An Inter-Piconet Scheduling Algorithm for Bluetooth Scatternets

Suman Saha, Mitsuji Matsumoto
GITS, Waseda University, Tokyo, Japan
suman@asagi.waseda.jp, mmatsumoto@waseda.jp

Abstract

Bluetooth is a wireless specification delivering short-range radio communication between electronic devices that are equipped with specialized Bluetooth chips. It lets nearly all devices talk to one another by creating a common language between them. The basic Bluetooth network topology (referred to as a piconet) is a collection of slave devices operating together with one master. A multi-hop ad-hoc network of piconets in which some of the devices are present in more than one piconet is referred to as a scatternet. Efficient scatternet data flow requires design of inter-piconet scheduling algorithms. This paper presents and evaluates a fair and efficient scheduling algorithm for inter-piconet based on the Bluetooth hold mode. During intra-piconet scheduling, we have used soft-QoS based SFPQ algorithm. SFPQ is a fair and efficient polling algorithm. Finally, we evaluate the performance of proposed algorithm regarding efficient and fairness inter-piconet scheduling via simulation and show that the proposed algorithm outperforms Load Adaptive Scheduling Algorithm.

1. Introduction

Bluetooth is an emerging standard for Wireless Personal Area Networks (WPANs): short range, ad hoc wireless networks [1]. Originally, Bluetooth was envisaged as a wireless cable replacement technique, hence the basic RF range of Bluetooth devices is only 10 meters [2]. However, the number of possible uses of Bluetooth has increased to include different networking tasks between computers and computer-controlled devices such as PDAs, mobile phones, smart peripherals and others. Figure 1 illustrates the Bluetooth stack, designed by the Bluetooth Special Interest Group (SIG). I shall briefly present the layers:
Radio – Operating in the 2.4 GHz Industrial, Scientific and Medical (ISM) band provides the physical channel among Bluetooth devices with a gross bit rate of 1 Mb/s (for more information see [2] and [3]).
Baseband – Controls the physical link through the radio, assembling packets and controlling frequency hopping.
Link Manager Protocol (*LMP*) – Controls and configures links to other devices (for example negotiates hold timeout).
Logical Link Control and Adaptation (*L2CAP*) – Multiplexes data from higher layers and converts between different packet sizes.

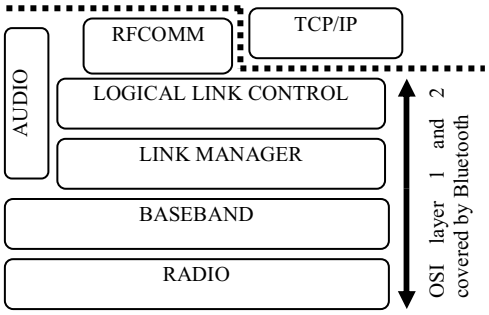

Figure 1. Bluetooth stack

The basic Bluetooth network topology (referred to as a *piconet*) is a collection of slave devices operating together with one master. A multihop ad-hoc network of piconets in which some of the devices are present in more than one piconet is referred to as a *scatternet* (see for example Figure 2). A device that is a member of more than one piconet (referred to as *bridge*) must schedule its presence in all piconets in which it is a member (it cannot be present in more than one piconet simultaneously). These presences of *bridge* are controlled by an inter-piconet scheduling algorithm.

In a piconet, transmissions can take place from master to slave (*downstream traffic*) or from slave to master (*upstream traffic*). In order to support full duplex transmissions, a Time Division Duplex (TDD), which divides each second into 1600 time slots, is adopted in the MAC layer located in the Baseband. The transmission of a Baseband packet usually covers a single slot but it may last up to five consecutive time slots.

Proceedings of the Advanced International Conference on Telecommunications and
International Conference on Internet and Web Applications and Services (AICT/ICIW 2006)
0-7695-2522-9/06 $20.00 © 2006 IEEE

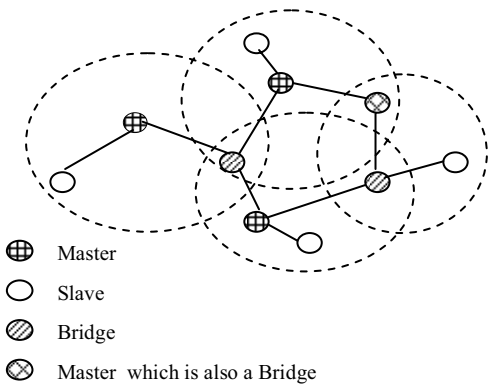

Figure 2. An example of a Bluetooth scatternet

The Bluetooth specification defines a few low power modes. Two of these modes, which are described below, can be used to enable inter-piconet communication:

Hold mode – A slave in this mode is inactive in the piconet for an agreed period. At the end of the period the slave becomes active and can be addressed by the master. The period is called *hold timeout* and its length is negotiated between the master and the slave.

Sniff mode – A slave in this mode is inactive in the piconet for agreed intervals (*sniff interval*).

When a bridge is inactive in a piconet it can be active in a neighboring piconet, and therefore, the hold and sniff modes can be used for inter-piconet communication. The main difference between the two modes is that the duration of the hold period is set every time the slave is placed in hold mode, whereas the parameters of the sniff mode are set once and can be used for many intervals. Thus, hold mode requires repeated negotiations that waste at least a slot pair while sniff mode requires a single negotiation.

2. Intra-piconet scheduling

Intra-piconet scheduling is obviously the main determinant of performance of Bluetooth piconets and one of the main determinants of performance of Bluetooth scatternets. As usual, the main performance indicator is the end-to-end packet delay, with lower delays being considered as better performance. There are, however, at least two other requirements to satisfy. First, the piconet master should try to maintain fairness among the slaves, so that all slaves in the piconet receive equal attention in some shorter or longer time frame (of course, their traffic load should be taken into account). Second, Bluetooth devices are by default low power devices, and the polling scheme should be sufficiently simple in terms of computational and memory requirements.

There are several intra-piconet scheduling algorithms have been proposed in the past (e.g. [15], [16]). In our paper, we have used SFPQ intra-piconet scheduling algorithm (for details see [6]).

FPQ is a fair and efficient polling algorithm with QoS support for Bluetooth piconet (for details see [5]). And SFPQ is soft-FPQ. That means, FPQ is the basis of SFPQ algorithm. Besides the indexes defined by FPQ, the SFPQ algorithm also estimates the satisfaction index of any data flow in the network. *Estimated satisfaction index*, $\hat{\xi}_i$ and *target satisfaction index*, ξ_i are used to define the satisfaction margin, for the *i*-th flow:

$$\psi_i = \hat{\xi}_i - \xi_i \tag{1}$$

The priority of the generic slave *i* in the SFPQ algorithm is given by

$$pr_i = \beta \cdot (\alpha \cdot p_{data_i} + (1-\alpha) \cdot n_i) + (1-\beta) \cdot \overline{\psi_i} \tag{2}$$

where p_{data_i}, n_i, $\overline{\psi_i}$ represent *normalized instantaneous data probability*, *normalized number of slots* and *normalized satisfaction margin* for the *i*-th flow. According to [5] and [6], α and β are set to 0.8 and 0.7 respectively. The slave analyzer selects a slave on higher basis pr_i (for details see [6]).

3. Inter-piconet scheduling

In this section, we propose a fair and efficient algorithm for inter-Piconet scheduling for Bluetooth scatternets. The Bluetooth specification defines a few low power modes (e.g. hold mode, sniff mode; see section 1). The hold and sniff modes can be used for inter-piconet communication. Our proposed algorithm, an Efficient and Fair inter-Piconet Scheduling Algorithm abbreviated as EFSA, works on hold mode. Various proposals regarding inter-piconet scheduling [7,8,9,10,11,12] have been proposed which are based on the concepts of rendezvous point. In [13], Load Adaptive Algorithm for inter-piconet scheduling was described utilizing hold mode. In past a few scheduling algorithms (e.g. [7], [14]) utilize the sniff mode. In following of this section, we present our hold mode based inter-piconet scheduling algorithm EFSA. And then, its performance evaluation is presented.

Proceedings of the Advanced International Conference on Telecommunications and
International Conference on Internet and Web Applications and Services (AICT/ICIW 2006)
0-7695-2522-9/06 $20.00 © 2006 IEEE

3.1. Principles and descriptions

The proposed algorithm EFSA has a similarity with FPQ intra-piconet scheduling algorithm (for details see [5]). But in this algorithm we consider two terms:

1) Instantaneous probability of having data in a piconet queue of bridge.
2) A Number of Slots since Last Poll for upstream traffic.

The first term contributes in scheduling efficiency. That means, it works for load adaptation. And the second term gives a support for fairness. In algorithm Q_i denotes the queue of piconet *i*, which remains in the bridge. Notation for the above two terms:

1) An instantaneous probability of having data in Q_i (P'_{data_i}).
2) A Number of Slots since Last Poll (*NSLP*) for upstream traffic (N'_i).

where $0 \le P'_{data_i} \le 1$ and $N'_i \ge 0$. In order to obtain the following normalized parameters:

$$p'_{data_i} = \frac{P'_{data_i}}{\sum_j P'_{data_j}} \quad (3)$$

$$n'_i = \frac{N'_i}{\sum_j N'_j} \quad (4)$$

where $0 \le p'_{data_i} \le 1$ and $0 \le n'_i \le 1$. Here efficiency is represented by p'_{data_i}, where as fairness is represented by n'_i. In order to control the tradeoff between efficiency and fairness, we define a control variable, γ. Then the equation for switching piconet is:

$$SwP_i = \gamma \cdot p'_{data_i} + (1-\gamma) \cdot n'_i \quad (5)$$

The bridge checks the SwP_i value of each piconet *i* and switches to the highest SwP_i value piconet. After various simulations, we achieve a value for γ which gives a good compromise between fairness and efficiency and the value is 0.7.

3.2. Computation of P'_{data_i} and N'_i

3.2.1. P'_{data_i} Calculation. The algorithm assumes that the arrival times of AP packets from the application follow a poison process. Looking at the states of Q_i, we may determine number slots in each *i*-th queue of bridge. But this approach does not differentiate packets, which correspond to the first segment of an AP packet from the other packets. This is not able to provide any priority to transmissions of AP packets, which are in progress.

We define $HoLP_i$ the Head-of-Line packet of Q_i (i.e. the first packet of Q_i) and *FSA*, a packet with the first segment of an AP packet (i.e. first Baseband packet of the transmission of an AP packet). We distinguish three possibilities, depending on the state of Q_i :

1) Q_i is empty.
2) The $HoLP_i$ is not a *FSA* **AND** number slots in Q_i $\ge$ *max queue length*.
3) The $HoLP_i$ is not a *FSA* **AND** number slots in Q_i $\le$ *max queue length.*
4) The $HoLP_i$ is a *FSA*.

We set P'_{data_i} to 0 in the first case. In second and third case, we consider queue length of respective piconet. We have to mention that in simulation we consider infinite queue. But there max queue length is a particular length of that queue. In these two cases, we set P'_{data_i} to 1 and 0.8 respectively. In case four, we only consider $HoLP_i$ and its P'_{data_i} is 0.5.

3.2.2. N'_i Calculation. N'_i is denoted as a fairness parameter. In this algorithm, it is used to estimate the amount of time between each opportunity the bridge has to begin the transmission of an AP packet with *i*-th piconet. Thus, bridge tries to distribute its time among the piconet. While a bridge is in transmission mode in *i*-th piconet, we set N'_i to 0. Otherwise, it will be incremented at each slot.

3.3. Performance evaluation

The performance of our proposed inter-piconet scheduling algorithm, EFSA and that of Load Adaptive Algorithm (e.g. [13]) have been evaluated via simulation. In this section, we present simulation results obtained for the scatternet illustrated in Figure 3. The scatternet consists of two piconets connected by a Bridge. At each node, packets are generated according to Poisson arrival process. Arrival rate in ***A*** piconet is more than that of in ***B*** piconet.

Simulation results are presented in Figure 4 and Figure 5. EFSA gives a better compromise between efficiency and fairness of scatternets. But in Load Adaptive Algorithm performance of fairness is very low. In terms of efficiency, Load Adaptive Algorithm has performed better than our proposed algorithm, EFSA. But fairness is also a vital part for scatternets. In that case, our proposed algorithm outperforms Load Adaptive Scheduling Algorithm.

Load Adaptive Algorithm only works for a smaller scatternet, which is formed with two piconets. But our algorithm works for larger piconet also. In Figure 6,

we present bridge utilization in a scatternet formed with three piconets. In simulation, we maintain a packet-arrival rate order among three piconets: *A*>*B*>*C*. In the scatternet formed with three piconets, data rate in *A* piconet and that in *C* piconet are highest and lowest respectively.

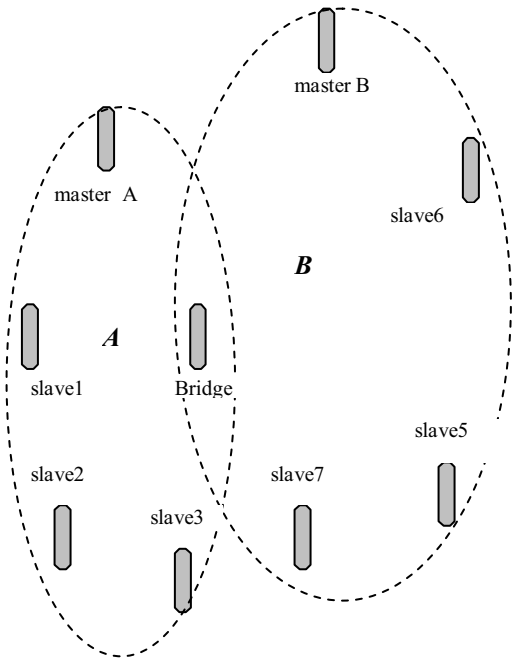

Figure 3. Scatternet consisting of 2 piconets

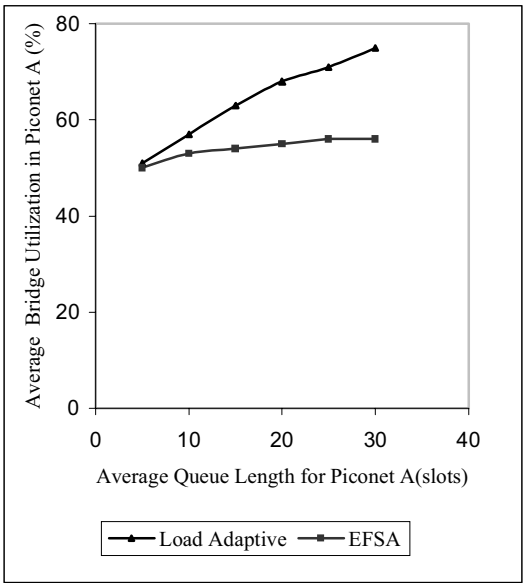

Figure 4. Performance of EFSA and Load Adaptive Algorithm in terms of piconet *A*

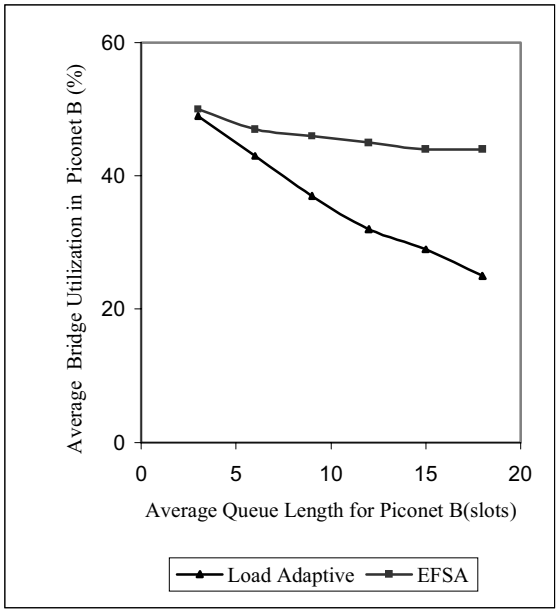

Figure 5. Performance of EFSA and Load Adaptive Algorithm in terms of piconet *B*

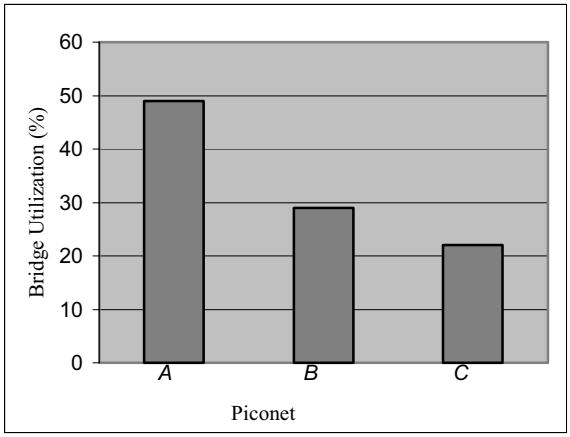

Figure 6. Bridge utilization in a scatternet formed with three piconets

4. Conclusion

In this paper, we use the SFPQ algorithm for intra-piconet purpose. We propose an inter- piconet algorithm for Bluetooth scatternets. From performance evaluation, this algorithm maintains a good compromise between efficiency and fairness of scatternets. These characteristics of our algorithm make its position in higher, in order to remain fair and efficient scatternets. This algorithm is also applicable for larger piconets.

Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW 2006)
0-7695-2522-9/06 $20.00 © 2006 **IEEE**

The computing complexity of four parameters of SFPQ algorithm is more. Our future interest is to reduce the computing complexity of the parameters.

5. References

[1] Wireless PAN medium access control MAC and physical layer PHY specification, IEEE standard 802.15, *IEEE*, New York, NY, 2002.

[2] Bluetooth SIG. Specification of the Bluetooth System, *Version 1.1*, Feb. 2001.

[3] J. Bray and C. Sturman, *Bluetooth 1.1 connect without cables*, Prentice Hall, 2001.

[4] J.B. Lapeyrie and T. Turletti, "Adding QoS support for Bluetooth piconet", *INRIA*, *Tech. Rep.* 4514, July 2002.

[5] J.B. Lapeyrie and T. Turletti, "FPQ: a fair and efficient polling algorithm with QoS support for Bluetooth piconet", *Proc. INFOCOM*, San Francisco, US, 2003.

[6] Andrea Zanella, Daniele Miorandi, Silvano Pupolin, and Christian Andreola, "A Soft-QoS Scheduling Algorithm for Bluetooth Piconets", *Proc. WPMC'03*, October 19-22 2003, Yokosuka, Kanagawa, Japan.

[7] S. Baatz, M. Frank, C. Kuhl, P. Martini, and C. Scholz. "Bluetooth scatternets: An enhanced adaptive scheduling scheme", *Proc. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies IEEE INFOCOM 2002,* New York, June 2002, pp. 782–790.

[8] N. Johansson, F. Alriksson, and U. Jonsson, "JUMP mode – a dynamic window-based scheduling framework for Bluetooth scatternets", *Proc. 2001 ACM International Symposium on Mobile ad hoc networking & computing,* Long Beach, CA, Oct. 2001, pp. 204–211.

[9] N. Johansson, U. Korner, and L. Tassiulas, "A distributed scheduling algorithm for a Bluetooth scatternet", *Proc. of the International Teletraffic Congress – ITC-17*, Salvador de Bahia, Brazil, Sept. 2001, pp. 61–72.

[10] A. Racz, G. Miklos, F. Kubinszky, and A. Valko, "A pseudo random coordinated scheduling algorithm for Bluetooth scatternets", *Proc. 2001 ACM International Symposium on Mobile ad hoc networking & computing,* Long Beach, CA, Oct. 2001, pp. 193–203.

[11] G. Tan and J. Guttag, "A locally coordinated scatternet scheduling algorithm", *Proc. of the 26th Annual Conference on Local Computer Networks LCN 2002,*Tampa, FL, Nov. 2002, pp. 293–303.

[12] W. Zhang and G. Cao, "A Fexible scatternet-wide scheduling algorithm for Bluetooth networks", *Proc. 21st IEEE International Performance, Computing, and Communications Conference IPCCC 2002,* Phoenix, AZ, Apr. 2002.

[13] Liron Har-Shai, Ronen Kofman, Gil Zussman, and Adrian Segall, "Inter-Piconet Scheduling in Bluetooth Scatternets", *Proc. of the OPNETWORK 2002 Conference*, Aug. 2002.

[14] J P. Johansson, R. Kapoor, M. Kazantzidis, and M. Gerla, "Rendezvous Scheduling in Bluetooth Scatternets", *Proc. IEEE ICC'02*, Apr. 2002.

[15] A. Capone, M. Gerla, and R. Kapoor, "Efficient Polling Schemes for Bluetooth Picocells", *Proc. IEEE ICC'0*1, June 2001.

[16] A. Das, A. Ghose, A. Razdan, H. Saran, and R. Shorey, "Enhancing Performance of Asynchronous Data Traffic over the Bluetooth Wireless Ad-hoc Network", *Proc. IEEE INFOCOM'0*1, Apr. 2001.