MAC Scheduling Policies with Reduced Power Consumption and Bounded Packet Delays for Centrally Controlled TDD Wireless Networks

Indraneel Chakraborty, Abhishek Kashyap, Apurva Kumar, Anupam Rastogi,

Huzur Saran & Rajeev Shorey

IBM India Research Laboratory,

Block 1, Indian Institute of Technology,

Hauz Khas, New Delhi 110016, India.

Email: `saran@cse.iitd.ernet.in, srajeev@in.ibm.com`

Phone: 91-11-6861100, Fax: 91-11-6861555.

Abstract—**In networks with mobile nodes, power consumption is an important constraint due to the limited battery power available at the mobile nodes. Reduction of power consumption in mobile devices can be achieved by keeping these devices in a low power (standby) mode during periods of inactivity. This may, however, result in increased end-to-end delays leading to violation of the Quality of Service (QoS) parameters. In this paper, we propose a Medium Access Control (MAC) scheduling policies for centrally controlled TDD wireless systems. The proposed policies reduce the power consumption of mobile devices by putting them into low power mode intelligently, using probabilistic estimates of inactivity based on the previous traffic arrival pattern. The policies ensure that the QoS parameters such as end-to-end packet delays are not violated.**

Keywords— **Wireless Networks, Bluetooth Technology, Scheduling, Power Aware Scheduling, TDMA/TDD, MAC.**

I. INTRODUCTION

Mobile devices typically have limited energy for computing and communication because of the limited battery life-times. Conserving battery power in mobile devices is an important consideration in designing protocols for networks with mobile nodes. This issue should be considered through all the layers of the protocol stack, including the application layer [5], [4].

The chief sources of energy consumption in a mobile unit are the CPU, the transmitter, and the receiver. CPU usage in mobile devices may be reduced by relegating most of the high-complexity computation (related to media access) to the stationary network. The focus of work in this paper is on the power usage at the transceiver (i.e., transmitter, receiver).

A radio can operate in three modes: standby, receive and transmit. We will refer to the mode in which the devices can receive and transmit data as *active* mode. In general, the radio consumes more power during transmission than during reception, and consumes the least power in the standby mode. For example, the GEC Plessey DE6003 [6] 2.4 GHz radio requires 1.8 W in transmit mode, 0.6 W in receive and 0.05 W in standby mode. The power consumption for Lucent's 15 dBm 2.4 GHz Wavelan radio is 1.725 W in transmit mode, 1.475 W in receive mode, and 0.08 W in standby mode [7]. Frequent use of standby mode can reduce the power consumption in mobile devices.

The MAC scheduling algorithm in wireless networks has to be such that the mobile devices remain in standby mode when there is no data to transmit or receive. The constraint of switching a device to the standby mode is that the end-to-end delays

may increase and thereby violate the Quality of Service (QoS) parameters. Therefore, an efficient scheduling algorithm has to be such that the QoS parameters are not violated. Moreover, frequent switching from one mode to another may itself lead to consumption of power. The need for minimizing such transitions requires that the device should move to standby mode after determining the expected overhead in switching and comparing it to the power it saves by going in to the standby mode.

Motivated by emerging standards for low power, low cost, short range indoor wireless networks (such as Bluetooth [1], HomeRF [2]), in this paper, we study efficient MAC scheduling algorithms in centrally controlled Time Division Duplex (TDD) wireless networks. We attempt to address how a scheduler should respond to different traffic types at the mobile nodes such that the power consumption of mobile devices in a wireless network is reduced without violating the QoS constraints.

We can best capture the requirements for MAC scheduling in a wireless network by listing properties that the scheduler should meet. The results in this paper are based upon the following principles: (i) the power consumption of a mobile device should be in proportion to the traffic at the connection, (ii) the end-to-end packet delays should not violate the specified QoS constraints, (iii) the bandwidth wastage in uplink and downlink *polling* in the absence of data is minimum, (iv) the capacity unused by one flow should be distributed uniformly among other flows. In the proposed scheduling policy, the scheduler tries to learn the nature of the traffic at the mobile nodes and polls a mobile device only when the device has data thus eliminating unnecessary polling. The overhead of switching a device to low power mode is compared to the expected power savings and a device is transferred to low power mode only if the overhead is less as compared to the expected power savings for that device.

This paper is organized as follows. Section II discusses the network model. Section III describes some simple scheduling algorithms. Section IV describes the proposed scheduling algorithm: Adaptive Probability based Polling Interval (APPI). The simulation results are presented in Section V and Section VI has the conclusions.

II. The Network Model

We focus on MAC scheduling in indoor short range wireless networks. An example of such a system is a Bluetooth piconet [1]. Another example is the HomeRF Shared Wireless Access Protocol (SWAP) [2] which is designed to support both TDMA and CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance). We assume that the scheduler views traffic as a set of flows. Flows can be individual, e.g., a single TCP connection, or aggregate, e.g., all traffic to a specific host.

A. Power Modes in Bluetooth Technology

Bluetooth [1] is a fast frequency hopping (1600 hops/sec) master driven Time Division Duplex (TDD) MAC wireless network, comprising of at most seven slaves connected to the master which receive and transmit baseband packets.

A slave can transmit data in the slot following the one in which the master has polled it. Bluetooth technology defines the *active* mode for a slave as the mode in which the slave has to listen to the channel for master transmissions at all times. On receiving a packet from the master, every active slave reads the destination slave address and packet length from the packet header. If the packet is not addressed to a slave, it stops scanning the channel for the duration of the packet length (in the packet header). The addressed slave will reply in the following reverse slot. If the master has no data to send during a slot when it polls a slave, it sends a *null* packet; the slave replies to the packet received from the master since the reply contains an acknowledgement for the received packet.

In Bluetooth technology, in addition to the *active* mode, there are three low power modes, *sniff*, *hold* and *park*. In this paper, for simplicity, we consider two power modes: the *active* and the *sniff* mode.

In the *sniff* mode (see Figure 1), the duty cycle of the listen activity for a slave can be reduced. In this mode, the master can only start transmission in specified time slots, thus a slave does not need to listen to the channel at all times. The sniff slots are placed at an interval T_{sniff}. A slave in sniff mode listens for a transmission every sniff period, T_{sniff}, for a specified number of slots (denoted by $N_{\mathrm{sniff-attempt}}$). We call the interval T_{sniff} as the *polling interval* for a slave in the *sniff* state.
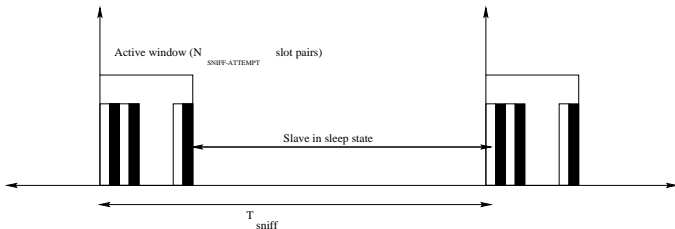

Fig. 1. The Sniff mode in Bluetooth Technology

III. The Scheduling Policies

In a Bluetooth picocell, the master performs the task of packet scheduling for both the uplink and downlink flows. However, the master has only limited knowledge of the arrival processes of uplink flows. Power optimization is required due to the limited battery power at the mobile hosts. Furthermore, the network must provide sustained quality of service (QoS) to the packet flows. This is important since if the packet delay is large, there may be re-transmissions due to a higher level protocol (such as a timeout in TCP) that leads to a further wastage of power.

Mobile hosts with limited battery power must conserve energy. Hence they are put in a low power mode when they do not have any traffic. The scheduling policy at the master should have criteria to decide the power mode of each mobile host connected to it so that the least amount of power is consumed.

To satisfy the QoS parameters, the end-to-end packet delay should be minimized. This means that the polling frequency of the mobile host by the master (in anticipation of traffic arrival) must be high. However, this will consume more power in transmission and reception and will also lead to wastage of bandwidth in unnecessary polling. Thus there are two contradictory requirements that need to be satisfied by a scheduling policy at the master: (i) low power consumption at the mobile nodes, (ii) appropriate polling frequency that yields low end-to-end delays.

To begin with, we describe some simple scheduling policies that estimate the next polling interval each time the mobile host is polled. These policies optimize either the QoS parameters or the power consumption at the mobile nodes. A brief overview of each policy is provided below.

A. Always Active Mode (AAM)

In this policy, the slaves are in *active* mode at all times and the master implements Earliest Deadline First (EDF) [8] scheduling for polling the slaves as this is known to yield superior results compared to other policies with respect to throughput and QoS parameters. The maximum time that a slave cannot be serviced is the *deadline* of service for that slave. Keeping each slave always active ensures that no slave misses the QoS guarantees, however, this leads to a wastage of battery power and bandwidth.

B. Fixed Polling Interval (FPI)

In this policy, the master transfers a connection to the *sniff* mode with a fixed *polling interval* when there is no data enqueued at its buffer and transfers the connection back to *active* mode when packets arrive. This policy uses EDF scheduling for the *active* slaves when they have data to send. FPI has an advantage over the previous policy in terms of power consumption, however, the end-to-end delays are high.

C. Mean Policy (MEAN)

In this policy, we compute the mean inter-arrival time of packet bursts and set it equal to the polling interval. This policy works better than FPI in terms of power savings but increases the average end-to-end delay of packets.

D. Off-line Optimum Policy

In this policy, the master schedules the connections based on the previous knowledge of the arrival of data, thus transferring connections into *sniff* mode and *active* mode based on exact power and end-to-end delay calculations, thereby optimizing both of them. Note that this is an ideal, unimplementable policy.

IV. The proposed policy: APPI

Adaptive Probability based Polling Interval (APPI) is an easy to implement algorithm for deciding the polling time for connections in a low power (*sniff*) mode (we recall that the polling time is the T_{sniff} parameter). APPI makes the intuitive assumption that the inter-arrival time till the next packet is drawn from the same distribution as the inter-arrival times that have been observed so far. Hence the inter-arrival times can be observed to obtain the expected time of arrival of a packet.

To learn the distribution D of the traffic for a given connection, a separate learning function H of the observed inter-arrival times of data bursts is kept for the forward (master to slave) and the reverse (slave to master) data traffic. The inter-arrival times of data bursts observed and recorded in the learning function correspond to the first packet in each burst. The remaining packets of a burst are excluded from the observation as they normally arrive within negligible time interval of the first packet. For each time interval $I \in 0...(m-1)$, $H(i)$ is the number of observed inter-arrival times in the interval $[(i)M/m, (i+1)M/m)$, where the parameter m is the number of entries in H and M is the maximum inter-arrival time for the observation.
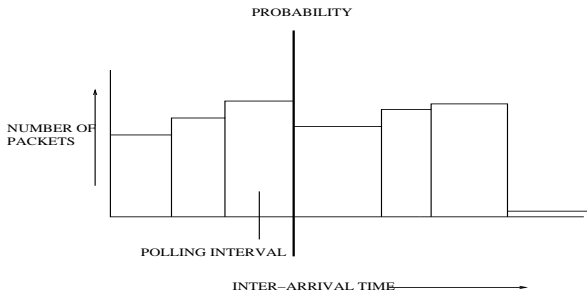

Fig. 2. The Learning Function

A. The Learning Function

We approximate the distribution of the incoming traffic at a connection by a learning function H of the inter-arrival times of the data packets. This is the histogram of the inter-arrival times of packets. The learning function stores the number of data-bursts that have arrived in particular ranges of inter-arrival times in the entry corresponding to those inter-arrival ranges. We call the inter-arrival ranges as columns in this paper. Figure 2 shows a learning function with the x-axis representing the inter-arrival time and y-axis corresponding to the number of bursts that have arrived within the inter-arrival ranges.

B. Condition for Sniffing

When a connection is in *active* mode and a data-burst ends, we calculate inter-arrival time T_{as}. This parameter should be chosen such that the probability of an arrival of the next data-burst of sufficient length within T_{as} that can force (due to QoS requirements over individual packet delay) the connection into *active* mode (if it is put into *sniff* mode) is less than a threshold

P_{as}. This prediction is done with the help of a learning function H.

If the probability $P(t)$ of the arrival of such a packet crosses a threshold P_{as} when integrated upto a time T_{as}, and the power overhead P_{overhead} of putting the connection into *sniff* mode and reverting it back to *active* mode is less than the power saved in the time T_{as}, the connection is put into *sniff* mode. Note that P_{as} depends upon the acceptable values of the QoS parameters.

Let f_D be the probability density function of the traffic distribution D, T_{as} is the inter-arrival time drawn from D. If the transmit power per slot is $P_{transmit}$, the receive power per slot is $P_{receive}$ and the power in sniff state per slot is P_{sniff}, then,

$$P(t) = \int_0^{T_{as}} f_D(x)dx \leq P_{as} \qquad \text{and}$$

$$(T_{as} - T_{as}/\text{deadline})\, P_{receive} + (T_{as}/\text{deadline})\, P_{transmit}$$
$$-T_{as}\, P_{sniff} \geq\ P_{\text{overhead}}$$

Note that *deadline* refers to the deadline of service for the slaves in *active* mode. From the above, it is clear that if the second condition is satisfied for the connection, then the slave is switched to *sniff* mode.

C. Deciding the Polling Interval in Sniff Mode

The second aim is to find a time-interval such that the expected time interval before the arrival of the next burst is greater than a probability P_B which reflects the tolerance of the connection to delayed packets. Thus,

$$P(t) = \int_0^{T_P} f_D(x)dx \leq P_B$$

where T_P is the polling interval of the slave in *sniff* mode, T_{sniff} (see Figure 1). In Figure 2, we represent the procedure of deciding the polling interval. The number of packets in different inter-arrival ranges are added over the inter-arrival ranges starting from the beginning till their sum becomes equal to P_B (≤ 1) times the total number of bursts. The probability P_B is shown as *PROBABILITY* in Figure 2. P_B refers to the fraction of packet bursts arrived till a particular inter-arrival range. We will find the last inter-arrival range (column) upto which P_B fraction of packet bursts have arrived. The mean of this column is taken to be the polling interval.

D. Deciding the Criterion of Switching from Sniff to Active Mode

If a packet being served in the *sniff* mode is estimated to get a delay higher than a threshold (depending upon the QoS), then the connection is immediately switched into the *active* mode. This is done by measuring the burst length b, the sniff-interval T_{sniff} and thereby estimating the maximum delay d of the last packet in the queue, where the estimated delay is given by $(b-1)\, T_{\text{sniff}}$. If the delay is less than d at the master, the master puts the slave in *active* mode, else the slave will request for a change of mode if the condition is satisfied at the slave.

E. Overview of the Algorithm

The following is a description of the various scheduler events at the master for the Adaptive Probability based Polling Interval

with Fixed Resolution (APPI-FR).

Arrival of a burst at a node (master or slave): whenever a new burst arrives at the queue for a particular stream for scheduling at the master (slave), the inter-arrival time between the last burst and the present burst is noted in a histogram $H_k{}^M$ ($H_k{}^S$) where M (S) indicates that the histogram is for the data-traffic from the master to the slave (slave to the master) and k indicates the slave index.

An active slave is scheduled: if a connection has data to transmit, then the connection is serviced. However, if the connection has no data in the queue, then if the condition of sniffing (as described in Section IV-B) is satisfied, by both the master and slave histograms H^M and H^S, then the connection is put into *sniff* mode.

A slave in sniff mode is scheduled: when a connection with a slave in *sniff* mode is scheduled, the MAC at the master and the MAC at the slave check the criteria of switching from *sniff* to *active*. If it is satisfied, the connection is switched to *active* state. Otherwise, the slave is serviced and the master MAC informs the slave of the new polling interval based on H^M and H^S both of which are at the master. The polling interval chosen is the minimum of the two arrived at by using H^M and H^S.

F. Adaptive Policy with Adaptive Resolution (APPI-AR)

The policy described above works well with a large number of columns in the histogram H. This leads to computational overheads at each polling interval. Hence, for reasons of efficiency, we decrease the number of columns in the histogram. This may lead to an incorrect selection of polling interval which may not be acceptable.

A new policy is therefore suggested in which the number of columns in the histogram is kept small. This policy handles the distribution by keeping smaller histogram columns in the zone where the expected probability of data arrival is high. The columns cover large intervals in the other zones where the expected probability of data arrival is low.

The advantages of this policy are evident. Though having lesser columns in the histogram H, this policy works out precise polling intervals as it is adaptive to the nature of the traffic. The total number of columns is also low, hence the computational overheads are avoided without any loss in the resolution of polling interval. All the advantages of APPI are still available.

In APPI-AR, whenever the learning function is updated at the arrival of a data-burst at the master or the slave, it is checked that the expected probability of data arrival at a polling interval (e.g., j) does not exceed a prefixed threshold P_r which is a function of the total number of columns N in the adaptive histogram. This condition can be written as $\frac{H(j)}{\sum_{i=0}^{N} H^M(i)} \geq P_r$. If such be the case, the column is bifurcated into two intervals. To keep the number of columns constant, two adjacent columns with the minimum sum of probability are joined together. This achieves higher resolution of the inter-arrival range when the data rate is higher while keeping the number of columns to a small and fixed number.

V. SIMULATION RESULTS

We have used the Bluetooth software stack developed at IBM for simulating the wireless network with a master driven Time Division Duplex MAC and one piconet. The Bluetooth software stack has been built using the Network Simulator (NS-2) [3]. We have used the following traffic traces: (i) TCP Traffic trace (downloaded from from *ee.lbl.gov/sigcomm/ITA*), (ii) TCP with HTTP as the application layer (using Network Simulator), (iii) TCP with FTP application layer (using Network Simulator), (iv) RealPlay Audio traffic trace, (v) CBR traffic.

The parameters used in the simulation are as follows: the polling interval in low power mode is adaptive and is between 100-500 slots, the duration of each slot is 625 microseconds, DEADLINE is 40 slots, polling interval for FPI is 250 slots, P_{sniff} is equal to 0.05 units per slot, P_{receive} is equal to 0.5 units per slot, P_{transmit} is equal to 1 unit per slot, P_{overhead} is $2P_{\text{transmit}} + 2P_{\text{receive}} = 3$ power units, P_{as} is 0.3 and P_B is 0.3.

The power levels above are the worst case figures taken from Lucent's WaveLAN card. The simulations were performed with the above parameters for different optimization policies: AAM, FPI, MEAN and APPI-AR. The results for all the policies have been normalized with respect to the off-line policy.

The simulation was run for 110,000 Bluetooth slots (each slot is of 625 μsec duration) and the results were noted down from 50,000 slots onwards, thus ignoring the time taken by the histogram to adapt to the nature of the traffic.

The graphs showing the power consumed by different policies normalized with respect to the off-line optimum are shown below. The off-line policy has prior information about data arrival and schedules accordingly. The average end-to-end link delays, maximum end-to-end delays and their jitter are also shown in the graphs. However the delays should be compared with the delay with respect to the AAM policy since the off-line delays are unimplementable.

The number of columns in the histograms for APPI-AR have been taken to be 5 with equally spaced boundaries. The boundaries of the columns are then adapted to the particular traffic which requires a finer resolution at specific inter-arrival times as per the concerned traffic distribution.
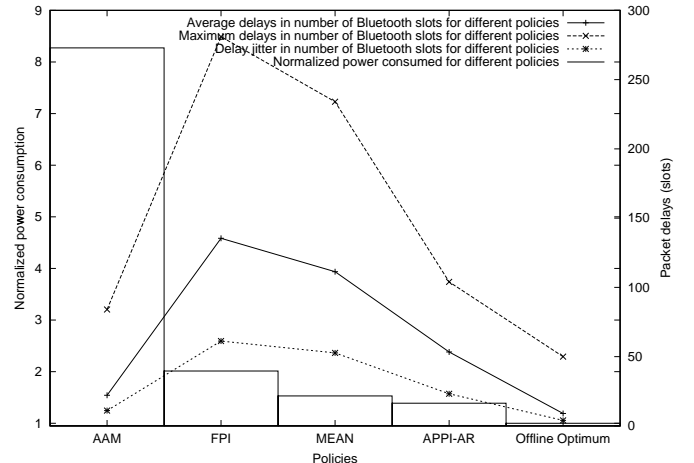

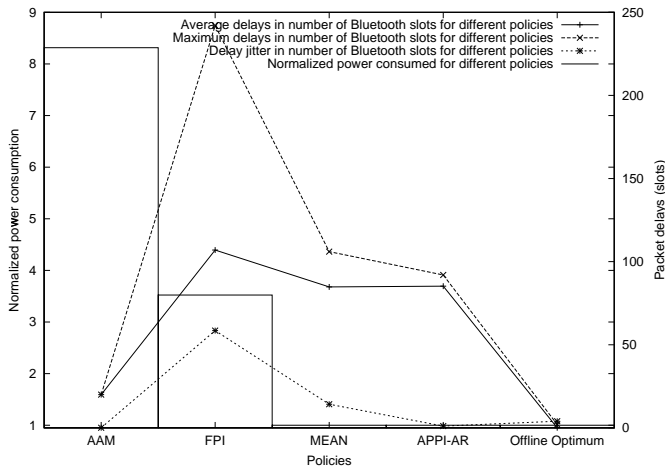Fig. 3. The Power and Delay for HTTP over TCP
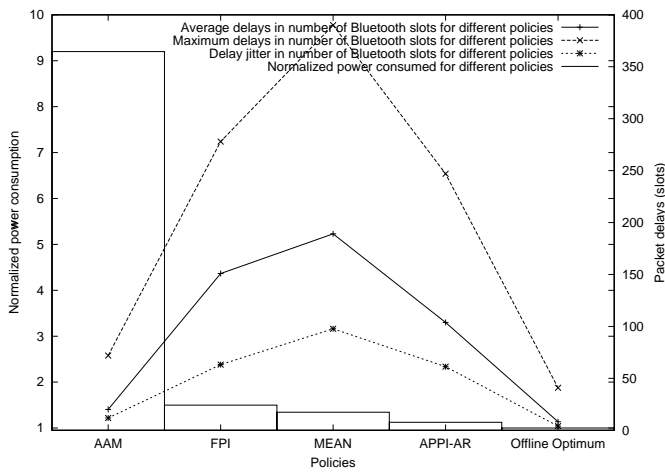
Fig. 4. The Power and Delay for CBR over UDP

Fig. 6. The Power and Delay for Real Audio from http://www.indiafm.com at 16Kbps

Fig. 5. The Power and Delay for TCP trace from http://ee.lbl.gov/sigcomm/ITA
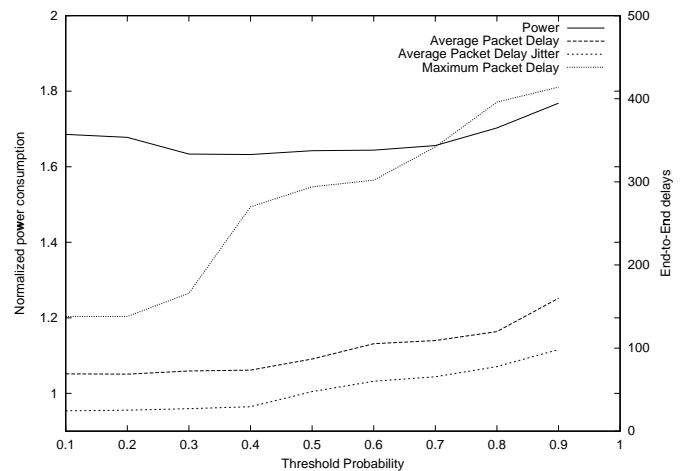
Fig. 7. The performance of APPI-AR with different threshold probabilities

Figures 3, 5 show the power consumed at a node and average, maximum and variance (jitter) of the end-to-end packet delays for HTTP over TCP (from network simulator) and TCP trace (from the site *ee.lbl.gov*). The graphs show that APPI-AR works better than both the power saving policies, FPI and MEAN, both in terms of power consumption and end-to-end delays. Figure 6 shows the power consumption and packet delays for a TCPdump trace from a realplayer playing 16Kbps audio from a website. The delay in audio play corresponds to the maximum end-to-end delay. The graph shows that maximum delays are very high for FPI and MEAN. Figure 4 shows the power consumption and delays for a CBR traffic with a packet arrival every 120 time slots. As can be seen from the graph, both our policy and the mean policy adapts to the inter-arrival rate of packets exactly as it is fixed. The graph shown in Figure 7 plots the power consumed, average and maximum delay and the jitter for APPI for different probabilities P_B. From the graph, we infer that a good range for P_B is between 0.1 and 0.3.

VI. CONCLUSION

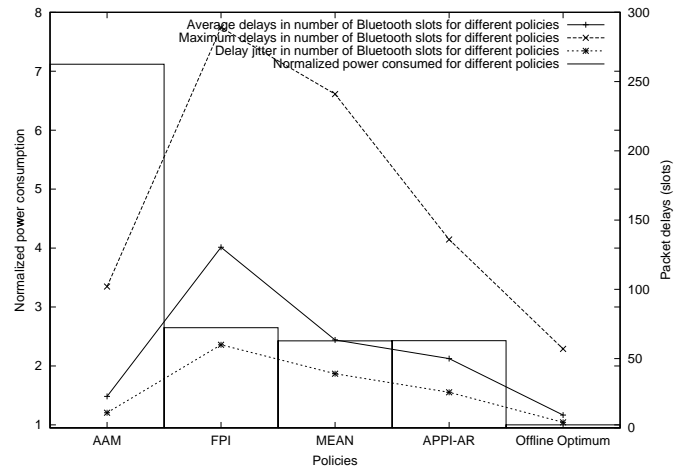We have proposed scheduling methodologies for master driven TDD wireless systems, such as Bluetooth, that optimize power consumption at the mobile devices and that simultaneously bound the end-to-end packet delays. An adaptive policy has been proposed for deciding the polling interval for connections based on their traffic distribution and QoS parameters. Further research is required to study MAC scheduling in such systems and to study other important QoS parameters such as the packet loss probability and jitter.

REFERENCES

[1] Bluetooth Special Interest Group. http://www.bluetooth.com
[2] http://www.homeRF.org
[3] http://www.isi.edu/nsnam/ns/
[4] J. M. Rulnick and N. Bambos, "Mobile power management for maximum battery life in wireless communication networks", In *Proc. of IEEE INFOCOM*, 1996.
[5] J. C. Chen, K. M. Sivalingam, P. Agrawal and S. Kishore, "A Comparison of MAC Protocols for Wireless Local Networks Based on Battery Power Consumption", In *Proc. of IEEE INFOCOM*, 1998.
[6] http://www.networks.digital.com/npb/html/products_guide /roamwir2.html, Jan 14, 1998.
[7] M. Stemm, P. Gauthier and D. Harada, "Reducing power consumption of network interfaces in hand-held devices", In 3rd International Workshop on Mobile Multimedia Communications, September, 1996.
[8] J. Hong, X. Tan and D. Towsley, "A performance analysis of minimum laxity and earliest deadline scheduling in a real-time system", *IEEE Transactions on Computers, Volume 38, Issue 12*, Dec, 1989