Interference Aware Bluetooth Packet Scheduling

N. Golmie, N. Chevrollier and I. ElBakkouri
National Institute of Standards and Technology
Gaithersburg, Maryland 20899

Abstract— **Bluetooth is a radio technology for Wireless Personal Area Networks operating in the 2.4 GHz ISM band. Since both Bluetooth and IEEE 802.11 devices use the same frequency band and may likely come together in a laptop or may be close together at a desktop, interference may lead to significant performance degradation. The main goal of this paper is to propose a scheduling algorithm aimed at reducing the impact of interference. This algorithm takes advantage of the fact that devices in the same piconet will not be subject to the same levels of interference on all channels of the band. The basic idea is to utilize the Bluetooth frequency hopping pattern and distribute channels to devices such that to maximize their throughput while ensuring fairness of access among users. Simulation results are given for selected scenarios and configurations of interest.**

Keywords— **WPANs, Bluetooth, Interference, max-min fairness, scheduling.**

I. Introduction

An important requirement in the design of a scheduling mechanism for the Bluetooth technology is the support of heterogeneous traffic, a mix of voice and data applications such as email, ftp, remote login, and video with a wide range of delay, packet loss and throughput constraints.

Another key challenge in the design of a Bluetooth scheduling algorithm is probably the adaptiveness to a noisy environment. Today most radio technologies considered by Wireless Personal Area Network (WPAN) industry consortia and standard groups including the Bluetooth Special Interest Group [1], HomeRF [2], and the IEEE 802.15, employ the 2.4 GHz ISM frequency band. In addition both WPANs and Wireless Local Area Network (WLAN) devices implementing the IEEE 802.11 standard specifications [3] will be sharing the same frequency band. Thus, WLAN devices operating in proximity to Bluetooth devices can significantly impact the performance of Bluetooth devices and vice versa as shown in [4][5][6].

Our goal in this paper is to propose a fair packet scheduling algorithm for Bluetooth that reduces the impact of interference. In Bluetooth, the master device controls both *downstream* (master-to-slave), and *upstream* (slave-to-master) traffic directions. The master can use odd numbered slots to send data *downstream* while slaves have to wait to be "polled" by the master in order to send data *upstream* in even numbered slots. Although in this paper, we do not make a specific distinction between *upstream* and *downstream* traffic, we focus on a scheduling policy for polling slaves in order to allow them to access the channel. However our strategy or a similar policy can be used for either traffic directions. Furthermore, we assume that the source of interference to the Bluetooth system is an IEEE 802.11 system operating in a Direct Sequence Spread Spectrum (DSSS) mode. Note that our technique can be adapted to any other interference environment as well.

Recently, the issue of meeting different quality of service requirements in a wireless environment has been receiving more attention in the literature.

Fragouli, et. al. [7] proposed a strategy that combines class-based queuing [8] with channel-state based scheduling [9] that eliminates the Head of Line problem caused by FIFO queuing when certain devices suffer from a bad link. In [7], link sharing guidelines are provided to maximize channel utilization and limit the access of misbehaving sources.

Furthermore, a number of algorithms have been proposed on fair scheduling [10][11][12]. While there may be some differences in implementation and complexity, the basic idea in all these algorithms, is for sources experiencing a bad wireless link to relinquish the unutilized bandwidth to other sources that can take advantage of it. Compensation in bandwidth occurs when the channel conditions improve in order to achieve the so-called *Long Term Fairness* objective.

While the problem that we are trying to solve bears some resemblance with the problem addressed previously ([12] [7] [11][10]), we are more interested in an instantaneous measure of fairness rather than a *Long Term Fairness* objective. The reason is as follows. All previous work uses a two state Markov channel model for each link. The transition probabilities between the good and bad states are in the order of several seconds to account for periods of fading, multipath and various other wireless effects. The situation in our case is somewhat different due to the hopping nature of the Bluetooth device that uses a different frequency every 625 μs interval. Since different Bluetooth devices in a piconet will be subject to different interference levels due to parameters such as geometry and transmitted power, not all frequencies will be equally good to all devices. Therefore, our goal is to optimally assign frequencies such as to maximize channel utilization and guarantee fairness among the devices.

This paper is organized as follows. In section II we give some general insights on the Bluetooth protocol operation. In sections III and IV, we describe the scheduling mechanism and discuss its fairness properties respectively. In section V, we give simulation results and concluding remarks are offered in section VI.

II. Bluetooth Protocol Overview

In this section, we give a brief overview of the Bluetooth protocol [1]. Bluetooth is a short range (0 m - 10 m) wireless link technology aimed at replacing non-interoperable proprietary cables that connect phones, laptops, PDAs and other portable devices together. Bluetooth operates in the ISM frequency band starting at 2.402 GHz and ending at 2.483 GHz in the USA, and Europe. 79 RF channels of 1 MHz width are defined. The raw data rate is defined at 1 Mbits/s. A Time Division Multiplexing (TDM) technique divides the channel into 625 μs slots. Transmission occurs in packets that occupy an odd number of slots (up to 5). Each packet is transmitted on a different hop frequency with a maximum frequency hopping rate of 1600 hops/s.

Two or more units communicating on the same channel form a piconet, where one unit operates as a master and the others (a maximum of seven active at the same time) act as slaves. A channel is defined as a unique pseudo-random frequency hopping sequence derived from the master device's 48-bit address and its Bluetooth clock value. Slaves in the piconet synchronize their timing and frequency hopping to the master upon connection establishment. In the connection mode, the master controls the access to the channel using a polling scheme where master and slave transmissions alternate. The master uses even numbered slots while odd numbered slots are reserved for slave transmissions.

There are two types of link connections that can be established between a master and a slave: the Synchronous Connection-Oriented (SCO), and the Asynchronous Connection-Less (ACL) link. The SCO link is a symmetric point-to-point connection between a master and a slave defined to carry 64 kbits/s of a voice stream.

In this paper, we focus on a scheduling strategy used for transmitting data on the ACL link that defines an asymmetric point-to-point connection between a master and active slaves in the piconet. While the master can send data to a slave in the piconet on any even numbered slots, a slave has to be polled before it can transmit data. Therefore, the slave to master data rate is negotiated using Link Manager Protocol (LMP) messages at connection setup. The negotiated rate is usually defined in terms of a poll interval, and a packet length. Additional Quality of Service (QOS) parameters can be exchanged in Link Layer Control Adaptation Protocol (L2CAP) messages and include parameters such as peak bandwidth, latency and delay variation.

Several packet formats are defined for ACL, namely DM or DH packets that occupy either 1, 3, or 5 time slots. DM packets use Forward Error Correction (FEC) while DH packets do not have any FEC in the payload. An Automatic Repeat Request (ARQ) procedure is applied to ACL packets where packets are retransmitted in case of loss until a positive acknowledgement (ACK) is received at the source. The ACK is piggy-backed in the header of the returned packet where an ARQN bit is set to either 1 or 0 depending on whether the previous packet was successfully received or not. In addition, a sequence number (SEQN) bit is used in the packet header in order to provide a sequential ordering of data packets in a stream and filter out retransmissions at the destination.

In addition to ACL and SCO packets, the master and slave message exchange includes short POLL and NULL packets. POLL messages can be sent by the master and require an ACK while NULL messages can be sent by either the master or the slave and do not require an ACK.

III. Bluetooth Interference Aware Scheduling (BIAS)

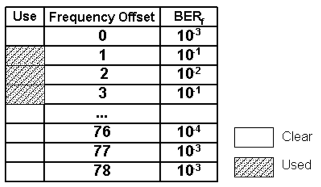
In this section, we present the Bluetooth Interference Aware Scheduling (BIAS) algorithm. Our main objective is to alleviate the impact of interference while maintaining fairness and supporting different Quality of Service (QoS).

In this sequel, we assume that the traffic from slave S_i to the master is characterized by a data rate, r_i, equal to $\frac{l_i}{p_i}$ where l_i is the packet length in slots (1, 3 or 5 slots depending on the packet type), and p_i is the poll interval in (master/slave) slot pairs. In addition, we assume the following transmission rules for the master and slave.

Master - The master polls slave S_i every p_i in order to guarantee r_i in the upstream direction. A poll message can be either a data or NULL message. A data packet is sent to slave S_i if there is a packet in the queue for slave S_i. This packet contains the ACK of the previous packet received from slave S_i. In case there is no data to transmit and the master needs to ACK a previous slave transmission, it sends a NULL packet to slave S_i.
Slave S_i **-** Upon receipt of a packet from the master, the slave can transmit a data packet. This data packet contains the ACK information of the master to slave packet transmission. In case the slave does not have any data to send, it sends a NULL packet in order to ACK the previous packet reception from the master. No ACK is required for a NULL message from the master.

Our algorithm consists of several components, namely, a channel estimation procedure, a procedure that assigns weights to devices in order to determine a channel access priority, and a resource credit function that allocates bandwidth to each device according to its service requirements and the state of the channel.

The *estimate_channel()* procedure is used to detect the presence of interference in the frequency band. Thus, each Bluetooth receiver maintains a *Frequency Usage Table* where a bit error rate measurement, BER_f, is associated to each frequency as shown in Figure 1. Frequencies are classified according to a criteria that measure the level of interference in the channel and are marked *used* or *clear* depending on whether their corresponding BER is above or below a threshold value, BER^T, respectively. Note that, other criteria such as frame error rate, packet loss, or the received signal strength can be used in addition to the bit error measurement to detect a high level of interference in a specific frequency band.



Use	Frequency Offset	BER_f
	0	10^{-3}
▨	1	10^{-1}
▨	2	10^{-2}
▨	3	10^{-1}
	...	
	76	10^{-4}
	77	10^{-3}
	78	10^{-3}

Fig. 1. Frequency Usage Table

Since the master device controls all transmissions in the piconet, the slaves need to send their *Frequency Usage Table* in the form of status update messages. The scheduler at the master can then make use of the measurements collected during the *Channel Estimation* phase in order to optimize the frequency allocation on each time slot and avoid a packet transmission in a receiving channel with a high level of interference. Figure 2 illustrates the frequency allocation that occurs at the master. In this case, frequency 78 is used to communicate with slave S_i, while frequencies 76, 1 and 0 are assigned to slaves S_i, S_{i+1}

and S_{i+1} in that order. Observe that the pattern of frequencies corresponds to the receiving frequencies. Thus, an M marks a receiving slot for the master device while an S is a receiving frequency for a slave device. Although, the master scheduler attempts to maximize channel utilization, it intentionally leaves certain slot pairs empty if either the master or the slave receiving frequency is *used*. Thus, in Figure 2, frequencies 16, 2, 7 and 77 are not used since frequencies 2 and 77 are not *clear* for the master.

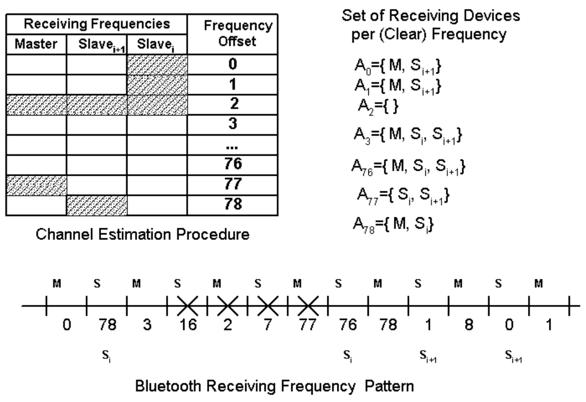


Fig. 2. Master Frequency Allocation Scheduling

The basic idea in the credit system is to control the bandwidth allocated to each device in order to ensure that no device gets more than its fair share of the available bandwidth. Thus, devices with a positive credit counter, c_i, are allowed to send data. There can be several ways to compute credits. Our method is based on the max-min fairness criteria [13]. Given r_i, we let u_i be the probability that a pair of slots (master/slave) are *clear*. Thus, u_i represents the available spectrum to slave i. Therefore, we write:

$$\begin{aligned} u_i &= P(slave\, i\, has\, a\, clear\, receiving\, frequency) \\ &\quad \times P(master\, has\, a\, clear\, receiving\, frequency) \end{aligned} \qquad (1)$$

where

$$P(device\, i\, has\, a\, clear\, receiving\, frequency) = \frac{Number\_of\_clear\_Channels_i}{Total\_Number\_of\_Channels} \qquad (2)$$

We then define μ_i as

$$\mu_i = \min(u_i, r_i) \qquad (3)$$

where μ_i is the minimum guaranteed rate for device i. Thus, in the case device i has a requested rate r_i, such that $r_i > u_i$, but is experiencing interference so it is not able to utilize more that u_i of the spectrum, and its rate is limited to u_i. We define a *constrained* device to be a device that is not able to use the entire frequency spectrum, such that $r_i > u_i$, while an *unconstrained* device is such that $r_i \le u_i$. The next step is to reallocate the leftover bandwidth that is unused by the *constrained* devices and let g_i be the actual rate given to device i:

$$g_i = \begin{cases} \mu_i + \frac{r_i(B-\mu)}{\sum_{j \in Unconstrained} r_j} & \text{if } r_i < u_i \\ \mu_i \quad otherwise \end{cases} \qquad (4)$$

where $\mu = \sum_i \mu_i$ and $B = 1 - \frac{Number\_of\_Used\_Channels_F}{Total\_Number\_of\_Channels}$. $Number\_of\_Used\_Channels_F$ is the number of frequencies that are marked *used* for all devices (in other words frequencies that can not be used by any device in the piconet). In essence, Equation 4 redistributes the leftover bandwidth to *unconstrained* devices proportionally to their service rate as in the Generalized Processor Scheduling (GPS) [14]. Thus, the *compute_credits()* function consists of computing the credits according to:

$$c_i = g_i \times N \qquad (5)$$

where N is the number of slot pairs considered in the allocation.

The other component of the algorithm is to actually give the ”right of way” or a priority of access to certain devices. We choose to give devices with fewer number of good channels a higher priority over other devices that have more channels available. Thus, in *compute_weights()* we set w_i as follows:

$$w_i = \min(\epsilon, P(slave\, i\, has\, a\, used\, receiving\, frequency)) \qquad (6)$$

where we define

$$P(slave\, i\, has\, a\, used\, receiving\, frequency) = \frac{Number\_of\_used\_Channels_i}{Total\_Number\_of\_Channels} \qquad (7)$$

and assume ϵ takes on values in $]0, \frac{1}{Total\_Number\_of\_Channels}]$ in the case all channels are *clear*. Finally, priority are assigned according to the ”send factor”, α_i, given by:

$$\alpha_i = w_i \cdot c_i \qquad (8)$$

A. BIAS Pseudocode

The algorithm’s pseudocode is as follows.

```
Every N slots
    estimate_channel();
    compute_weights();
    compute_credits();
    Every Even $TS_f$ // Master Transmission Slot
     if $TS_f + 1$ is clear // Master can receive in next slot
        $A_f$ = { set of slaves that can receive on frequency $f$ }
        $i = \max_{A_f}(\alpha_i)$ // Select device $i$ with the largest send factor
        if $\exists$ $i$ s.t. $qsize_i > 0$ and $\alpha_i > 0$
          $c_i$--;     //decrement credit counter
          $\alpha_i = w_i \times c_i$;     // update send factor
          transmit packet for slave $i$
```

Table I summarizes the parameters used in the algorithm and their definition.

TABLE I

DEFINITION OF PARAMETERS USED IN THE SCHEDULING ALGORITHM

Parameters	Definition
B	available spectrum
r_i	rate negotiated for device i
w_i	weight for device i
c_i	credit for device i
g_i	rate allocated for device i
α_i	send factor for device i
u_i	available frequency usage for device i
μ_i	minimum guaranteed rate for device i
ϵ	weight assigned to devices with $u_i = 1$

B. Numerical Example

Let's consider the *Frequency Usage Table* given in Figure 3 as an example. We consider 10 receiving frequencies, $f \in [0, 9]$. In order to keep the discussion simple, we show the frequency pattern for the *downstream* traffic and assume all 10 frequencies are *clear* for the master. We assume that there are 3 slaves in the piconet and each slave has a service rate equal to $r_1 = r_2 = r_3 = 1/3$ i.e. each slave gets polled every 6 slots. Since frequency 2 is marked *used* for all 3 slaves, noone can use it and $B = 9/10$. We compute u_i according to Equation 1. Slave, S_1, can use 2 out the 10 frequencies, therefore $u_1 = 2/10$. $u_2 = 7/10$ and $u_3 = 3/10$ for S_2 and S_3 respectively.

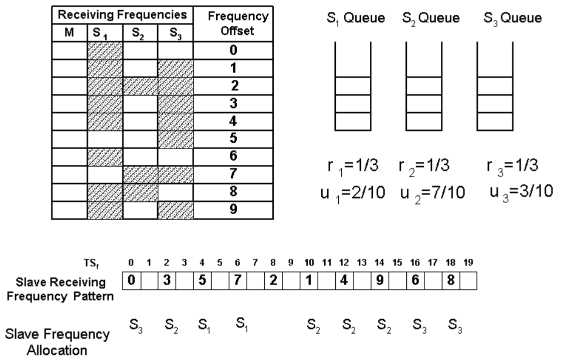


Fig. 3. Frequency Allocation Example

Similarly, $\mu_1 = 2/10$, $\mu_2 = 1/3$ and $\mu_3 = 3/10$. $\mu = \sum_{i=1}^{3} \mu_i = 25/30$. Since S_2 is *unconstrained*, it is a candidate device for receiving the leftover bandwidth. Therefore, $g_1 = 0.2$, $g_2 = \mu_2 + B - \mu = 0.4$, and $g_3 = 0.3$. The credits are $c_1 = 2$, $c_2 = 4$ and $c_3 = 3$ slots for $N = 10$. The weights are $w_1 = 8/10$, $w_2 = 3/10$ and $w_3 = 7/10$. At time $TS = 0$, $\alpha_1 = 2 * 8/10 = 8/5$ while $\alpha_2 = 4 * 3/10 = 6/5$ and $\alpha_3 = 3 * 7/10 = 27/10$. Therefore, S_3 is serviced at time $TS = 0$. A similar calculation determines the service order for S_1 on $TS = 4, 6$ and S_2 on $TS = 2, 10, 12, 14$, and S_3 on $TS = 16, 18$.

IV. BIAS PROPERTIES

In this section we summarize the features of the BIAS algorithm and highlight some of its fairness properties. Specifically, Theorem 1 states that error-free connections are serviced according to their negotiated rate and gives an upper bound derivation for the initial delay. Theorem 2 says that *clear* channels are shared among error-free and error-prone devices according to their proportional rate.

The BIAS algorithm properties are summarized as follows.
P1. Service guarantees are provided to error-free connections including delay bounds and throughput. Although, error-prone connections are given a higher priority of access, interference-free devices are not affected by the interference conditions subsiding on some devices in the piconet.
P2. The scheduling policy is work conserving since no slots will be left idle if there is at least one device with a positive credit counter.
P3. Short term max-min fairness is guaranteed since the leftover bandwidth unused by the error prone sessions is redistributed to error-free sessions proportionally to their negotiated service rate.
P4. The sharing of *clear* channels is proportional to each session's negotiated rate regardless of whether they are error-free or error-prone.

Theorem 1
The number of slots allocated to an error-free session i over an interval of N slot pairs is equal to at least $r_i \times N$. The initial delay (in slot pairs), Δ, for an error free session, k, to send its first packet over an interval of $1.25 \times N$ms is at most equal to:

$$\Delta = \sum_{i=1}^{k} \min(x_i, c_i) \qquad (9)$$

where i represents the index of slave i,and c_i is the credit given to device i. x_i is defined as:

$$x_i = \lceil \frac{c_i \cdot w_i - c_{i-1} \cdot w_{i-1}}{w_i} \rceil \qquad (10)$$

where w_i is the weight of device i, and indices are assigned to devices such that $w_1 \cdot c_1 > w_2 \cdot c_2 > w_k \cdot c_k > ... > w_n \cdot c_n$, for all n devices in the piconet.

Proof
The minimum number of slots allocated to an error free connection follows directly from Equation 4. Since error free connections are not constrained by their spectrum usage, their allocation is greater or equal to their negotiated service rate, r_i.

In order to prove the initial access delay bound, we consider two devices i and j such that $w_i \cdot c_i > w_j \cdot c_j$. Since $\alpha_j < \alpha_i$ ($\alpha_i = w_i \cdot c_i$), device j will only be allowed to access the channel after device i has transmitted at least $x_i = \lceil \frac{c_i \cdot w_i - c_j \cdot w_j)}{w_i} \rceil$. Note that

$$w_i(c_i - x_i) \leq w_j \cdot c_i \qquad (11)$$

and device j is allowed to transmit after device i has transmitted at least x_i packets. Also, observe that after the initial Δ slot pairs, both devices i and j will be serviced in a Round Robin (RR) fashion. This represents an upper bound since device i will keep its priority of access only if all Δ slot pairs are *clear*. In case, a *used* slot pair is encountered during the first Δ slot pairs and it is *clear* for device j, then j is allowed to transmit. □

Theorem 2
All devices sharing a set or subset of *clear* channels are serviced according to their allocated rate.

Proof
In case the devices sharing a set of *clear* frequencies have equal αs, they are serviced according to a RR policy. In the case devices have different αs, the device with the maximum α is serviced first. Assuming equal credit counters, and without loss of generality, a high value for α indicates that the device has limited usage of the spectrum and is therefore self-constrained. That is, the device will not be able to use every slot in the set and thus will not deny service to other devices sharing the same subset with a smaller value of α. □

V. SIMULATION RESULTS

In this section, we present simulation results to evaluate the performance of BIAS. The results obtained are compared with Round Robin (RR) scheduling. Our simulation environment is based on a detailed simulation environment consisting of the MAC, PHY and channel models for Bluetooth and IEEE 802.11 (WLAN) as described in [?]. We use the topology illustrated in Figure 4, and the simulation parameters presented in Table II.

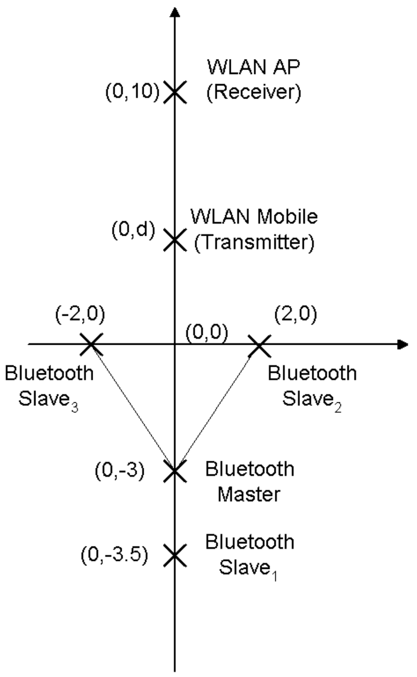


Fig. 4. Experiment Topology

We assume that WLAN is operating in the Direct Sequence Spread Spectrum (DSSS) mode. We vary the traffic distributions for WLAN and Bluetooth as follows. We assume that the WLAN Mobile device is transmitting data packets to the Access Point (AP) device which is responding with ACKs. The WLAN packet payload is set to 7776 bits transmitted at 11 Mbits/s, while the packet header is set to 224 bits transmitted at 1 Mbits/s. We assume that the WLAN packet interarrival rate is exponentially distributed with a mean of 1.86 ms corresponding to 50% of the offered load. For Bluetooth, we assume that the master device is transmitting DM1 packets with a mean arrival rate of λ where $\lambda = \frac{2*0.000625}{l} - 2*0.000625$ seconds, and $l = 25$ is the offered load in percent of the channel capacity. There are 3 slaves in the topology and they are sharing 25% of the total capacity. Thus, the offered load for each slave is set to 8.33%. The parameters used in the setup are summarized in Table II. Statistics are collected at the Bluetooth slave devices.

TABLE II
SIMULATION PARAMETERS

Bluetooth Parameters	**Values**
ACL Baseband Packet Encapsulation	DM1 , 366 bits
Packet Interarrival Time for the master	2.91 ms
Transmitted Power	1 mW
Slave 1 Coordinates	(0, -3.5)
Slave 2 Coordinates	(2,0)
Slave 3 Coordinates	(-2,0)
Master Coordinates	(0,-3)
WLAN Parameters	**Values**
Packet Interarrival Time	1.86 ms
Offered Load	50 % of Channel Capacity
Transmitted Power	25 mW
Data Rate	11 Mbits/s
AP Coordinates	(0,10)
Mobile Coordinates	(0,d)
Packet Header	224 bits
Payload Size	7776 bits

The performance metrics that we use include the packet loss, the mean access delay and the fairness index. The packet loss is the probability that a packet is dropped at a device due to interference. The access delay measures the time it takes to transmit a packet from the time it is passed to the MAC layer until it is successfully received at the destination.The delay is measured at the L2CAP layer. We define the fairness index for device i, $F_i = \frac{Number\_of\_Packets\_Received}{Expected\_Number\_of\_Packets\_Received}$. The $Number\_of\_Packets\_Received$ is the number of packets sent minus the number of packets dropped.

Figure 5 gives the Bluetooth slave packet loss with respect to d, the y-coordinate of the WLAN transmitter. As the WLAN transmitter moves further away from the Bluetooth piconet (d increases), the packet loss measured at each of the slave devices decreases for RR scheduling and is kept at zero for BIAS. At $d = 0$, the WLAN transmitter is 2m away from slaves 2 and 3 and 3 meters away from slave 3, the packet loss is 1%, 17.3%, and 17.8% for slaves 1, 2 and 3 respectively for RR and 0% for BIAS. Even when $d = 8$m (i.e. the WLAN device is at 11.18m away from slaves 2 and 3, the packet loss is still in the order of 6.7% for slaves 2 and 3 with RR.

Figure 6 gives the mean access delay for servicing the Bluetooth slaves. With RR, the mean access delay is around 1.6ms for $d = 0$m for all 3 slaves. With BIAS, the delay increases to ~ 3.6 ms for slaves 2 and 3 and 5.2ms for slave 1. This increase in delay is expected since there is a trade-off between packet loss and delay.Thus, in order to bring the packet loss to zero, bad frequencies are skipped at the expense of increasing

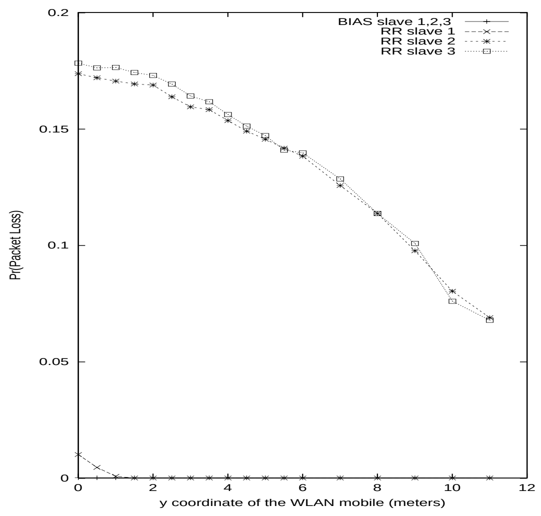


Fig. 5. Effect of Scheduling on Packet Loss.

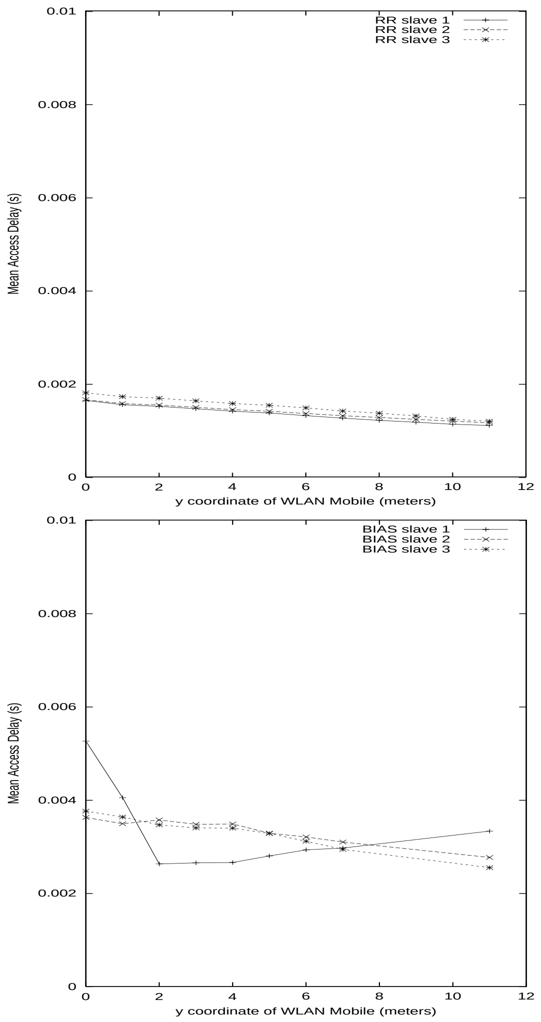


Fig. 6. $\boxed{\frac{(a)}{(b)}}$ Effect of Scheduling on the Mean Access Delay. (a) Round Robin Scheduling. (b) BIAS Scheduling

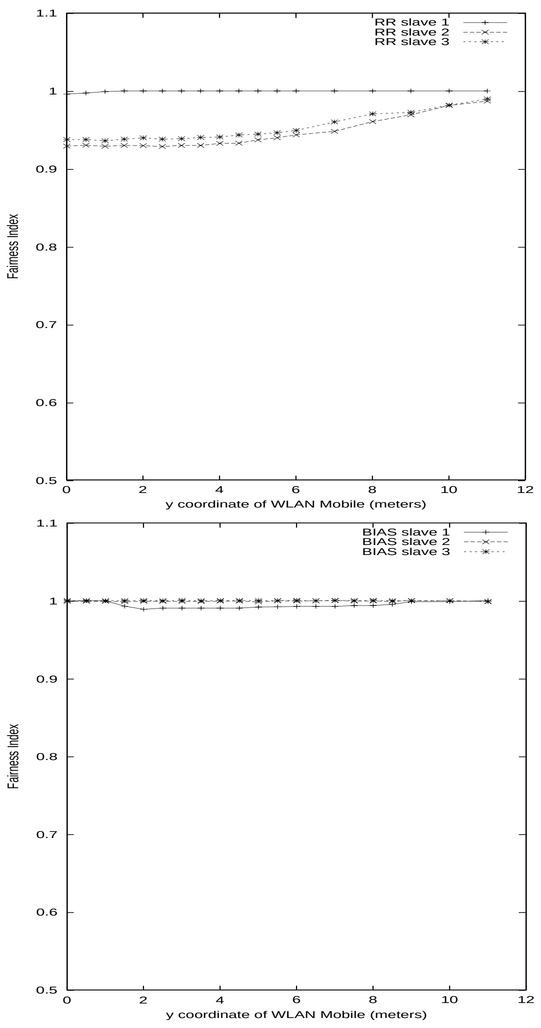


Fig. 7. $\boxed{\frac{(a)}{(b)}}$ Effect of Scheduling on Fairness. (a) Round Robin Scheduling. (b) BIAS Scheduling

the mean access delay. We verified that this result does not apply to multi-slot packets where the transmission time represents a larger fraction in the delay calculation. Therefore, reducing the packet loss when multi-slot packets are used reduces the mean access delay as well.

Figure 7 gives the fairness index with RR and BIAS. We note that all 3 slaves get the same number of packets with BIAS and their fairness index is 1 regardless of the position of the WLAN transmitter. For RR, the fairness index is 0.99 for slave 1, while it is 0.92 and 0.93 for slaves 2 and 3 respectively for $d = 0$m. Also, note that slaves 2 and 3 experience 17% of packet loss. Thus, more packets are being sent to slaves 2 and 3 with RR and the channel utilization is as not as efficient as with BIAS. As the offered load increases, we expect this effect to be magnified and lead to unfairness and degradation in servicing slave 1, which the error-free device in this case. Also, on the topic of channel utilization, we observe that with RR scheduling the number of NULL packets transmitted is 12% higher than with BIAS. This is mainly due to the packet loss that leads to retransmissions and therefore the number of NULL packets to ACK data transmissions is higher.

VI. Concluding Remarks

In this paper we present, BIAS, a scheduling technique for alleviating the impact of interference on the Bluetooth performance. This technique attempts to redistribute the bandwidth unused by the interference-prone sessions to other error-free connections that can take advantage of it. Our goal is to guarantee fairness in scheduling while maximizing the channel utilization.

Our simulation results indicate that BIAS can significantly lower the probability of packet loss for sessions experiencing interference without much increase in the mean access delay for the worst case scenario. Furthermore, we demonstrate that BIAS can provide *Short Term Fairness* where error-free sessions are still serviced according to their negotiated rate regardless of the channel conditions of other devices.

Our current work is focused on extending BIAS and investigating the use of combined approaches such as packet encapsulation and flow control in order to support QoS in a Bluetooth environment.

References

[1] Bluetooth Special Interest Group, "Specifications of the Bluetooth System, vol. 1, v.1.0B 'Core' and vol. 2 v1.0B 'Profiles'," December 1999.

[2] K. J. Negus, A. P. Stephens, and J. Lansford, "HomeRF: Wireless Networking for the Connected Home," in *IEEE Personal Communications*, February 2000, pp. 20–27.

[3] IEEE Std. 802-11, "IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification ," June 1997.

[4] N. Golmie and F. Mouveaux, "Interference in the 2.4 GHz ISM band: Impact on the Bluetooth access control performance," in *Proceedings of IEEE ICC'01*, 2001.

[5] A. Soltanian and R. E. Van Dyck, "Physical layer performance for coexistence of Bluetooth and IEEE 802.11b," in *in Virginia Tech. Symposium on Wireless Personal Communications*, June 2001.

[6] N. Golmie, R.E. Van Dyck and A. Soltanian, "Interference of Bluetooth and IEEE 802.11: Simulation Modeling and Performance Evaluation," in *Proceedings of the Fourth ACM International Workshop on Modeling, Analysis, and Simulation of Wireless and Mobile Systems, MSWIM'01*, Rome, Italy, July 2001.

[7] C. Fragouli, V. Sivaraman, and M. B. Srivastava, "Controlled Multimedia wireless link sharing via enhanced class-based queuing with channel-state-dependent packet scheduling," in *Proceedings of IEEE INFOCOM'98*, San Fransisco, CA, March 1998, pp. 572–580.

[8] S. Flyod and V. Jacobson, "Link Sharing and resource management models for packet networks," in *ACM/IEEE Transactions on Networking*, August 1995, vol. 3, pp. 365–386.

[9] P. Bhagwat, P. Bhattacharya, A. Krishna, and S. Tripathi, "Enhancing Throughput over Wireless LANs using Channel State Dependent Packet Scheduling," in *Proceedings of IEEE INFOCOM'96*, 1996, vol. 3, pp. 1133–1140.

[10] S. Lu, V. Bharghawan, and R. Srikant, "Fair Scheduling in wireless packet networks," in *Proceedings of ACM SIGCOMM'97*, September 1997, pp. 63–74.

[11] P. Ramanathan and P. Agrawal, "Adapting packet fair algorithms to wireless networks," in *Proceedings of ACM/IEEE MOBICOM'98*, October 1998, pp. 1–9.

[12] T.S.E. Ng, I. Stoica, and H. Zhang, "Packet fair queueing algorithms for wireless networks with location dependent errors," in *Proceedings of INFOCOM'98*, March 1998, pp. 1103–1111.

[13] D. Bertsekas and R. Gallager, *Data Networks, 2nd Ed.*, Prentice Hall, 1992.

[14] A. Parekh and R. Gallager, "A generalized processor sharing approach to flow control - the single node case," in *ACM/IEEE Transactions on Networking*, June 1993, vol. 1, pp. 344–357.