung Cheng University, Taiwan, in 1993. He is currently a Graduate Student for the PhD Degree in the Department of Computer Science and Information Engineering, National Chung Cheng University. His research interests include mobile computing, wireless sensor networks, WCDMA Code Assignment and Capacity Planning.

1 Introduction

Several untethered small handheld electronic *personal area network* (PAN) devices including mobile phones, cell phones, *personal digital assistants* (PDAs), mp3 players, headphones, digital cameras, sensors, and their peripherals will become part of our daily lives. The tremendous growth of these popular heterogenous PAN devices is increasingly creating requirements for efficient communications. Bluetooth *wireless personal area network* (WPAN) technologies are continually increasing in interest for their ubiquitous mobile connections and their ability to provide new personal communication opportunities and services (Joo et al., 2003). Different WPAN infrastructures based on Bluetooth can provide many different communication services and can be interconnected to enable sharing of information to allow interactions with the physical environment. The smallest Bluetooth enabled device in a WPAN, called a piconet, is established by at most eight nodes. Member devices of piconets may become member devices of other piconets, thus forming a large network called a scatternet (Bhargava and Gruenbacher, 2004; Bluetooth Special Interest Group, 2003; Whiteaker et al., 2004). Different Bluetooth devices may have various traffic characteristics; the uniform distribution of service opportunities is not an efficient policy from the perspective of obtaining the best possible quality of service (QoS) and the most efficient allocation of wireless resources. Slaves are not always listening to the master in the listening mode of the scatternet scheduling mechanism. If the source and destination nodes are located in distinct piconets, the weakness of the interpiconet scheduling mechanism is still a problem. Addressing the ondemand QoS routing and interpiconet scheduling problems, Chen et al. (2004) designed a slave-master-slave *credit-based QoS* (CQ) routing protocol under the considerations of different Bluetooth packet types and different bandwidth utilisations. Bandwidth utilisations in Bluetooth scatternets and the solutions of interpiconet scheduling problems are improved by the CQ routing protocol. This protocol still has drawbacks of no direct link existing between any two slaves within the same piconet, and the master having to forward the exchanged packets between two slaves. The heavier the loading of the master is, the lower the system performance of the scatternet will be. Due to the weakness of the interpiconet scheduling mechanism, the slave-master-slave system still possesses communication QoS problems. Joo et al. (2003) developed a power efficient mechanism for Bluetooth scatternets to guarantee the various QoS requirements of Bluetooth devices. This MAC scheduling scheme reduces the number of data exchanges in order to gain low power consumption. Cordeiro et al. (2004) designed two protocols and a direct slave-to-slave communication model to provide their QoS requirements. The master in the piconet dynamically assigns slots to slaves so as to allow them to directly communicate with each other without any intervention from the master. Reducing the network power consumption and enhancing the Bluetooth QoS performance in terms of delay, bandwidth utilisation, overhead, and aggregate throughput are the main advantages of Cordeiro et al.'s schemes. These scatternet scheduling schemes use the sniff mode mechanism to achieve scheduling without needing to modify the Bluetooth specifications. Recently, Zhang et al. (2001, 2002) proposed a new scatternet QoS approach, called *time-slot leasing* (TSL) and an enhanced version, called *enhanced TSL* (ETSL), to address the problems associated with the slave-master-slave model. The TSL approach establishes temporarily leased slots from the constructed piconet to support slave-to-slave communication. Because the master centrally controls the usage of the time slots, the slaves are stateless and no collisions occur between them. In this paper, the TSL based schemes were adopted, and each temporary piconet was allowed to have its own dedicated channel without permanently changing the scatternet structure, thus producing no negative effects on interpiconet communication. The drawbacks with the slave-master-slave model are resolved by the idea of dynamic structures and role management of the piconet under TSL based strategies.

We investigate a new efficient QoS routing protocol based on the time-slot leasing mechanism over Bluetooth WPANs in this paper. In the Bluetooth scatternet, a QoS route path is constructed through a series of QoS slave-master-slave communications. The 'transmission holding' problem is incurred because the master node is the communication bottleneck for each slave-master-slave communication. To alleviate this problem, the time-slot leasing scheme is adaptively incorporated into our scheme to provide a completely new QoS routing protocol. This QoS routing protocol can additionally offer extra slave-to-slave QoS communication capability to effectively reduce the workload of master node and significantly promote the success rate of finding a QoS route. In our proposed QoS routing protocol, QoS slave-master-slave and slave-to-slave communication mechanisms are simultaneously considered in order to achieve a high success rate of QoS routing from the source to the destination nodes. Centralised and distributed QoS routing protocols are respectively presented. Finally, simulation results demonstrate that the time-slot leasing-based QoS routing protocol can significantly improve the success ratio, delay time, throughput, and bandwidth utilisation when compared to other existing QoS routing protocols.

The rest of this paper is organised as follows. Section 2 introduces some preliminaries and basic ideas. Section 3 develops the *centralised time-slot leasing-based QoS* (CTQ) routing protocol. The *distributed time-slot leasing-based QoS* (DTQ) routing protocol is given in Section 4. In Section 5, we discuss the experimental results and performances of our *time-slot leasing-based QoS* (TQ) routing protocol. Finally, Section 6 concludes this paper.

2 Preliminaries and basic ideas

Bluetooth technology was brought into existence with the motive to replace cable wires between any communicating devices and support wireless networking with different types of devices. The devices that use Bluetooth technologies are normally formed piconets which communicate with each other in a slave-master-slave configuration. A piconet consists of a single master device and up to seven active slave devices (Bluetooth Special Interest Group, 2003). Bluetooth radios issue two types of packets in the connected state, named the *synchronous connection oriented* (SCO) type and *asynchronous connectionless* (ACL) type. The link of the ACL type is packet oriented under both symmetrical and asymmetrical traffic and is a packetswitched asynchronous connection between two devices created on the *link manager protocol* (LMP) level. The main function of the LMP is for link setup and link control with the piconet. In the following, QoS issues in links of the ACL type are investigated. Packets of the ACL type are created with odd numbered time slots and include two packet types: the medium rate data DM type packet that includes *forward error correction* (FEC) and the high rate data DH type packet that excludes FEC (Chen et al., 2004; Bluetooth Special Interest Group, 2003). The bandwidth utilisation is calculated by the amount of the used data pay load per number of time slots; the bandwidth utilisations of $DM1$, $DM3$, $DM5$, $DH1$, $DH3$

and *DH*5 are 17.0 (17/1), 40.3 (121/3), 44.8 (224/5), 27.0 (27/1), 61.0 (183/3) and 67.8 (339/5) bytes/slot, respectively (Chen et al., 2004). Using comparisons of the degree of bandwidth utilisation, it is quite obvious that *DM*5 > *DM*3 > *DM*1 and *DH*5 > *DH*3 > *DH*1. This fact motivated us to develop an efficient QoS routing scheme by taking the factor of different packet types with different bandwidth utilisations into account. In the TQ routing protocol, only an errorprone environment is adopted, and three packets, DM1, DM3 and DM5 packets, are considered. For an errorfree environment, our TQ routing protocol can easily be tallied with the *DH*1, *DH*3 and *DH*5 packets.

Bluetooth is a promising short range wireless communication technology, but the fact of the 'transmission holding' problem that no direct connection exists between any two slave nodes in a piconet is still a drawback. Any slave-to-slave communications must go through the master node in the same piconet, and the master node has to use extra bandwidth to exchange packets between two slave nodes (Zhang et al., 2001). For example, in Figure 1(a), if slave node, S_2, wants to transmit packets to another slave node, S_1, then two transmissions are needed. One is from S_2 to the master, denoted S_2*-to-master* and the other is from the master node to the slave node, S_1, denoted *master-to-*S_1. Use of S_2-to-master and master-to-S_1 doubles the bandwidth consumption and communication delay time. The ideal throughput for slave-to-slave model communication can be doubled compared to the standard slave-master-slave model. For example, in Figure 1(a), slave node, S_2 and slave node, S_1, switch their roles and become a temporary master node, denoted *Temp-master* and a temporary slave node, denoted *Temp-slave*, respectively and then a new temporary piconet with lower master throughput and a shorter communication delay is constructed. The RS approaches can create a new scatternet topology and provide a solution with slave-to-slave topologies. Nevertheless, the relay actions of the role switching strategy require some communication resources and share half the resources of time slots. The TSL approach need not permanently change the basic piconet structure and has no negative effects on interpiconet communications (Zhang et al., 2002). For example, as illustrated in Figure 1(b), if slave node, S_1, needs to send a large file to another slave node, S_2, then S_1 requests the master to lease some time slots for direct transmission between S_1 and S_2. Direct packet transmissions between S_2 and S_1 are accomplished without changing the roles of S_1 and S_2. Figure 1(a) also shows the RS setup procedure, and Figure 1(b) also shows the TSL setup procedure. Because the RS approach needs to switch the master-to-slave or slave-to-master roles, the time slots are averages shared by the two roles. By limiting modifications to traditional master-to-slave scatternet models and without permanently changing the piconet network topologies, TSL can achieve good system performance. Average time slot usages of TSL complemented packet communications are less than those of normal slave-master-slave packet communications, even though the TSL mechanism requires more startup time slots. Figure 2 shows a comparison of time slot usages between normal piconet transmissions and TSL complemented piconet transmissions. The new QoS connection requirement is 51 bytes/cycle time and the DM3 packet type is adopted to send the data packets. The total time slot usages of a new QoS connection requirement in normal piconet communication and TSL complemented piconet communication are 32 and 24 time slots, respectively, as displayed in Figure 2. The performance of time slot usages in the TSL complemented scheme is better than that of the nonTSL complemented scheme. In this paper, we modified the TSL approach and Chen et al. (2004)'s CQ routing protocol to form our TQ routing protocol. The QoS scheduling results extend the algorithm of Chen et al. The algorithms of our time-slot leasing-based protocol are illustrated in Figure 3. Figure 3(a) shows an example of a scatternet, and Figure 3(b) shows the new QoS connection requirements. In this case, the approach in Chen et al. will fail at the link, $\overrightarrow{aM}$ and the connection request will be rejected, as shown in Figure 3(c). Figure 3(d) shows that our time-slot leasing-based QoS routing protocol can overcome this problem and complete the new QoS connection request. In our TQ routing protocol, the time-slot leasing-based scheme is started up when the time slot usage of slave-master-slave communication is underway. The slave-to-slave time-slot leasing-based scheme is started up to replace the slave-master-slave scatternet communication. The replaced time-slot leasing-based scheme increases the system throughput and decreases the system delay time. Figure 4 shows the basic idea of time-slot leasing-based QoS route-discovery operations in our TQ routing protocol. If a new QoS connection request arrives with source node, *S* and destination node, *D* then the QoS requirement will be *n* bytes/cycle time. Figure 4(a) shows the case when master node, *M* has insufficient free ($< n$) time slots in the $\overrightarrow{aM}$ and $\overrightarrow{cM}$ input links and $\overrightarrow{Mb}$ and $\overrightarrow{Md}$ output links, to service the new QoS requirement. In this period, the time-slot leasing-based routing scheme is started up in order to find sufficient ($\geq n$) time slots for the slave-to-slave link. Figure 4(b) shows that the usable slave-to-slave link, $\overrightarrow{ab}$, can replace links $\overrightarrow{aM}$ and $\overrightarrow{Mb}$ using the time-slot leasing-based routing scheme to form the final routing path, $S \to a \to b \to D$. Figure 4(c) shows that $S \to a \to b \to D$ is the other final routing path.

Figure 1 Architectures of (a) RS and (b) TSL and setup procedures of (c) RS and (d) TSL

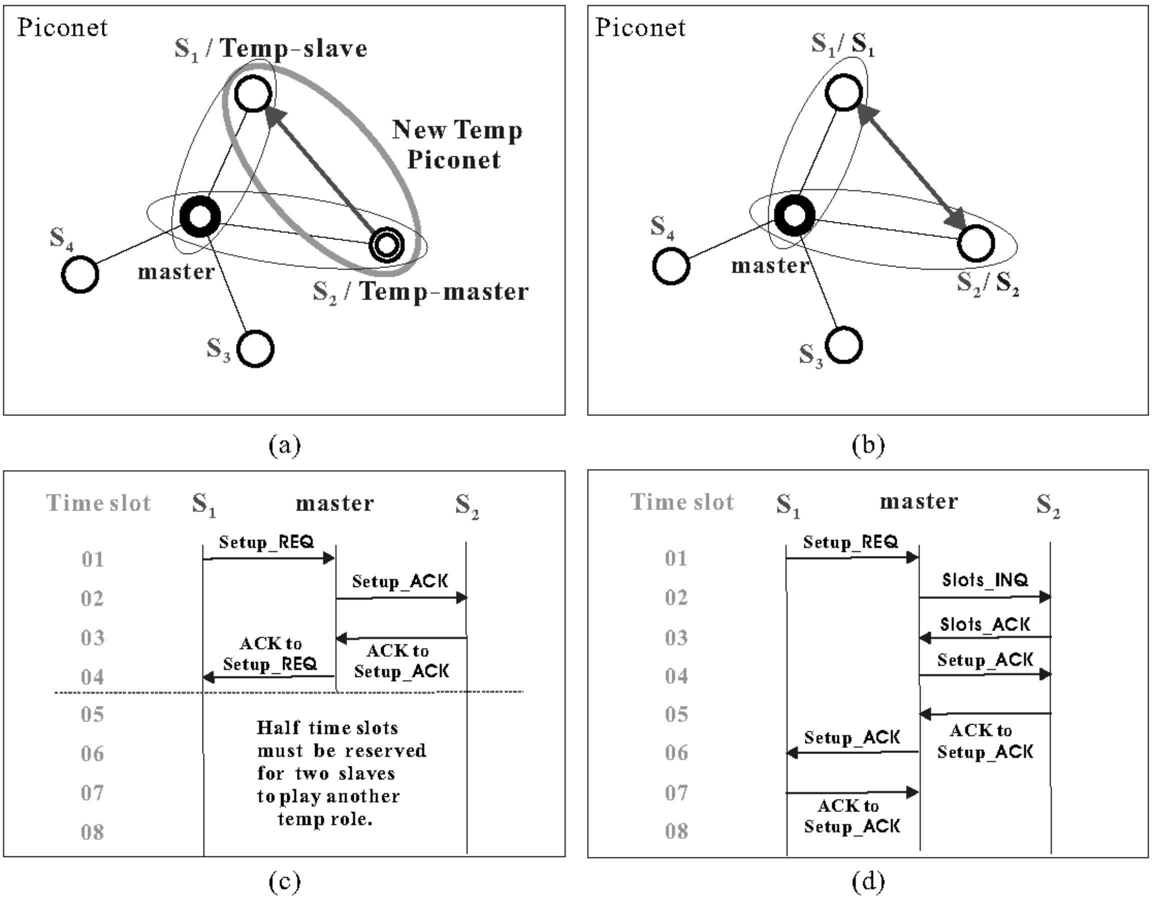


Figure 2 Time slot usages of (a) normal and (b) TSLcomplemented piconet transmission

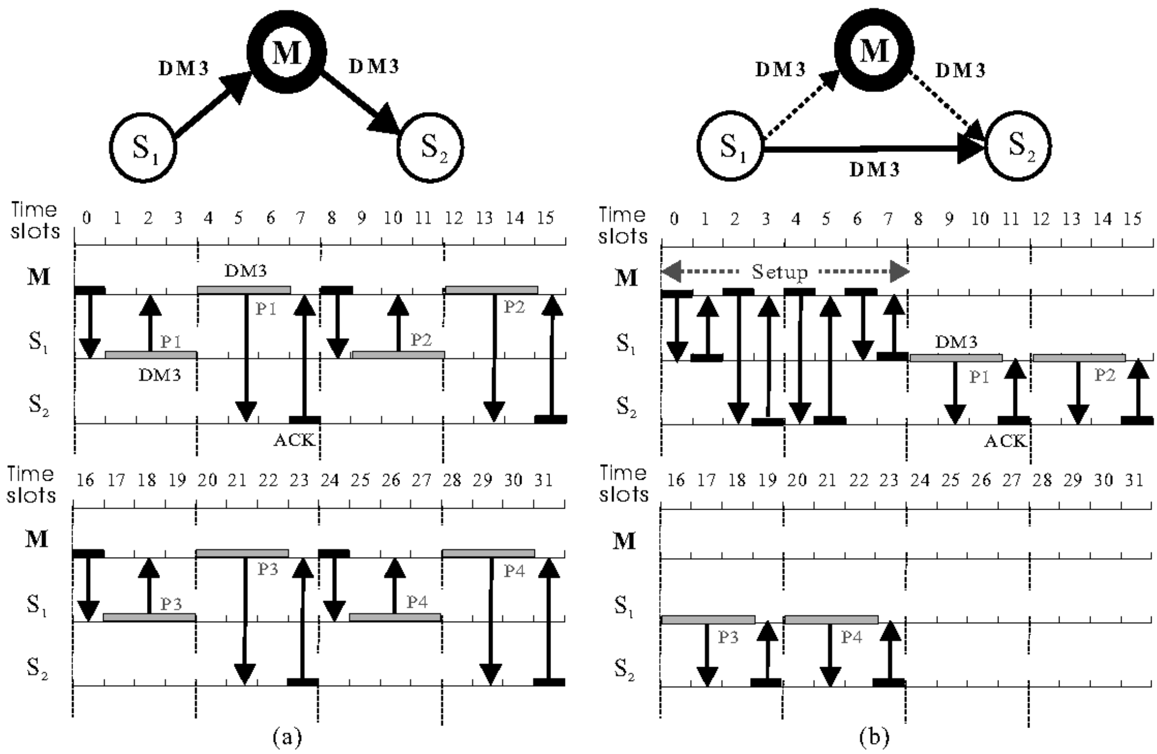


Figure 3 Comparison of QoS scheduling results

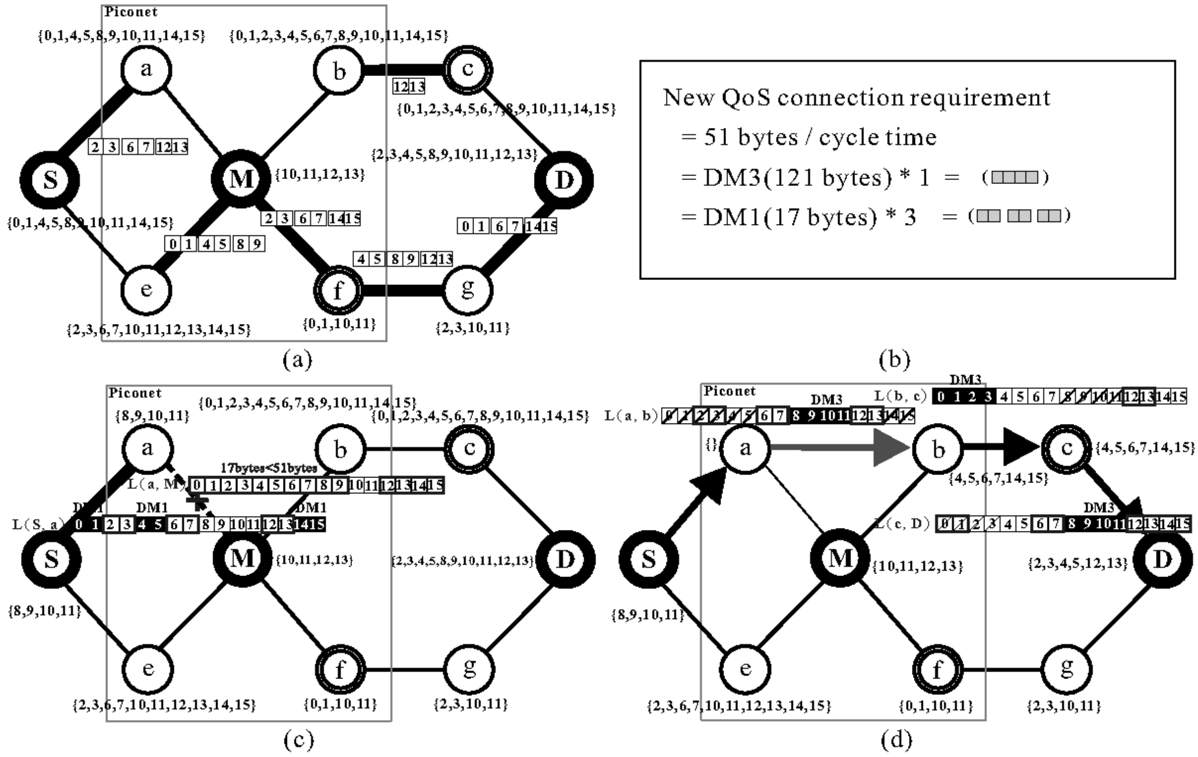
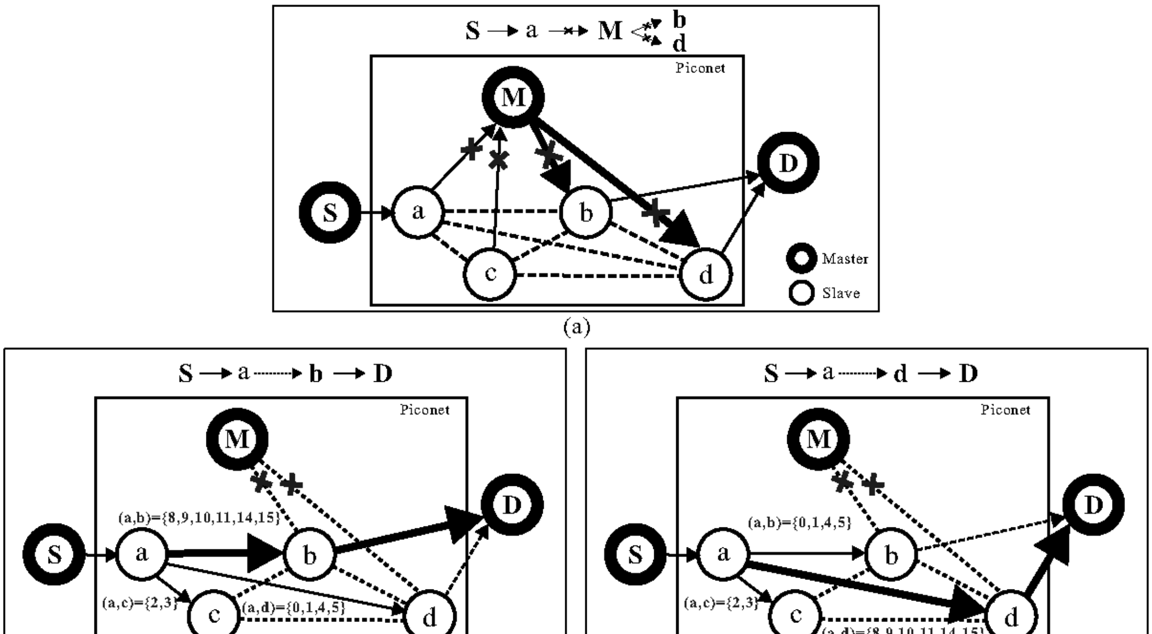


Figure 4 Basic idea of time-slot leasing-based QoS routediscovery operations



3 CTQ (centralised time-slot leasing-based QoS) routing protocol

To improve the system performance, two versions of algorithms, centralised and distributed algorithms, were incorporated into our TQ routing protocol. In Section 3, the CTQ (centralised time-slot leasing-based QoS) routing protocol is proposed. Over the preformed Bluetooth scatternet, the centralised QoS routing algorithm constructs a routing path from the source node to the destination node to satisfy the QoS requirement. Our TQ routing protocol is an efficient synthesised QoS routing protocol combining a priority based scheme, i.e., a time-slot leasing-based scheme and taking the factor of different packet types with different levels of bandwidth utilisation into account. In a comparison with the algorithms of Chen et al. (2004), our CTQ routing protocol was achieved by developing an *available QoS* *routing-path search phase, a time-slot leasing-based path recovery phase* and a *time-slot reservation phase*. First, in the available QoS routing-path search phase, information on free time slots is collected among all of the routing paths constructed from the designated source node to the designated destination node. In the time-slot leasing-based path-recovery phase, backup paths meeting the QoS requirement are found when the request for a normal QoS requirement connection fails. Finally, in the time-slot reservation phase, priority based, time-slot leasing-based strategies and the optimal solution of QoS time-slot reservation are given.

3.1 Available QoS routing-path search phase

Collection of the required QoS free time-slot information is performed between all routing paths from the designated source node to the designated destination node. First, the designated source node broadcasts a QoS request packet, named QoS_REQ, into the Bluetooth scatternet and waits for the *route reply* (RREP) to return from the designated destination node. The QoS_REQ packet is defined as QoS_REQ (DS_ADR, DD_ADR, QoS_RQ, Cur_Free_TS, Path_List, Free_TS_link, Time_to_Live, Min_FTP), where the detailed definitions of arguments in the QoS_REQ packet are illustrated in Table 1.

Table 1 Definitions of the arguments in the QoS_REQ packet

Packet field	*Field description*
DS_ADR	Address of the designated source node
DD_ADR	Address of the designated destination node
QoS_RQ	Value of the designated QoS requirement
Cur_Free_TS	Free time slots of the current traversed node
Path_List	List of nodes that records the path from the designated source node to the current traversed node
Free_TS_Link	Free time slot list of corresponding links; each item on the list records the shared free time slots of an indicated link among Path_List
Time_to_Live	Limitation of hop numbers in the designated search path
Min_FTP	Minimum packets of available free time slots among the Path_List

Next, let $\{\alpha_1, \alpha_2, ..., \alpha_n\}$ denote a set of free time slots for our Bluetooth scatternet environment. For example as displayed in Figures 5(a) and 6(a), {2, 3, 4, 5, 6, 7, 8, 9, 12, 13} is the set of free time slots of node *a* and {10, 11, 12, 13} is the set of free time slots of node *M*. Then, $\overleftrightarrow{aM}$ denotes the link between adjacent nodes *a* and *M*. An intersection function is executed for link $\overleftrightarrow{aM}$ to calculate the shared free time slots between nodes *a* and *M*. The intersection function is illustrated as $\{\alpha_1, \alpha_2, ..., \alpha_{n_1}\} \cap \{\beta_1, \beta_2, ..., \beta_{n_2}\} = \{\gamma_1, \gamma_2, ..., \gamma_{n_3}\}$, where $\{\gamma_1, \gamma_2, ..., \gamma_{n_3}\} \in \{\alpha_1, \alpha_2, ..., \alpha_{n_1}\}, \{\beta_1, \beta_2, ..., \beta_{n_2}\}$ and $n_3 \leq \min(n_1, n_2)$. The algorithm of the available QoS routing-path search phase is described as follows.

A1 The designated source node initially broadcasts the QoS_REQ (DS_ADR = *S*, DD_ADR = *D*, DQ_REQ = *QR*, Cur_Free_TS = $\{\alpha_1, \alpha_2, ..., \alpha_{n_1}\}$, Path_List = {}, Free_TS_Link = {}, Time_to_Live = *HN*, Min_FTP = 0) into the Bluetooth scatternet, where *S* is the designated source node, *D* is the designated destination node, *QR* is the QoS requirement and *HN* is the hop numbers of the time to live.

A2 If an intermediate node between nodes *S* and *D* receives the QoS_REQ (*S*, *D*, *QR*, *Cur_FT*, *Cur_PL*, *Cur_FTL*, *Cur_HN*, *MinPK*) from the other nodes in the scatternet, where *Cur_FT* is the currently free time slots, *Cur_PL* is the current path list, *Cur_FTL* is the currently free time slots of link, *Cur_HN* are the remaining useable hop numbers and *MinPK* are the packet numbers of the current minimum available free time slots. Then, the current node checks the *Cur_HN* and *D* and the cases below follow.

- B1 If the current node is destination node, *D*, then jump to A3.
- B2 If the current node is not destination node, *D* and the remaining hop number *Cur_HN* is equal to zero, then drop the current QoS_REQ.
- B3 If the current node, assumed to be N_c, is not the destination node, and the shared free time slots between N_c and the corresponding node, assumed to be N_c in the *Cur_FTL* cannot satisfy the QoS value *QR*, then keep the failed link, $\overleftrightarrow{N_c N_{c'}}$ and jump to the time-slot leasing-based recovery phase (C1).
- B4 If the current node is not destination node, *D*, then add the information of the current node to the QoS_REQ and flood the new modified QoS_REQ into the Bluetooth scatternet, where the new QoS_REQ is (*S*, *D*, *QR*, {*free time slots of current node*}, {*Cur_PL*} ∪ {*current node*}, [{*Cur_FTL*} ∪ {*shared free time slots of current node*}], *Cur_HN*-1, *new updated MinPK*).

A3 The designated destination node waits for a reasonable time period to receive the different QoS_REQ packets from the designated source node. If the reasonable time period passes, then the list of QoS_REQ packets passes into the *time-slot reservation phase* (E1) and this phase is finished.

Figure 5 One hop CTQ approach

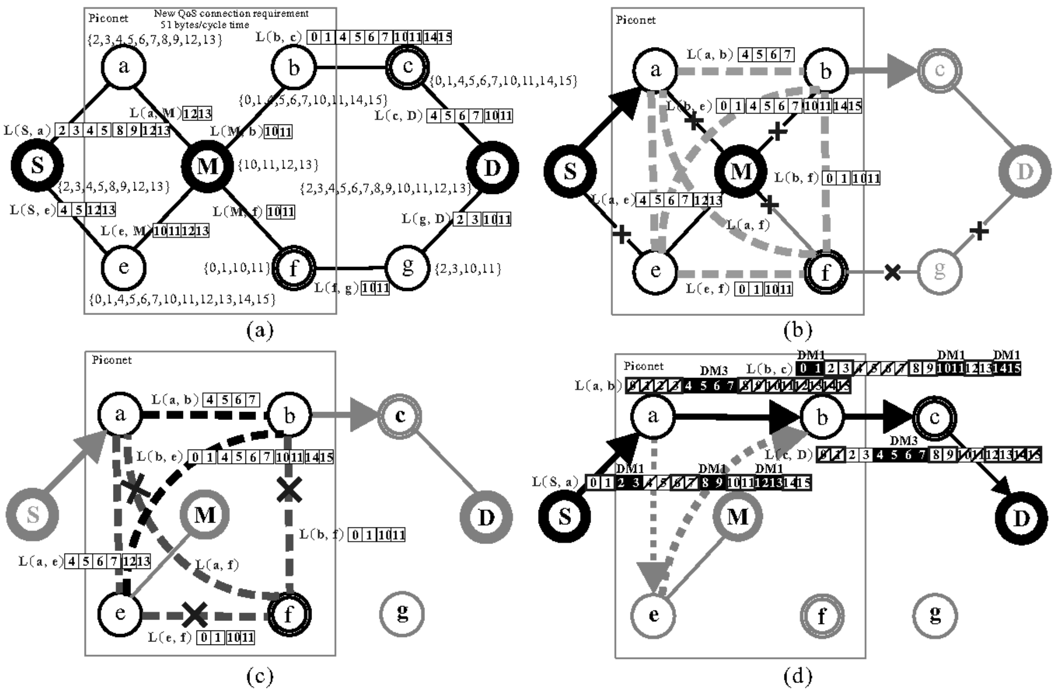
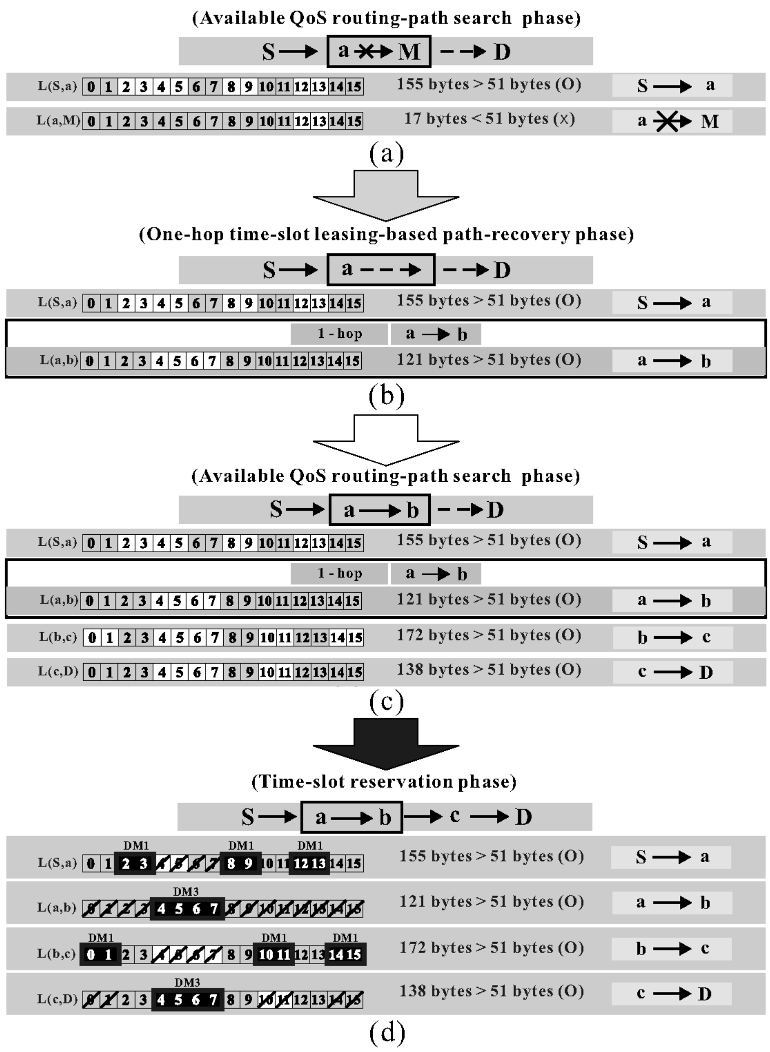


Figure 6 Time slot reservation steps using the one hop CTQ approach



3.2 Time-slot leasing-based path-recovery phase

To increase the system performance, the time-slot leasing-based scheme is started up when the routing path of the traditional free time-slot information collection phase fails. In our TQ routing protocol, the time-slot leasing-based slave-to-slave architecture replaces the traditional slave-master-slave architecture to increase system throughput and decrease system delay times. The algorithm of the time-slot leasing-based path-recovery phase is described as follows.

C1 If the failed link is $\overleftrightarrow{N_cN_{c'}}$, then all of the slave-to-slave information is collected in the following substeps.

D1 If N_c is the master node, then link information is collected of all links from all other slave nodes to slave node, $N_{c'}$, in the same piconet with master node, N_c. For example as illustrated in Figures 5(b) and 6, if the failed link is $\overleftrightarrow{Mb}$, then the information of $\{\overleftrightarrow{ab},(\overleftrightarrow{ae},\overleftrightarrow{eb}),(\overleftrightarrow{af},\overleftrightarrow{fb}),\overleftrightarrow{ae},\overleftrightarrow{ef},\overleftrightarrow{fb})\}$ will first be collected.

D2 If $N_{c'}$ is the master node, then link information is collected of all links from slave node, N_c, to other slave nodes in the same piconet with master node, $N_{c'}$. For example as shown in Figures 5(b) and 6, if the failed link is $\overleftrightarrow{aM}$ then the information of $\{\overleftrightarrow{ab},(\overleftrightarrow{ae},\overleftrightarrow{eb}),(\overleftrightarrow{ae},\overleftrightarrow{ef}),\overleftrightarrow{af}\}$ will first be collected.

C2 If all of the links have failed, then the QoS_REQ is dropped and the algorithm is broken by returning the message of the failed QoS connection requirement.

C3 The ‘fewer hops first’ strategy will preferentially adopt a choice of time-slot leasing-based replacement paths. If more than one available link exists, then the group possessing the fewest number of hops is chosen.

C4 When the group with fewer hop numbers is chosen, the ‘maximum free packet first’ strategy continuously chooses the time-slot leasing-based replacement path. If more than one available link exists, in the chosen group, then the link with the maximum free packets is chosen. Figure 7 shows examples of the strategy of twohop time-slot leasing-based ondemand QoS routing replacement. The new QoS connection requirement is 51 bytes/cycle time. A complete twohop CTQ approach is shown in Figures 8 and 9.

C5 When the replacement link is chosen, then return to the previous *available QoS routing-path search phase*.

Figure 7 Example of a twohop time-slot leasing-based ondemand QoS routing replacement

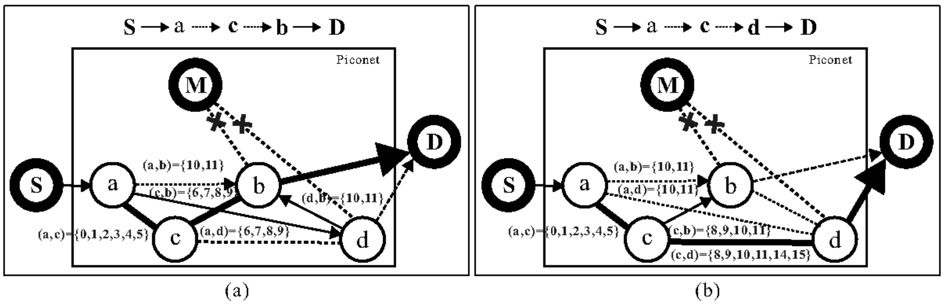


Figure 8 Twohop CTQ approach

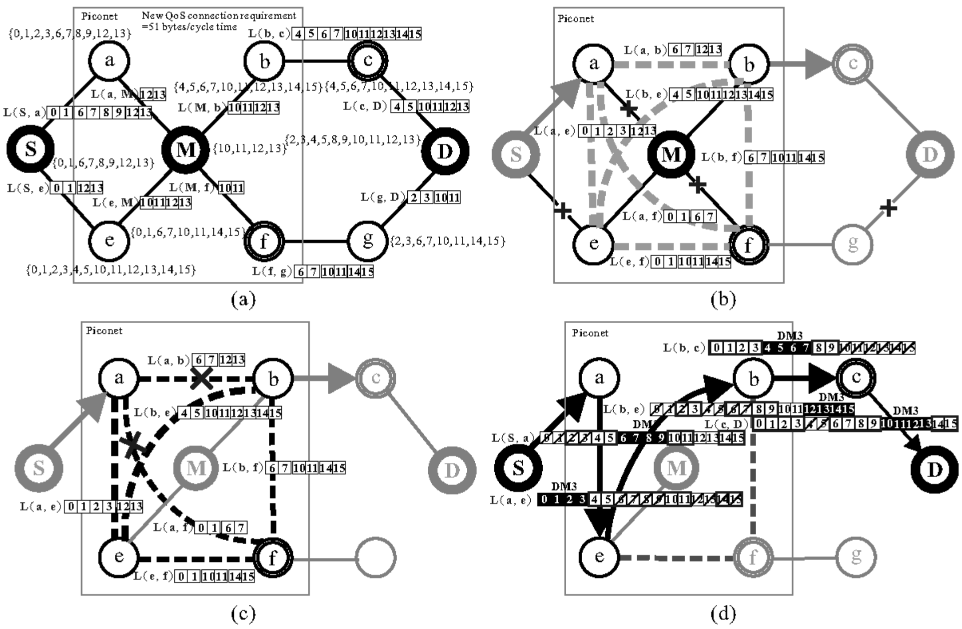
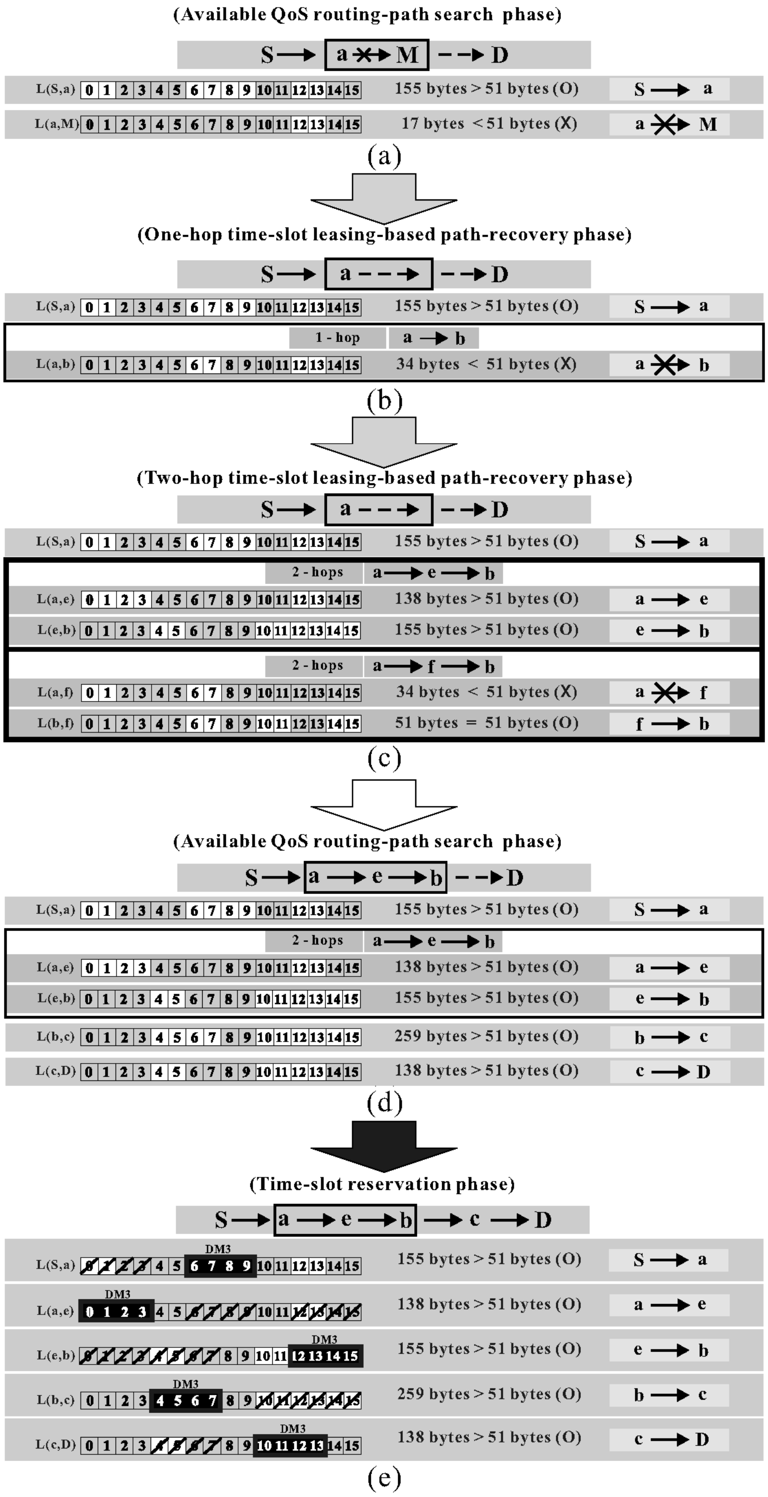


Figure 9 Time-slot reservation steps using the twohop CTQ approach



3.3 Time-slot reservation phase

Finally, the time-slot reservation phase is proposed to reserve the time slots to support the QoS requirement. In this phase, a priority based QoS routing scheme is also proposed to increase the high bandwidth utilisation by assigning time slots with different packet types into each link. The main purpose of the priority based algorithm is developed based on the priority value to find a lower influence QoS route. In comparison with the definition of Chen et al. (2004), we describe our priority based definition clearly as follows.

Let BTL(X, Y) and FTL(X, Y) denote the time slots with a busy status of link $\overrightarrow{XY}$ and time slots with a free status of link $\overrightarrow{XY}$, respectively. For example as displayed in Figure 10(a), the busy slots of $L(S, a)$ are slots 2, 3, 14 and 15. Therefore, BTL(S, a) = {2, 3, 14, 15} and FTL(S, a) = {0, 1, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13}. L(X, Y, Z) denotes the union of the two links, $\overrightarrow{XY}$ and $\overrightarrow{YZ}$ and the links are from node X to node Z through node Y. The status of time slot i of L(A, B) is denoted $TS(i)_{L(A,B)}$. A priority credit value list $CN(\delta)_{L(X,Y,Z)}$ is constructed by priority credit values $CN(\delta_i)_{L(X,Y,Z)}$, where $0 \leq i \leq$ the polling interval. We show the rules of the time-slot reservation phase as follows.

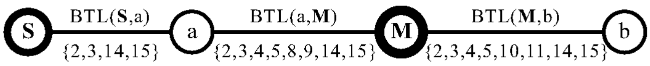
E1 If $TS(i)_{L(Y,Z)}$ is in BTL(Y, Z), then set $CN(i)_{L(X,Y,Z)}$ to 0. This means that the current link, $L(Y, Z)$, is busy and unusable, even though the prefix link, L(X, Y), is free. We give the priority credit value 0 to the connection reference $CN(i)_{L(X,Y,Z)}$. For example as shown in Figure 10(a), because $TS(5)_{L(a,M)}$ is in the busy list, BTL(a, M), $CN(5)_{L(S,a,M)}$ is set to 0.

E2 If $TS(i)_{L(Y,Z)}$ and $TS(i)_{L(X,Y)}$ are in the FTL(Y, Z) and FTL(X, Y), respectively, then set $CN(i)_{L(X,Y,Z)}$ to 1. This means that the current link, $L(X, Y, Z)$, is free and we can use this time slot for a QoS connection. We give the priority credit a value of 1 to the connection reference $CN(i)_{L(X,Y,Z)}$. For example as illustrated in Figure 10(b), because $TS(6)_{L(a,M)}$ is in the free list, FTL(a, M) and $TS(6)_{L(M,b)}$ is in the free list, FTL(M, b), the value of $CN(6)_{L(a,M,b)}$ is set to 1.

E3 If $TS(i)_{L(Y,Z)}$ is in the FTL(Y, Z) and $TS(i)_{L(X,Y)}$ is in the BTL(X, Y), then set $CN(i)_{L(X,Y,Z)}$ to 2. This means that the prefix link, $L(X, Y)$, is unusable and the current link, $L(Y, Z)$, is still unusable even though the current link, $L(Y, Z)$, is free. We give the priority credit value 2 to the connection reference, $CN(i)_{L(X,Y,Z)}$. For example as displayed in Figure 10(b), if $TS(10)_{L(a,M)}$ is in the free list, FTL(a, M) and $TS(10)_{L(M,b)}$ is in the busy list, BTL(M, b), $CN(10)_{L(a,M,b)}$ is set to 2.

E4 If $PL(i)_{L(W,X,Y,Z)}$ is denoted the priority list of the 'time-slot chosen' strategy, then, $PL(i)_{L(W,X,Y,Z)} = CN(i)_{L(W,X,Y)} + CN(i)_{L(X,Y,Z)}$. The higher value in the sum number list, $PL(i)_{L(W,X,Y,Z)}$, will be chosen first by the property of lower influence capability. For example as shown in Figure 10(c), $TS(10)_{L(a,M)}$ and $TS(11)_{L(a,M)}$ are the highest priority objects.

Figure 10 Example of the computation of the priority credit value



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
L(**S**, a)	F	F	B	B	F	F	F	F	F	F	F	F	F	F	B	B
L(a,**M**)	F	F	B	B	B	B	F	F	B	B	F	F	F	F	B	B
$CN(\delta)_{L(S,a,M)}$	1	1	0	0	0	0	1	1	0	0	1	1	1	1	0	0

(a)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
L(**M**,b)	F	F	B	B	B	B	F	F	F	F	B	B	F	F	B	B
L(a,**M**)	F	F	B	B	B	B	F	F	B	B	F	F	F	F	B	B
$CN(\delta)_{L(a,M,b)}$	1	1	0	0	0	0	1	1	0	0	2	2	1	1	0	0

(b)

$CN(\delta)_{L(S,a,M)} + CN(\delta)_{L(a,M,b)}$	2	2	0	0	0	0	2	2	0	0	3	3	2	2	0	0

(c)

As per specifications of Bluetooth scatternets (Bluetooth Special Interest Group, 2003; Kardach, 2000; Whiteaker et al., 2004), the capacity definitions of $DM5$, $DM3$ and $DM1$ packet types are 224, 121, and 17 bytes, respectively. If the QoS connection requirement is 200 bytes/cycle time and the three inequalities ($1 \times 224 = 224 > 200 > 0 \times 224 = 0$), ($2 \times 121 = 242 > 200 > 1 \times 121 = 121$) and ($12 \times 17 = 204 > 200 > 11 \times 17 = 187$) are established, then we can use either one packet of the $DM5$ packet type, two packets of the $DM3$ packet type, or twelve packets of the $DM1$ packet type to satisfy the QoS requirement under the three inequalities. In particular, the $DM1$ and $DM3$ packet types of all the examples in this paper are the only chosen types that are adopted to support the QoS connection requirement, as illustrated in Figure 3(b). If the new QoS connection requirement is 51 bytes/cycle time, then one $DM3$ packet type and three $DM1$ packet types are available for the request. Now, a path, (M_0, s_0, M_1, s_1, ..., W, X, Y, Z, ..., M_i), named PA and a priority list, $PL(i)_{L(W,X,Y,Z)}$, are given. PA is received by the designated destination node. Then, the time-slot reservation algorithm is described as follows.

F1 Randomly select a link, $L(X, Y)$, with minimum shared free time-slot s from PA and then divide the PA into two subpaths, $PA_1 = (M_0, ..., X)$ and $PA_2 = (Y, ..., M_i)$, where $PA = PA_1 + PA_2$. For example as displayed in Figure 6(d), $L(a, b)$ and $L(c, D)$ are the links that share a minimum number of free time slots. Then, we can randomly choose one link from those two links, and link L(a, b) is chosen in this case.

F2 For the QoS connection requirement, the priority list, $PL(i)_{L(W,X,Y,Z)}$, is used and the $DM5$ packet type is first adopted to try the QoS connection requirement if the number of free time-slot s is sufficient. Continuing, the $DM3$ packet type is tried to see if it can satisfy the QoS requirement if the $DM5$ packet type failed. If it still does not satisfy the QoS requirement, the $DM1$ packet type is tried to see if it can satisfy the QoS requirement.

F3 Recursively perform the time-slot reservation operations of steps F1 and F2 until all of the subpaths are processed and all of the paths that are received by the designated destination node are processed.

When the time-slot reservation phase is recursively performed and the time slots with the QoS connection requirement are reserved, the designated destination node sends the acknowledgement packet (RREP) back to the designated source node to reserve the time slots along the chosen path. All unselected time slots of all existing routing paths that were received by the designated destination node are released. The detailed optimal priority based algorithm of the time-slot reservation phase follows the rules of Chen et al. (2004). Without loss of generality, a path (M_0, s_0, M_1, s_1, ..., W, X, Y, Z, ..., M_i), named PA and a priority list, $PL(i)_{L(W,X,Y,Z)}$, are also given and the modified rules are presented as follows.

G1 Randomly select link L(X, Y) with minimum shared free time-slot s from PA and then divide PA into two subpaths, $PA_1 = (M_0, ..., X)$ and $PA_2 = (Y, ..., M_i)$, where $PA = PA_1 + PA_2$.

G2 Assign a different priority value that indicates the affected degree with neighbouring links to each link collected in the preceding two phases, the *available QoS routing-path search phase* and time-slot *leasing-based path-recovery phase.*

G3 Apply the priority based scheme to reserve time slots on link $L(X, Y)$.

G4 Recursively execute the time-slot reservation operations of steps G1 and G2 on all paths received by the destination node until all of the links have been selected for time-slot reservation.

Figure 11 shows a traversal tree, also called a time-slot *reservation tree* (TSR-tree), to present all of the recursive operations without using the time-slot leasing-based approach. Figure 12(a) shows the onehop time-slot leasing-based TSR tree and routing graph to present all of the recursive operations when the onehop time-slot leasing-based approach is adopted. Figure 12(b) shows the twohop time-slot leasing-based TSRtree and its routing graph.

Figure 11 Conditions of time-slot reservation with a normal optimal algorithm

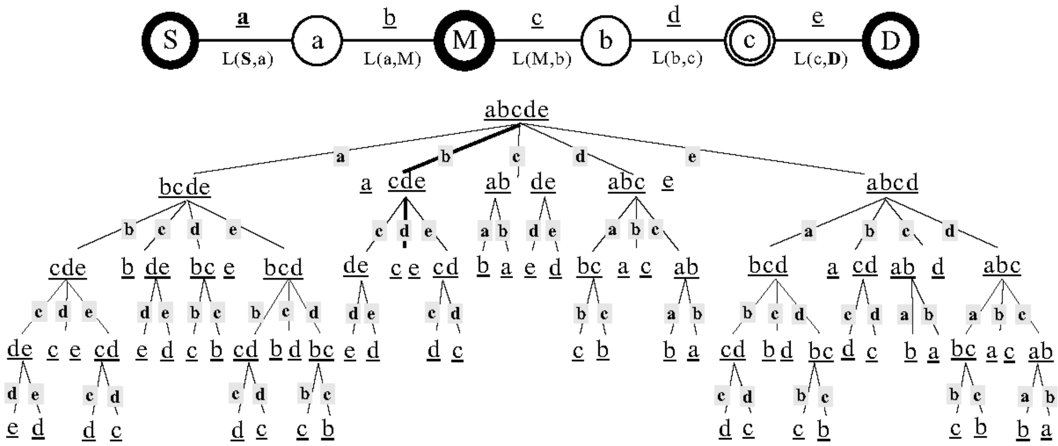
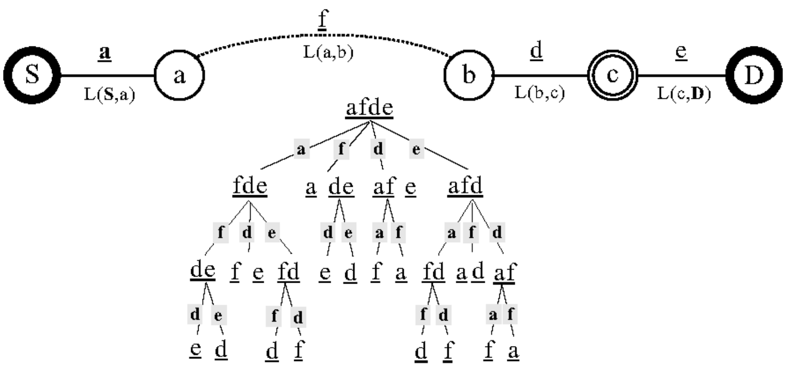
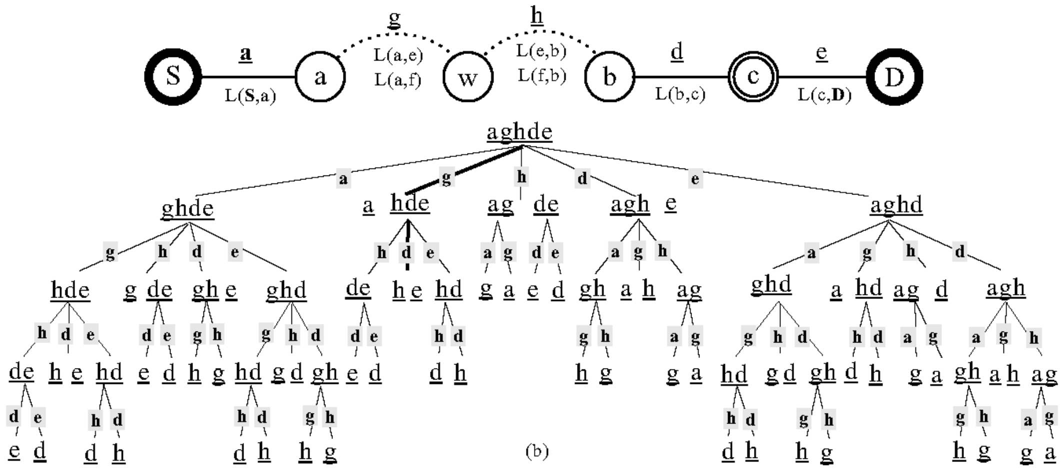


Figure 12 Conditions of time-slot reservation with the (a) onehop and (b) twohop TSLbased optimal algorithm





4 DTQ (Distributed time-slot leasing-based QoS) routing protocol

Due to the usable time-slot information stored in the QoS_REQ lists and kept in the designated destination node, our CTQ routing protocol can support the optimal algorithm of time-slot reservation. To truly become the optimal algorithm, we still have a problem which must be solved. The fact that all of the usable paths need to be stored in the designated destination node has been confronted with the new scalability problem. To reduce the scalability problem, we propose a *distributed* time-slot *leasing-based QoS* (DTQ) routing protocol. The DTQ routing protocol adopts the hop by hop priority based routing algorithm to overcome the scalability problem of the CTQ routing protocol. In comparison with the CTQ routing protocol, the DTQ routing protocol first floods the QoS_REQ packet to the designated hops and performs the time-slot *reservation* (TSR) operation. In this paper, we adopt a threehop flooding strategy to resolve the scalability problem. For every threeroute path passed, the *time-slot reservation packet* (TSRP) is replied back to the preceding nodes to confirm the TSR message, as defined in Chen et al. (2004). A QoS routing path is completely constructed when the operations are repeated and arrive at the designated destination node. In the DTQ routing protocol, the new parameter, DHC (*distributed hops counter*) is added to the QoS_REQ packet to limit the hop numbers in the routing path of each run. For example as adopted in this paper, if the parameter DHC in the DTQ routing protocol is set to 3, then each action of the QoS consideration floods three hops. The QoS_REQ packet of the DTQ routing protocol is defined as QoS_REQ (DS_ADR, DD_ADR, QoS_RQ, Cur_Free_TS, Path_List, Free_TS_link, Time_to_Live, Min_FTP, DHC). In addition to QoS_REQ, another replied packet, TSRP, is modified from Chen et al. (2004)'s definition and defined as TSRP (CN_ADR, PN_ADR_List, PN_Info, P_TS_Info, Time_to_Live). The detailed definition of the TSRP packet is shown in Table 2. In this section, we combine the three phases of the CTQ algorithm and form the DTQ algorithm. The detailed steps of the DTQ algorithm are formally described as follows.

Table 2 Detailed definition of the TSRP packet

Packet field	*Field description*
CN_ADR	Address of the current node
PN_ADR_List	List of nodes in each round that is acquired from the Path_List
PN_Info	Node information of PN_ADR_List
P_TS_Info	Information of reserved time slots for the preceding nodes of the same round
Time_to_Live	Limitation of hop numbers within the designated DHC hops

A'1 The designated source node initially broadcasts the QoS_REQ (DS_ADR = *S*, DD_ADR = *D*, DQ_REQ = *QR*, Cur_Free_TS = $\{\alpha_1, \alpha_2, ..., \alpha_{n_1}\}$, Path_List = {}, Free_TS_Link = {}, Time_to_Live = *HN*, Min_FTP = 0, DHC = 0) into a Bluetooth scatternet, where *S* is the designated source node, *D* is the designated destination node, *QR* is the QoS requirement and *HN* is the hop number of the 'time to live'.

A'2 If the intermediate node, N_c, within the designated DHC hops receives the QoS_REQ (*S*, *D*, *QR*, *Cur_FT*, *Cur_PL*, *Cur_FTL*, *Cur_HN*, *MinPK*, *Cur_DHC*), where *Cur_FT* is the currently free time slots, *Cur_PL* is the current path list, *Cur_FTL* is currently free time slots of the link, *Cur_HN* is the remaining useable hop number, *MinPK* is the packet number of the current minimum available free time slots, and *Cur_DHC* is the current value of DHC, then, the current node, N_c, checks the values of *Cur_DHC*, *Cur_HN* and *D* and then follows the suitable cases step by step as described below.

B'1 If current node N_c is the destination node, *D*, then jump to A'3.

B'2 If current node N_c is not the destination node, *D* and the remaining hop number, *Cur_HN*, is equal to zero, then drop the current QoS_REQ packet and reject the algorithm.

B'3 If current node N_c is not the destination node and the shared free time slots $\{\alpha_1, \alpha_2, ..., \alpha_{n_1}\}$ between N_c and the corresponding node, assumed to be $N_{c'}$, in the *Cur_FTL* cannot satisfy the QoS value, *QR*, then keep the failed link, $\overrightarrow{N_c N_{c'}}$ and jump to B'6.

B'4 If the *Cur_DHC* is smaller than DHC, current node N_c calculates the shared free time slots $\{\alpha_1, \alpha_2, ..., \alpha_{n_1}\}$ of link $\overrightarrow{N_c N_{c'}}$ and adds useable slot information into the field of Free_TS_Link. The value of *Cur_DHC* is increased by 1. Moreover, the values of (*S*, *D*, *QR*, {the free time slots of current node N_c}, *Cur_PL* ∪ {N_c}, *Cur_FTL* ∪ $\{\alpha_1, \alpha_2, ..., \alpha_{n_1}\}$, *Cur_HN*-1, *new updated MinPK, Cur_DHC* + 1) are updated to the QoS_REQ packet and the updated QoS_REQ packet is broadcast to all neighbouring nodes of node N_c.

B'5 If the *Cur_DHC* is equal to DHC, in this paper the value of DHC is set to 3, then current node N_c jumps and performs steps A'3 and A'4 to execute time-slot reservations. If this step is finished then jump to step A'3. Return to step B'1 to continue the process.

B'6 If the failed link is $\overleftrightarrow{N_c N_{c'}}$, then collect all of the slave-to-slave information by the following sub-steps.

C'1 If N_c is the master node, then collect the link information of all links from all other slave nodes to slave node $N_{c'}$ in the same piconet with master node N_c.

C'2 If $N_{c'}$ is the master node, then collect the link information of all links from slave node N_c to the other slave nodes in the same piconet with master node $N_{c'}$.

B'7 If all of the links have failed then drop the QoS_REQ and break the algorithm by returning the message of failed QoS connection requirement.

B'8 The 'fewer hops first strategy' will be first adopted to choose the time-slot leasing-based replacement path. If more than one available link exists then choose the group that possesses the lower hop number.

B'9 When the group with a fewer hop number is chosen, the 'maximum free packet first' strategy will continuously choose the time-slot leasing-based replacement path. If more than one available link exists in the chosen group, then the link with the maximum number of free packets is chosen.

B'10 When the replacement link is chosen, return to B'1.

A'3 With using the Free_TS_Link of QoS_REQ, all free time slot information in every round can be acquired by the current node, *Cur_DHC.* The time-slot reservation operations in every round are described as follows.

D'1 Randomly select a link, $L(X, Y)$, from PA with a minimum number of shared free time slots and then divide the PA into two subpaths, PA_1 and PA_2, where $PA = PA_1 + PA_2$. Because the DHN is set to 3 in this paper, the equation of PA is $PA = L(W, X) + L(X, Y) + L(Y, Z)$. By the way, the three links, $L(S, a)$, $L(a, M)$ and $L(M, b)$, form one of the threehop links in the first round of time slot reservation, as shown in Figure 13(a). Another example is also illustrated in Figure 13(a), in which the three links, $L(S, e)$, $L(e, M)$ and $L(M, f)$, still form one of the threehop links in the first round of time-slot reservation.

D'2 The priority credit value of $PL(i)_{L(W,X,Y,Z)} = CN(i)_{L(W,X,Y)} + CN(i)_{L(X,Y,Z)}$ is still repeatedly calculated to perform the time-slot reservation.

A'4 After the TSR (*time-slot reservation*) steps are successful reserved, the following operations are continually executed and repeated until all the information is completely processed.

E'1 The current node, N_c, recursively updates the values of (S, D, QR, {the free time slots of current node N_c}, $Cur\_PL \cup \{N_c\}$, $Cur\_FTL \cup \{\alpha_1, \alpha_2, ..., \alpha_{n_1}\}$, Cur_HN-1, *new updated MinPK, Cur_DHC* + 1) to the QoS_REQ packet and broadcasts it to the next nodes until the designed destination node receives the final QoS_REQ. When the Time_to_Live is equal to zero, then the recursively updating operation is terminated.

E'2 The current node, N_c, replies the TSRP packet back to the preceding nodes, which in this paper is three nodes, to confirm time-slot reservation.

The completed processing steps of the onehop DTQ algorithm are graphically presented and tabulated as displayed in Figures 13 and 14. In Figures 13 and 14, the DHC is set to 3, and the final routing path is $S \rightarrow a \rightarrow b \rightarrow c \rightarrow D$.

5 Performance analysis and comparison results

In this section, we implement five algorithms to verify our TQ routing protocol's analytic observations. In the following simulator, we use 'CTQ', 'DTQ', 'CCQ', 'DCQ' and 'KIM' to denote our CTQ algorithm, our DTQ algorithm, Chen et al. (2004)'s CCQ algorithm, Chen et al. (2004)'s DCQ algorithm and Kim et al., (2003) algorithm, respectively. C++ simulation programs were developed to achieve the requirements of five algorithms. The simulation programs use the same environments of Network Simulator (ns-2) and BlueHoc (IBM Bluetooth simulator). The system parameters are given in Table 3. The performance metrics of the simulations are given as follows.

- *success rate*: the value of the sum of successful QoS route requests divided by the sum of total QoS route requests
- *throughput*: the value of all data bytes received by all devices per unit time.

Figure 13 Onehop DTQ approach

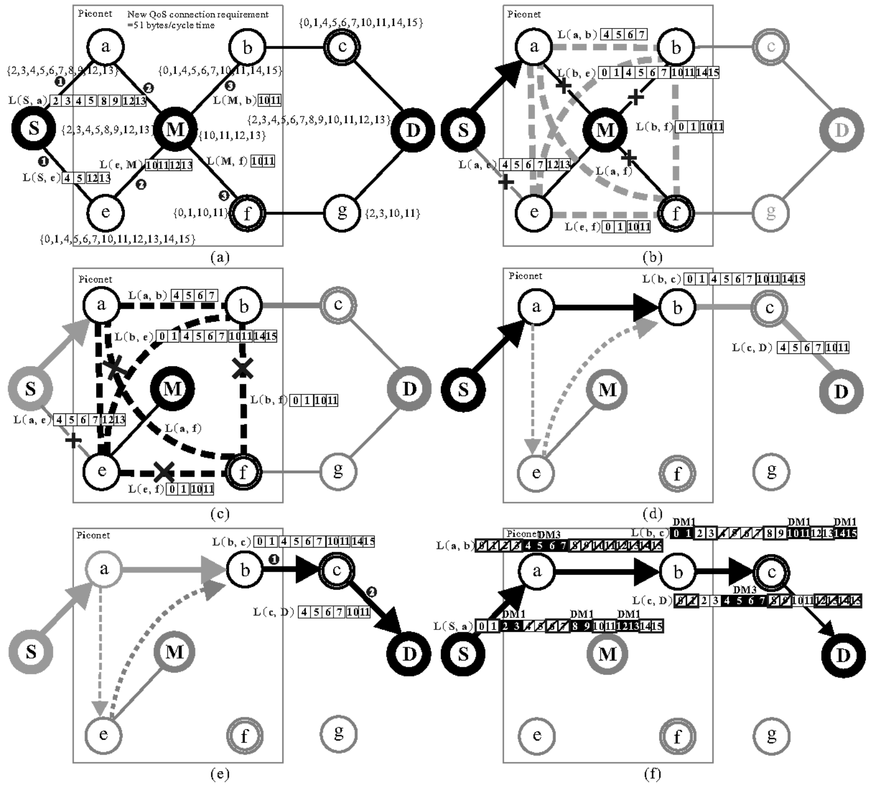


Figure 14 Time-slot reservation steps by the onehop DTQ approach

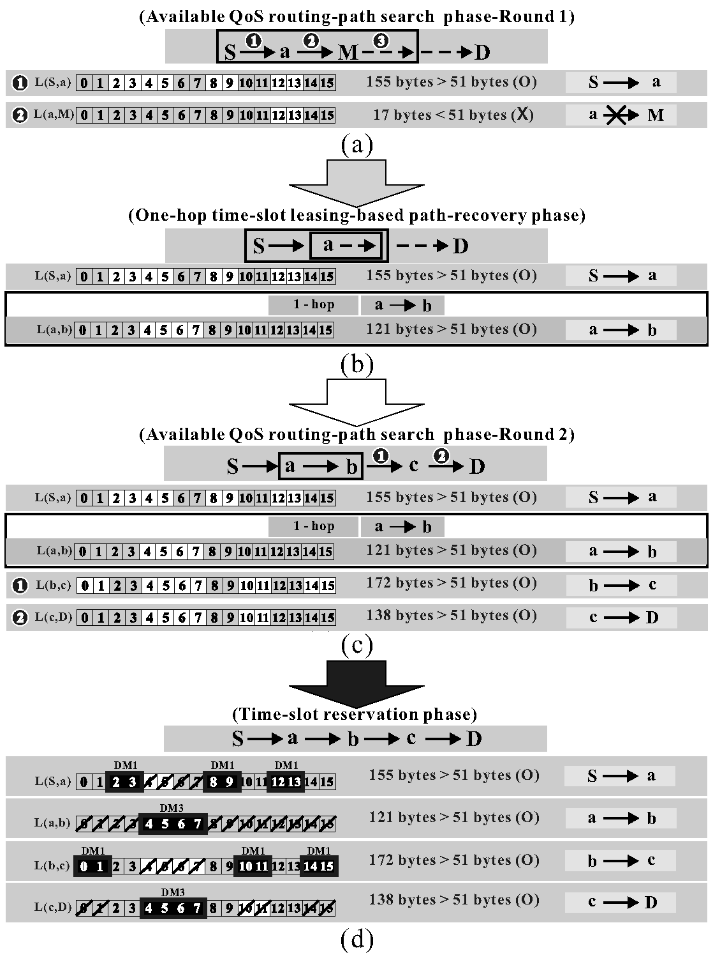


Table 3 Detailed simulation parameters

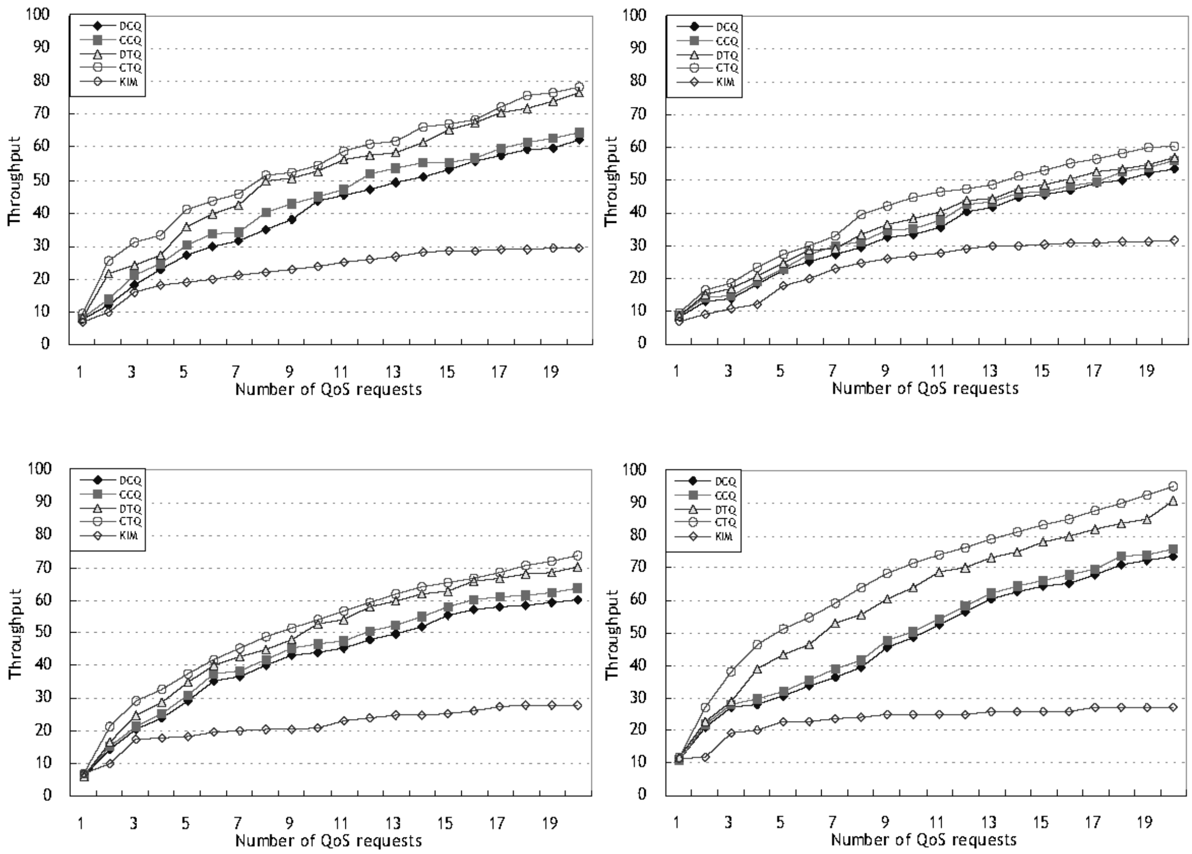
Parameter	*Value*
Number of devices	18
Network region	17×17 m
Radio propagation range	10 m
Mobility	No
Cycle time	16 time slots
Packet type	*DM*1 or *DM*3 or *DM*5
QoS requirement	$DM1:DM3:DM5 = 1:1:1$
	$DM1:DM3:DM5 = 3:1:1$
	$DM1:DM3:DM5 = 1:3:1$
	$DM1:DM3:DM5 = 1:1:3$

A high success rate, high throughput, high bandwidth utilisation, and low slot occupation are the main issues of an efficient QoS routing protocol in Bluetooth WPANs. In the following, we demonstrate that our time-slot leasing-based QoS routing protocol can significantly improve the success rate, throughput, slot occupation and bandwidth efficiency from several prospects.

5.1 Performance of success rate

We investigate the effects of various numbers of QoS requests on the success rate. As illustrated in Figures 15(a)–(d), the four QoS requirement scenarios are *DM*1:*DM*3:*DM*5 = 1:1:1, 3:1:1, 1:3:1 and 1:1:3, respectively. The same compared results of CCQ, DCQ, and KIM were described in Chen et al. (2004)'s simulations. CCQ and DCQ had higher success rates than KIM and the success rate of KIM rapidly decreased. This is because KIM wastes too many POLL time-slot s by using the *DM*1 packet, which consumes most of the time slots (Chen et al., 2004). In general, the success rate of our CTQ was greater than that of Chen et al. (2004)'s CCQ. Similarly, the success rate of our DTQ was greater than that of Chen et al. (2004)'s DCQ. This is because that our CTQ routing protocol additionally applied the slave-master-slave network to construct the QoS routing path. In case the connection request fails, the routing is immediately switched to the backup slave-to-slave time-slot leasing-based routing paths to support the transmission QoS. The success rate of CTQ > that of DTQ > that of CCQ > that of DCQ > that of KIM, as illustrated in Figure 15. On the other hand, most time slots were free for request and the success rate was almost 100% in the initial stage of simulation. Higher QoS requests produced lower success rates. Figures 15(b)–(d) show that the values of success rates in average case are 3:1:1 > 1:3:1 > 1:1:3. This is because, if the frequency of a high traffic pattern is higher than that of a low traffic pattern, then the system throughput increases, the time slot occupation increases, and the success rate decreases.

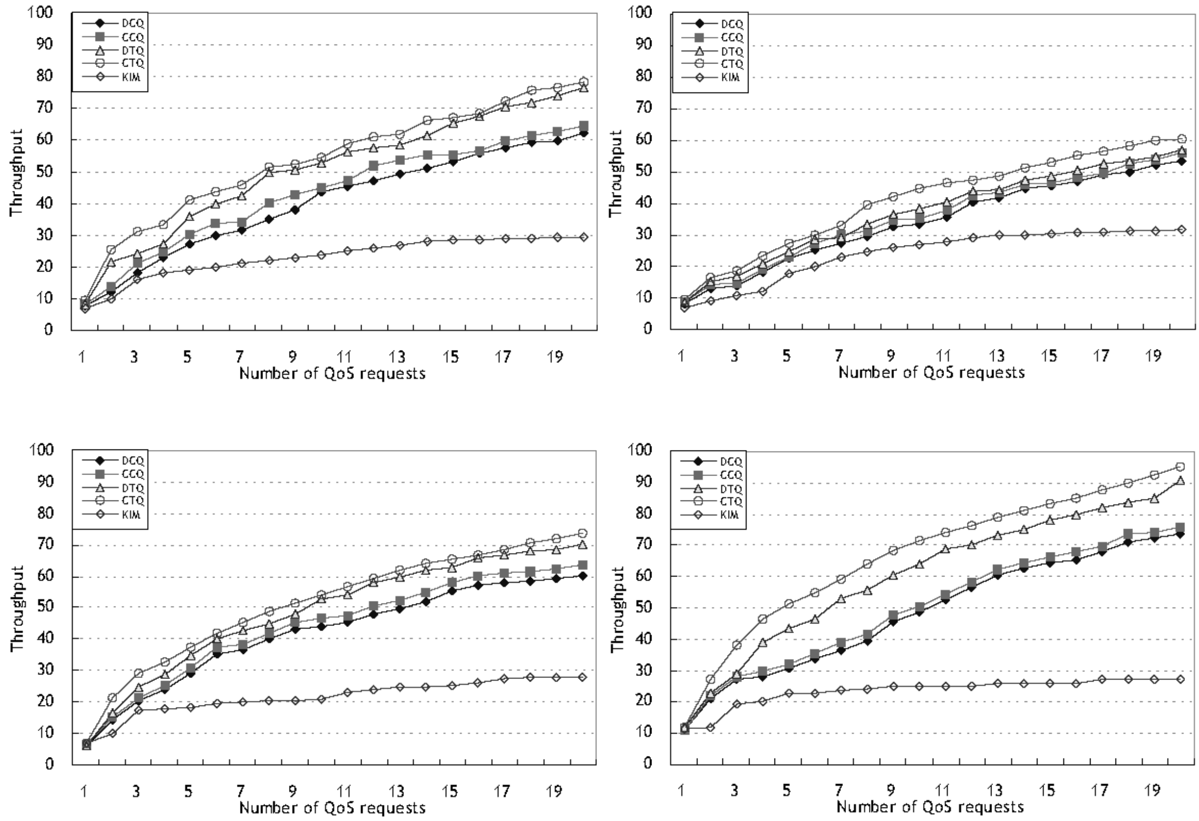
Figure 15 Success rate vs. the number of QoS requests (a) $DM1:DM3:DM5 = 1:1:1$, (b) $DM1:DM3:DM5 = 3:1:1$, (c) $DM1:DM3:DM5 = 1:3:1$ and (d) $DM1:DM3:DM5 = 1:1:3$



5.2 Performance of throughput

Throughput is the value of all the data bytes received by the entire device per unit time. The throughput of our CTQ was greater than that of CCQ, and the throughput of our DTQ was also greater than that of DCQ, as illustrated in Figure 16. This is because the success rate of our CTQ is greater than that of CCQ and the success rate of our DTQ is greater than the success rate of DCQ, as displayed in Figure 15. For instance as shown in Figure 15(a), when the number of QoS requests exceeds than ten, the success rate of CTQ is trivially greater than that of CCQ, then the throughput of CTQ is still greater than that of CCQ as illustrated in Figure 16(a). On the other hand, in the four QoS requirement scenarios, the values of throughput in an average case are 3:1:1 < 1:3:1 < 1:1:3, as displayed in Figures 16(b)–(d). This is because the numbers of transferred packets are 3:1:1 < 1:3:1 < 1:1:3. For example, as shown in Figure 16(b), the ratio of the traffic pattern is *DM*1:*DM*3:*DM*5 = 3:1:1. The throughput of CTQ is nearly 60 kbytes/s and this value is greater than the 55 kbytes/s of CCQ when the number of QoS requests is equal to 20. Figure 16(d) shows that the ratio of the traffic pattern is *DM*1:*DM*3:*DM*5 = 1:1:3. The throughput of CTQ is nearly 95 kbytes/s and is greater than the CTQ values of Figure 16(b) and (c). Generally speaking, the results of throughput are CTQ > DTQ > CCQ > DCQ > KIM and *DM*1:*DM*3:*DM*5 = 3:1:1 < *DM*1:*DM*3:*DM*5 = 1:3:1 < *DM*1:*DM*3:DM5 = 1:1:3.

Figure 16 Throughput vs. the number of QoS requests (a) *DM*1:*DM*3:*DM*5 = 1:1:1, (b) *DM*1:*DM*3:*DM*5 = 3:1:1, (c) *DM*1:*DM*3:*DM*5 = 1:3:1 and (d) *DM*1:*DM*3:*DM*5 = 1:1:3



6 Conclusions

To provide the QoS service, our integrated interpiconet scheduling approach, named the TQ routing protocol, which offers QoS guarantees of Bluetooth scatternets in WPANs, is presented in this paper. The slave-master-slave communication is the regular data transmission operation in Bluetooth networks and the master node is the communication bottleneck. To alleviate the 'transmission holding' problem, we have proposed a new time-slot leasing-based scheme to provide extra slave-to-slave QoS communication capability to reduce the workload of master nodes and lower the missing rate of QoS requests. Finally, our simulation results have demonstrated that the time-slot leasing-based QoS routing protocol can significantly improve the success ratio, delay time, throughput, and bandwidth utilisation.

References

Bhargava, A.S. and Gruenbacher, D.M. (2004) 'Using piconet avoidance techniques to reduce interference in Bluetooth networks', *Proceedings of IEEE Consumer Communications and Networking Conference (CCNC)*, 5–8th January, pp.65–70, Las Vegas, Nevada, USA.

Bluetooth Special Interest Group (2003) *The Bluetooth Specification Version 1.2*, http://www.bluetooth.com.

Chen, Y.S., Liu, K.S. and Lin, Y.W. (2004) 'A credit-based on-demand QoS routing protocol over Bluetooth WPANs', *Proceedings of the 10th IEEE Symposium on Computers and Communications (IEEE ISCC 2005)*, to appear.

Cordeiro, C., Abhyanker, S. and Agrawal, D.P. (2004) 'Design and implementation of QoS-driven dynamic slot assignment and piconet partitioning algorithms over Bluetooth WPANs', *Proceedings of the Twenty-third Annual Joint Conference of the IEEE Computer and Communications Society (INFOCOM)*, 7–11th March, Hong Kong, China.

Joo, Y.I., Lee, T.J., Eom, D.S., Lee, Y. and Tchah, K.H. (2003) 'Power-efficient and QoS-aware scheduling in Bluetooth scatternet for wireless PANs', *IEEE Transactions on Consumer Electronics*, Vol. 49, No. 4, pp.1067–1072.

Kardach, J. (2000) 'Bluetooth architecture overview', *Intel Technology Journal*, May, Vol. 4, No. 2, pp.1–7.

Kim, Y.M., Lai, T.H. and Arora, A. (2003) 'A QoS-aware scheduling algorithm for Bluetooth scatternets', *Proceedings of IEEE the International Conference on Parallel Processing*, 6–9 October, pp. 455–462.

Whiteaker, R.M., Hodge, L.E. and Chlamtac, I. (2004) 'Bluetooth scatternet formation: a surve', *The Journal of Ad Hoc Networks*, July, Vol. 3, No. 4, pp.403–450.

Zhang, W., Zhu, H. and Cao, G. (2001) *On Improving the Performance of Bluetooth Networks Through Dynamic Role Management*', *Technical Report*, CSE-01-018 Pennsylvania State University, also available at http://www.cse.psu.edu/gcao/paper/bluetooth.ps.

Zhang, W., Zhu, H. and Cao, G. (2002) 'Improving Bluetooth network performance through a time-slot leasing approach', *Proceedings of Wireless Communications and Networking Conference (WCNC)*, Vol. 2, pp.592–596.