

# Rendezvous Scheduling in Bluetooth Scatternets

P. Johansson\*, R. Kapoor\*\*, M. Kazantzidis\*\*, M. Gerla\*\*  
Ericsson Corporate Research\*      Computer Science Department, UCLA\*\*  
(Per.Johansson@ericsson.com)      (rohitk, kazantz, gerla@cs.ucla.edu)

## Abstract

Bluetooth scatternets are formed by interconnecting Bluetooth piconets in an ad-hoc fashion and consists typically of the handheld electronic devices from one or more user(s). Thus, scatternets may be used to form and interconnect ad-hoc Bluetooth personal area networks (PANs). The inter-piconet Bluetooth units, i.e. the gateways interconnecting the piconets in a scatternet, need to time division multiplex their presence in each of their piconets. This requires an inter-piconet scheduling (IPS) algorithm, operating in each inter-piconet unit in order to efficiently coordinate its presence with other Bluetooth units. In this paper an IPS algorithm based on periodic rendezvous points is proposed and analyzed with simulations. The algorithm is called Maximum Distance Rendezvous Point (MDRP) and utilizes the Bluetooth SNIFF functionality to establish the periodic rendezvous points between gateways and their peer nodes. The Bluetooth simulator used in the study is based on NS-2, augmented with a scatternet architecture and a Bluetooth MAC layer model. The simulations obtain results regarding TCP behavior and voice delays. The results show that TCP works well with MDRP, but the large round trip delays caused by the inter-piconet gateway nodes give a slowly growing flow control window for TCP. The latter will in particular have impact on the TCP performance in scatternets hosting "thin" (embedded) clients with limited dynamic memory capacity.

## 1. Introduction

The Bluetooth wireless technology [1] is a low power, low cost, short range RF technology that focuses on replacement of cables between electronic devices. It uses a fast frequency-hopping scheme and operates in the unlicensed Industrial Scientific-Medical (ISM) band at 2.4 GHz.

Even though pure cable replacement with point-to-point radio links is perhaps the most obvious use for Bluetooth, the capability to form Bluetooth networks opens up a whole new arena for applications. Networks of small handheld electronic devices will enable an environment where information may be accessed, exchanged and shared seamlessly among the devices in the network. Typically, such a network could consist of a cellular phone, a PDA, a notebook PC, an mp3 player, a DVD player etc., all of which are devices that a person carries around in his every day life both for work and pleasure [2]. Often, this kind of network is referred to as a personal area network (PAN). However, a PAN may, from time to time, also include devices that are not carried along with the user, e.g., an access point for Internet access or sensors located in a room. Moreover, devices from other users' PANs may also be interconnected to enable sharing of information, which could be anything from business cards to multiplayer game interaction.

The Bluetooth network architecture is based on the piconet, a star topology, which is defined by two or more Bluetooth units that share the same frequency hopping channel. The central unit in a piconet (the master) may control up to seven units (slaves). Any Bluetooth unit is capable of becoming the master of a piconet.

The nature of PANs in general implies a minimum of preconfiguration, i.e. it should be possible to establish the network in an ad-hoc fashion with minimum user intervention. In this context, Bluetooth has been tailored to provide such functionality through its inherent ad-hoc discovery and connectivity capabilities. Also, it is likely that most PAN devices will be battery-driven, which makes efficient use of power an important issue. The Bluetooth piconet architecture enables the controlling master unit to apply a strict medium access control through the use of a polling scheme. This gives a better control over power consumption and bandwidth usage compared to a random access scheme since slave units transmit only when scheduled by the master unit. Hence, the Bluetooth wireless technology provides functions that make it a natural choice for PAN connectivity.

In order to further enhance the networking capabilities of Bluetooth, piconets may be interconnected into scatternets, which requires some units to be present in two or more piconets. However, since a Bluetooth unit is expected to host only one radio transceiver, this presence needs to be handled in a time division manner, i.e. the inter-piconet unit will switch between the piconets. As the unit switches between the piconets, it can act as slave in one or more, but as master in at most one piconet. The inter-piconet units may also forward traffic between the piconets, i.e. operate as gateways between piconets. Since the inter-piconet unit cannot receive information from

more than one piconet at a time, the need to co-ordinate the presence of masters and inter-piconet devices in each piconet is necessary to achieve a controlled performance. The work in this paper focuses on describing the inter-piconet scheduling issue in a scatternet. In addition, the performance of a proposed scheduling scheme is analyzed through simulation experiments. Results are presented for a mix of TCP and voice traffic traversing across different scatternet scenarios.

Most of the published work on Bluetooth so far has focused on the single piconet and issues related to scheduling of units within one piconet, e.g., [3] [6]. Some work has also looked at the task of actual formation of Bluetooth piconets and scatternets, [4] [5] [7]. In a scatternet resulting from any formation algorithm, the inter-piconet units still need to be scheduled in an efficient way. Thus, the inter-piconet scheduler (IPS) is crucial for the operation of scatternets. Very little work on inter-piconet scheduling issues in scatternets can be found in the literature. In [8] an inter-piconet scheduler is analyzed for a case where central control of the entire scatternet is assumed, which is costly to achieve. The work herein will focus on analyzing inter-piconet schedulers that operate in a distributed manner in scatternets and thus eliminate the need for scatternet-wide coordination.

The work presented in this paper is organized as follows: Section 2 describes scatternet usage scenarios for Bluetooth PANs. Section 3 gives an overview of issues related to scatternet scheduling. Section 4 describes the proposed IPS algorithm MDRP. Section 5 explains the simulation model and Section 6 provides simulation results.

## 2. A Scatternet PAN Scenario

A Bluetooth unit present in multiple piconets may either operate independently in the piconets, or function as a gateway between the piconets, forwarding data between them. When the inter-piconet units forward packet between piconets, the Bluetooth ad-hoc PANs pertain to the class of multihop ad hoc networks. These scatternet based PANs may be used when information needs to be spread among PANs through a “reasonable” number of radio hops. This number of hops depends heavily on the scatternet application. When the scatternet is used to interconnect, for instance, a large network of Bluetooth equipped sensors, the flow of information will have low data rates and may sustain high delays and could tolerate a large hop count. On the other hand, a high quality interactive video application within a PAN may not be able to handle the delay imposed by more than one Bluetooth inter-piconet hop.

Figure 1 shows a scenario in which a GPRS phone provides Internet access to all the notebook computers in the scatternet. The notebook computer in PAN 1 operates as an inter-piconet gateway, forwarding packets between PAN 1 and the notebook computers of the other piconet.

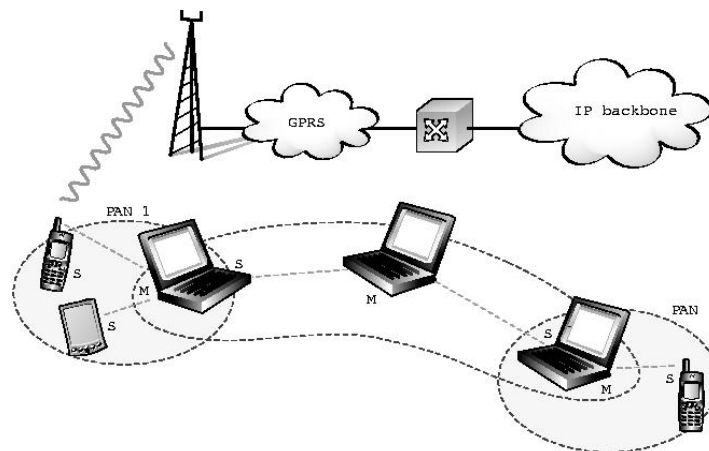


Illustration: Claes-Göran Andersson

**Figure 1. The master of PAN 1 forwards packets from the GPRS phone to the rest of the notebook computers, providing internet access to the entire scatternet.**

The scatternet functionality may also be used to improve the performance of a group of Bluetooth units that are either already part of a scatternet, or part of separate piconets. The roles of the units in such a group may be rearranged to adapt to a new traffic distribution among the devices by changing the allocation of masters, slaves and inter-piconet units. For instance, if two slave units need to communicate, it might be wiser to create a new

piconet that contains only these two units. Since the frequency hopping channel combined with fast ARQ makes Bluetooth very robust against interference, new piconets gain substantially more capacity than they lose due to increased interference between them.

### 3. Inter-piconet scheduling

The main issue with inter-piconet scheduling (IPS) is to coordinate the simultaneous presence of inter-piconet units in a scatternet. In a piconet, the master unit always expects a slave unit to be present when it sends a data or poll packet to it. If the slave unit is not present, the master may choose to disconnect the slave after some predefined time-out period since it may be seen as a case of an erroneous radio link. However, an inter-piconet slave will be visiting another piconets and is not simultaneously present at its masters as part of its normal operation. Thus, this intermittent presence of an inter-piconet node must be considered by all its masters as part of an inter-piconet scheduling algorithm.

Moreover, the time-slots of the piconet channels are, in general, not synchronized between different piconets, which introduces a delay of one or two slots (0.625 ms each) each time a unit switches between piconets. This results in a piconet switching overhead that should be taken into account in the design of an IPS algorithm.

In addition to the IPS algorithm, the master unit of each piconet will also host a scheduling function that controls the polling of the slaves within the piconet, i.e. an intra-piconet scheduling (IRPS) function. This combination of IPS and IRPS schedulers in the scatternet should be coordinated in order to give an efficient scheduling of the units of the scatternet. For instance, an inter-piconet node may be removed from the scheduling list of the intra-piconet scheduler for the period it is not present, but re-installed in the list for more frequent scheduling once it is back in the piconet. In Figure 2 a scatternet consisting of two piconets with one inter-piconet unit (double slave role) is depicted. The two time lines in the figure show how the time division multiplexing for the inter-piconet unit needs to take the phase shift into account when jumping between the two piconets. When the inter-piconet unit is active in one of the piconets, the IRPS algorithm of that piconet determines the amount of polling it will receive.

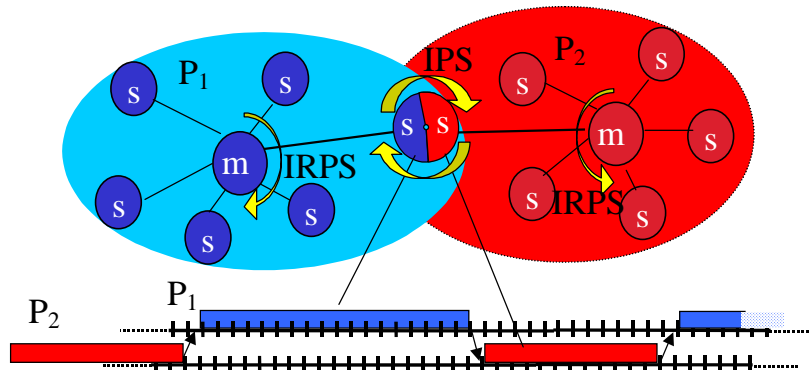


Figure 2. A scatternet with one inter-piconet unit that divides its time between the two piconets.

#### 3.1. Rendezvous point scheduling

One approach to implement the inter-piconet coordination is by introducing *rendezvous points*. A rendezvous point (RP) is a slot at which a master and an inter-piconet unit have decided to meet, i.e., at this slot, the master has agreed to address a packet to the inter-piconet unit and the inter-piconet unit has agreed to listen to the master.

Two major issues that need to be addressed by a Rendezvous Point inter-piconet algorithm are:

- The rendezvous point (RP): how do the master and slave decide on the RP and how strict is the commitment to this RP? In a strict algorithm, the master and slave units will always honor a RP. In fact, a wide variety of rules on honoring RPs may be defined. The distribution of RPs over time could be periodic, decided upon each visit, or spread out in a pseudo-random sequence (known by both nodes). The delay property of an inter-piconet algorithm will, to a great extent, depend on the time between mutually honored rendezvous points.

- The rendezvous window (RW): given that the master and slave units are both present at a RP, how much data will they be able to exchange? This depends upon the duration of the period that the units are mutually present and how much of this time is used to send data. In a strict algorithm, a time window, called a rendezvous window, could be defined in which both master and gateway must be present and exchange data in every available slot. A relaxed algorithm would not require any of the master or slave units to stay any pre-determined time. The achieved throughput for the master slave pair will depend on the duration of these rendezvous windows.

In the following an IPS algorithm, called Maximum Distance Rendezvous Point (MDRP) is studied. It is using committed RPs and the RW sizes are agreed between both master and slave.

## 4. The Maximum Distance Rendezvous Point IPS Algorithm

The basic idea behind the MDRP algorithm is that RPs should be as far from each other as possible, i.e., the distance between the RPs should be maximized. In the following analysis, the inter-piconet node is assumed to operate as a gateway between piconets and henceforth the term *gateway* is used to denote the inter-piconet unit. It is assumed that the inter-piconet node is never a master of a piconet, i.e., it can only be a slave in more than one piconet. The MDRP uses the notion of a periodic *superframe* that is common to all nodes involved in the IPS and denotes the time period between two RPs for any master-gateway pair. For a gateway, the distance between two RPs associated with different masters denotes the time that it spends in a piconet, i.e. the RW, before switching to the next. The MDRP algorithm strives to divide the superframe equally between the piconets that the gateway belongs to by maximizing the distance between each RP of a gateway.

Consider a master node that wants to assign a rendezvous point to a gateway. Suppose the gateway node belongs to  $i$  other piconets and has RPs  $r_1, r_2, \dots, r_i$  with these piconets. The master node itself has  $j$  other gateways to which it has assigned RPs  $r_{i+1}, r_{i+2}, \dots, r_{i+j}$ . These RPs are actually slot numbers in the superframe and repeated in every superframe period. The master considers the ordered list of all the  $i+j$  RPs already established and assigns a new RP that maximizes the distance with the previous ones. Thus, the RP is chosen as the middle slot of the largest interval between successive RPs. This gives a very simple and robust allocation of RPs and helps to create a stable IPS performance. However, this simplicity and robustness comes at a price: the algorithm cannot adapt to temporary traffic changes in the network since the RW allocation only depends on the number of piconets a gateway is associated with.

The MDRP algorithm makes use of the Bluetooth SNIFF mode to implement the superframe in the nodes involved in the IPS. The SNIFF mode is one of the power saving modes specified in Bluetooth that allows a slave unit to reduce the duty cycle at which it needs to listen to the master unit. Instead of listening to the master in every downlink slot, the SNIFF mode allows a slave unit to listen to the master only in every  $T_{sniff}$  slot, which forms a SNIFF period. Once the slave listens at one of the periodic SNIFF slots it needs to continue listening for at least  $N_{sniff\ attempt}$  slots. After every received packet from the master unit, it will keep listening for  $N_{sniff\ timeout}$  even if the master does not address any packets to it. However, it will keep listening as long as it keeps receiving packets from the master. Hence,  $N_{sniff\ timeout} > 0$  means that the slave will listen as long as the master keeps sending packets, while  $N_{sniff\ timeout} = 0$  makes the slave listen only for  $N_{sniff\ attempt}$  slots. Thus, the SNIFF mode relieves a slave unit from its duty to listen for the master unit in every slot and may be used to allow a gateway to switch between piconets. The SNIFF feature can be used to implement an IPS algorithm, since the superframe period can be mapped to the  $T_{sniff}$  parameter, the RPs to the starting points of a  $T_{sniff}$  between a gateway-master pair, and the RW to the  $N_{sniff\ attempt}$  parameter.

### 4.1. MDRP Example RP Allocations

Consider the topology shown in Figure 3 where the masters and gateways are shown. No RPs have been assigned yet and the superframe ( $T_{sniff}$ ) period is 120 slots. This superframe period will give approximately 1-2 percent overhead caused by the switching between piconets (on average two slots are lost at each piconet switch instance). Suppose the gateways are assigned RPs in the following order:

- 1) gateway1, master1 – Assigned RP = 0 (no other RPs are assigned yet, any RP could be chosen; choose 0).
- 2) gateway2, master1 – Assigned RP = 60 (most distant from earlier RP, 0).
- 3) gateway1, master2 – Assigned RP = 60 (most distant from earlier RP, 0).
- 4) gateway3, master1 – Assigned RP = 90. Actually, both 90 and 30 are slots that are at a maximum distance from 0 and 60. The tie is broken at random.
- 5) gateway 2, master 3 – Assigned RP = 0 (most distant from earlier RP, 60).

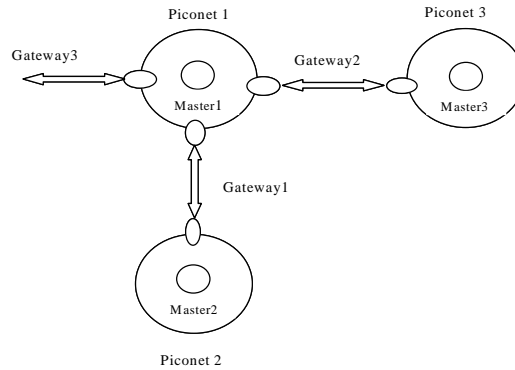


Figure 3. Topology showing gateways and masters in a Scatternet

## 4.2. IRPS and IPS interaction

The polling scheme in [3] has been shown to be efficient for intra-piconet scheduling (IRPS), but is further enhanced with the feature described below to make it interact with the inter-piconet communication present in a scatternet case.

Instant Attention to gateway: Since a gateway spends only a fraction of the time in a piconet, it is important that the gateway gets polled when it is a part of the piconet. Thus, the master should poll the gateway as soon as it enters its piconet. This may suspend the polling of another slave, which can be resumed when the gateway has been polled.

## 4.3. Support for Ad-Hoc Routing

Ad-hoc routing protocols, e.g. the Ad-hoc On-demand Distance Vector routing protocol (AODV) [10], rely on a broadcast functionality at the MAC layer to propagate the routing packets. In Bluetooth, packets can be broadcast inside a single piconet using the piconet broadcast address of 0. In order to support broadcast of routing packets to outside of a piconet, the notion of MAC layer broadcast in Bluetooth is defined as follows:

- i) If a non-gateway slave node needs to broadcast a packet, it sends the packet to the master of the piconet it belongs to.
- ii) If a master node needs to broadcast a packet, it broadcasts the packet to all the non-gateway slaves in its piconet using the piconet broadcast address of 0 and unicasts the packet to each of the gateway slaves when they visit the piconet.
- iii) A gateway slave node wanting to broadcast a packet sends it in a unicast manner to the master of each piconet it belongs to when it visits the piconet.

## 5. Simulation Model

A Bluetooth simulation model was developed as an extension to NS-2 (Network Simulator) [9]. The simulation model implements most of the features of the Bluetooth baseband layer like frequency hopping, time division duplexing, ARQ (automatic retransmission query), multi-slot packets, fragmentation and reassembly of packets. In addition, the simulator includes support for Bluetooth scatternets and inter-piconet communication. The simulator can define gateway nodes that can do forwarding between piconets, as described earlier.

Another important feature of the simulator is the channel model. The frequency hopping is modeled as a pure pseudo-random sequence. If two or more transmissions occur on the same frequency, the SIR (Signal-to-Interference Ratio) is evaluated using the gain factor  $g$  of each radio channel. The factor  $g$  is considered constant during the packet transmission and its value is obtained by considering the following factors:

- *path loss due to distance*:  $d^{-\eta}$ , where  $d$  is the distance and  $\eta$  ranges between 2 and 4.
- *shadowing*: log-normal random variable  $s=10^{0.1\epsilon}$ , where  $\epsilon$  is a Gaussian variable with  $\sigma$  standard deviation.
- *fading*: exponential random variable (Rayleigh fading) with mean = 1.

At the receiver  $i$  the SIR is evaluated as:

$$SIR = \frac{P_t \cdot g_{ii}}{P_n + \sum P_t \cdot g_{ji}}$$

where  $g_{ij}$  is the gain factor between transmitter  $j$  and receiver  $i$ ,  $P_t$  is the transmitted power and  $P_n$  is the noise power in the signal bandwidth.

The receiver model is based on the SIR value. In a segment, errors occur independently with the probability of error obtained from the SIR, taking into consideration the modulation adopted and the FEC coding (if adopted). The SIR is re-computed any time a collision occurs, while it is considered constant for a colliding period. Note that, since the simulator assumes that piconets are synchronized in time, each collision persists for an integral number of Bluetooth slots.

## 6. Simulations

This section presents scatternet experiments using the MRDP algorithm. The simulated scatternet topologies carries a mix of TCP and voice traffic. In the experiments, the link layer queue size is 30 kbytes, unless otherwise mentioned. The packet size for TCP is 1000 bytes. Voice connections are modeled according to the Brady model [11]. In particular, the voice connections are "on-off" sources, with "on" time equal to 1 s and "off" time 1.35 s, and "on-off" times exponentially distributed. The voice-coding rate is 8 kbit/s and the packetization period is 20 ms, which gives a payload size of 20 bytes. Header compression is assumed for voice packets in Bluetooth and the total packet size is 30 bytes. Voice packets are sent using RTP over UDP. The routing protocol used is AODV [10] and the superframe ( $T_{sniff}$ ) was set to 100 slots.

### 6.1. Simple Scatternet Topology

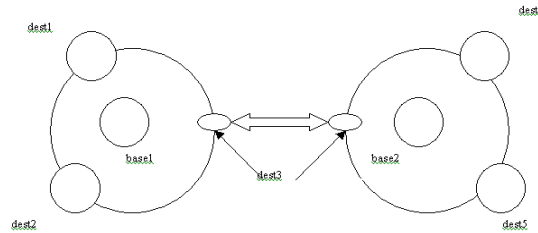


Figure 4. Scatternet consisting of 2 piconets

In the first set of experiments, a scatternet consisting of two piconets is considered, connected by a gateway as shown in Figure 4 (base1 and base2 are the masters, dest1, dest2, dest4 and dest5 are the non-gateway slaves and dest3 is a gateway slave which is a slave in both piconets).

#### 6.1.1. Single TCP – Congestion Window Behavior

First, a single TCP connection was run between dest1 and dest4 in order to study the congestion window behavior of a TCP connection going through a gateway. The simulation was run for 35 seconds and Figure 5 shows the congestion window of the TCP connection versus time. One important observation regarding the window behavior is that the round-trip-delay for the TCP connection is of the order of 100 msec, which causes a slowly growing window and has implications in the choice of the buffer size. The latter is further explored in 6.1.3.

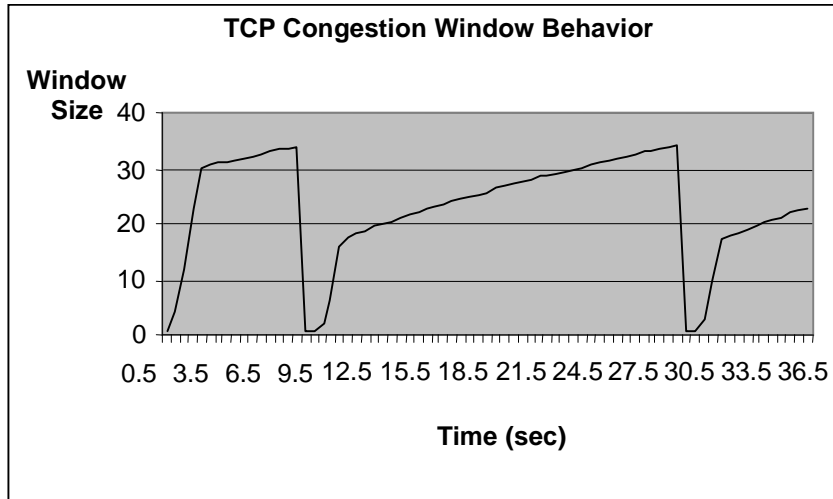


Figure 5. Congestion Window Behavior of TCP

### 6.1.2. Two TCP connections

Secondly, two TCP connections were run. The first was run from dest1 to dest4 and the second from dest2 to dest5. The two TCP connections start at the same time and run until the end of the simulation. The simulation was run for 50 seconds. Both these connections go through the gateway dest3 and thus, share the gateway bandwidth. The motivation behind this experiment was to see how one TCP connection affected another going through the same gateway. Figure 6 shows the data transferred for the two TCP connections versus time. It can be seen that the two TCP connections share the gateway in an even manner.

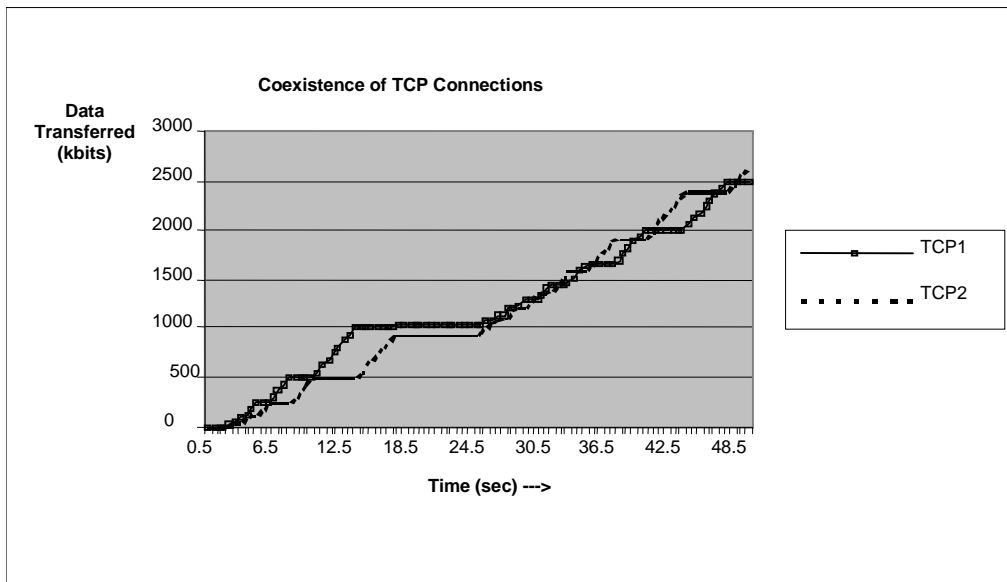


Figure 6. Sharing of gateway by 2 TCP connections

### 6.1.3. Effect of Buffer Size on TCP

As discussed earlier, the switching between piconets in the gateway node causes a relatively long end-to-end delay between nodes in separate piconets. Such long delays may have an adverse effect on the performance of TCP unless the buffers along the end-to-end path are large enough to allow TCP to utilize the available link rate. Hence, the required buffer capacity in a gateway node will depend on the superframe size selected for the system. In order to study the relation between superframe and buffer capacity, the TCP performance (one

connection) against the gateway buffer size was studied for three different superframe sizes (60, 90 and 120). The roundtrip delays for the TCP flows were approximately 75 ms, 112 ms, and 150 ms for superframe sizes 60 ms, 90 ms, and 120 ms respectively. This gives a bandwidth-delay product (BW\*RTT) of 27 kbits, 41 kbits, and 54 kbits respectively for the three cases. Note that the available bandwidth (BW) is  $720/2=360$  kbit/s since the gateway node is time sharing its capacity between the two piconets. As can be seen calculated values agree fairly well with the results obtained in the simulations presented in Figure 7.

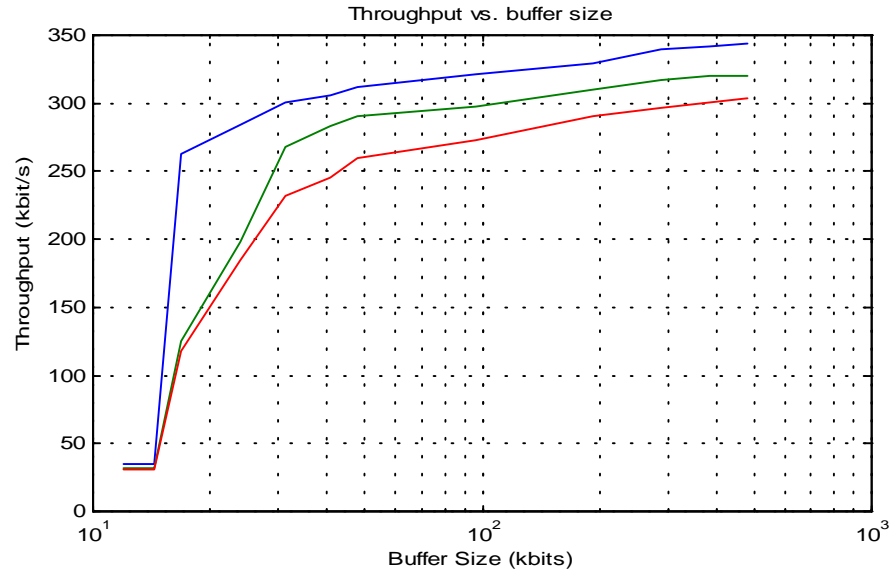
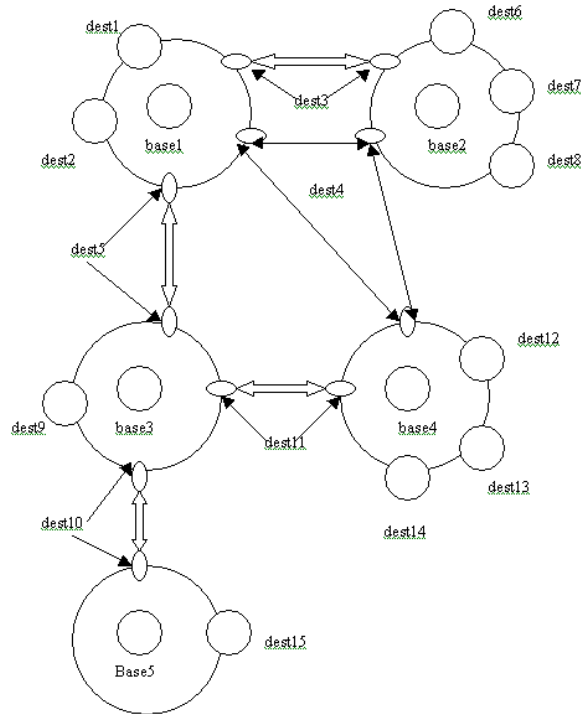


Figure 7. Effect of Buffer Size on TCP Throughput (superframe=60 (upper), 90 (middle), and 120 (lower)).

A clearly defined improvement of the TCP performance is attained if the buffers are large enough to accommodate the bandwidth-delay product, which is a well known result regarding TCP performance in relation to, e.g., satellite links. Nevertheless, the same property becomes a design issue for “thin” low price Bluetooth clients, where even a few kbytes of buffer space may be difficult to achieve. In these devices the dynamic memory area is often scarce and needs to be shared by all other programs running on the processor of the device.

## 6.2. Scatternet Consisting of 5 Piconets

In the second experiment, a bigger scatternet is considered, consisting of 5 piconets as shown in Figure 8. The base(1-5) nodes are the masters and the dest(1-15) nodes are the slaves. The dest3, dest4, dest5, dest10 and dest11 are the gateway nodes. Also, dest4 is a gateway belonging to the 3 piconets whose masters are base1, base2 and base4. A combination of TCP and voice traffic was run and the TCP throughput and voice delays were monitored. Each experiment lasts 30 seconds of simulation time. The source and destination nodes for each connection (TCP or voice) are chosen randomly. All the connections start at the same time (0.5 sec) and run till the end of the simulation.



**Figure 8. Scatternet consisting of 5 piconets.**

In this experiment, two TCP and two voice connections were run along the following paths:

TCP 1: base1 – dest3 (gateway) – base2 – dest6

TCP 2 : dest9 – base3 – dest5 (gateway) – base1 – dest1

Voice 1: dest2 – base1 – dest5 (gateway) – base3 – dest10 (gateway) – base5

Voice 2: base2 – dest3 (gateway) – base1 – dest2

The source destination pairs were randomly chosen and the paths determined by the AODV routing protocol.

Some of the experiments suggested that voice packets suffered extremely large delays (of the order of 2000 msec), when voice and TCP connections share the same gateway. In order to provide better support for voice in a scatternet, a head-of-line priority for voice packets was introduced in the gateway queues, i.e. voice packets were always placed in front of TCP packets in the queues. This is a very simple method for providing priorities for real-time traffic like voice, which would otherwise suffer huge delays in the presence of TCP connections.

The throughputs achieved by the TCP connections are 309.665 kbit/sec and 242.277 kbit/sec. Note that the maximum throughput achievable for a gateway that is a slave in two piconets is  $721/2$  kbit/sec (= 360 kbit/sec), if it spends its time equally in the two piconets. In this simulation, the paths traversed by the two TCP connections both contain such a gateway (dest3 and dest5). This explains the throughputs achieved by the TCP connections. The lower throughput of TCP 2 is explained by the fact that dest5 (which is the gateway in TCP2's path) is a slave in a piconet which contains 3 gateways; its share of the piconet bandwidth is, thus, reduced. Also, TCP1 achieves a throughput slightly less than the 360 kbit/sec ideal throughput. This is due to the high round-trip-delay of TCP (Section 6.1.1), which results in a slow initial increase of the TCP window.

The complementary cumulative delay distribution of the “prioritized” voice packets for both the voice connections is shown in Figure 9. It can be seen that about 95% of the voice packets have a delay of less than 300 msec, which is quite acceptable. In addition, very few (around 0.5%) voice packets are dropped due to queue overflow.

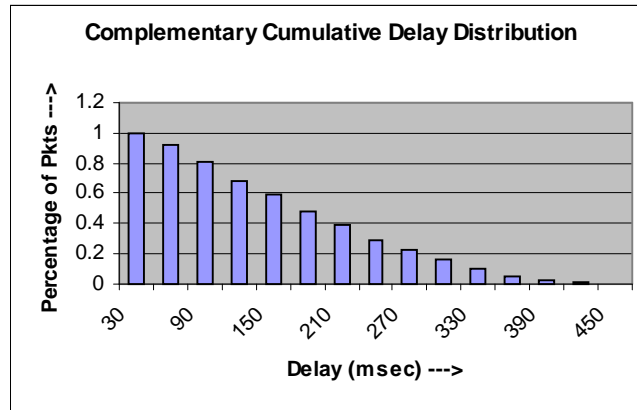


Figure 9. Complementary Cumulative Delay Distribution for the two voice connections

## 7. Conclusions

The introduction of Bluetooth scatternets enables a wider and more flexible use of Bluetooth network as means to create local and short range connectivity. However, the scatternet also presents a technical challenge with respect to network formation and scheduling of traffic within a Bluetooth network. In this paper the focus was set on the scheduling issues in general and on the inter-piconet scheduling (IPS) in particular. An overall description of the issues related to scheduling in scatternets was given. For instance, the intra-piconet scheduler (IRPS) and the IPS need to be able to interact in order to provide an efficient scatternet scheduler. A periodic rendezvous IPS algorithm was proposed, called the Maximum Distance Rendezvous Point algorithm (MDRP). MDRP uses a periodic superframe in which the rendezvous points are distributed according to a maximum spread approach. The MDRP algorithm may utilize the Bluetooth SNIFF functionality to set-up and maintain the periodic rendezvous points between the gateways and their peer nodes. MDRP was analyzed by means of simulations based on NS-2, enhanced with a Bluetooth scatternet model. The simulations covered experiments on an ad-hoc network of Bluetooth nodes using AODV as the ad-hoc routing protocol.

The simulations presented herein showed that the TCP behavior is affected by the large round trip delay imposed by the inter-piconet gateways, causing TCP to grow its window slowly. Moreover, the simulations showed that the delays for voice packets, in presence of other TCP connections, are acceptable if head-of-line priority is applied to the voice packets in the gateway nodes. The TCP simulations also confirmed a well known relation between the bandwidth-delay product and the available buffer capacity in the bottleneck nodes, which in the inter-piconet case is the gateway nodes. Hence, the choice of time granularity for the inter-piconet time division multiplexing may heavily depend on the buffering capacity in the Bluetooth devices. A fact that will affect the ability for “thin” clients to efficiently operate as gateway nodes in a scatternet.

## References

- [1] Specification of the Bluetooth System - Core vol.1 v1.1, [www.bluetooth.com](http://www.bluetooth.com)
- [2] J.C. Haartsen, “Bluetooth: a New Radio Interface Providing Ubiquitous Connectivity”, IEEE VTC 2000 – Spring, pp. 107-111.
- [3] A. Capone, R. Kapoor and M. Gerla: “Efficient Polling Schemes for Bluetooth Picocells”, ICC 2001.
- [4] T. Salonidis, P. Bhagwat, L. Tassiulas, R. LaMaire, “Distributed Topology Construction of Bluetooth Personal Area Networks”, In the proceedings of IEEE INFOCOM 2001, Anchorage, Alaska, USA, April 22-26, 2001.
- [5] S. Basagni, I. Chlamtac, G. V. Záruba, “Bluetrees – Scatternet Formation and Routing in Bluetooth-Based Ad Hoc Networks”, In the proceedings of IEEE INFOCOM 2001, Anchorage, Alaska, USA, April 22-26, 2001.
- [6] A. Das, A. Ghose, A. Razdan, H. Saran, R. Shorey, “Enhancing Performance of Asynchronous Data Traffic over the Bluetooth Wireless Ad-Hoc Network”, In the proceedings of IEEE INFOCOM 2001, Anchorage, Alaska, USA, April 22-26, 2001.
- [7] G. Miklós, A. Rác, Z. Turányi, A. Valkó, P. Johansson, “Performance Aspects of Bluetooth Scatternet Formation”, Poster session Mobihoc 2000, Boston, MA, August 11, 2000.

- [8] N. Johansson, U. Körner, L. Tassiulas, "A Distributed Scheduling Algorithm for a Bluetooth Scatternet," to appear at the Seventeenth International Teletraffic Congress, ITC'17, Salvador da Bahia, Brazil, September 24-28, 2001.
- [9] "Network Simulator (NS-2)", [www-mash.cs.berkeley.edu/ns/](http://www-mash.cs.berkeley.edu/ns/)
- [10] C. Perkins, E. Royer, "Ad-hoc on-demand distance vector routing", Mobile Computing Systems and Applications, 1999. Proceedings. WMCSA ' 99. Second IEEE Workshop on , 1999 , Page(s): 90 –100.
- [11] P.T. Brady, "A model for generating on-off speech patterns in two-way conversation", Bell System Technical Journal, Sept. 1969, pp. 2445-2471.