在干擾環境下具適應性的藍芽封包
選擇及排程策略

學生：施辰翰　　　指導教授：王國禎 博士

國立交通大學資訊科學系

摘 要

藍芽是一個應用於無線個人區域網路的技術。它旨在消除各種裝置之間裝設電纜和連接器的需求。它可應用在個人電腦及其週邊設備，隨身聽及其連接的耳機等。藍芽提供具強韌性、安全性且支援數據和聲音的無線電通訊，而這些裝置不需要彼此都在視線之內。藍芽使用2.4 GHz ISM 頻帶，與IEEE 802.11系列的無線區域網路共用同一頻帶。此兩種網路本是互補而非互相競爭關係，然而它們互相干擾可能會不預期發生，此將嚴重降低彼此效能。在本論文中，我們提出根據頻道狀況進行藍芽資料分割和重組的方案(CSD-SAR)及根據佇列

狀況作排程的策略(QSD-PR)。CSD-SAR會建構一個接收頻道表(receiving frequency table)，並根據此表去預測頻道狀況及選擇最適合的封包格式和封包大小來傳輸資料。透過這個方法，不僅可以在不延遲傳輸下，避用不好的頻道，並且在易於發生錯誤的環境下有較好的鏈路使用率及較好的效能表現。此外，QSD-PR也利用這個接收頻道表去避免在不好的頻道傳送封包，且給予彼此之間有較多資料要傳送的主從對（master-slave pair）較高的優先權，以避免時槽的浪費。常見的排程策略如輪詢排程(RR)，在分時雙工(TDD)的存取控制通訊協定下，不能提供較好的效能，此造成時槽的浪費，也不能確保公平性。模擬結果顯示，與RR比較，無論在無錯誤或易於發生錯誤的環境下，我們提出的封包選擇及排程策略能達到較好的鏈路使用率及較高的效能。我們提出的方案因為避免使用其他網路所佔用的頻道，所以也能消除對其他共用同一頻帶的無線區域網路之干擾。

關鍵詞：藍芽、適應性封包選擇方案、根據頻道狀況排程。

An Adaptive Bluetooth Packet Selection and Scheduling Scheme in Interference Environments

Student：Chen-Han Shih **Advisor：Dr. Kuochen Wang**

Department of Computer and Information Science
National Chiao Tung University

Abstract

Bluetooth is a new technology for Wireless Personal Area Networks (WPANs). It intends to eliminate the need of wires and connectors between a variety of devices, like PCs and their peripherals, walkmans and their earphones, and etc. Bluetooth provides robust and secure wireless radio communication of both data and voice, even when the devices are not within line-of-sight. Bluetooth employs the 2.4 GHz ISM band, sharing the same band with the Wireless LAN (WLAN) implementing the IEEE 802.11 series standard. While WLANs and WPANs are complementary rather than competing technologies, the likelihood of mutual interference may occur unexpectedly, which may impact the performance of either severely. In this thesis, we propose a Bluetooth *channel state dependent data segmentation and reassembly* (CSD-SAR) scheme and a *queue state dependent priority* (QSD-PR) scheduling policy. The CSD-SAR maintains a receiving frequency table to predict channel conditions and selects the best packet type and packet size to transmit data. In this way, it not only masks bad frequencies without delaying transmission but also leads to the best performance with high link utilization in error-prone environments. In addition, the

QSD-PR also uses the receiving frequency table to avoid bad frequencies and gives a selected master-slave pair, which has more queued data to send between each other, a higher priority to eliminate the wastage of slots. The conventional scheduling policy, *Round Robin* (RR), yields poor performance with the *time division duplex* (TDD) based MAC protocol and results in slot wastage and may not ensure fairness. Simulation results show that our proposed scheme achieves better link utilization and higher throughput with bounded delay compared to the RR scheme in error-free and error-prone environments. Our scheme can also eliminate interference to other wireless networks that share the same spectrum, such as WLANs, by avoiding selecting channels occupied by other networks.

Keywords：Bluetooth, adaptive packet selection, channel state dependent packet scheduling, interference environment.

Acknowledgements

Many people have helped me with this thesis. I am in debt of gratitude to my thesis advisor, Dr. Kuochen Wang, for his intensive advice and instruction. I would like to thank all the classmates in *Mobile Computing and Broadband Networking Laboratory* for their invaluable assistance and suggestions. The support by the National Science Council under Grant NSC90-2213-E-009-162 is also grateful acknowledged.

Finally, my families deserve special mention. This thesis is dedicated to them for their support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the need for new mobility arises, the devices of Wireless Personal Area Networks (WPANs) and Wireless Local Area Networks (WLANs) will increase in a rapid pace. The WPAN category is led by a short-range radio technology called Bluetooth [1][2], which was designed primarily for cable replacement applications. The WLAN category has several technologies competing for dominance, like IEEE 802.11a/b/g [3], HomeRF [4], HiperLAN/2 [5], and etc. Bluetooth and existing Wireless LANs (only IEEE 802.11b is discussed in this thesis owing to its popularity) have a number of distinctive features. Bluetooth uses the Frequency Hopping Spread Spectrum (FHSS) scheme and hops over 79 1-MHz-wide channels by 1600 times per second while IEEE 802.11b uses the Direct Sequence Spread Spectrum (DSSS) scheme and occupies one 22-MHz-wide static channel across the acceptable 83.5 MHz of the 2.4 GHz ISM band. Bluetooth was designed for personal area networking that transmits at power level of about 1 mW and IEEE 802.11b was designed for wireless local area networking with power level from 30 to 100 mW.

Both WPANs and WLANs share the same 2.4 GHz unlicensed frequency band and provide complimentary wireless solutions for connectivity. This complimentary nature of the services could enhance the use of both protocols at the same physical location and provide an incentive for their adoption.

Recently, the issue of designing coexistence mechanisms between WLANs and WPANs has received much attention because both may suffer strong interference from each other [6][7]. Some interference reduction techniques such as power control adjustments [8], channel state dependent error avoidance schemes [9][10], collaborative schemes, and adaptive frequency hopping [11] were proposed. A scheduling algorithm was proposed in [9] that used a Frequency Usage Table to distribute channels to devices and ensures fairness of access among users by means of max-min fairness criteria. In [10], a Link State History based scheme was proposed to achieve high accuracy in identifying the good and bad periods of the channels. However, the packet selection scheme in Bluetooth also has a significant effect on data scheduling and network performance. It controls the distribution of packet types and packet sizes that may result in different probabilities of packet loss. For this reason, we propose a channel state dependent packet selection scheme and a simple priority scheduling policy that takes queue states and channel conditions into account to maximize link utilization while ensuring a high throughput and low packet error rate in interference environments. In addition, the simulation models in [9] and [10] are restricted to the link layer and is not optimized for transport layer sessions. We extend the simulation model to include not only the core Bluetooth protocol layers but also TCP/IP.

This thesis is organized as follows. Chapter 2 gives general insights on the Bluetooth technology. Chapter 3 presents our packet selection and scheduling scheme.

Chapter 4 shows simulation scenarios and simulation results of our proposed scheme and the performance is then evaluated. Finally, concluding remarks and future work are presented in Chapter 5.

Chapter 2

The Bluetooth System

Bluetooth was designed with the objective of small size, low power consumption, and low cost. Bluetooth has a range of 10 meters and provides a nominal data rate of 1 Mbit/s for wireless communications in a small area network. Two or more Bluetooth devices communicating on the same channel form a *piconet* [12], where one device operates as a *master* (generally means the unit that establishes the piconet) and the others act as the *slaves*. Up to seven slaves can be active in the piconet and the master is always responsible for defining and synchronizing the frequency hop pattern of the piconet.

2.1 Medium Access Control in Bluetooth

As shown in Fig. 2.1, the Time Division Duplex (TDD) scheme is used in the Bluetooth for resolving contention over wireless links. The master device controls data transmission through a polling procedure: periodically polls slave devices for information and only after receiving such a poll a slave is allowed to transmit. Thus, it is the master that determines which slave is scheduled when and how often.

The channel is divided into time slots, each 625 microseconds in length. The master is required to always start transmission on an even numbered slot while a specific slave on an odd numbered slot. The time slots, where each slot corresponds to an RF hop frequency, are numbered according to the Bluetooth clock of the piconet master. It should be noted that the Bluetooth clock has no relation to the time of day. Since transmission and reception take place at different time slots, transmission and reception also take place at different hop carriers. In order to support asymmetric links, devices have the option of transmitting a single packet lasting as much as five slots. The center frequency used for each packet does not change until that packet has ended, regardless of the number of slots the packet occupies and depends on the frequency at the time when the master begins sending the packet.

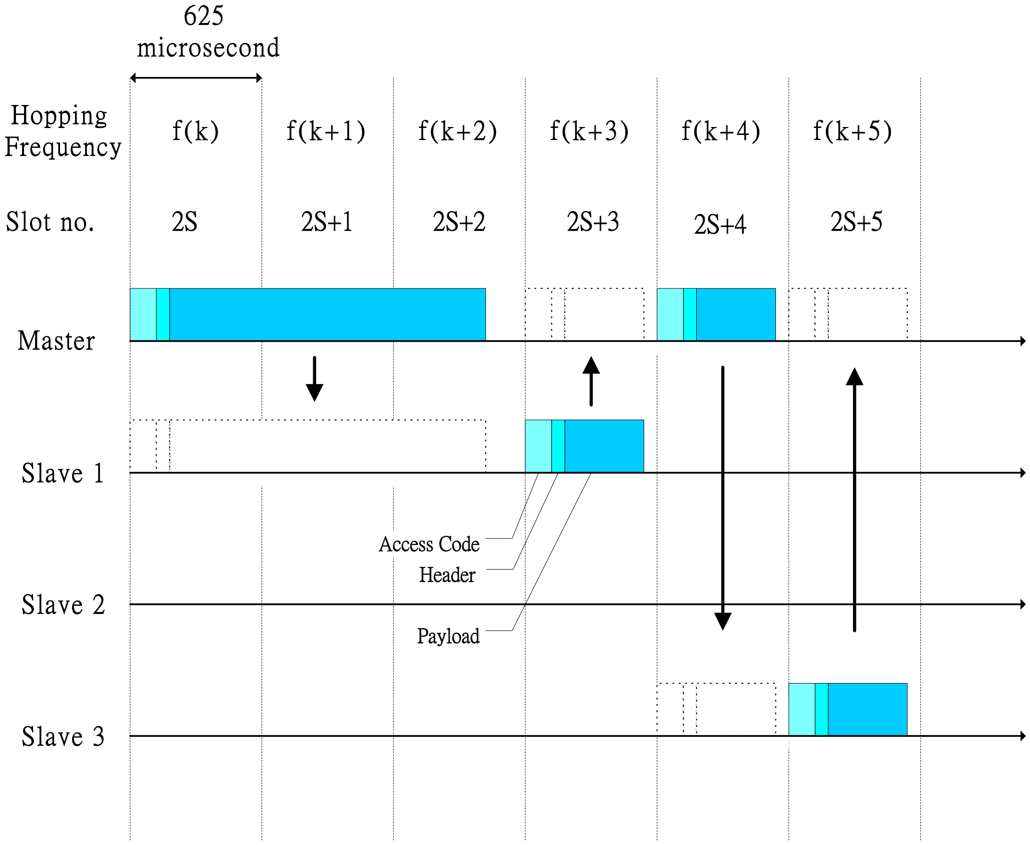

Fig. 2.1: Bluetooth channel structure.

2.2 Packet-Based Communications

The Bluetooth system uses packet-based transmission: the information stream is fragmented into packets. Packets can reserve one, three or five consecutive time slots for transmission. The standard packet format is shown in Fig. 2.2. Each packet has the same format, starting with an *access code*, followed by a packet *header*, and ending with the *payload*. The access code (72-bits) is used for synchronization and to identify packets in a piconet. The packet header consists of 18 bits and is encoded with a rate 1/3 FEC (Forward Error Correction) resulting in a 54-bit header. All packets sent in the same piconet are preceded by the same channel access code.

Bits	72	54	0-2745
	Access Code	Header	Payload

Fig. 2.2: Bluetooth standard packet format.

Bluetooth links support both synchronous services such as voice traffic and asynchronous services such as bursty data traffic. There are two types of physical links that can be established between the master and a slave [2]:

The *Synchronous Connection-Oriented* (SCO) link is designed to support real-time isochronous applications. The SCO link is a point-to-point link between the master and a specific slave. The link is established by reservation of duplex slots at regular intervals without being polled.

The *Asynchronous Connectionless* (ACL) link is used to exchange data in non-time-critical applications. The ACL link is a point-to-multipoint link between the master and all slaves on the piconet and can use all of the remaining slots on the channel not used for the SCO link. The traffic over the ACL link is scheduled by the master with the polling mechanism.

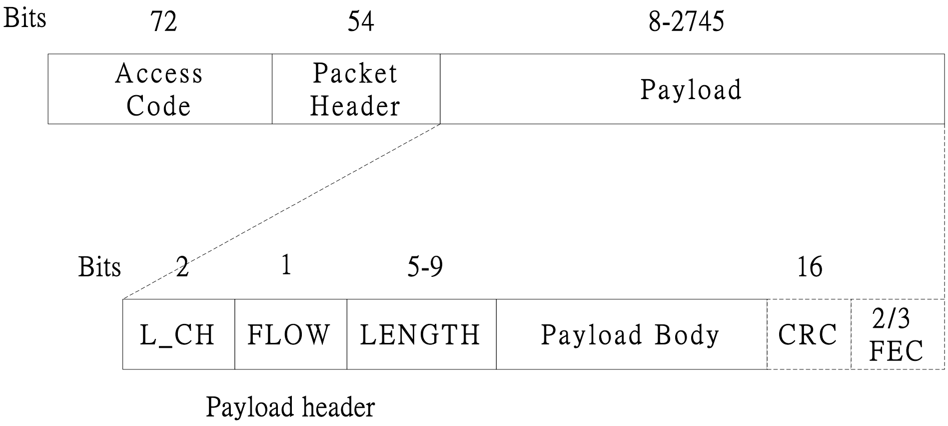

Fig. 2.3: ACL packet format.

In this thesis, the packet selection and scheduling only applies at the ACL link. Unlike SCO packets have a fixed payload length 240 bits and no payload header is present, the ACL packets have three segments in the payload: a *payload header*, a *payload body*, and possibly a *CRC code,* as shown in the Fig. 2.3. The payload header specifies the logical channel (2-bit L_CH indication), controls the flow on the logical channels (1-bit FLOW indication), and has a payload length indicator (5-bit LENGTH indication for single time-slot packets, 9-bit LENGTH indication for multi-slot packets) [2].

The ACL packets can be classified into two categories. (1) *DH packets* stand for Data-High rate packets and did not incorporate FEC code, and (2) *DM packet*s stand for Data-Medium rate packets and are protected with 2/3 FEC code to resist interference. That is, unlike DM packets, DH packets are not protected by the FEC code. The only error recovery used by DH packets is error detection through a 16-bit CRC combined with the ARQ (Automatic Repeat Request) scheme. The packet types of ACL packets are described as follows [2]:

DM1: The DM1 packet is a packet that carries data information only. The payload contains 18 information bytes, one of which is a payload header and added 16-bit CRC code. The payload in this packet type is protected by the (15, 10) Hamming code. DM1 packets occupy only one time slot.

DM3: DM3 packets are very similar to DM1 packets. The only difference is that, unlike the DM1 packet, the DM3 packet occupies three time slots. The extra length allows a DM3 packet to carry 123 data bytes, including a 2-byte header, with an added 16-bit CRC code.

DM5: DM5 packets are a variation of DM1 packet. The only difference between DM5 and DM1 is that DM5 packets occupy five time slots, allowing it to carry up to 226 information bytes, consisting a 2-byte header, 224 bytes of data, and 16-bit CRC code.

DH1: The DH1 packet can carry up to 28 information bytes (including the 1 byte payload header) plus a 16-bit CRC code. The DH1 packet occupies a single slot.

DH3: DH3 packets occupy 3 time slots and are very similar to DH1 packets. DH3 packets carry up to 185 information bytes, including a 2-byte payload header as well as a 16-bit CRC code.

DH5: DH5 packets are also a variation of DH1 packets. The only difference between DH1 and DH5 packets is that DH5 packets occupy five time slots. Its payload contains

2-byte header, 339 data bytes, and 16-bit CRC code.

Finally, a summary of the packets and their characteristics is shown in Table 2.1 [2]. The user payload represents the packet payload excluding FEC, CRC, and payload header.

Table 2.1: Summary of ACL packets

Type	Payload Header (bytes)	User Payload (bytes)	FEC	CRC	Symmetric Max. Rate (*Kb/s*)	Asymmetric Max. Rate (*Kb/s*)	
						Forward	Reverse
DM1	1	0-17	2/3	Yes	108.8	108.8	108.8
DH1	1	0-27	No	Yes	172.8	172.8	172.8
DM3	3	0-121	2/3	Yes	258.1	387.2	54.4
DH3	3	0-183	No	Yes	390.4	585.6	86.4
DM5	5	0-224	2/3	Yes	286.7	477.8	36.3
DH5	5	0-339	No	Yes	433.9	723.2	57.6

2.3 Bluetooth Protocol Stack

Fig. 2.4 shows the Bluetooth protocol stack [13]. The *Bluetooth Baseband* enables adjacent Bluetooth units to form a piconet. Bluetooth provides two different kinds of physical links (SCO link and ACL link) with their corresponding baseband packets. Note that one of the basic limitations of the Bluetooth Baseband protocol is that the packets that make up its transport service are size-limited. The Bluetooth *Logical Link Control and Adaptation Protocol* (L2CAP) layer adapts upper layer protocols over the

Baseband and provides *Segmentation and Reassembly* (SAR) operations to improve efficiency by supporting a *maximum transmission unit* (MTU) size larger than the largest baseband packet. The L2CAP permits higher-level protocols and applications to transmit and receive L2CAP data packets up to 64 kilobytes in length. This reduces overhead by spreading the network and transport packets used by higher layer protocols over several baseband packets. The primary data buffers in Bluetooth are at the L2CAP and at the Bluetooth Baseband. When the L2CAP fragments L2CAP packets into baseband packets, there is a separate ACL buffer for each slave at the master, and the scheduler decides which packet to send and how often.

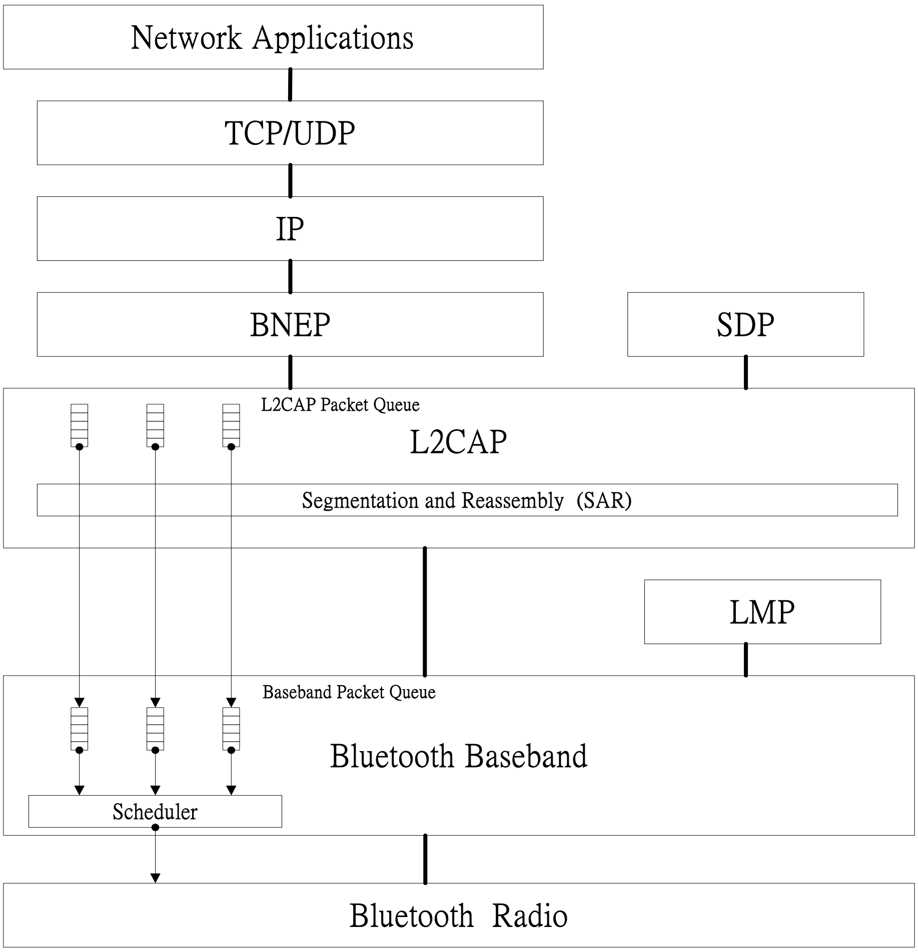

Fig. 2.4: Bluetooth protocol stack.

The *Link Manager Protocol* (LMP) in Fig. 2.4 is responsible for link-setup between Bluetooth devices. Furthermore, it controls the power modes and the connection states of a Bluetooth unit in a piconet. Discovery services are a crucial part of the Bluetooth framework. Using the Service Discovery Protocol (SDP), device information, services and the characteristics of the services can be queries and after that, a connection between two or more Bluetooth devices can be established [13].

Note that the Bluetooth Network Encapsulation Protocol (BNEP) [14] can encapsulate packets from various networking protocols, which are transported directly over the Bluetooth L2CAP protocol. The BNEP is used primarily in the Bluetooth Personal Area Networking Profile [15] to provide networking capabilities for Bluetooth devices.

Chapter 3

Proposed Packet Selection and Scheduling Scheme

3.1 Basic Idea

Using different packet types with different lengths and error protection properties results in different packet error rates in the same channel status. In an error-free environment, the DH5 packet would give the best performance [16] since it carries the most information bits per unit time. However, as the bit error rate increases, the resulting network performance will depend on the degree of forward error correction (FEC) and packet length [17][18][19]. The *packet error rate* (PER) of different ACL DH packet types can be expressed in terms of the *bit error rate* (BER) (assume the event of a bit error is independent of others):

$$PER(X) = 1 - (1 - BER)^m \tag{1}$$

where BER is the current bit error rate and m is the number of payload bits in packet type X, m = 240 for DH1, m = 1496 for DH3, and m = 2744 for DH5.

The payload of DM packets is protected by a (15, 10) Hamming code, which is capable of correcting one bit error per 15 bits code block. Similarly, we can also estimate the PER of DM packets from the BER as follows:

$$PER(X) = 1-((1-BER)^{15} + 15 \times BER \times (1-BER)^{14})^{M} \qquad (2)$$

where M =16 for DM1, M = 100 for DM3, and M = 183 for DM5.

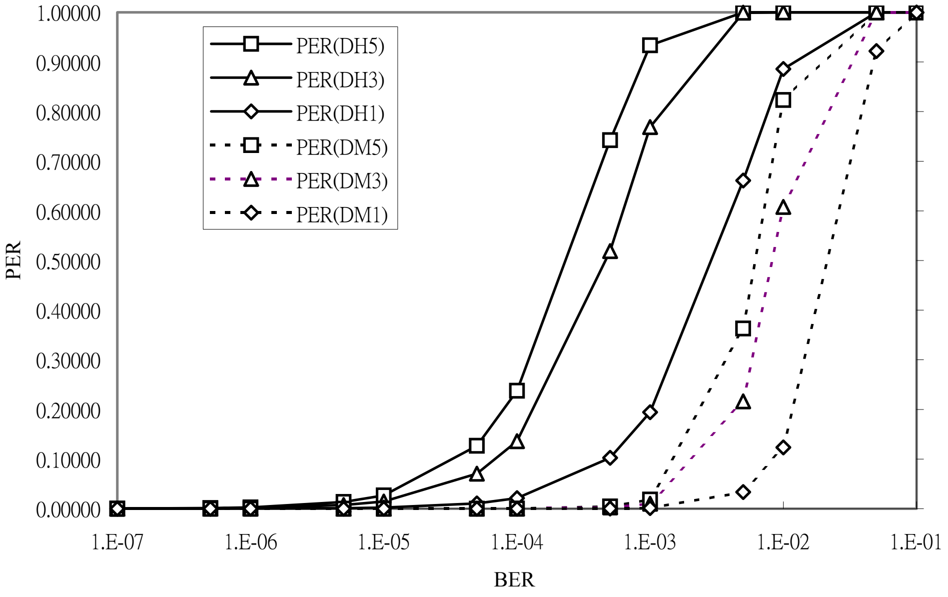

Fig. 3.1: Packet error rate of different packet types vs. bit error rate.

Figure 3.1 plots the PER of different ACL packet types (include DH and DM packets) with respect to the uniform BER based on equations (1) and (2). However, as shown in the figure, when the BER increases from 10^{-4} to 10^{-3}, the PER of DH packets increases rapidly while the PER of DM packets still increases slowly. Thus, on one hand we can transmit DH packets when the BER is lower than a threshold value, BER^{H}, and on the other hand, transmit DM packets when the BER is lower than a threshold value, BER^{M}. Note that we can mark a channel's state as *Better*, *Good*, or

Bad, not only by the BER but the PER [10] or *received signal strength indication* (RSSI) [11], and etc. In addition, these two thresholds are not fixed, which can be adjusted dynamically.

Receiving Frequency Status				Frequency Offset
Master	Slave 1	Slave 2	Slave 3	
Better	Bad	Good	Bad	0
Good	Bad	Better	Better	1
Bad	Better	Good	Bad	2
				…
Bad	Better	Bad	Bad	77
Good	Good	Better	Good	78

Fig. 3.2: Receiving frequency table.

Since different Bluetooth devices in a piconet have different interference levels due to location-dependent errors and the bit error rates seen by different frequencies in the hopping spectrum are significantly different from each other [20]. The master maintains a receiving-frequency table as shown in Fig. 3.2, which is a $(n+1)\times h$ matrix M, where $n+1$ represents the master plus n slaves and h represents the number of operating RF channels. At receiving frequency k, the channel state of the master, M_{0k}, or the slave i, M_{ik} (located at the column i+1 and row k of the matrix M), is classified according to the BER measured in each channel and is marked as *Bad*, *Good*, or *Better*. Note that the slaves should send its link status to the master at a regular interval to update the receiving frequency table. It is no enough to consider each master-slave connection as an independent channel in the interference environments especially when Bluetooth uses the Frequency Hopping (FH) scheme, thus we define one element of the matrix *M* to be one **channel** [10].

The Bluetooth specification did not specify any scheduling policy that the master should adopt for medium access control. The Round Robin (RR) scheduling is the simplest strategy for scheduling in Bluetooth. However, it not only leads to low link utilization and low throughput, but also is unsuitable for traffic sources with different data rates. Thus, we proposes a Queue State Dependent Priority (QSD-PR) policy to schedule packets based on the queue backlogs at the master queue and the slave queue to provide higher link utilization and hence higher throughput, and lower end-to-end delay. Note that the QSD-PR also takes the channel conditions into account and avoids bad channels by using the receiving-frequency table at the master.

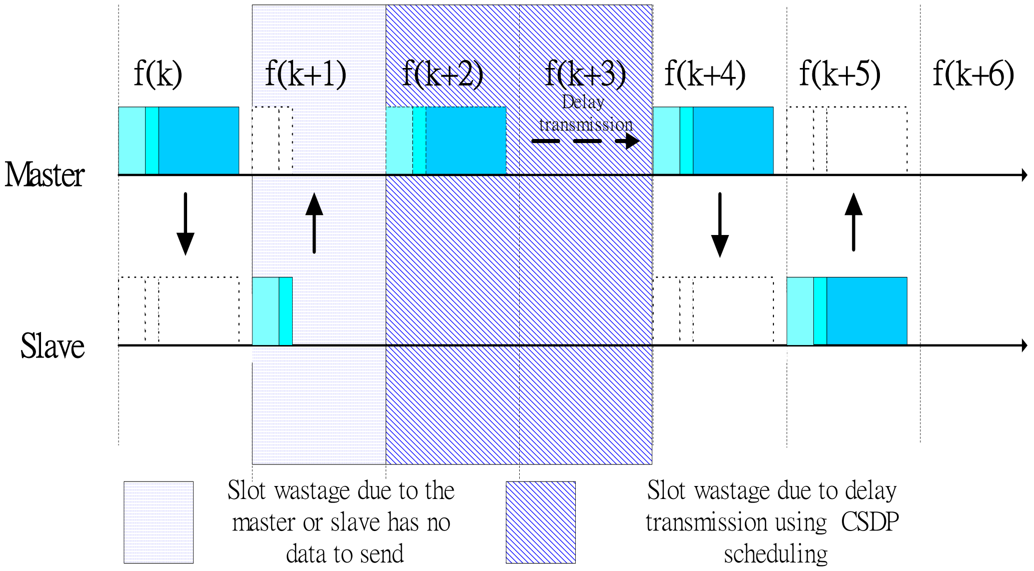

Fig. 3.3: Slot wastage scenario in Bluetooth.

As shown in Figure 3.3, when applying channel state dependent packet scheduling, a slot gets wasted primarily from two situations: (1) the master or slave has no data to send and (2) delay transmission in channel state dependent packet scheduling. Based on this observation, we also propose a Channel State Dependent Segmentation and Reassembly (CSD-SAR) scheme to maximize the link utilization by using multi-slot packets to mask bad frequencies. In an interference-limited environment, small size packets and incorporate FEC protection will cause Bluetooth devices to generate more packets and thus more interference to the 802.11b network. Oppositely, in a range-limited environment, larger size packets without incorporating FEC protection will result in a high packet error rate. Therefore, we will transmit the largest packets to the utmost on *Better* or *Good* channels and avoid transmitting on *Bad* channels according to the receiving frequency table. The detailed packet selection and packet scheduling scheme will be illustrated in the next section.

3.2 Packet Selection Scheme

There is a degree of flexibility in the choice of packet type: incorporating FEC or not. We select an appropriate packet type DH or DM to a specific slave according to the ratio of the total number of *Good* and *Better* channels between the master and a specific slave to the total number of frequencies, i.e. based on the location-dependent channel conditions for the specific slave. Frequently switching between protected (DM) and unprotected (DH) packets is inefficient due to message-passing overheads [21]. Therefore, according to the channel condition of the slave, we need to decide to use either DH or DM packets during each period of communication. The multi-slot packet uses lesser time to transmit the same amount of data that will result in higher throughput, lower end-to-end-delay and hence higher link utilization in either error-free or error-prone environments [17][18][19][22]. Thus, after determining the DM or DH packet type to send, we select the packet size as large as possible.

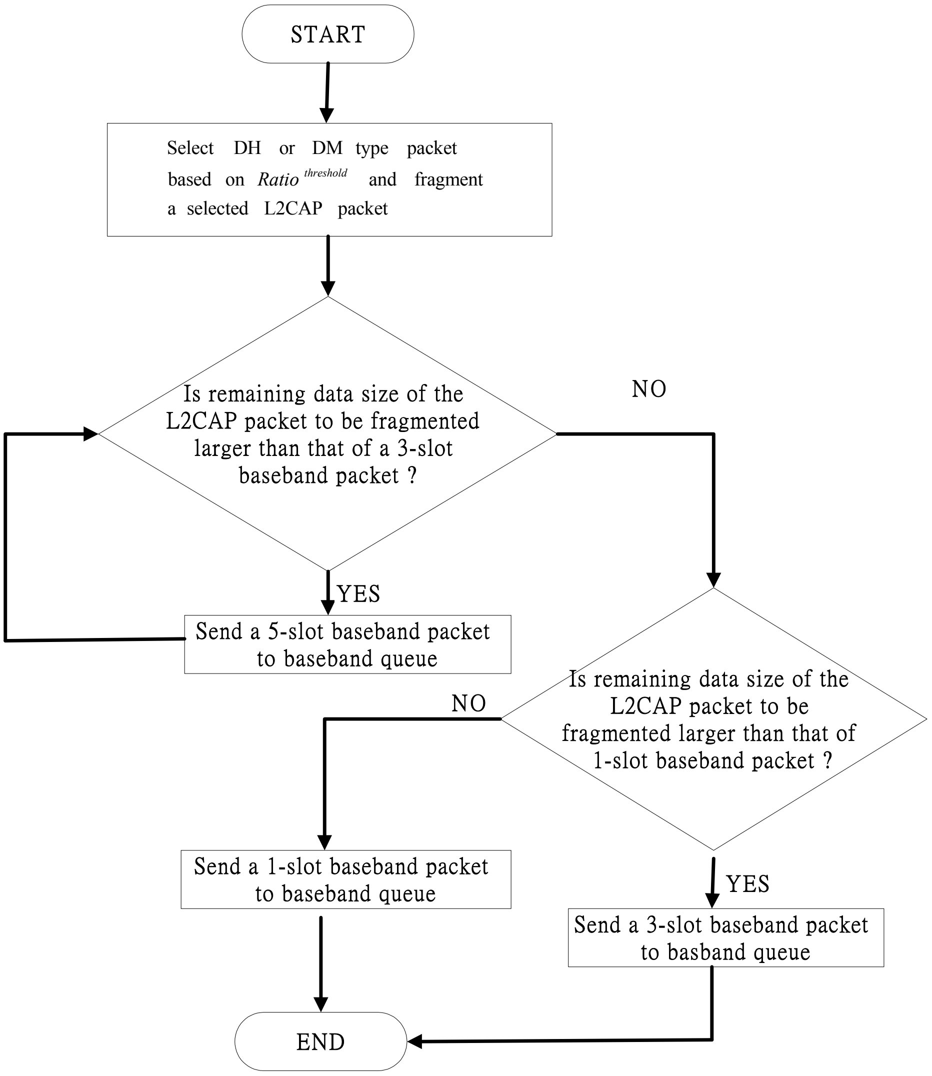

Fig. 3.4: Adaptive packet selection scheme.

The adaptive packet selection scheme is shown in Fig. 3.4. Regardless of the *Bad* channels of a specific slave, if the ratio of *Better* channels to *Good* channels exceeds a threshold, $Ratio^{threshold}$, we will send the DH packets in the *Better* channels, and select packet size based on the remaining data size to fragment. However, if the ratio of

Better channels to *Good* channels below a threshold, $Ratio^{threshold}$, we will use the *Better* channels and *Good* channels to transmit DM packets and select the packets as large as possible, such as DM5 packets. That is, we will use the largest packet type to transmit on the *Better* or *Good* channels to mask bad frequencies. In this way, we can solve the wastage of slots due to delay transmission, as illustrated in Figure 3.3. For example, with the L2CAP packet size of 500 bytes, if we decide to transmit DM packets, we will fragment the L2CAP packet into two DM5 packet of 224 bytes each and one DM3 packet of 52 bytes.

3.3 Scheduling Policy

We will give each slave a priority based on the sum of queue backlogs (the number of data packets at the master and the slave queues), $Q_{back\log}$, and the waiting time T_{wait} since the salve has been scheduled to send packets previously. Thus, we can give each slave a priority as follows:

$$P = \gamma \left(\frac{Q_{\text{backlog}}}{Q_{\max}}\right) + (1-\gamma)\left(\frac{T_{\text{wait}}}{T_{\max}}\right), \quad \gamma \le 1 \tag{3}$$

where $Q_{\max}$ is the sum of maximum queue size at the master and a specific slave, and $T_{\max}$ is the maximum time that a specific slave can wait, which is negotiated during the master-slave connection setup based on QoS requirements. In this priority scheduling policy, we give the slave that has data to receive/send from/to the master a higher priority, and the lowest priority was given to the salve that has no queue backlogs at the master and the slave. $T_{\max}$ is specific to each slave guarantees a bounded delay.

Based on the priority scheme, each time we can select the next slave queue from which a packet should be sent.

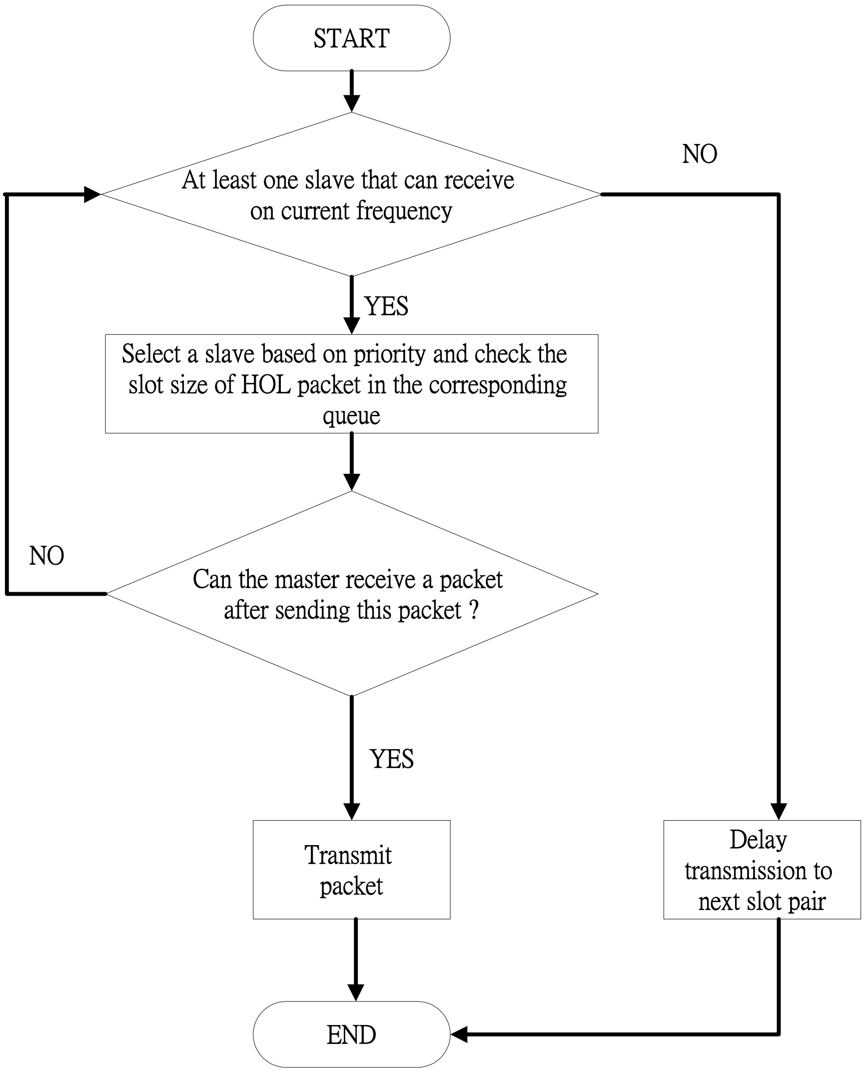

Fig. 3.5: Packet scheduling policy.

The scheduling policy is shown in Figure 3.5. The master selects a slave *i* to transmit a packet based on the priority from the set of slaves that can receive on current frequency *k*. Since the present Bluetooth architecture does not support packet reordering, we check the Head of Line (HOL) packet size at the queue corresponding to slave *i*. If the packet size is 5 time slots, we assume the slave will respond on frequency *k_5*, and we need to check the channel state of the receiving frequency of the master, M_{0k_5}, in the receiving frequency table. We will send this packet to slave *i* if either the packet is a DH packet and the channel state is *Better* or the packet is a DM packet and the channel condition is not in the *Bad* state. If the HOL packet size is 3 time slots or 1 time slot, we also check the channel state of the receiving frequency, which is similar to the procedure described above.

Finally, if the HOL packet at the queue for a specific slave cannot be sent because of channel conditions, we will select another slave queue based on the priority to send its packet. If all slaves are unsuitable to send its HOL packet, we will delay the transmission to the next slot pair. Note that the delay rule is only implemented at the master side.

In the coexistence scenario, like IEEE 802.11b and Bluetooth, the primary reason for packet drop is due to the interference between them, not the random bit errors caused by noise or the distance between devices. In the case of such persistent errors that occupy certain static frequencies for much larger duration may range from minutes to even several hours and cause severe interference to each other when Bluetooth hops over these infected frequencies.

The design goal of our packet selection and scheduling scheme is to generate fewer packets by using larger packet size and schedules the packets in a way to avoid bad frequencies. Thus, the coexistence problems can be solved and the impact of interference can be reduced.

3.4 Compared to BIAS [9]

Table 3.1 The proposed scheme in comparison with BIAS

	BIAS	Proposed scheme
ACL link	DM1	ALL ACL Packets
Fairness	Short term max-min fairness Unit: slot pair	Bounded delay
Restriction	Best when master's data rate is equal to each slave's	No
Link utilization	Low	High
Throughput	Low	High
Simulation model	MAC, PHY	TCP/IP, MAC, PHY

Using channel state dependent packet scheduling to improve performance in error-prone environments is not new, as indicated by literatures such as [9][23][24]. Unlike [23][24], which were applied to Wireless LANs, [9] considered the hopping nature of the Bluetooth devices and distributed channels to devices in order to ensure fairness of access among users by means of max-min fairness criteria. It assumes an essential unit is a (master/salve) slot pair and distributes the bandwidth unused by the interference-prone sessions to other error-free connections [9]. Consequently, only the DM1 packet is used in the simulation environment. However, when the downstream (master-to-slave) traffic is not equal to the upstream (slave-to-master) traffic, this scheme will cause the wastage of slots. The objective of the proposed scheme is to maximize link utilization and throughput while that of the BIAS is to ensure fairness. Since the objectives of these two schemes are different, we only compare these two schemes, qualitatively, as shown in Table 3.1. Since the BIAS only considers the DM1 packet, it will results in low link utilization and low throughput. In the next section, we will compare our scheme with the RR only.

Chapter 4

Evaluation and Discussion

4.1 Simulation Setup

This chapter explains the details of simulation setup: topology, traffic sources, and the error characteristics of RF channels. We use *ns-2* [25] and a Bluetooth extension [26] to simulate our proposed scheme. In the simulation, every element in the receiving frequency table is considered separately as a **channel**. A channel is considered as either *clear* or *interference-affected*. If a channel is clear, we assume the channel is in the *Better* state and the packets sent on this channel to a specific slave will not be corrupted. That is, the packets are always received successfully. An interference-affected channel maybe in the *Good* or *Bad* state that can be modeled as a two-state Markov channel. According to Fig. 3.1, we assume the PER is not the same for all Bluetooth data packets. We calculate the PER of each packet type based on equations (1) and (2) when BER is 1×10^{-3}: 0.934 for DH5, 0.769 for DH3, 0.194 for DH1, and 0.018 for DM5, 0.010 for DM3 and 0.001 for DM1. Also, we assume that in the *Bad* state it is not always destructive, and we set the PER is 0.95 for all packets. We compare the performance of different packet selection and scheduling schemes by increasing the number of interference-affected channels gradually.

One study in [27] has proved that the finite state Markovian model can be used to effectively characterize the bursty bit error behavior of wireless links. Previous work on CSDP scheduling [23][24] uses a two-state (Good-Bad) Markov process [28][29] to model the wireless link, as shown in Fig. 4.1. In the *good* state the BER, P_G, is low and in the *bad* state the BER, P_B, is high. Transitions between the two states occur according to the corresponding state transition probability of μ_G for transferring from the *good* state to the *bad* state and μ_B for transferring from the *bad* state to the *good* state.

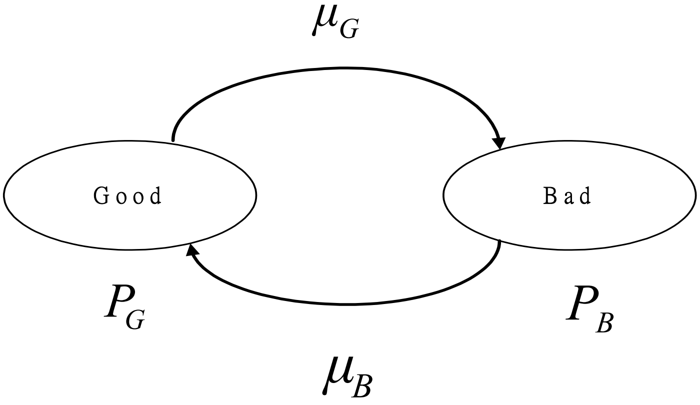

Fig. 4.1: Two-state Markov model.

However, our objective here is not to derive an accurate characterization of the channels. The parameters of the two-state Markov model that are used in the simulation are to illustrate the behavior of the transport sessions when packets are subject to burst loss. We assume that the time spent in the Good and Bad periods are exponentially distributed, with different mean values, that is, different rates of state transition μ_G and μ_B. According to the properties of exponentially distributed random variables, the average time between state transitions can be expressed by $X_G = 1/\mu_G$ and $X_B = 1/\mu_B$ [22]. We set the parameter values $X_G = 500\,ms, X_B = 500\,ms$ in scenario 1, and compare the performance improvements using our proposed scheme with different values of X_G and X_B in scenario 2.

The performance metrics that we used include *packet loss*, *end-to-end delay, link utilization*, and *transport layer throughput*. The packet loss is the probability that a packet is discarded at the MAC layer due to interference. It is expressed as the number of packets lost divided by the total number of packets sent during the simulation time. The end-to-end delay measures the elapsed time from the packet that is enqueued in the buffer until it is successfully received at the destination slave. The delay is measured at the L2CAP layer. Link utilization quantifies the percentage of total slots that are successfully used to transmit. That is, we did not take retransmission packets and NULL packets into account. The NULL packet has no payload and occupies one time slot [2]. Transport layer throughput is an indication of how much data that the user can receive per second.

The network topology used in the simulation is shown in Fig. 4.2. It includes one Bluetooth piconet that contains one master and three slaves. In the simulation, data packets flows consist only from the master to the salves and on the reverse direction only NULL packets for acknowledgements are returned. The traffic model used in the simulation has tried to capture a variety of traffic sources.

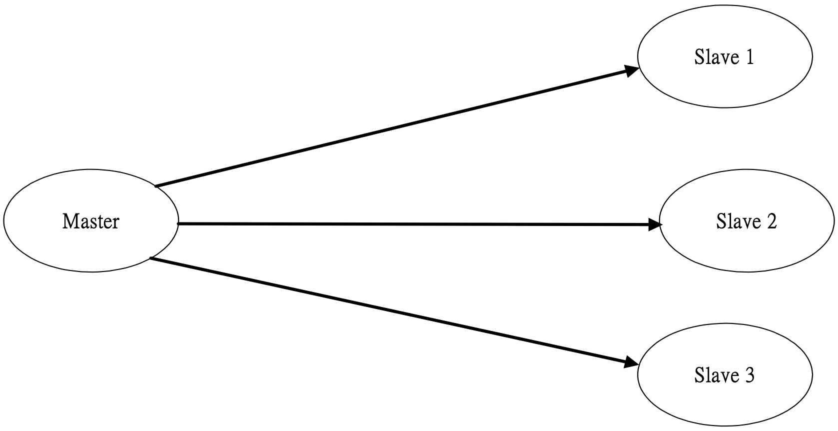

Fig. 4.2: Network topology used in the simulation.

However, while we study this traffic model, we look into the effect of all sources rather than a single one and hence we use the performance metrics described above to reflect the overall data performance rather than the performance of a single slave in a piconet [22].

4.2 Simulation Results

We compare our scheme with RR under light offered load (scenario 1) and heavy offered load (scenario 2).

4.2.1 Scenario 1: Light Offered Load

Table 4.1: Properties of the data flows used in scenario 1.

Property / Slave no.	Traffic Type	Data Rate	Packet Size	Transport Layer	Burst Time	Idle Time
Slave 1	CBR	100 *Kbps*	500 bytes	UDP		
Slave 2	CBR	100 *Kbps*	500 bytes	UDP		
Slave 3	Exponential Traffic	64 *Kbps*	500 bytes	UDP	500 *ms*	500 *ms*

In order to understand the packet loss and end-to-end delay, throughput and link utilization in interference environments, we setup simulation scenario 1 (light offered load) which is listed in Table 4.1. There are three data flows in this scenario: two 100 Kbps *Constant Bit Rate* (CBR) flow and one exponential distributed data flow (exponential traffic). The data flows were all run over the transport layer and the UDP packet size is 500 bytes. The two CBR flows are guaranteed flows. Note that the exponential distributed data flow generates traffic according to an exponential *On*/*Off* distribution. Packets are sent at a fixed rate during *on* periods (*Burst time*), and no packets are sent during *off* periods (*Idle time*). Both on and off periods are taken from an exponential distribution [23].

Figure 4.3 shows the packet loss when applying the Round Robin and our

QSD-PR scheduling with different SAR schemes, respectively. The SAR schemes include the Random-SAR [30], which select data packet sizes (i.e., 1, 3 or 5) randomly and our proposed CSD-SAR scheme. Note that *R-SAR (DH)* stands for Random-SAR with DH type packets and *R-SAR (DM)* stands for Random-SAR with DM type packets. We set $r = 0.5$ for QSD-PR and $Ratio^{threshold} = 1$ for CSD-SAR in scenario 1. When the number of interference-affected channels increases, we can see that using the RR scheduling policy with R-SAR (DH) and R-SAR (DM) result in higher packet loss. The DM packets that incorporate FEC code can effectively reduce the percentages of packet loss. But the percentage of packet loss using RR still increases in proportion to the number of interference-affected channels. By taking channel conditions into consideration and using the receiving frequency table to avoid *Bad* channels, the percentage of packet loss is almost kept at zero when applying our QSD-PR scheduling policy with either R-SAR or CSD-SAR scheme.

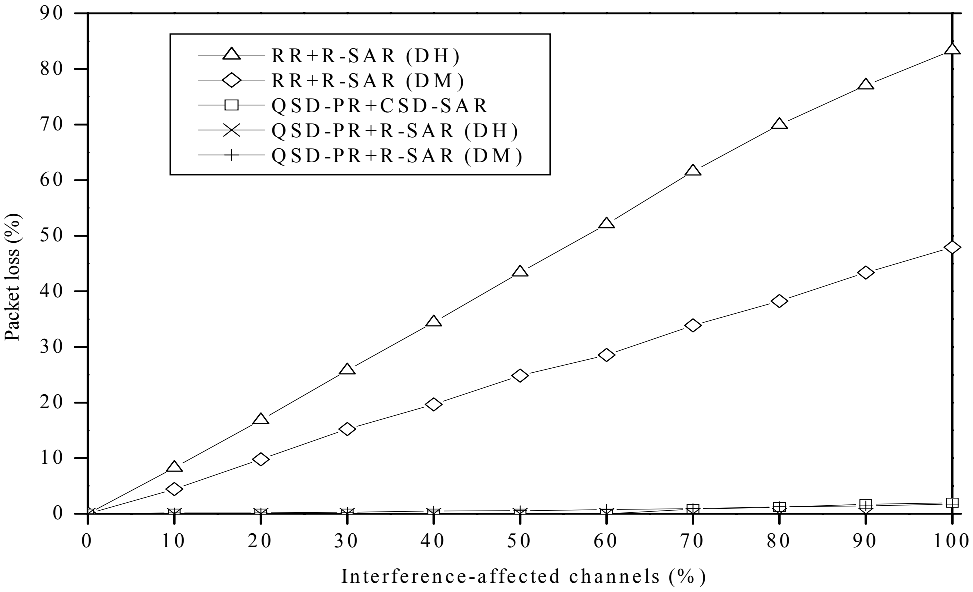

Fig. 4.3: Packet loss vs. interference-affected channels (%) in scenario 1.

Without loss of generality, we only analyze the simulation results for slave 1. The slave 1's end-to-end delay (experienced by the UDP packets) is shown in Fig. 4.4. Note that the interference-affected channels were distributed to the 3 slaves uniformly. When all the channels were clear and the QSD-PR and RR were applied the same R-SAR scheme, we can see that the end-to-end delay of QSD-PR is lower than RR. This is because the three data flows did not have equal data rates and resulted in the wastage of slots in RR. We can also see that the QSD-PR with CSD-SAR or R-SAR (DM) scheme can achieve very low end-to-end delay even when all the channels were interference-affected because the QSD-PR can use the *Good* periods of the channels (according to the receiving frequency table) to transmit DM packets and guarantee low end-to-end delay.

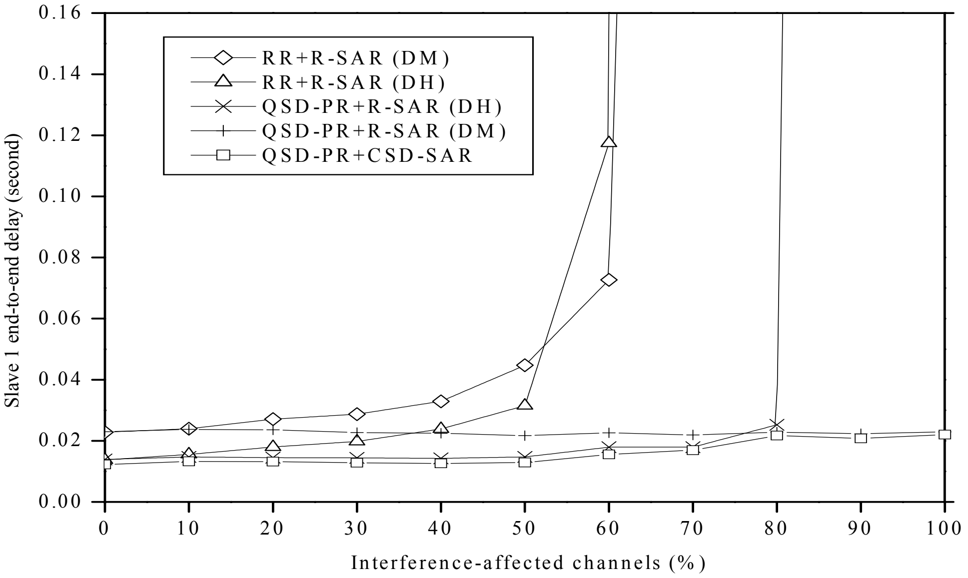

Fig. 4.4: End-to-end delay vs. interference-affected channels (%) in scenario 1.

On the contrary, the end-to-end delay in the RR scheduling increased in a rapid pace when the number of interference-affected channels is greater than 50% of the total channels. Also, the end-to-end delay in the QSD-PR with R-SAR (DH) scheme also increases rapidly when the number of interference-affected channels is greater than 80% of the total channels. When the number of interference-affected channels increases, the end-to-end delay increase rapidly in the RR is due to frequent retransmission of packets, while the end-to-end delay increases rapidly in the QSD-PR with RSAR (DH) is due to no enough *Better* channels to transmit DH packets and results in frequent delayed transmission of packets.

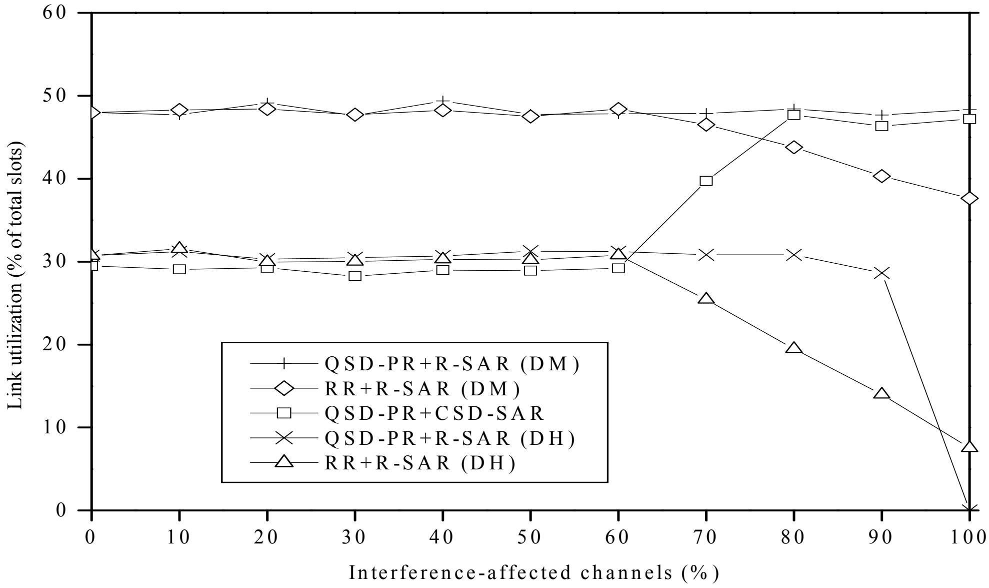

Fig. 4.5: Link utilization vs. interference affected channels (%) in scenario 1.

Fig. 4.5 shows the link utilization vs. interference-affected *channels*. First, we can see that the link utilization is about 31% of the total slots when using DH packets to transmit data. However, the link utilization can achieve up to 48% of the total slots when using DM packets to transmit data. Since the DM packet incorporates (15,10) Hamming code, the DM packet only carries about 2/3 of the data compared to the DH packet. That is, using the DM packet will need to generate more packets in order to transmit the same amount of data. Note that when the number of interference-affected channels increases up to 70% of the total channels, the ratio of *Better* channels to *Good* channels is below the threshold $Ratio^{threshold}$. Thus, CSD-SAR using DM packets to fragment the transport layer packets instead of DH packets to resist the interference. This increases the link utilization from 30% to about 50% of the total slots.

Fig. 4.5 also shows that using the RR scheduling that did not take channel state information into consideration has lower link utilization when the percentage of interference-affected channels increases. The link utilization of the QSD-PR with R-SAR (DH) scheme also decreases when the percentage of interference-affected channels increases up to 90% of the total channels. The decrease of link utilization in RR was primarily due to frequent packet retransmission while in QSD-PR it was primarily due to delayed transmission to avoid *Bad* channels.

Fig. 4.6 shows the throughput of the piconet while using different packet selection and scheduling schemes. The throughput shows how much available bandwidth that is actually being used. We can see that the throughput kept constant at about 230 *Kbps* (close to the offered data rate) regardless of the number of interference-affected channels of the total channels when applying the QSD-PR with CSD-SAR scheme. In contrast, the RR with RSAR scheme would cause the allocated slots not able to satisfy the required data rates of the three flows.

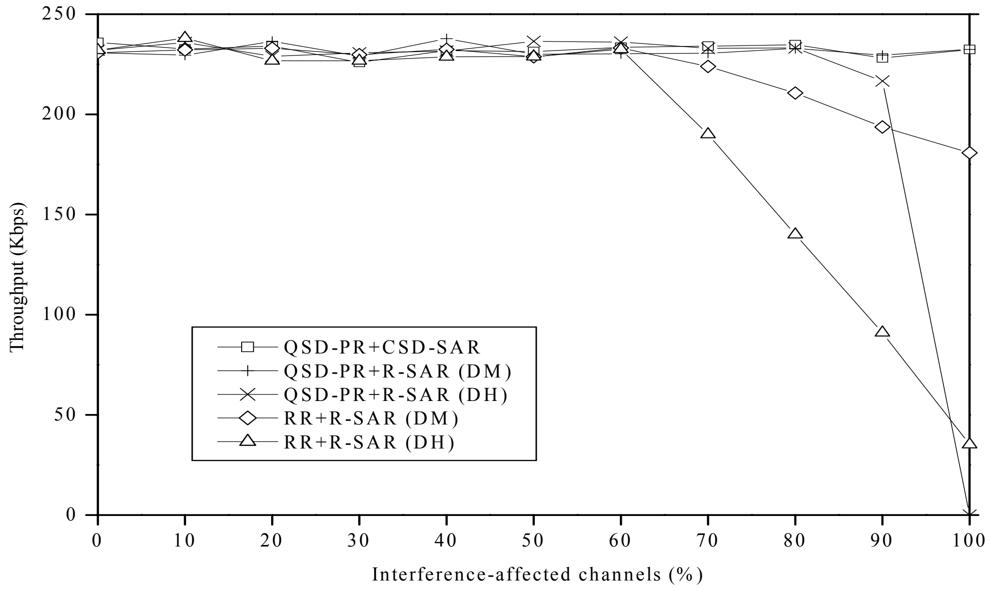

Fig. 4.6: Throughput vs. interference-affected channels (%) in scenario 1.

In sum, the performance obtained from scenario 1 show that a good packet selection scheme is important when applying a channel state dependent packet scheduling policy. When applying QSD-PR, the performance of CSD-SAR is outstanding compared to that of R-SAR. The CSD-SAR can select the best packet type and packet size according to channel conditions and guarantee higher throughput and lower end-to-end delay. In the next scenario, we applied a heavy offered load to the simulation network, and we will use the CSD-SAR scheme instead of the R-SAR scheme while using QSD-PR.

4.2.2 Scenario 2: Heavy Offered Load

Table 4.2: Properties of the data flows used in scenario 2.

Property / Slave no.	Traffic Type	Data Rate	Transport Layer	Packet Size	Burst Time	Idle Time
Slave 1	FTP	ABR	TCP	500 bytes, 40 bytes ACK		
Slave 2	FTP	ABR	TCP	500 bytes, 40 bytes ACK		
Slave 3	Exponential Traffic	64 *Kbps*	UDP	500 bytes	500 *ms*	500 *ms*

There are three data flows in scenario 2: two FTP flows and one exponential distributed data flow (exponential traffic). The FTP flow simulates bulk data transfer and will occupy as much as bandwidth as possible (ABR). The specifications of scenario 2 are listed in Table 4.2. In this scenario, the offered load was close to 100% of the total capacity and the link utilization was almost full all the time. That is, the throughput degradation is more susceptible by increasing the number of interference-affected channels. We also set $r = 0.5$ for QSD-PR and compare the link utilization and throughput with different $Ratio^{threshold}$ in scenario 2.

Fig. 4.7 shows that the link utilization when the percentage of interference-affected channels increase from 0% to 100%. When the percentage of interference-affected channels increases from 0% to about 40%, the QSD-PR with CSD-SAR could still maintain high link utilization and had high throughput. This is because the QSD-PR with CSD-SAR uses multi-slot packets to mask interference-affected channels. However, when the percentage of interference-affected channels increases over 40% of the total channels, the link utilization decreases gradually because of delayed transmission. In Fig. 4.8, it shows that by applying our QSD-PR with CSD-SAR scheme, we can achieve higher throughput in either error-free or error-prone environments. Note that the threshold value $Ratio^{threshold}$ in CSD-SAR can be optimized and further enhance the throughput.

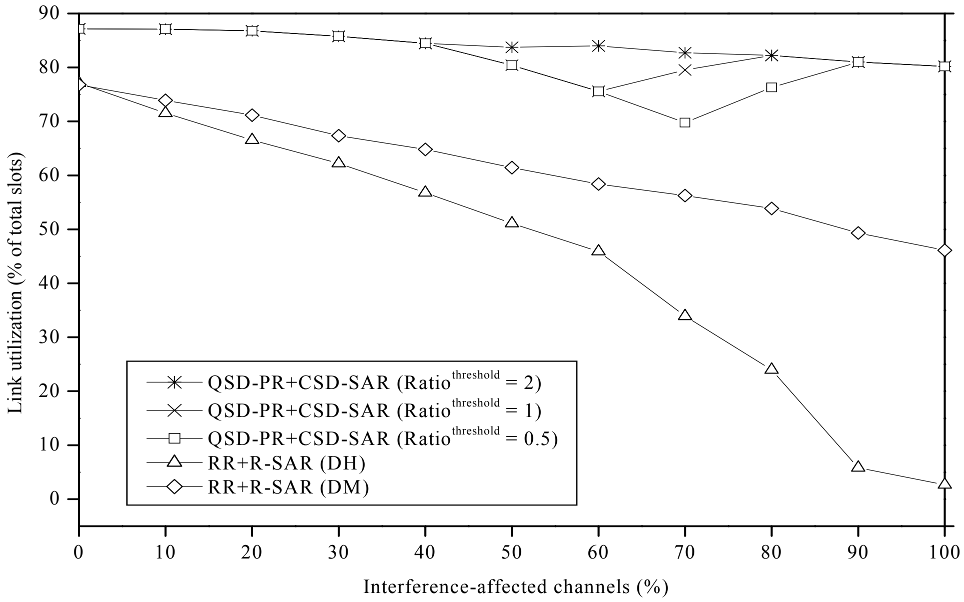

Fig. 4.7: Link utilization vs. interference-affected channels (%) in scenario 2.

In scenario 2, the performance of our proposed scheme is significantly better than the RR with R-SAR scheme because we use the receiving frequency table to avoid bad channels and we fragment the transport layer packets as large as possible to mask bad frequencies by using multi-slot packets. In addition, using the DM and DH packets based on channel conditions can efficiently reduce the packet error rate and guarantee high throughput in error-prone environments. Simulation results show that the QSD-PR with CSD-SAR scheme can adapt to error-prone environments under high load.

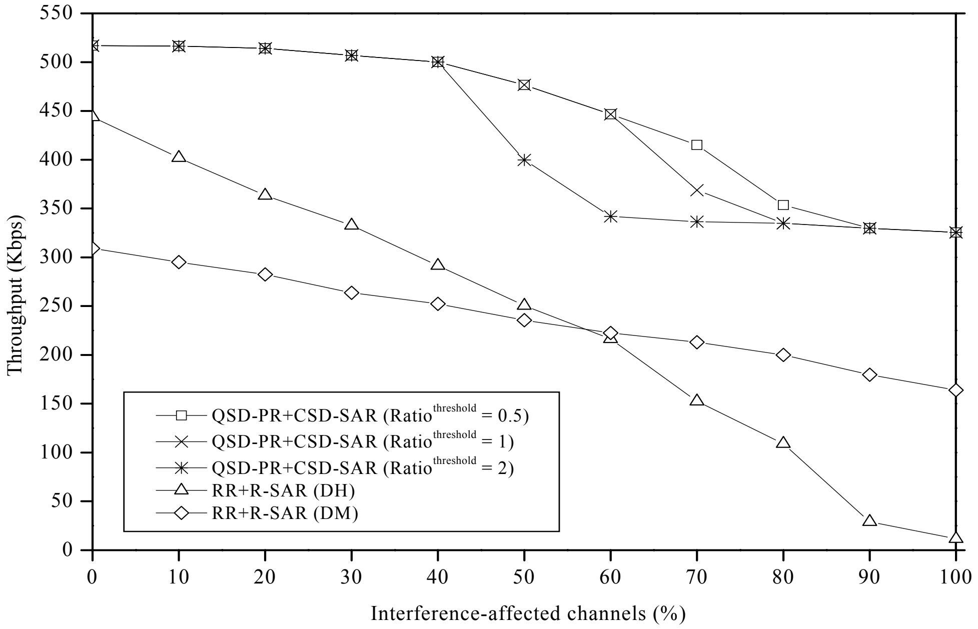

Fig. 4.8: Throughput vs. interference-affected channels (%) in scenario 2.

According to the results from Fig. 4.8, we set $Ratio^{threshold} = 0.5$ and gave the mean state residency time X_G and X_B with different values and observed the throughput improvements using the proposed scheme. In Fig. 4.9, it shows that our QSD-PR with CSD-SAR scheme can offer throughput improvements as high as 195% compared to the RR with R-SAR (DH) scheme (when the percentage of interference-affected channels increases from 0% to 70%). In addition, the QSD-PR with CSD-SAR allows the Bluetooth system to remain usable even when all the channels are interference-affected. Note that when channels are more error prone, the more improvement can be obtained by using our proposed scheme.

Finally, we investigate the effect of number of slaves on throughput improvements by increasing the number of slaves from 3 slaves up to 7. The traffic type of slaves 3 through 7 is the exponential traffic that specified in Table 4.2. A piconet has a limit on the maximum number of active slaves. In Fig. 4.10, by increasing the number of slaves in the piconet, we can see that the more throughput improvements can also be obtained by using our proposed scheme. This is because each slave in the piconet has a different data input rate and the RR scheduling scheme will waste more baseband slots by polling sources with low data input rates. It results in lower link utilization and thus lower throughput using the RR.

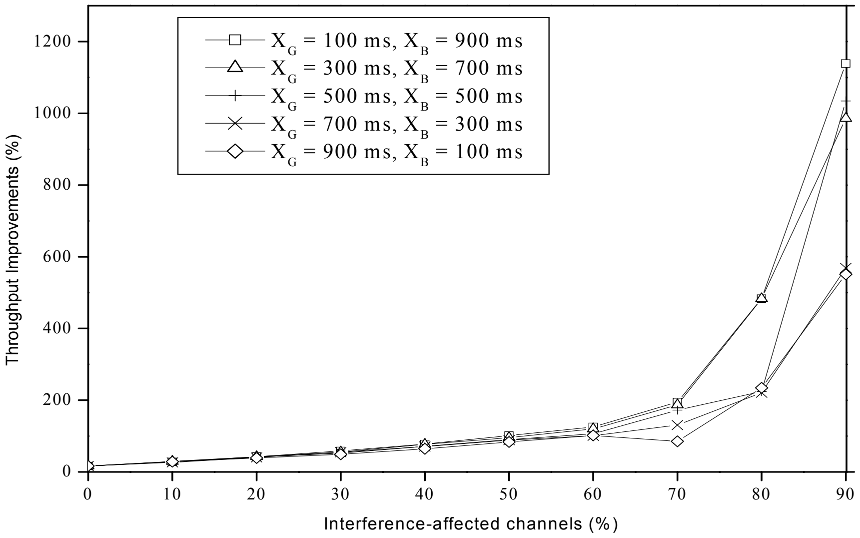

Fig. 4.9: Throughput improvements vs. different X_G and X_B.

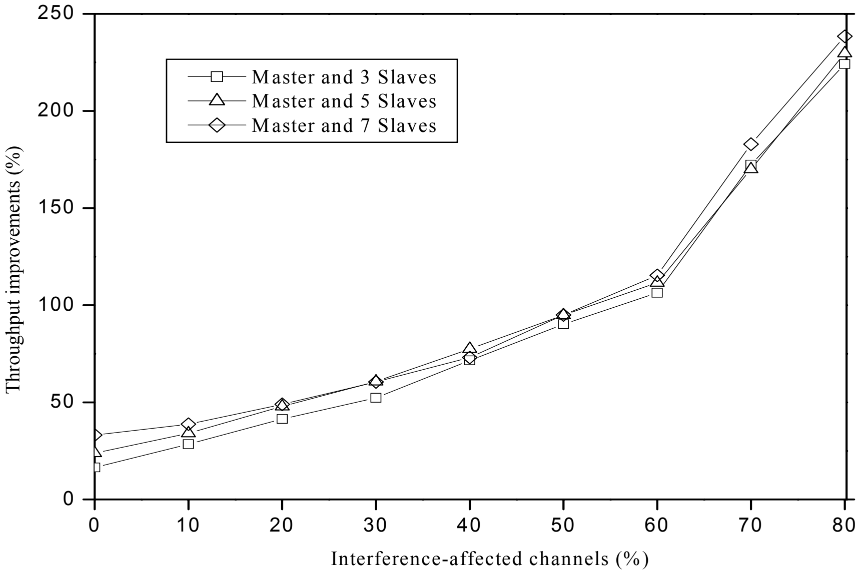

Fig. 4.10: Throughput improvements with various numbers of slaves in a piconet.

Chapter 5

Conclusions and Future Work

5.1 Concluding Remarks

The market is rapidly moving toward resolving the coexistence concerns surrounding the IEEE 802.11b and Bluetooth [31]. Our proposed approach and other approaches have addressed the issue before it ever affects the end-user. As a result, market forecast for Bluetooth and IEEE 802.11b will remain strong, and the need for effective, multi-standard, coexistence solutions will only increase as wireless devices proliferate and simultaneous operation usage models become pervasive [31].

Simulation results have shown that out packet selection and scheduling scheme based on the channel state and queue state can have higher link utilization and higher throughput compared to the Round Robin packet scheduling scheme in an interference environment. In addition, the scheduling policy that delays transmission to avoid bad frequencies occupied by other devices will alleviate the impact of interference on the other systems significantly [8]. Note that our scheme can also be adapted and used in other centrally controlled TDD wireless systems, such as IEEE 802.15.1.

5.2 Future Work

We will look for additional scenarios for a variety of traffic sources and take the SCO link into consideration and study the performance of real-time applications (e.g. voice) using our proposed packet selection and scheduling scheme. In addition, the parameters of QSD-PR and CSD-SAR, such as r and $Ratio^{threshold}$, will be further optimized by mathematical analysis or simulation.

Bibliography

[1] The Bluetooth Web Site, *http://www.bluetooth.com.*

[2] Bluetooth Special Interest Group, Specification of the Bluetooth System, Version 1.1, Feb. 2001.

[3] S. Kapp, “802.11: leaving the wire behind,” *IEEE Internet Computing*, vol. 6, issue 1, pp. 82-85, Jan.-Feb. 2002.

[4] K. J. Negus, A. P. Stephens, and J. Lansford, “HomeRF: Wireless networking for the connected home,” *IEEE Personal Communications*, vol. 7, issue 1, pp. 20-27, Feb. 2000.

[5] ETSI, “Broadband Radio Access Networks (BRAN): HIPERLAN type 2 technical specification,” April 2000.

[6] J. Lansford, A. Stephens, and R. Nevo, “Wi-Fi (802.11b) and Bluetooth: enabling coexistence,” *IEEE Network*, vol. 15, issue 5, pp. 20-27, Sept.-Oct. 2001.

[7] Javier del Prado and Sunghyun Choi, “Experimental study on coexistence of 802.11b with alien devices,” in *Proc. of the IEEE 54th Vehicular Technology Conference*, Oct. 2001, vol. 2, pp. 977-981.

[8] N. Golmie and N. Chevrollier, "Techniques to improve Bluetooth performance in interference environments," in *Proc. of IEEE Military Communications Conference*, Oct. 2001, vol. 1, pp. 581-585.

[9] N. Golmie, N.Chevrollier and I. EIBakkouri, "Interference aware Bluetooth packet scheduling," in *Proc. of IEEE Global Telecommunications Conference*, Nov. 2001, vol. 5, pp. 2857-2863.

[10] S. Deb, M. Kapoor, and A. Sarkar, "Error avoidance in wireless networks using link state history," in *Proc. of IEEE INFOCOM*, April 2001, vol. 2, pp. 785-795.

[11] Y.-H. You, et al., "Adaptive frequency hopping scheme for interference-limited WPAN applications," *Electronic Letters,* vol. 37, issue 15, pp. 976-978, July 2001.

[12] J. C. Haartsen, "The Bluetooth radio system," *IEEE Personal Communications,* vol. 7, pp. 28-36, Feb. 2000.

[13] Bluetooth Special Interest Group, "Bluetooth protocol architecture," *White paper*, Aug. 1999.

[14] Bluetooth Special Interest Group, "Bluetooth Networking Encapsulation Protocol (BNEP) Specification," Specification of the Bluetooth System, Version 0.95, May 2001.

[15] Bluetooth Special Interest Group, "Bluetooth Personal Area Networking Profiles," Specification of the Bluetooth System, Version 0.95, May 6, 2001.

[16] S. Zurbes, "Considerations on link and system throughput of Bluetooth networks," in *Proc. of the 11th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Sept. 2000, vol. 2, pp. 1315-1319.

[17] J. Kim, Y. Lim et al., "An adaptive segmentation scheme for the Bluetooth-based wireless channel," in *Proc. of the 10th International Conference on Computer Communications and Networks*, Oct. 2001, pp. 440-445.

[18] M. C. Valenti, M. Robert, and J. H. Reed, "On the throughput of Bluetooth data transmission," in *Proc. of IEEE Wireless Communications and Networking Conference*, Mar. 2002, vol. 1, pp. 119-123.

[19] T.Y. Chui, F. Thaler and W.G. Scanlon, "Bit error rate related load constraints for Bluetooth baseband packets," *Electronic Letters*, vol. 38, issue 3, pp. 137-138, Jan. 2002.

[20] J. Zander and J. Malmgren, "Adaptive frequency hopping in HF communications," *IEEE Proceedings – Communications*, vol. 142, issue 2, pp. 99-105, April 1995.

[21] J. Bray, and C.F. Sturman, *Bluetooth: Connect without cables*. Prentice Hall PTR, 2001, pp. 348-350.

[22] A. Das, A. Ghose, A. Razdan, H. Saran, and R. Shoery, "Enhancing performance of asynchronous data traffic over the Bluetooth wireless ad-hoc network," in *Proc of IEEE INFOCOM 2001*, April 2001, vol. 1, pp. 591-600.

[23] P. Bhagwat, P., et al., “Enhance throughput over wireless LANs using channel state dependent packet scheduling,” in *Proc. of IEEE INFOCOM*, Mar. 1996, vol. 3, pp. 1133-1140.

[24] C. Fragouli, V. Sivaraman, M. Srivastava, “Controlled multimedia wireless link sharing via enhanced class based queuing with channel state dependent queuing,” in *Proc. of IEEE INFOCOM 1998*, Mar. 1998, vol. 2, pp. 572-580.

[25] ns Notes and Documentation, the VINT Projects, UC Berkeley, LBL, USI/ISI and Xerox PARC, *http://www.isi.edu/nsnam/ns/.*

[26] Bluehoc simulator, a Bluetooth extension to NS, *http://www-124.ibm.com/developerworks/opensource/bluehoc/.*

[27] H. Wang, and N. Moayeri, “Finite state Markov channel – a useful model for radio communications, “ *IEEE Trans. On Vehicular Technology,* vol. 44, no. 1, pp. 163-171, Feb. 1995.

[28] E. O. Elliott, “Estimates of error rate for codes on burst-noise channels,” *Bell System Technical Journal*, vol. 42, pp.1977-1997, Sept. 1963.

[29] E. N. Gilbert, “Capacity of burst-noise channel,” *Bell System Technical Journal*, vol. 39, pp. 1253-1266, Sept. 1960.

[30] M. Kalia, D. Bansal, R. Shorey, “Data Scheduling and SAR for Bluetooth MAC,” in *Proc. of Vehicular Technology Conference*, May 2000, vol. 2, pp. 716-720.

[31] Mobilian Corp., “Wi-Fi and Bluetooth: An examination of coexistence approaches,” *http://www.mobilian.com/documents/WinHEC_whitepaper.pdf.*