Bluetooth Scatternet Formation and Scheduling: An Integrated Solution

Stefano Basagni *Senior Member, IEEE*, Maurizio A. Nanni, Chiara Petrioli *Senior Member, IEEE*

Abstract— **Building and deploying *multi-hop* networks of Bluetooth devices (aka *scatternets*) concerns devising methods for forming *piconets*, connecting them through shared *gateways*, and scheduling the presence of these gateways among the piconets they interconnect (*inter-piconet scheduling*). There are several types of gateways, and their efficient scheduling is affected by the gateway type. Scatternet formation and scheduling have been dealt with separately in the past. This leads to network performance degradation because of the missed opportunity of designing scatternet formation protocols that best address scheduling requirements and vice-versa. In this paper we propose SS-Blue, an integrated mechanism for the joint design of scatternet formation and scheduling. Specifically, we enhance an efficient scatternet formation protocol by adopting methods for piconets interconnection that favor types of gateways which will result in a better performing inter-piconet scheduling. At the same time, a fair and traffic adaptive mechanism is proposed to schedule all types of gateways (inter-piconet scheduling), and for managing *intra-piconet scheduling*, i.e., for scheduling traffic transmission within a piconet. Our solution is evaluated through extensive simulations. Our results show that SS-Blue succeeds in producing scatternets whose gateways are efficiently scheduled. The combination of the proposed intra- and inter-piconet scheduling is shown to be remarkably effective in favoring packet forwarding with very low end-to-end latency.**

I. Introduction

The Bluetooth (BT) technology [1] has established itself as one of the most commercially viable for interconnecting devices without cables. In order to offer widespread BT access to these multiple services the need arises for an efficient, integrated solution for BT devices self-organization and data transmission. Defining this integrated solution is the purpose of this paper.

According to the BT specifications, a Bluetooth device is elected to be the *master* of the communication. The master can communicate with at most 7 active *slaves* at the same time. This basic communication structure is called a *piconet*. Some devices (called *gateways*) can time share among different piconets, allowing the possibility of forming multi-hop networks, called *scatternets*. While piconet formation is well described by the BT specifications, no details are given about forming scatternets. This task has been recently tackled with by many researchers [2]–[6]. Once the scatternet has been formed, equally important tasks concern deciding how to schedule traffic within a piconet (*intra-piconet scheduling*), and deciding when and for how long the gateways are active and in which piconet (*inter-piconet scheduling*).

Traditionally, scatternet formation and scheduling have been dealt with separately. Solutions for gateway scheduling have been proposed that mostly deal with scatternets where piconets are interconnected only by common slaves (*slave gateways*). This assumption clashes with current scatternet formation protocols that generate scatternets whose piconets are joined also by *intermediate gateways*. These are (slave) devices that in order to interconnect two piconets form an *extra* piconet in which one is the slave and the other becomes the master. It is quite natural to have scatternets with both slave and intermediate gateways, and most of the scatternet formation protocols have both type of gateways. Furthermore, the design of protocols for scatternet formation has been driven by desirable scatternet features such as 1) generating connected scatternets whenever the network topology is connected; 2) limiting the piconet size to ≤ 8 devices (thus avoiding the time consuming procedures for *parking* and *unparking* the slaves in excess to 7), and 3) generating meshed scatternets with short routes. No attention has been given to the identification of those scatternet properties that would favor efficient intra-piconet and inter-piconet scheduling. The only identified desirable feature which is related to scheduling, i.e., limiting the number of piconets to which a gateway belongs, has been shown in this work to only partially capture what is really critical for scheduling performance.

In this paper we present a solution for building multi-hop networks of BT devices that takes into consideration the efficiency requirements of inter-piconet scheduling while designing a scatternet formation protocol. Our integrated solution, which we term Scatternet-formation and Scheduling for Bluetooth networks, or *SS-Blue* for short, comprises the following contributions. We define a fair and traffic adaptive scheduling solution that works with intermediate gateways. This scheme, inspired by the *adaptive credit scheme* presented in [7], is also applied to intra-piconet scheduling instead of the usual round robin scheme. To the best of our knowledge, this is the first proposal for scheduling gateways that are not slave gateways as well as for using the credit scheme scheduling within a piconet. Our investigation leads to the clear understanding that managing intermediate gateways introduces non-negligible overhead and significant data latency. Therefore, we set to investigate ways of forming connected scatternets with a small number of extra piconets: We consider a protocol for scatternet formation, *BluePleiades* [8], and enhance it with an efficient mechanism for piconet interconnection that remarkably reduces the number of intermediate gateways.

S. Basagni is with the ECE Department at Northeastern University, Boston, MA. M. A. Nanni and C. Petrioli are with the Dipartimento di Informatica, Università di Roma, "La Sapienza." This work was partially supported by the MIUR FIRB project VICOM (Virtual Immersive COMmunications) and by the International FIRB RBIN047MH9.

SS-Blue, which combines the enhancement to BluePleiades with the credit based scheme for scheduling re-defined for intermediate gateways and for intra-piconet scheduling, is demonstrated via thorough simulations. A new extension to the ns2-simulator, *BlueBrick*, is also presented that includes all the primitives of the BT protocol stacks that are necessary for both scatternet formation and inter-piconet scheduling. Our simulations concern scatternet formation (from device discovery, to piconet formation and interconnection), as well as inter-piconet scheduling experiments based on shortest path-like routing. SS-Blue scatternet formation is compared to BluePleiades with respect to the average number of piconets generated, the percentage of intermediate gateways, the average number of gateways per piconet, the average routes length, and other relevant metrics. Experiments on intra- and inter-piconet scheduling consider metrics such as: End-to-end packet latency, the length of the BT L2CAP queue, which is a good indication of the effectiveness of the scheduling mechanism in being able to efficiently forward the packets, and the per-hop packet latency. Tests performed in several different topologies, on scatternets with both gateway slaves and intermediate gateways, have shown that the credit scheme here specifically adapted to scatternets with intermediate gateways is able to fairly forward inter-piconet traffic thus making it possible to build and deploy multi-hop networks of BT devices.

The rest of the paper is organized as follows. We briefly survey works on scatternet formation and scheduling in Section II, where we also describe BluePleiades and the credit scheme used for scheduling. In Section III we describe SS-Blue. Our solution is demonstrated through extensive simulations in Section IV. Finally, Section V concludes the paper.

II. Related Work

Scatternet formation for multi-hop networks of Bluetooth devices begins with the protocol described in [2] that, starting from a designated node (the *blueroot*) generates a tree-like scatternet. Other solutions for scatternet formation that create topologies different from a tree are those introduced in [3], [4], [5] and [9]. Limits of these protocols include the need for nodal location awareness [3], the formation of connected scatternet whose piconets have an unbounded number of slaves (*BlueStars* [4]), and potentially non-connected scatternets (*BlueNet* [5]). A deterministic protocol that produces connected scatternets with no more than 7 slaves per piconet has been presented in [6], termed *BlueMesh*. In order to form a connected scatternet, the protocol proceeds in successive iterations through which connectivity is achieved progressively. (For further considerations and performance comparison of scatternet formation protocols see [10].)

BluePleiades [8] comes on top of this multitude of solutions for scatternet formation with the aim of combining the device discovery phase (by far the most time consuming in all solution seen so far) and the following scatternet formation phases. Each BT device tries, for at most ρ times, to establish a link only with a constant number c of its neighbors, chosen randomly and uniformly. A node that has already established c^* links (having discovered or having been discovered by neighbors) stops trying and does not accept any further connection. Via a simple geometric random graph model it is proven that both c and c^* can be constant for the resulting discovered topology to maintain the network topology connectivity properties and that, in network with n BT devices, $\rho \in O(\log n)$. A node that has discovered c neighbors moves to the following phase of piconet formation and interconnection which follow closely the rule in [4], i.e., piconets are joined via gateway slaves and also by intermediate gateways (extra piconets). Extensive simulations in practical BT scenarios show that $c = c^* = 6$ or at most 7 are excellent choices that guarantee the connectivity of a BluePleiades scatternet. Once devices are discovered according to the BluePleiades manner devices are partitioned into piconets and interconnected into a connected scatternet as seen for BlueStars [4]. In particular, each node is associated a weight reflecting how suitable it is to perform resource-consuming roles (i.e., master and gateway). Each node makes a decision on its role only after its bigger neighbors (neighbors with bigger weight) have. If the node is invited by a bigger master it happily joins its piconet. Otherwise it will become itself a master. Scatternet connectivity is achieved by interconnecting through gateways all piconets which are two or three hops away [11].

Scheduling gateway nodes to the corresponding piconets has also received considerable attention. The reader is referred to the many seminal papers in this research area, where basic concepts and methods are defined that obtain fair scheduling that are also traffic adaptive [12]–[17]. An inter-piconet scheduling method that achieves fairness as well as traffic adaptiveness has been introduced by Baatz et al. in [7]. The method, called *credit scheme* works as follows. A gateway node maintains a credit account for each of the piconets it belongs to. The gateway spends a credit unit from the account of piconet P for every slot (625μs) it stays with P. This credit is transferred to a temporary account. When this account reaches the value k, k being the number of piconets to which the gateway belongs, the value of the temporary account is reset and one credit is given to any of the gateway's accounts. Every time the gateway reaches a point in time when it *could* switch from the current piconet C to a new one N (i.e., it reaches a *presence point*) the actual decision on whether to switch or not is taken based on credits. More specifically, when the presence point for switching to piconet N is reached, the gateway compares the credit corresponding to C to that associated to N. If the credit with N is bigger than C's, the gateway switches to N. Otherwise, it stays with C. Since every time a gateway switches to another piconet two slots are necessary for synchronization, the number of switches should be limited. To this aim a "switch threshold" h has been introduced according to which the gateway switches from C to N if N's credit exceeds C's by h. Fairness is guaranteed by the temporary account mechanism that increases the credits of piconets the gateway is not staying at. In order to guarantee traffic adaptiveness every time a gateway has no more packets to send/receive in the piconet C where it currently resides (i.e., when a POLL/NULL sequence occurs), the credits x of C's account are redistributed to the other piconets. In particular, let

y be the minimum number of credits associated to one of the other accounts. $r = \lceil k(x-y)/k-1 \rceil$ credits are redistributed from C's account to the other piconets. $r/(k-1)$ credits are added to all the other accounts. In this way the number of credits in C's account equals the minimum number of credits in the other accounts after redistribution.

III. SS-BLUE

We start by describing the enhancement to BluePleiades (BP for short) where the gateway selection is performed so that intermediate gateways are used only when strictly needed (for obtaining a connected scatternet), and preference is given to slave gateways interconnections. We term the protocol resulting from enriching BP with such a mechanism for optimized gateway selection *BluePleiades star*, or BP* for short.

The credit scheme scheduling method for inter-piconet scheduling, here also defined and used for intra-piconet scheduling, is then defined for working on scatternet resulting from BP*.

A. Enhancement to BP

We assume that BP has run over a set of Bluetooth devices as described in [8] up to the second phase, i.e., until devices are discovered and piconets are formed. We recall that after the second phase every master has the list of all its mNeighbors, i.e., of all the masters that are two and three hops away from it. Differently from BP, where each master would establish connections with *all* its mNeighbors, BluePleiades star (BP*) starts by selecting gateways according to the following rules. All (and only) those masters that are *mInit* masters, i.e., that have the highest weight among all its mNeighbors establish connections with all their mNeighbors. As in BP, in case multiple paths are possible between a pair of masters, preference is given to two-hops connections, and in case paths have the same number of hops those with the highest weight are selected (possible ties are broken via the devices' unique ID). At this time, the mInit device sends to all its mNeighbors the list of the *two-hop* connections just established. The generic non-mInit master m waits to receive the lists of connection from all its bigger (i.e., with higher weight) mNeighbors. Based on these lists m establishes connections to all its own smaller mNeighbors. If a two-hop connection to a smaller mNeighbor v is possible, this is the connection that is established. Otherwise, connections to v are established according to the following two rules. 1) If node v appears in at least one of the lists received by m then v is reachable by m via a four hop long path passing through two slave gateways. Therefore, there is no need to establish a new inter-piconet connection through intermediate gateways, i.e., there is no need to establish an extra piconet. 2) If node v does not appear in any of the lists received by m, the creation of an extra piconet is unavoidable for guaranteeing the connectivity of the scatternet. For example in Fig. 1 masters m2 and m3, having received the information on the masters with which master m1 has decided to connect, decide that an extra piconet for joining them is not necessary. The communication between devices in m2 and m3 piconets will occur through m1's piconet. Notice also that, although a packet from piconet m2 to piconet m3 has to travel through four links instead of three, it has to pass through the same number of piconets (namely, three).

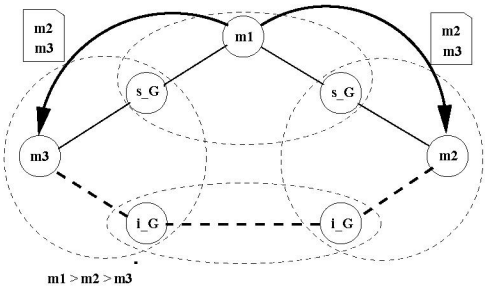

Fig. 1. Avoiding building extra piconets

It is clear that the scatternet connectivity guaranteed by the rule described in Section II [11] still holds.

B. Credit scheme-based scheduling for intermediate gateways

In case of slave gateway interconnections, gateway scheduling follows the fair and traffic adaptive scheme defined in [7], termed credit scheme. Here we define an extension of the credit scheme that is suitable for intermediate gateways. In a slave gateway scenario the gateway node assumes the constant presence of the masters of the piconets to which it belongs. When a gateway belongs to an extra piconet in which it is slave, it cannot assume the master of the extra piconet to be always present (since it is also affiliated to other piconets). Therefore, it is necessary that the two negotiate a common presence point. The slave gateway performs a sniff request to its master proposing a presence point that it calculates locally according to the *maximum distance rendezvous point* (MDRP) algorithm [16]. In other words, the slave computes a presence point in its superframe that is as far as possible from its other presence points and sends the corresponding LMP packet (LMP_sniff_req) to the master. This choice is the best for the slave. If the master has not scheduled other presence points that overlap with the newly received one, it agrees to allocate the requested common presence point and confirms it to the slave by sending a corresponding LMP_accepted packet. In case there is overlap, the master will make a counter-offer to the slave by transmitting a new LMP_sniff_req with a presence point the closest possible to the one originally requested.

Once a common presence point has been agreed upon between the two intermediate gateways, we need to ensure that the two nodes actually meet at that time. This is obtained by modifying the credit scheme method so that an intermediate gateway spends its credit only if it meets the other gateway as scheduled. This guarantees that if the two gateways do not meet for a certain time the credits corresponding to their link in the extra piconet will be much higher that the credits on other links. This will force them to switch to the common piconet at the next corresponding presence point.

IV. Experiments

In order to demonstrate the efficiency of the combined solution for scatternet formation and scheduling we have implemented SS-Blue by using *BlueBrick*, the extension to the ns2 simulator that we have implemented explicitly for this purpose.

We have performed two sets of simulations. The first set concerns a comparative performance evaluation of BP and BP* that shows the effectiveness of BP* in forming scatternets with comparable connectivity and robustness and a considerably lower number of extra piconets (intermediate gateways). The second set of experiments concerns the implementation and testing of the credit scheme in scatternets where data packets are traveling through BP* scatternets.

A. BP* vs. BP

We consider networks where $n = 30, \ldots, 110$ Power Class 3 BT nodes (i.e., nodes with maximum transmission radius of 10 meters) are randomly and uniformly scattered in a geographic area which is a square of side 30m. This allows us to test our protocol on increasingly dense networks, from (moderately) sparse networks, ($n = 30$), to highly dense networks. The average degree ranges from 26.9 for $n = 110$ down to 7.2 when $n = 30$. We call *visibility graphs* the topologies generated by drawing an edge between each pair of BT devices that are in each other transmission range (radio visibility). As the density increases, the average shortest path length in the visibility graph slightly decreases (10%) from 2.37 to 2.14.

All the nodes enter the network within 100ms and run the BP and BP* (with $c = 7$) to form a connected BT scatternet. The value assigned to every node's clock at simulation start is drawn randomly and uniformly in the range 0 and $2^{27} - 1$. (This allows us to evaluate the performance of device discovery in the worst case in which nodes in inquiry use frequencies from both trains A and B [10].) All results presented in this section were obtained by running the two protocols on 500 connected visibility graphs.

Our comparison concerns the following metrics. **(a)** Number of gateways per piconets. **(b)** Number of gateways. **(c)** Percentage of gateways that belongs to at least an extra piconet. **(d)** Number of gateway roles. **(e)** Number of piconets. **(f)** Scatternet robustness. **(g)** Route length.

The average number of gateways per piconet and the average number of slaves per piconet increases with the number of nodes and hence with the network density. This is because each piconet has to interconnect with a higher number of adjacent piconets. As a consequence, the number of gateways also increases. In networks with 110 nodes the number of gateways per piconet approaches the number of slaves per piconet: Almost all the nodes in the piconet are gateways to one or more piconets. We observe that BP* is effective in reducing the number of gateways and the number of gateways per piconet. The higher the density, the more likely it is that two piconets can communicate via intermediate piconets. This justifies the further, evident reduction of gateways in dense networks. When $n = 110$, BP*-built scatternets have about 14% less gateways than BP's. This reduction is even more evident when considering the percentage of gateways that belong to at least an extra piconet. BP* reduces the fraction of intermediate gateways from 75.57% (48.81%) to 43.01% (40.21%) when $n = 110$ ($n = 30$). This is a remarkable improvement since handling intermediate gateway scheduling is more complex and often results into a waste of available resources (as there is no guarantee that a gateway switching to an extra piconet will meet its peer intermediate gateway). Also packets going through extra piconets may incur longer delays.

Fig. 2 investigates another critical performance figure for scheduling efficiency, i.e., the number of piconets to which each gateway belongs.

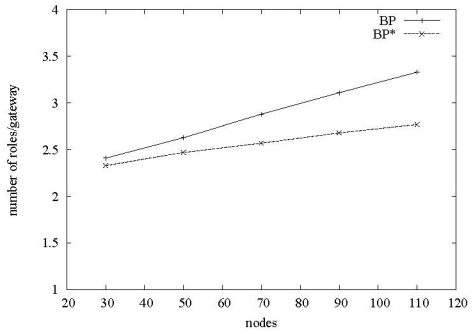

Fig. 2. Gateway roles

By reducing the number of extra piconets, BP* also decreases the number of piconets to which a gateway belongs. In dense networks ($n = 110$) a gateway belongs to an average of 3.33 BP piconets. This number decreases to 2.77 in BP* scatternets.

Fig. 3 shows the number of piconets in the scatternets generated by BP and BP*.

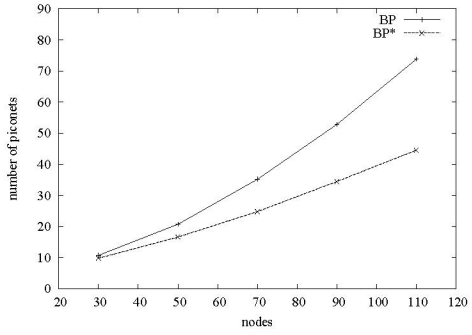

Fig. 3. Number of piconets

Consistently with the results shown above, the number of piconets in BP* scatternet is significantly lower (up to 40%) than that in BP scatternets, and most of this reduction is due to the decreased number of extra piconets.

The price to pay is the reduced percentage of gateway failures that can be tolerated without disconnecting the scatternet, i.e., a lower *scatternet robustness* (Fig. 4).

We notice however that the scatternets generated by BP* are still very robust, being able to tolerate 35.85% (71.42%) gateway failures when $n = 30$ ($n = 110$).

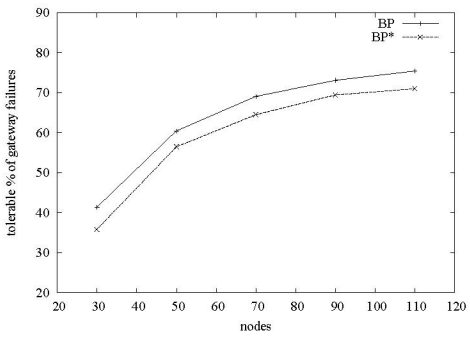

Fig. 4. Scatternet robustness

Finally, the BP* slimmer scatternet shows an expected increase of the average shortest path lengths (Fig. 5).

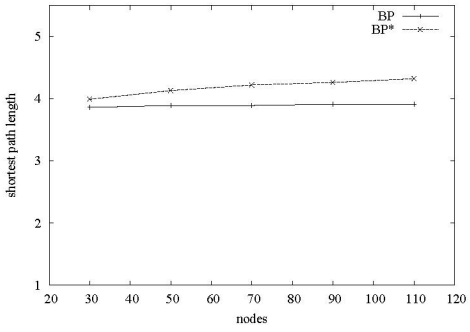

Fig. 5. Route lengths

This increase is however quite contained, since, independently of the network size, BP* average shortest route length is never more than 10% longer than BP's.

B. Intra- and inter-piconet scheduling

We have tested the proposed use of the credit scheme scheduling method in two different scenarios, derived from what occurs in the scatternets generated by BP*. The first scenario comprises 3 piconets with 6 slaves each, three of which are gateways. Fig. 6 shows the scatternet used in this experiment (we show only the nodes that are involved in the packet traffic).

Nodes that are source (SRC_i) and destination (DST_i) of a communication flow are non-gateway nodes (all slaves).

We have performed experiments with two data packet flows from SRC_1 in piconet 1 to DST_1 in piconet 2 and from SRC_2 in piconet 1 to DST_2 in piconet 3. We have tested both intra- and inter-piconet scheduling. Intra-piconet scheduling has been implemented for achieving fairness and traffic adaptiveness. To this purpose we have developed a credit scheme-based solution according to which gateways are polled by the master before any other non-gateway slave. Competitions among slaves and among gateways are resolved by using the credit scheme. In this case each slave has associated a credit account which is managed in a centralized way by the piconet master. Every time the master addresses a slave (giving it the chance to answer back) a credit is removed

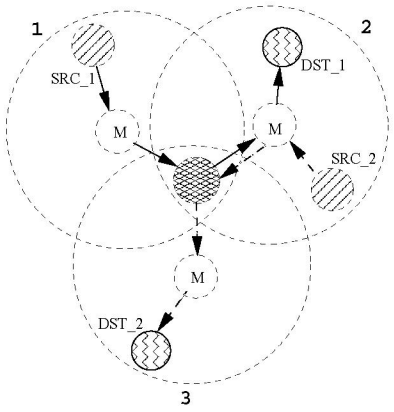

Fig. 6. Three piconet scenario

from its account. When a polled slave does not have anything to transmit (i.e., a POLL/NULL sequence occurs) the slave credits are redistributed to the other slaves. For the way presence points are allocated within the master superframe, i.e., with no overlapping (Section III), the only case when two gateways are simultaneously active in the piconet is when the first being served in the superframe keeps sending packets until the presence point of the second arrives. In this case credits are used to decide between the two gateways. To the best of our knowledge, there is no previous investigation of this scheme for intra-piconet scheduling.

We have compared the credit scheme-based intra-piconet scheduling with the pure round robin scheme where gateways first, and slaves then are polled cyclically, each for the same amount of time. (This is what the Bluetooth specifications indicates as the default intra-piconet scheduler.) As expected, in a scenario where many slaves are inactive, the credit scheme outperforms the round robin remarkably. In the case of a relatively high data flow (50Kb/s, using DH1 packets which implies a theoretical maximum data rate of 172.8Kb/s) from node SRC_1 in piconet 1 to node DST_1 in piconet 2, we observe an average latency from SRC_1 to its master of 1.52s using the round robin scheduler vs. a latency of 0.11s obtained with the credit scheme. The higher delay incurred by the round robin method is due to the presence of 6 slaves in the piconet. Despite there is no traffic from 5 of the slaves to the master, they are polled anyway (unnecessarily). For scatternet formation algorithms like BP*, that generates piconet with an average of 5 slaves, the round robin scheduler is quite inadequate. Given that the credit-based intra-piconet scheduling clearly outperforms the pure round robin scheduler, in the following we show only results where intra-piconet scheduling is credit-based.

The experiments we have performed in the scenario depicted in Fig. 6 aim at confirming the suitability of the credit scheme as the inter-piconet scheduling mechanism of choice in case of slave gateways. We compare the credit scheme scheduling with the exhaustive round robin scheduler (ERR) according to which the piconets to which the gateway belongs are visited cyclically until a POLL/NULL sequence is encountered. Our test lasts 20s. The first 4 seconds are needed to create the scatternet. Then, traffic is generated at the nodes for the following

11 seconds. We consider three different scenarios depending on the data rates with which traffic is transmitted over the two source/destination pairs. In the first scenario traffic generated at both source nodes is relatively low (30Kb/s). Such packet generation rate increases to (50kb/s) in the second scenario.

We have compared the two inter-piconet schemes with respect to the average hop-by-hop packet latency and the gateway queue occupancy (Fig. 7 and Fig. 8). In particular, the figures depicts the queue occupancy. Traffic flow 1 (TF1) refers to the number of backlogged packets generated by SRC_1 and addressed to DST_1, while traffic flow 2 (TF2) refers to the amount of packets generated by SRC_2 and intended for DST_2, currently buffered at the gateway.

When the data flow is relatively low, we observe that both scheduling methods perform comparably. This is because each gateway is able to transmit all the packets it has in queue for the master (and is able to receive all the packets buffered for it at the master) before the next presence point is encountered. This renders the behavior of the credit scheme similar to that of ERR.

In case of higher traffic (50Kb/s) the advantage of using the credit scheme scheduling becomes more evident. Table I shows the end-to-end (EtE) as well as hop-by-hop latency incurred by a packet traveling from SRC_1(2) to DST_1(2) (Traffic Flow 1 and 2 are labeled TF1 and TF2, respectively).

TABLE I

HOP BY HOP DATA LATENCY

	EtE	I hop	II hop	III hop	IV hop
TF1/ERR	1.51	0.11	0.41	0.33	0.66
TF1/CS	1.04	0.11	0.52	0.17	0.24
TF2/ERR	1.35	0.22	0.39	0.37	0.37
TF2/CS	0.76	0.12	0.25	0.2	0.19

We notice that using the credit scheme improves the end-to-end data latency experienced by packets belonging to the first flow (second flow) of over 30% (40%). More noticeably, the use of credit scheme scheduling reduces the latency to and from the gateway (II and III hops), being able to better balance the latency over all links the gateway has in the piconets to which it belongs. For instance, the combined latency on the two most critical links (II and III hops) for TF1 (TF2) is 0.69s (0.45s) vs. the 0.74s (0.76s) of ERR.

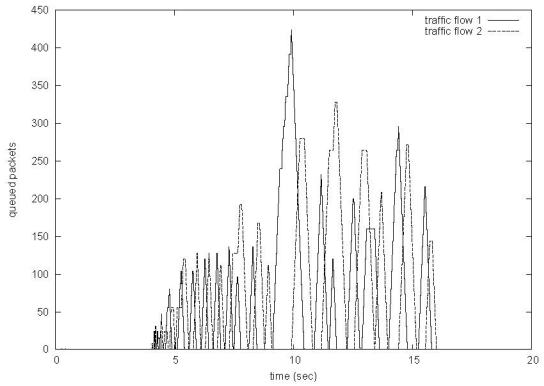

Fig. 7. Three piconets scenario: ERR, data rate $50Kbps$

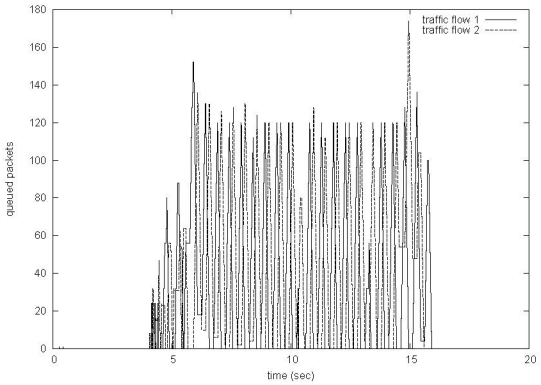

Fig. 8. Three piconets scenario: CS, data rate $50Kbps$

We have finally performed experiments to demonstrate the use of our extended credit scheme with intermediate gateways. The considered scenario is depicted in Fig. 9 (which shows only the nodes that are involved in packet exchange). Each piconet contains 6 slaves, three of which are gateways. Sources and destinations are all non-gateway nodes. (We display only one gateway per piconet; the other two gateways do not have any traffic to transmit).

We have performed experiments with three data packet flows, namely, from SRC_1 in piconet 3 to DST_1 in piconet 1; from SRC_2 in piconet 2 to DST_2 in piconet 4, and from SRC_3 in piconet 1 to DST_3 in piconet 2. The third flow traverses the extra piconet 5. Our attention is focused on the packets belonging to this flow (the other two flows are introduced with the sole purpose of challenging the intermediate gateways ability to meet on the extra piconet). A first experiment concerns the case where all the flows have the same data packet generation rate, namely, 30Kbps (moderate traffic). By using the ERR inter-piconet scheduler data latency of the packets of the third flow results to be more than twice than that of the packets of the other two flows. The credit scheme treats the three flows quite fairly. If we compare the time needed by the credit scheme (CS) and by ERR for forwarding a packet of the third flow (TF3) from the first intermediate gateway to the second, we observe that this time is 50% higher in ERR than in CS. The lower performance of ERR is due to the low likelihood of the two gateways switching to the extra piconets at the same time. The CS naturally enforces this switching because the intermediate gateways accumulate credits if they fail to meet. This effect becomes more evident when we increase the data rate generated by the first two flows. Fig. 10 and Fig. 11 display the gateways queue occupancy over time. Traffic flow 1 refers to the number of backlogged packets at the first intermediate gateway which have been generated by SRC_1. Traffic flow 2 identifies the number of backlogged packets at the second intermediate gateway generated by SRC_2. Finally, traffic flow 3 shows the number of backlogged packets at the first intermediate gateway which have been generated by source SRC_3 (and are awaiting to be transmitted over the extra piconet). In this second experiment we increased the traffic flow between SRC_2 and DST_2 to 45Kbps. The results displayed in Fig. 10 clearly show that ERR is no longer able to

successfully deliver the packets which belong to the third flow. Packets are stacked in the gateway queue until no more traffic is generated (15s). Then the queue empties slowly. Even in this higher traffic scenario CS is able to treat the three traffic flows fairly (Fig. 11), and to deliver all packets within reasonable time (the average end-to-end latency is around 0.73s).

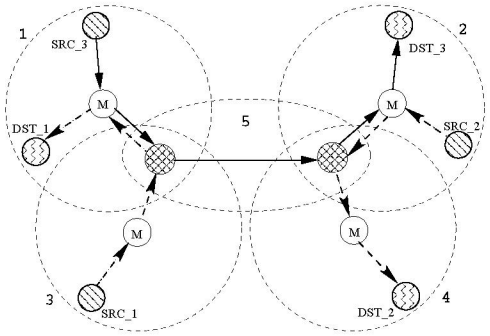

Fig. 9. Five piconet scenario

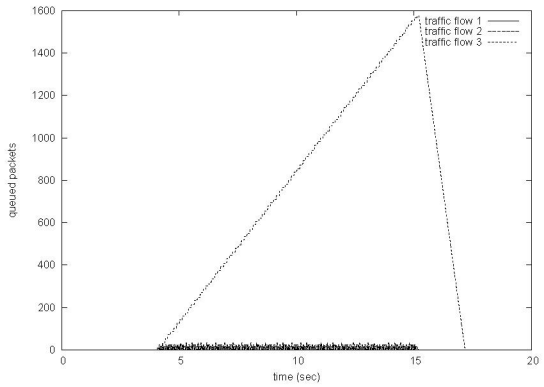

Fig. 10. Five piconets scenario: ERR

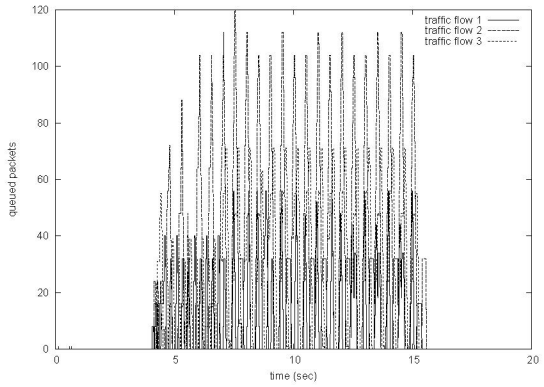

Fig. 11. Five piconets scenario: Credit scheme

V. Conclusions

We have presented an integrated mechanism for scatternet formation and scheduling, SS-Blue, that jointly considers building multi-hop networks of Bluetooth devices and the problem of scheduling their gateways, both slave and intermediate gateways. Experiments have shown that our proposed mechanism is effective in greatly reducing the number of intermediate gateways, tougher to schedule, without losing connectivity or increasing route length sensibly. Furthermore, the extension of the credit scheme to both intra- and inter-piconet scheduling of intermediate gateway, proves to be effective in reducing data latency.

References

[1] http://www.bluetooth.com, *Specification of the Bluetooth System, Volume 1, Core*. Version 2.0 + EDR, November 8 2004.

[2] G. Záruba, S. Basagni, and I. Chlamtac, "BlueTrees—Scatternet formation to enable Bluetooth-based personal area networks," in *Proceedings of the IEEE International Conference on Communications, ICC 2001*, vol. 1, Helsinki, Finland, June 11–14 2001, pp. 273–277.

[3] X. Li and I. Stojmenovic, "Partial Delaunay triangulation and degree-limited localized Bluetooth scatternet formation," in *Proceedings of AD-HOC NetwOrks and Wireless, ADHOC-NOW 2002*, Fields Institute, Toronto, Canada, September 20–21 2002.

[4] C. Petrioli, S. Basagni, and I. Chlamtac, "Configuring BlueStars: Multi-hop scatternet formation for Bluetooth networks," *IEEE Transactions on Computers, Special Issue on Wireless Internet (Y.-B. Lin and Y.-C. Tseng, editors)*, vol. 52, no. 6, pp. 779–790, June 2003.

[5] Z. Wang, R. J. Thomas, and Z. Haas, "BlueNet—A new scatternet formation scheme," in *Proceedings of the 35th Hawaii International Conference on System Science (HICSS-35)*, Big Island, Hawaii, January 7–10 2002.

[6] C. Petrioli, S. Basagni, and I. Chlamtac, "BlueMesh: Degree-constrained multihop scatternet formation for Bluetooth networks," *ACM/Kluwer Journal on Special Topics in Mobile Networking and Applications (MONET), Special Issue on Advances in Research of Wireless Personal Area Networking and Bluetooth Enabled Networks (G. Zaruba and P. Johansson, editors)*, vol. 9, pp. 33–47, February 2004.

[7] S. Baatz, M. Frank, C. Kuhl, P. Martini, and C. Scholz, "Bluetooth scatternets: An enhanced adaptive scheduling scheme," in *Proceedings of IEEE Infocom 2002*, vol. 2, New York, NY, June 23–27 2002, pp. 782–790.

[8] D. Dubhashi, O. Häggström, G. Mambrini, A. Panconesi, and C. Petrioli, "Blue pleiades, a new solution for device discovery and scatternet formation in multi-hop bluetooth networks," *ACM/Kluwer Wireless Networks*, 2006, to appear.

[9] C. Petrioli and S. Basagni, "Degree-constrained multihop scatternet formation for Bluetooth networks," in *Proceedings of the IEEE Globecom 2002*, vol. 1, Taipei, Taiwan, R.O.C., November 17–21 2002, pp. 222–226.

[10] S. Basagni, R. Bruno, G. Mambrini, and C. Petrioli, "Comparative performance evaluation of scatternet formation protocols for networks of bluetooth devices," *ACM/Kluwer Wireless Networks*, vol. 10, no. 2, pp. 197–213, March 2004.

[11] I. Chlamtac and A. Faragó, "A new approach to the design and analysis of peer-to-peer mobile networks," *Wireless Networks*, vol. 5, no. 3, pp. 149–156, May 1999.

[12] R. Kapoor, A. Zanella, and M. Gerla, "A fair and traffic dependent scheduling algorithm for bluetooth scatternets," *Mobile Networks and Applications*, vol. 9, pp. 9–20, 2004.

[13] A. Racz, G. Miklos, F. Kubinszky, and A. Valko, "A pseudo random coordinated scheduling algorithm for bluetooth scatternets," in *Proc. 2001 ACM International Symposium an Mobile ad hoc networking and computing*, Long Beach, CA, October 2001, pp. 193–203.

[14] N. Johansson, F. Alriksson, and U. Jonsson, "Jump mode - a dynamic window-based scheduling framework for bluetooth scatternets," in *Proc. 2001 ACM International Symposium an Mobile ad hoc networking and computing*, Long Beach, CA, October 2001, pp. 204–211.

[15] S. Saha and M. Matsumoto, "An inter-piconet scheduling algorithm for bluetooth scatternets," in *Proc. of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT/ICIW 2006)*, Guadelupe, February 2006.

[16] P. Johansson, R. Kapoor, M. Kazantzidis, and M. Gerla, "Rendezvous scheduling in Bluetooth scatternets," in *Proceedings of IEEE ICC 2002*, vol. 1, New York, NY, April 28–May 2 2002, pp. 318–324.

[17] W. Zhang and G. Cao, "A flexible scatternet-wide scheduling algorithm for bluetooth networks," in *Proc. 21st IEEE International Performance, Computing, and Communications Conference IPCCC 2002*, Phoenix, AZ, April 2002, pp. 291–298.