Uplink Downlink Limited Round Robin Scheduling Algorithms

Rawya Rizk
Electrical Engineering Department, Suez Canal University, Egypt

Abstract: *Several polling algorithms have been proposed to overcome the slot wastage problem due to the exchange of POLL or NULL packets in the case of no data to transmit. However, most of these algorithms suffer from achieving the fairness among all slaves. This paper proposes two scheduling algorithms, Uplink Downlink Limited Round Robin (LRR) with fixed cycle time and Uplink Downlink LRR with adaptive cycle time. They utilize the status of the uplink queues in addition to the downlink queues to schedule the slots effectively. The proposed algorithms combine the simplicity and fairness of the LRR algorithm with the efficiency of the queue state dependent packet algorithms. Simulation results show that the proposed scheduling algorithms achieve higher utilization, lower delay and reasonable fairness among the slaves compared to the existed LRR algorithm. These results are proved practically. Experimental results confirm that, the proposed algorithms achieve more than 14% improvement in the utilization compared to the LRR algorithm in asymmetric traffic conditions.*

Keywords: *Bluetooth, intra piconet scheduling algorithms, polling, Round Robin, segmentation and reassembly.*

Received July 20, 2008; accepted November 31, 2008

1. Introduction

Bluetooth enables the design of low-power, low-cost, and small-size radios. It is going to play an important role in communication among small electronic devices and the access to wired networking infrastructure. The transmission rate is up to 1 Mbps. The channel is divided into time slots of $625\mu s$. A Time-Division Duplex (TDD) scheme is used for full-duplex communication [1, 4].

Bluetooth provides two types of services. First, Synchronous Connection-Oriented (SCO) are provided for real-time delay-sensitive traffic such as voice. Second, Asynchronous Connection-Less (ACL) are provided for data. The ACL links are classified based on the data rates they carry, namely High Data (DH) rate packets and Medium Data (DM) rate packets. The packets are also classified into three types based on their lengths that include 1, 3, or 5 time slots [2, 3].

Bluetooth devices are organized into piconets, where a central device acts as the master, which can manage up to seven active slaves. The master fully controls the traffic in the piconet. The master-to-slave communication is called downlink and the slave-to-master communication is called uplink. A slave is allowed to start transmission in a given slot if the master has addressed it in the previous slot. The master addresses a slave by sending a 1-slot poll packet. Then, the slave must respond by sending a data packet or a 1-slot null packet if it has nothing to send [12]. Thus, the performance of data traffic in the Bluetooth piconet is critically dependent on the manner and the sequence in which the master visits or polls its slaves- the scheduling algorithm. Several scheduling algorithms have been proposed and developed in the recent years. These algorithms broadly fall under two categories: Round Robin (RR) algorithms and queue state dependent packet scheduling algorithms such as Priority Policy (PP) and K-Fairness Policy (KFP).

In the RR, the master polls each slave sequentially. The Pure Round Robin (PRR) is the original polling algorithm in which the master visits each slave for exactly one frame [13]. It has a slot wastage problem since many slots are wasted by poll-null exchanges when polled pairs have no data in the queues. The Exhaustive Round Robin (ERR) exhausts the payload of each master-slave pair in both sides at each cycle [7, 9]. It reduces the number of visits for idle queues. However, it does not provide fairness since the channel may be captured by higher traffic slaves. In the Gated Round Robin (GRR), only the packets buffered at the slave queue are served, while the packets that arrive during the service time wait to be served at the next cycle [8]. It reduces the capture effect, but provides increase of the average delay at asymmetric traffic conditions. In the Limited Round Robin (LRR), the master stays with a slave for a fixed number of frames [9]. It provides more fairness than ERR and GRR by solving the capture effect but still suffers from a slot wastage problem.

Queue state dependent packet scheduling algorithms perform preferential polling based on each pair's queue state [5, 10]. The PP assigns a priority class to each pair. In the KFP, a lower priority pair is sacrificed to a higher priority pair until a certain threshold. Once this threshold value is reached, the RR is resumed leading to degraded performance. These algorithms give

transmission opportunities to higher classes without considering the fairness among the pairs. In this paper, two new algorithms are proposed, uplink downlink - LRR algorithms. The proposed algorithms combine simplicity and fairness of the RR algorithms with the efficiency of the queue state dependent packet algorithms. The rest of the paper is organized as follows. Section 2 presents the two proposed uplink downlink LRR scheduling algorithms. Section 3 introduces the performance evaluation of the proposed algorithms compared with the conventional LRR algorithm. Section 4 shows an experimental evaluation of the proposed algorithms. Finally, section 5 presents the conclusion.

2. Proposed Scheduling Algorithms

In this section, the two proposed algorithms are presented. They combine the simplicity of RR algorithms with the priority feature to solve the slot wastage problem and remain fair and efficient in asymmetric flow rates. In the two proposed algorithms, the master-slave pairs are classified into three priority classes based on their queue states. Table 1 demonstrates the three priority classes. The states $M \to S$ or $S \to M$ indicate the presence of data in the downlink queue or uplink queue; respectively. The states $\overline{M} \to S$ or $\overline{S} \to M$ indicate that there is no data in the downlink queue or uplink queue; respectively. Therefore, the master needs to know the status of both the uplink and downlink queues at each cycle. Slaves can provide information about their queue states using the unused fields in the Bluetooth payload header. Some queue state dependent packet scheduling algorithms assume that the master has updated information on the uplink queues; however this assumption is not suitable for the current Bluetooth standards [6]. This information can only provided to the master when a slave send a packet to it. In the proposed scheduling algorithms, it is assumed that the master has information about the uplink queue status from the previous cycle. Therefore, the proposed algorithms use the current queue status for the master and the previous queue status for the slaves.

Table 1. Priority classes of master-slave connections.

Priority Classes	Queue Status	Description
Class 1: High	$M \to S$ and $S \to M$	The slave has traffic to AND from the master.
Class 2: Mid.	$M \to S$ and $\overline{S} \to M$	The slave has traffic only from the master.
	$\overline{M} \to S$ and $S \to M$	The slave has traffic only to the master.
Class 3: Low	$\overline{M} \to S$ and $\overline{S} \to M$	Other slaves.

Two main assumptions are considered to avoid starvation of the low priority pairs. First, the maximum number of transmissions per cycle for class 1, class 2 are 8 and 4; respectively. Thus, the maximum time of each pair $Max_t(C)$ depends on whether it belongs to class 1 (C =1) or class 2 (C =2):

$$Max_{t(C)} = \begin{cases} 8 \times \overline{L} = 24\, slots & if\ C=1 \\ 4 \times \overline{L} = 12\, slots & if\ C=2 \end{cases} \quad (1)$$

These values have a high effect on the performance. Using lower number of transmissions increases the delay and may cause a slot wastage problem since the algorithm will tend to PRR. While, higher number of transmissions causes capture effect by higher traffic slaves and thus starvation of the other slaves like ERR. Second, a pair of class 3 is avoided to be polled for only two cycles. After that, it takes the opportunity to be polled. The number of cycles that it must wait is increased by two each time it still inactive pair. Table 2 summarizes the notations used in the proposed algorithms.

Table 2. Notations.

Symbol	Meaning	Value
N_s	Number of slaves	1:7
N_a	Number of active slaves	
i	Number of a polled slave	
L	Packet length	1, 3, or 5 (slots)
$\overline{L}$	Mean packet length	3 (slots)
C	Class of each pair	1,2, or 3
$t(i)$	Time of each pair with maximum value = $Max_t(C)$	$Max_t(1)$= 24 (slots) $Max_t(2)$= 12 (slots)
T	Cycle time with maximum value = Max_T	At UD-FC: Max_T =60 (slots) At UD-AC: Max_T = 74 (slots) (at the first cycle)
$\overline{T}$	Average time of all polled pairs	(slots)
D_i	The packet delay in every connection between the master and each slave.	(*ms*)
$\overline{D}$	Average packet delay.	(*ms*)
F	Faireness	

2.1. Uplink Downlink LRR with Fixed Cycle Time

The Uplink Downlink LRR with Fixed Cycle Time (UD-FC) algorithm assumes that the master have an updated information on the status of the slave-to-master queues. It classifies the pairs according to the presence or absence of data in both uplink and downlink queues and considers the size of the queues. The UD-FC algorithm is described in Figure 1. It has the following steps:

a. At the beginning of each cycle, the master determines the polling order by:
   i. Arranging pairs according to three priority classes in an ascending order.
   ii. Prioritizing pairs in each class according to the sum of both uplink and downlink queues in a descending order.
b. The time of transmission of ith slave (t(i)) is

initialized and the data of this pair is still serviced until one of these two conditions comes true:

i. There is no more data in both uplink and downlink queues.
ii. The time of transmission exceeds the maximum time of transmissions of the class that this pair belongs to it.

c. The master polls the next slave if the cycle time (T) is still less than the maximum cycle time (Max_T); otherwise the next cycle begins.

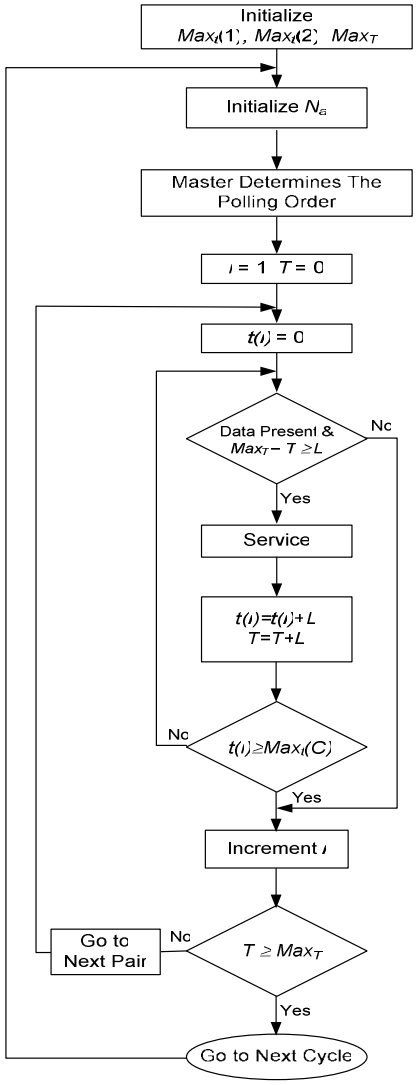


Figure 1. Flow chart of UD-FC.

2.2. Uplink Downlink LRR with Adaptive Cycle Time

One problem of the UD-FC algorithm is that the cycle time is constant. Therefore, the highest priority slave in the cycle may enjoy a large portion of the available cycle time and the lower priority slaves have little chance to be polled. The Uplink Downlink LRR with Adaptive Cycle Time (UD-AC) algorithm assumes that the first cycle time is defined in advance. At the end of each cycle, the time of the next cycle is adopted due to the number of the polled active slaves in the present cycle. The time of the next cycle may increase or decrease with respect to the present cycle if the number of polled slaves is less or more than the number of active slaves; respectively. Figure 2 describes the UD-AC algorithm. At the first cycle, the maximum cycle time is initialized as:

$$Max_T = Max_t(2) \times N_s \quad (2)$$

where, the maximum time of class 2 $(Max_t(2))$ is the average time of the three classes. At the end of each cycle, the maximum cycle time of the next cycle is adopted as:

$$Max_T = \begin{cases} Max_T - \bar{T}(i - N_a) & \text{if} \quad N_a < i \\ Max_T & \text{if} \quad N_a = i \\ Max_T + \bar{T}(N_a - i) & \text{if} \quad N_a > i \end{cases} \quad (3)$$

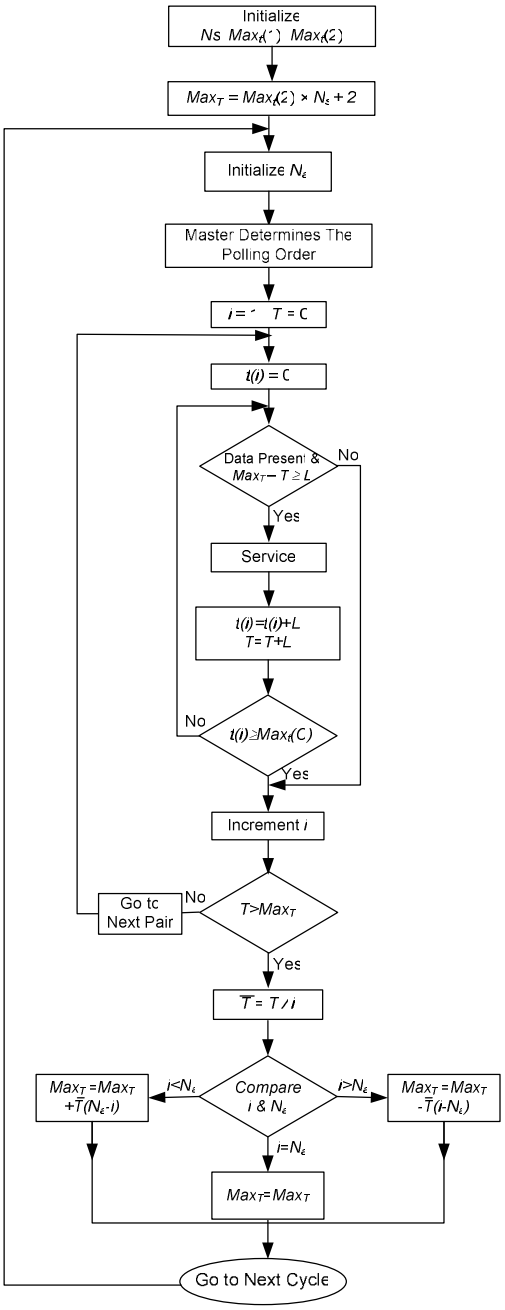


Figure 2. Flow chart of UD-AC

3. Performance Evaluation

3.1. Simulation Environment

The simulator is developed using C++. The simulated time is 200 seconds. It is assumed that all time parameters are multiple of Bluetooth slot ($625\mu s$). The master has a corresponding queue for each slave. Data traffic is generated independently for each queue according to a poisson process [11]. DH packets only are assumed. The number of slaves ranges from one to seven. Three performance metrics are evaluated, delay, channel utilization, and fairness. The channel utilization percentage is the total number of bytes delivered by the link during an interval to the total possible bytes that could have left the link in the same interval. The utilization is simply defined as the ratio of the utilized time and the total time. The fairness can be intended in two ways [9] a network can be fair in terms of bandwidth allocated to each user or in terms of the time delay each master-slave connection spent until its service is complete. Focusing on the delay since it has the most importance in real networks, the system is fair if the time which master spent at each slave is approximately equal. The Fairness always lies between 0 and 1. A higher fairness index indicates better fairness between connections. Given a set of delays $(D_1, D_2, \ldots, D_{Ns})$, the fairness index is calculated as follows [10]:

$$F(D_1, D_2, \ldots, D_{N_s}) = \frac{(\sum_{i=1}^{N_s} D_i)^2}{N_s \sum_{i=1}^{N_s} D_i^2} \qquad (4)$$

3.2. Simulation Results

Figures 3 and 8 show the simulation results of the two proposed algorithms, UD-FC and UD-AC compared with the conventional LRR. Figures 3 and 4 present the total packet delay with respect to the load and the number of slaves; respectively. It is clear that the proposed algorithms outperform the LRR algorithm. The average delay is the lowest for UD-AC since the adaptive cycle algorithm chooses the most appropriate cycle time, depending on the load and the number of active slaves.

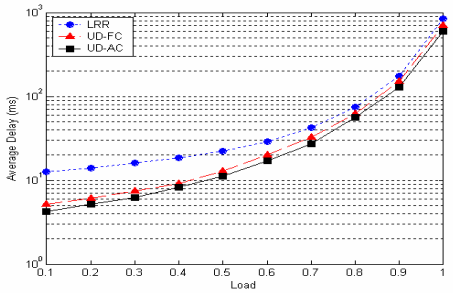


Figure 3. Average delay versus load in LRR, UD-FC and UD-AC algorithms.

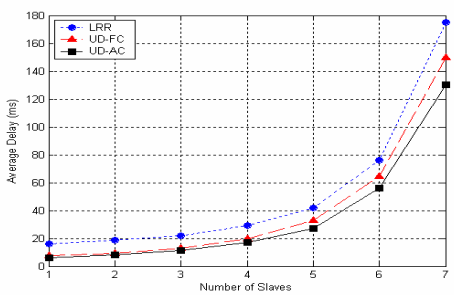


Figure 4. Average delay versus number of slaves in LRR, UD-FC and UD-AC algorithms.

Figures 5 and 6 show the channel utilization percentage with varying the load and the number of slaves; respectively. It is noticed that the utilization of UD-FC is higher than that in UD-AC under low load (less than 0.3) or low number of slaves (up to 2 slaves) since that all slaves have a chance for transmission. Therefore, under low load there is no need to the cycle time adaptation of UD-AC. However, UD-AC achieves channel utilization about 90% as opposed to only 80% for UD-FC and 60% for LRR at heavy load (load=0.8).

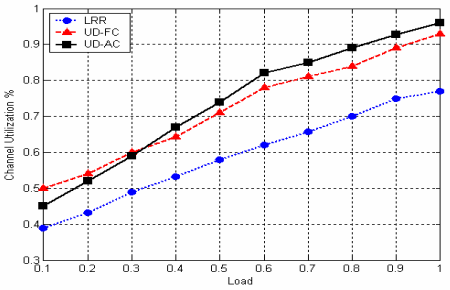


Figure 5. Channel utilization against load.

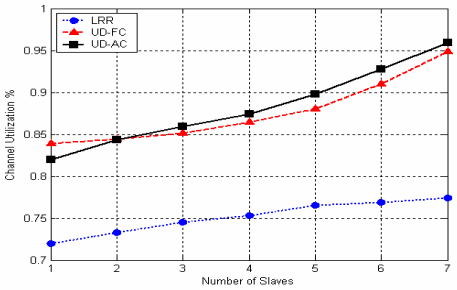
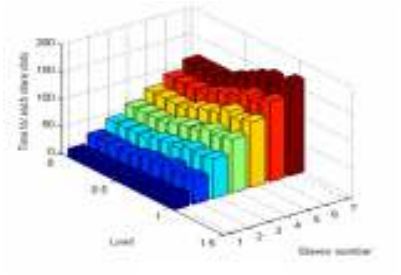
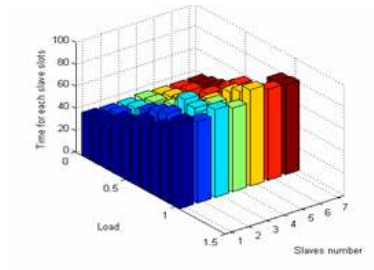


Figure 6. Channel utilization against number of slaves.

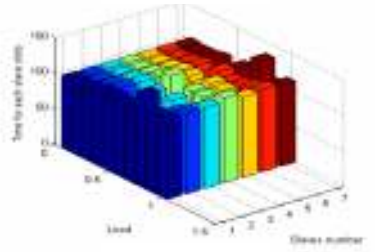
The fairness in the three algorithms is illustrated in Figures 7 and 8. Figure 7 shows the distribution of time among the slaves in the three algorithms. It is observed that how UD-AC achieves equal distribution of time among the seven slaves. While using LRR may cause an unfair sharing of time between the slaves. Figure 8 confirms this result. It is clear that UD-AC provides the highest fairness. The difference between the LRR algorithm and the two proposed algorithms decreases as the increase of load. This is because all the pairs will have data to transmit and each pair gets an equal opportunity for data transmission.



a. LRR



b. UD-FC



c. UD-AC

Figure 7. Distribution of time among slaves in LRR, UD-FC and UD-AC algorithms.

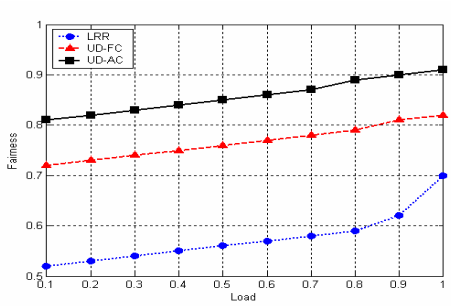


Figure 8. Fairness in LRR, UD-FC and UD-AC algorithms.

4. Experimental Evaluation

In this section, a practical experiment is presented as a case study to prove the correctness of the simulation results. The experiment includes a hardware Bluetooth piconet. The assumed scenario is that the piconet with 6 ACL slaves with asymmetric traffic (*i.e.*, variable packet arrival rates).

4.1. Experiment Environment

The case study consists of master and six slaves. The master is a Personal Computer (PC) and the slaves are mobile phones. The software used in transmission is BlueSender program [14]. The following characteristics are assumed:

a. The Packet Types

Since that, the time of the packet types DH1, DH3, and DH5 is measured by microseconds (μsec); it is difficult to measure the transmission time. So, relative packet sizes corresponding to DH1, DH3, and DH5 are used that their time can be measured by seconds (*sec*). As known in the Bluetooth standards [5], DH1 is 27 bytes, DH3 is 183 bytes, and DH5 is 339 bytes. Since that the ratio between DH1:DH3 is 1:7 and DH1:DH5 is 1:13, the packets used in the experiment have the same ratios. So, the packet types used in the experiment are assumed as:

- P1 = 3Mb with time = 7 *sec* corresponding to DH1 with time = 625μsec,
- P3 = 21Mb with time = 15 *sec* corresponding to DH3 with time = 1875μsec, and
- P5 = 39Mb with time = 19 *sec* corresponding to DH5 with time = 3125μsec.

b. The Queue Status

Table 3 describes the assumed traffic distribution in the first time slot.

Table 3. The queues status in the first time slot.

Slaves	$M \rightarrow S$	$S \rightarrow M$	Class	Queue Size (T)	Polling order
S_1	__	__	Class 3	0	__
S_2	2P5	__	Class 2	10	4
S_3	5P3	3P3	Class1	24	1
S_4	7P1	7P1	Class1	14	2
S_5	__	__	Class 3	0	__
S_6	__	3P5	Class 2	15	3

c. The Polling Algorithms

Three polling algorithms are used in the experiment: the traditional LRR and the two proposed algorithms UD-FC and UD-AC. The timing diagrams of the first cycle of the three algorithms are presented in Figures 9, 10, and 11; respectively. In the traditional LRR algorithm, the slaves' order is fixed in every cycle. Many slots are wasted by poll-null exchanges when polled pairs have no data in the queues. In every cycle, the master and the corresponding slave must exchange a poll–null frame to know the end of the transmitted data frames. It is shown from Figure 9 that poll-null frames waste four time slots since S_1 and S_5 have empty queues. In addition, S_6 is always has a lower chance for transmission since it is always the last polled slave. The UD-FC algorithm polls active slaves only. This leads to higher utilization than the traditional LRR. However, using fixed cycle time can't adapt to the bursty nature of the Bluetooth. It is shown from Figure 10 that, the transmission time of S_2 is less than the maximum transmission time of class 2 however it still has data. The lower chance of S_2 for transmission is due to the fixed cycle time. While, UD-AC algorithm can adapt its cycle time according to the variable traffic load.

Therefore, all the active slaves have the same chance for transmission. Figure 11 shows the timing diagram of the first cycle at UD-AC.

4.2. Experimental Results

In this section, the experimental results for LRR, UD-FC and UD-AC algorithms are compared. It is assumed that, the human effect is neglected during transmission. Two measures are taken into account, the wasted time and the utilization. The wasted time is obtained by counting the poll-null frames that occur during the visit to one slave at the time some other uplink or downlink queues are non-empty. The utilization is computed as the ratio between utilized time and the total time. Table 4 summarizes the results.

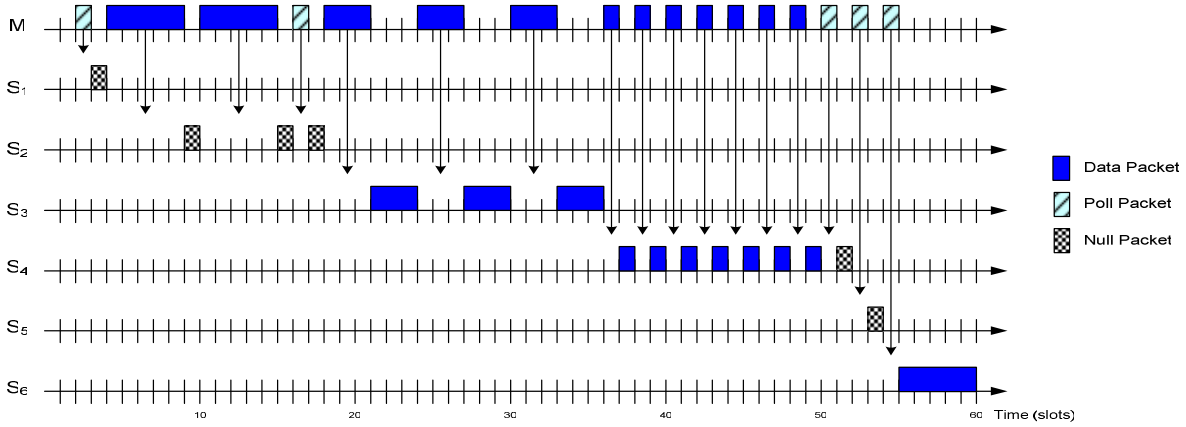


Figure 9. Timing diagram of the LRR algorithm.

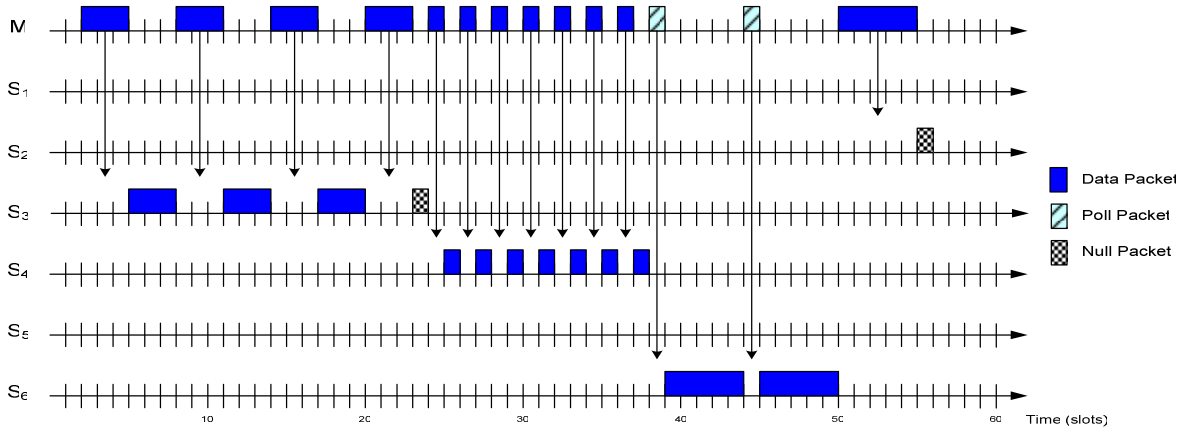


Figure 10. Timing diagram of the UD-FC algorithm.

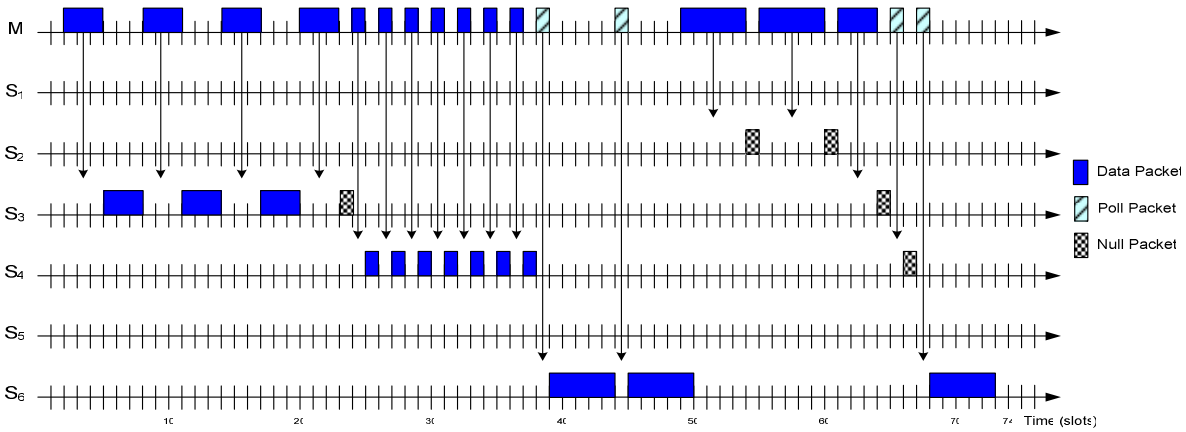


Figure 11. Timing diagram of the UD-AC algorithm.

Table 4. The experimental results.

	LRR algorithm	UD-FC algorithm	UD-AC algorithm (in the 1st time slot)	UD-AC algorithm (in the 3rd time slot)
The total time	322 sec.	288 sec.	376 sec.	
The utilized time	= 3×P5+6×P3+14×P1 = 245 sec. = 0.029 *sec* (w.r.t using DH packets)	= 3×P5+7×P3+14×P1 = 260 sec. = 0.031 *sec* (w.r.t using DH packets)	=5×P5+8×P3+14×P1 = 313 sec. = 0.039 *sec* (w.r.t using DH packets)	
The wasted time	= 11×p1 = 77 sec. = 6.875 *msec* (w.r.t using DH packets) = 23.9%	= 4×P1 = 28 sec = 2.50 *msec* (w.r.t using DH packets) = 9.7%	= 9×P1 = 63 sec. = 5.625 *msec* (w.r.t using DH packets) = 16.8%	7.6 %
The utilization	**76.1 %**	**90.3%**	**83.2 %**	**92.4%**

It is observed form Table 4 that, the proposed UD-FC algorithm saves more time than the traditional LRR algorithm. In the first time slot, it enhances both the wasted time and utilization parameters. However, the UD-AC algorithm can adapt to the variations of the load in the consequent time slots. Thus, it leads to improve in the performance over UD-FC after a small number of time slots. The wasted time and the utilization of the three algorithms are represented in Figures 12 and 13, respectively. These results confirm the simulation results. It is clear from the results that the wasted time in UD-AC and UD-FC is much lower than the wasted time in LRR. In addition, the two proposed algorithms achieve utilization above 90%, as opposed to only 76% for the LRR.

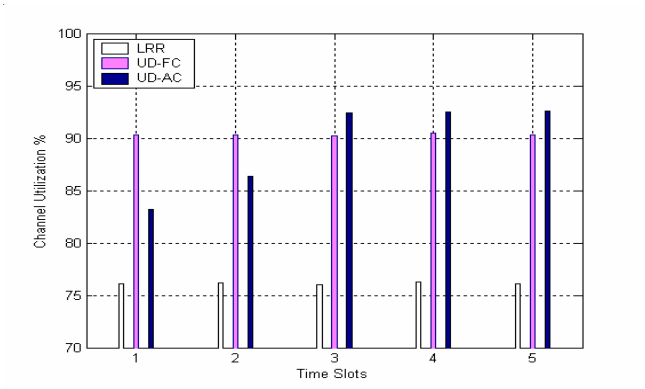


Figure 12. Channel utilization through five time slots.

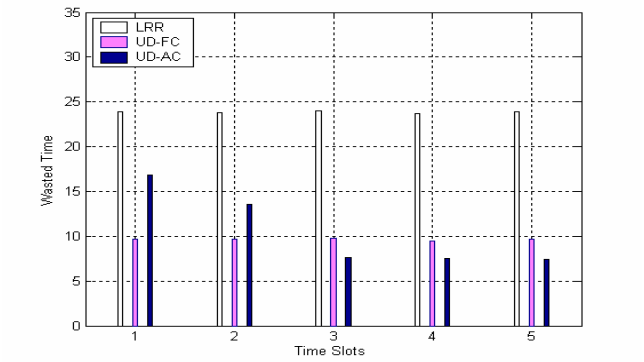


Figure 13. Wasted time through five time slots.

5. Conclusions

This paper considered the slot wastage and capture effect problems in designing efficient and fair scheduling algorithm in Bluetooth. Two efficient, fair and simple scheduling algorithms, Uplink Downlink LRR with Fixed Cycle Time (UD-FC) and Uplink Downlink LRR with Adaptive Cycle Time (UD-AC) are proposed in this paper. They enhance the performance of LRR by reducing the rate of visits to queues. They indicate the active slaves at the beginning of each cycle and prioritize these slaves according to the presence or absence of data and then according to the size of their uplink and downlink queues. The main difference between the two proposed algorithms is that the UD-AC adopts the time of each cycle based on the number of polled slaves in the previous cycle. This gives equal chances for transmission to all active slaves in asymmetric traffic load conditions. Simulation results prove that the proposed UD-FC and UD-AC algorithms are fair and capable of achieving high utilization of the piconet bandwidth in addition to low packet delay and wasted time. A practical experiment is presented to prove the improvement in the performance over the traditional LRR algorithm.

References

[1] Chakrabarti S., Vuong S., Sinha A., and Paul R., "Convergence in Bluetooth and 802.11 Networks," *in Proceedings of International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Croatia and Italy, 2004.

[2] Das A., Ghose A., Razdan R., Saran H., and Shorey R., "Enhancing Performance of Asynchronous Data Traffic Over the Bluetooth Wireless Ad-hoc Network," *in Proceedings of IEEE INFOCOM'01*, pp. 591-600, 2001.

[3] Haartsen J., "The Bluetooth Radio System," *IEEE Personal Communications Magazine*, vol. 7, no. 1, pp. 28-36, 2000.

[4] Jahansson P., Kazantzidis M., Kapoor R., and Gerla M., "Bluetooth: An Enabler for Personal Area Networking," *in Proceedings of IEEE Network*, pp. 28-37, 2001.

[5] Kalvala J. and Zaruba G., "Differentiated Priority Scheduling and Adaptive Segmentation for

Bluetooth Piconets," *in Proceedings of the 39th Hawaii International Conference on System Sciences*, pp. 233p, 2006.

[6] Lee S., Park B., and Latchman H., "Enhanced Limited Round Robin with a Priority Policy for Bluetooth," *in Proceedings of 9th International Conference on Advanced Communication Technology (ICACT)*, pp. 182-185, 2007.

[7] Mercier A., Minet P., and George L., "A Class-Based EDF Scheduling for Bluetooth Piconet," *Ad Hoc And Sensor Wireless Networks (AHSWN)* vol. 1, pp. 343-364, 2005.

[8] Miorandi D., Zanella A., and Pierobon G., "Performance Evaluation of Bluetooth Polling Schemes: An Analytical Approach," *Mobile Networks and Applications*, vol. 9, no. 1, pp. 63-72, 2004.

[9] Mišić V. and Mišić J., "Modeling Bluetooth Piconet Performance," *IEEE Communications Letters*, vol. 7, no. 1, 2003.

[10] Oh J., Joo Y., Kwon O., Kim Y., Lee T., and Tchah K., "Differentiated Fairness Guaranteeing Scheduling Policies for Bluetooth," *in Proceedings of IEEE 56th Vehicular Technology Conference (VTC),* vol. 2, pp. 923-926, 2002.

[11] Saidane L., Minet P., Ayed S., and Mercier A., "Deterministic and Probabilistic QoS Guarantees in a Bluetooth Piconet with 1-RR/FIFO Scheduling," *in Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW)*, Washington, 2006.

[12] Yang C. and Liu C., "A Bandwidth-Based Polling Scheme for QoS Support in Bluetooth," *Computer Communications*, vol. 27, pp. 1236-1247, 2004.

[13] Zussman G., Yechiali U., and Segall A., "Exact Probabilistic Analysis of the Limited Scheduling Algorithm for Symmetrical Bluetooth Piconets," *in Proceedings of IFIP-TC6 Personal Wireless Communications (PWC),* pp. 276-290, 2003.

[14] Available on http://www.bluesender.com /content/view/14/28.

Rawya Rizk is an assistant professor in Electrical Engineering Department, Suez Canal University, Egypt. She received her BS degree, MS degree, and PhD degree in computer and control engineering, Suez Canal University, in 1991, 1996, and 2001; respectively. Her research interests are in the area of computer networking, including mobile networking, ATM, QoS, traffic and congestion control, and handoffs.