Time-synchronised multi-piconet Bluetooth environments

I. Ashraf, A. Gkelias, M. Dohler and A.H. Aghvami

Abstract: Multiple Bluetooth piconets operating in the globally available 2.4 GHz industrial, scientific and medical (ISM) band are likely to co-exist in a physical environment, supporting applications such as wireless earphones, keyboards etc. An independently operating Bluetooth piconet will inevitably encounter the interference from collocated piconets, which results in both individual piconet and overall network performance degradation. We propose a simple time-synchronisation scheme among co-existing Bluetooth piconets that yields considerable performance improvements as compared to uncoordinated piconets. We introduce a new state to the Bluetooth link controller that does not require any changes to the existing Bluetooth hardware. The proposed scheme synchronises the collocated piconets in a totally blind fashion and incurs minimal overhead to the participating piconets. The performance of the scheme is measured by evaluating metrics such as packet error rate, individual piconet throughput and aggregate network throughput using an extensive analytical model. Furthermore, the effect of partial synchronisation among collocated piconets is also investigated.

1 Introduction

Bluetooth is a technology using short-range radio links, forming a small network among communicating nodes called a piconet [1, 2]. It is intended to replace the cable(s) connecting portable and/or fixed electronic devices. The usage of Bluetooth-enabled devices has increased enormously in recent years. This trend has facilitated an extensive deployment of low-power *ad hoc* wireless communication links between different electronic devices to transfer data, synchronise information and even connect to the Internet through Bluetooth. In short, Bluetooth has served the need of an increasingly mobile lifestyle by providing an independent personal area network (PAN) to the individual end-user.

Based on the deployment of the Bluetooth technology on such a rapid scale, multiple Bluetooth piconets are likely to co-exist in a physical environment. A typical example of this can be portrayed as a classroom where each student operates his/her own Bluetooth piconet to transfer or synchronise files. Hence, a typical classroom may easily have a large number (e.g. 100) of co-existing piconets, located within each other's transmission range and using the same 2.4 GHz industrial, scientific and medical (ISM) band. Therefore, it becomes vitally important to investigate the problem of co-channel interference arising from other Bluetooth piconets and propose a feasible solution to mitigate it.

Mutual interference between Bluetooth piconets was investigated in [3] and [4], where the authors addressed the problem by presenting simulation results considering fully and partially loaded piconets. In [5], the author presented an upper bound on the packet error rate of a Bluetooth asynchronous connectionless (ACL) link under co-channel interference. The work also showed the effects of mutual interference on network parameters such as aggregate network throughput as a function of the number of interfering piconets. The analysis, however, had two drawbacks. First, it failed to combine different Bluetooth baseband parameters, such as incorporating different packet types. Second, [5] assumed that the time slots of each co-existing piconet are always fully occupied by packets.

These considerations were successfully featured in [6], in which different packet types and traffic models were integrated into the collision analysis. The network performance measures were chosen as packet error probability, individual piconet throughput and overall network throughput. The analytical results were benchmarked against the results from simulations. In [7], the analytical approach of [6] was extended further by including the frequency-hopping guard time effect.

The aforementioned studies provide an analytical approach to the mutual interference problem in co-existing Bluetooth piconets. These studies, however, do not propose any interference mitigation approach. The already published works emphasise the fact that a large number of co-existing piconets severely degrades the individual piconet or overall network performance in terms of higher packet error rates and lower aggregate network throughput. The aim of this paper is to propose a simple interference mitigation scheme in co-existing Bluetooth piconets that can improve the network performance.

The quantification of Bluetooth and IEEE 802.11b co-existence has drawn much attention lately [8–11], and

IEE Proceedings online no. 20050510

doi:10.1049/ip-com:20050510

Paper first received 15th September and in final revised form 17th November 2005

I. Ashraf, A. Gkelias and A.H. Aghvami are with Centre for Telecommunications Research, King's College London, 26-29 Drury Lane, London, WC2B 5RL, UK

M. Dohler is with the France Telecom R&D, 28 Chemin du Vieux Chêne, 38243 Meylan Cedex, France

E-mail: imran.ashraf@kcl.ac.uk

an IEEE Co-existence Task Group (IEEE 802.15 Task Group 2) is working to devise co-existence methods for the two complimentary technologies. One of the outputs of Task Group 2 has been the adoption of a non-collaborative interference mitigation scheme, adaptive frequency hopping (AFH), in the Bluetooth specification v1.2 [2]. AFH is designed to reduce interference in the ISM band by identifying fixed sources of interference and excluding them from the list of available channels [12]. However, AFH is primarily intended to identify and exclude channels that are in continuous use by other devices or technologies, i.e. static sources of interference such as IEEE 802.11b/g. It might not be very well suited to devices using frequency hopping. This is because, based on channel assessments, the number of available frequency hops in AFH is reduced, thereby increasing the probability of collision in dense homogeneous wireless environments. It shall finally be noted that the Bluetooth special interest group (SIG) has defined an enhanced standard, referred to as Bluetooth v2.0 [13]. It offers a drastically increased data rate of up to 3 Mbps and enhanced features for multi-device piconet maintenance, but no additional mechanisms to mitigate the interference owing to co-existing piconets; our analysis is hence also applicable and useful to Bluetooth v2.0.

Although the co-existence issue between Bluetooth and IEEE 802.11b/g in the 2.4 GHz band is an important one, it is not the only interference problem to be addressed in the free ISM band. Research studies on the mutual interference scenario of multiple co-existing Bluetooth piconets are few and far between. With such a rapid and pervasive deployment of Bluetooth enabled electronic devices, this issue requires the need for a practical and simple approach towards interference mitigation in multiple co-existing piconets.

The technique presented in this paper is a simple time-synchronisation scheme that achieves synchronisation among co-existing piconets such that a packet transmission in a certain piconet is never interfered with by more than one colliding packet from another piconet. This substantially reduces the number of potential interfering packets from collocated piconets as compared to the unsynchronised case. Synchronisation is accomplished blindly in the whole network of piconets by assigning negligible overhead or responsibilities to the co-existing masters. The whole process does not interfere with the regular piconet activities of a master and can be tailored as per the event-scheduling needs of individual piconet controllers.

The scheme uses a time-domain approach for interference mitigation and is totally noncollaborative in nature. It is well understood that non-collaborative co-existence schemes offer more flexibility and practicality, and are applicable to more application scenarios as compared to their collaborative counterparts. Furthermore, the proposed scheme allows independent piconets to continue operating independently without any need to exchange control information that may place additional overheads.

The new scheme can improve the error performance, data rate, plus the individual and aggregate throughputs of collocated Bluetooth piconets. It also caters for the implementation considerations and presents a solution that requires minimal changes to the existing Bluetooth standard. In fact, the scheme does not pose any change to the hardware and can be simply implemented by a firmware upgrade. In short, the scheme can be effectively used in dense multi-piconets Bluetooth environments and, hence, is a simple and cost-effective way to improve network performance.

The rest of the paper is organised as follows. The Bluetooth system overview is described in Section 2. The underlying interference model is discussed in Section 3. In Section 4, the new piconet synchronisation scheme is explained. Analytical results are then presented in Section 5 and conclusions are drawn in Section 6.

2 Bluetooth

The entire Bluetooth protocol architecture is defined in [1]. A brief summary will be presented in this section with special emphasis on its PHY layer characteristics and parameters. A Bluetooth piconet essentially uses a master/slave architecture in which the master controls the traffic flow. All Bluetooth devices have identical hardware properties so that the master is only selected when the network is successfully established. The unit initiating the connection acts as the master and can organise the channel access for up to seven other active units, which are called slaves. The master carries out the polling in a time-division duplex (TDD) manner.

Bluetooth uses frequency hopping spread spectrum (FHSS) to transmit data packets and frequency hop range in most countries covers 79 carriers of 1 MHz bandwidth each. Binary baseband data is modulated using Gaussian frequency shift keying (GFSK) and the frequency synthesiser transmits each packet on a newly chosen frequency. The maximum hopping rate in a piconet is 1600 hops per second. During the packet transmission, the actual data is transmitted in part of the total packet transmission time, e.g. 366 μs out of 625 μs in the case of a single-slot packet transmission. The remaining time (259 μs, as shown in Fig. 1) is used to let the electronics stabilise to the next frequency hop, known as the frequency hopping guard time.

Two physical links are supported in Bluetooth: ACL for data traffic and synchronous connection-oriented (SCO) for time-bounded voice communication. SCO voice links always have a higher priority than ACL data connections. Table 1 summarises all the SCO and ACL supported packet types. Three SCO packets are defined: HV1, HV2 and HV3.

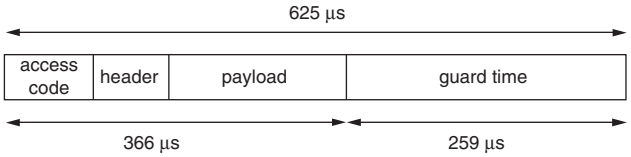

Fig. 1 *Frequency hopping guard time illustration for 1-slot packet transmission*

Table 1: ACL and SCO packet types

	Type	User payload (bytes)	FEC	CRC
ACL packets	DM1	0–17	2/3	yes
	DH1	0–27	no	yes
	DM3	0–121	2/3	yes
	DH3	0–183	no	yes
	DM5	0–224	2/3	yes
	DH5	0–183	no	yes
	AUX	0–29	no	no
SCO packets	HV1	10	1/3	yes
	HV2	20	2/3	yes
	HV3	30	no	yes

HV stands for high-quality voice. These packets are all single slot but vary in the amount of information they carry: HV1 carriers 10 bytes, HV2 carries 20 bytes and HV3 carries 30 bytes. The ACL link, however, allows 1-, 3- and 5-slot data packets with the optional use of forward error correction (FEC). DM stands for medium-speed data, and DH stands for high-speed data. DM packets are all 2/3-FEC encoded to tolerate possible transmission errors. Not encoded by FEC, DH packets are more error-vulnerable but can carry more information. DM1/DH1 packets occupy one time slot, while DM3/DH3 and DM5/DH5 packets occupy 3 and 5 time slots, respectively. A speci al single-slot ACL packet, AUX1, can also be used to transfer raw data between two Bluetooth devices. It does not employ a cyclic redundancy check (CRC) code and can be used to test the bit error rate (BER) across a channel.

This paper focuses on performance improvements in ACL connections. Only DH1, DH3 and DH5 packets are considered as these packets do not use FEC and are more susceptible to channel impairments and interference.

3 Interference model

We consider N Bluetooth piconets co-existing independently in a certain closed physical environment, as illustrated in Fig. 2. This leads to each piconet being suffered by $N-1$ potential interfering piconets. We assume that all nodes within a given piconet lie in the transmission range of all the other co-existing piconets. Thus, they are collocated sufficiently close in such a manner that if two or more piconets transmit a packet on the same frequency band at any instant, the corresponding colliding packets are considered corrupted and lost. All piconets use a frequency hop range of 79 frequency channels.

As explained in the previous section, three types of ACL packets are considered here. Each of these packets has a probability of arrival associated with it. The arrival probabilities per slot for 1, 3, and 5-slot packets are represented by ρ_1, ρ_3 and ρ_5, respectively. Time used in activities other than the data traffic or the idle time is modelled by assuming a single-slot dummy or empty packet that does not carry any data traffic but occurs with a certain probability. This is the same approach used by [6, 7, 14].

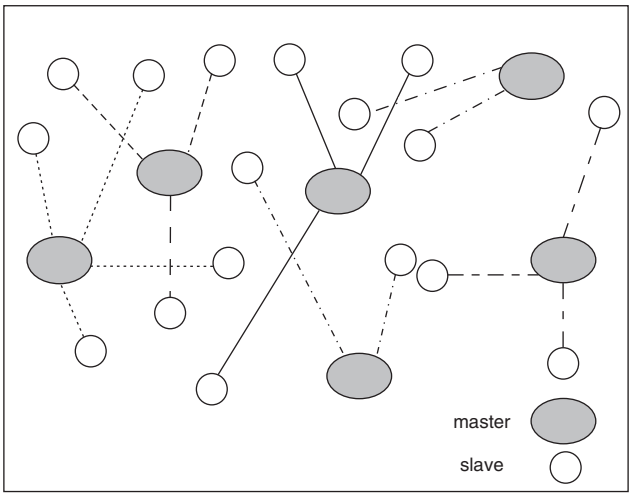

Fig. 2 *Network topology of co-existing Bluetooth piconets in a closed physical environment*
Each master can be serving several slaves independently using TDD scheduling and transmitting data using FHSS

The dummy packet is assigned the probability ρ_0, such that $\rho_0 = 1-(\rho_1+3\rho_3+5\rho_5)$. We explain in the later sections that ρ_0 also includes the slight overhead incurred by the proposed synchronisation scheme. Furthermore, in the case of a 3- or 5-slot packet, only the first slot counts as the arrival. We assume that all collocated piconets carry identical traffic with equal arrival probabilities ρ_1, ρ_3 and ρ_5. An extension of the presented analysis to the case of different arrival rates in all piconets is tedious but feasible.

4 New Bluetooth synchronisation scheme

In [7], the authors presented a closed form solution for packet error rate in unsynchronised collocated piconets taking into account the frequency-hopping guard time effect. In this section, we present a novel scheme to achieve synchronisation in piconets that is consequently shown to yield considerable performance improvements as compared to the unsynchronised case.

4.1 Background – Bluetooth piconet connection states

The Bluetooth link controller, which sits above the radio in the Bluetooth protocol stack, is a state machine that can adopt different states in a random fashion as illustrated in Fig. 3. There are various states associated with the Bluetooth piconet activity, followed by the specific methods for inquiry and paging. When a Bluetooth device is first powered up, it enters the STANDBY state, where its hardware and software are initialised. From this state, the master can either enter the INQUIRY or the PAGE state.

The INQUIRY state enables the prospective master to find out which other stations are within transmission range, and what their addresses and clock states are. From here, the master can either return to the STANDBY state or enter the PAGE state to establish a connection with the respondent (slave). Once the piconet is created, both the master and slave now appear in the CONNECT state. It is essential for the prospective slave to periodically move out of the STANDBY state to listen for inquiries and pages in the INQUIRY SCAN and PAGE SCAN states, respectively. If paged successfully, it transitions to the CONNECT state along with the master. Also, the master already in the

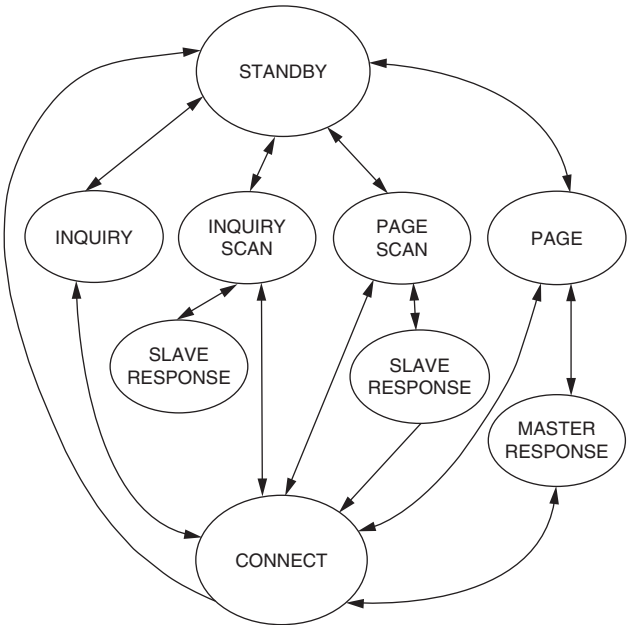

Fig. 3 *Bluetooth link controller connection states*

CONNECT state has to move back to the PAGE or INQUIRY states regularly. This enables the master to search for prospective slaves in the vicinity and let them join the piconet.

After the successful establishment of the piconet, the master centrally manages access to the channel using the TDD scheme. All Bluetooth devices have a native clock, called CLKN, from which various timing signals are derived to meet the timing requirements of the TDD process. Piconet timing is under the control of the master and all the slaves synchronise their clocks to that of the master. The piconet timing clock is called CLK, and for the master CLKN = CLK. Slaves achieve synchronisation by adding an offset to their CLKN to form CLK.

The Bluetooth device clock is implemented as a 28-bit counter that is set to 0 on power up, and is incremented every 312.5 μs or one-half of a time slot. The last two bits of the 28-bit counter determine when a master or a slave should start packet transmission. A master in the CONNECT state only starts transmission when the last two bits of its clock are 00, whereas a slave transmits only when the last two bits of its clock, synchronised to that of the master's, are set to 10. In this manner, the master always transmits in an even-numbered slot and the slave always transmits in an odd-numbered slot.

To communicate properly using FHSS, the Bluetooth devices must be properly synchronised so that they hop together from channel to channel. The frequency hopping sequence is also specified by the master and follows a pseudo-random number sequence which is calculated, using relatively complex rules, from the master's 48-bit device address, and is therefore unique to each piconet. This has been designed to support the operation of the maximum possible number of independent piconets in a small geographical area. However, with increasing number of piconets the probability of packet collisions in frequency also increases. The purpose of the scheme presented hereafter is to reduce the number of potential interfering packets to an ongoing packet transmission, by causing a minimal overhead on the network or individual masters.

4.2 Scheme description

We introduce a new state to the Bluetooth link controller, SYNC. The masters and their respective slaves have to be in the SYNC state regularly once they have successfully established the CONNECT state. A master within a certain piconet is generally unaware of the presence of masters in collocated piconets. The purpose of the SYNC state is to synchronise with co-existing piconets blindly and in a totally *ad hoc* fashion. The following tasks are associated with the SYNC process:

- All the collocated masters in the CONNECT state fetch a 10-bit uniformly distributed random number RAND, which has a value between 0 and 1023 (it is the same as the one used in the INQUIRY process [1]). This translates to a timer length of 0 to 639 ms, with an average value denoted by $\overline{T}_t$. We call it the synchronisation timer and a master, with all its slaves, moves from the CONNECT to the SYNC state after RAND number of time slots have expired. The master can broadcast the instruction to its slaves about the transition to the SYNC state in the next slot.
- Once the timer expires, a master and its slaves move from the CONNECT state to the SYNC state for a time duration denoted by T_S. The master and the slaves listen for a 68-bit ID packet on a pre-defined frequency in the SYNC state. This frequency should be different from the 32 frequencies used in the INQUIRY process. The structure of the ID packet remains the same as defined in the Bluetooth standard [1]. It contains the general inquiry access code (GIAC) that comprises of a 4-bit preamble and a 64-bit sync word. GIAC is used as the access code so that all the nodes can listen to the ID packet. The process of deriving the sync word remains the same as explained in [1].
- If the master and its slaves do receive the ID packet at any instant during the T_S period (this might have been sent by another master who came out of the random synchronisation timer earlier), they transit out of the SYNC state. They then update their clock information by resetting the last two bits to 10, once the reception is complete. This is because the last two bits of the clock of the master, which originally transmitted the ID packet, will also be 10 at the end of transmission. This ensures that the time slots in the participating piconets are synchronised and always start at the same time instant. The newly synchronised piconets might not return to the SYNC state together as each of them will enter into an independent synchronisation timer based on their respective RAND.
- If the master and its slaves do not receive the ID packet for the whole length of the T_S listening period, the master transmits its own ID packet at the end of T_S. After transmitting, the master comes out of the SYNC state with its slaves, resets the synchronisation timer and does not return for the length of the timer.
- There is an important point to be examined here. When the synchronisation timer expires, a master might not necessarily be always in the CONNECT state. It could be involved in inquiry and paging processes. In that case, the master can continue with the processes it is involved with, and reset the timer to come back to the SYNC state afterwards.

To gain further insight into the interaction between different piconets during the SYNC process, we consider an example of synchronisation among three collocated piconets X, Y and Z as shown in Fig. 4. The master from piconet X (M_X), along with its slaves, is the first to move from the CONNECT mode to the SYNC mode at point A. It starts its period of T_S time duration to listen for any incoming ID packets. The master in piconet Y (M_Y) along with its slaves enters the SYNC mode at point B and also starts listening on the pre-defined frequency. At point C, M_X finishes its listening period and transmits the ID packet. This packet is received by M_Y as it is still in the SYNC state at point C. M_Y will update its piconet clock based on the information

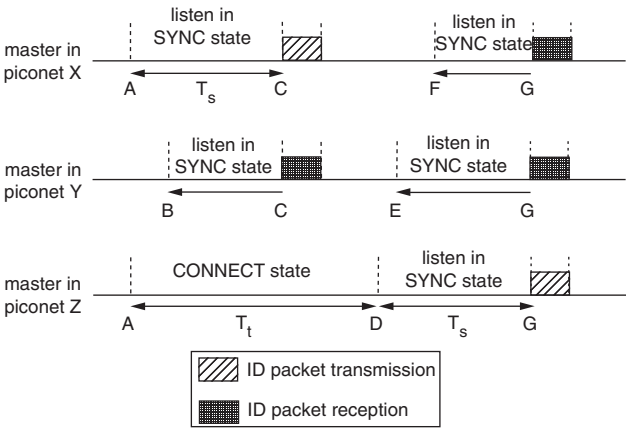

Fig. 4 *Synchronisation in three collocated piconets*

retrieved from the received packet. Hence, the piconets X and Y will be synchronised and both will enter in to their respective synchronisation timers independently. At point D, the master from piconet Z (M_Z) enters the SYNC state. The synchronisation timers for M_X and M_Y expire at point F and E respectively, and both piconets start listening for the ID packet. The ID packet transmitted by M_Z at G is received by both M_X and M_Y, which update their respective information accordingly. Hence all the three collocated piconets are synchronised at G. As an alternative, M_X may come back into the SYNC state at point D i.e. before M_Y and the unsynchronised master M_Z. In that case, it will reach point G first and transmit the ID packet. Y and Z will then synchronise their clocks accordingly.

Next, we dwell on the notion of convergence of all piconets to one synchronised state. We model our synchronisation architecture by a system consisting of k unsynchronised collocated piconets, denoted by Ω_k. We prove that as time approaches infinity, the system converges to Ω_1, or alternatively all piconets are synchronised. We specifically use an intuitive method to prove the convergence, and do not focus on how convergence is achieved or the delays associated with it. The system can retain a certain state Ω_i having i number of unsynchronised piconets with probability P_i. ϵ is defined as $\epsilon = 1 - P_i$, such that $0 < \epsilon \leq 1$. Since $\epsilon > 0$, the system always moves from a state with higher number of unsynchronised piconets to a state with a lower number. We denote by P_{Ω_i} the probability to remain in the state Ω_i after t number of synchronisation events have taken place. Thus, $P_{\Omega_i} = P_i^t$, for $i = 2 \ldots k$. Since the system can always change its state with probability ϵ, we have

$$\lim_{t \to \infty} P_{\Omega_i} = \lim_{t \to \infty} (1 - \epsilon)^t = 0 \qquad (1)$$

State transitions always happen from Ω_m to Ω_l, where $m > l$, the system therefore converges to Ω_1 with probability 1 as $t \to \infty$. The state transition diagram for a system initially consisting of k unsynchronised piconets is depicted in Fig. 5.

It is imperative here to calculate the time required for piconets to synchronise. We do this by running simulations on a C++ platform and finding the average time required for the new scheme to synchronise all co-existing piconets. Figure 6 depicts the effects on synchronisation time in dependency of the factor $\eta = \overline{T}_S / T_t$. In essence, η provides us with a measure of the time consumed in the new state SYNC as a percentage of the total time. We fix the value of RAND as 1023 ($\overline{T}_t = 320$ ms) and vary T_S to plot synchronisation time for different values of η. The results illustrate that although a higher value of η reflects a higher overhead factor, yet it provides faster synchronisation of the system. Expectedly, the synchronisation time increases with increasing values of N.

We observe that the only overhead the new scheme induces is the idle time slots that occur during the listening period. Presented in the next section, our analytical methodology incorporates this overhead in the form of occurrence of single-slot dummy packets, as explained in Section 3. Also, the whole synchronisation process is not affected by a dynamic network topology where new piconets are moving in and out of the geographical area under consideration. If a new piconet joins in, it will experience higher interference initially until it finally synchronises with the rest of the network of piconets.

4.3 Analysis

The performance of a Bluetooth network can be evaluated using various quantitative metrics such as packet error rate

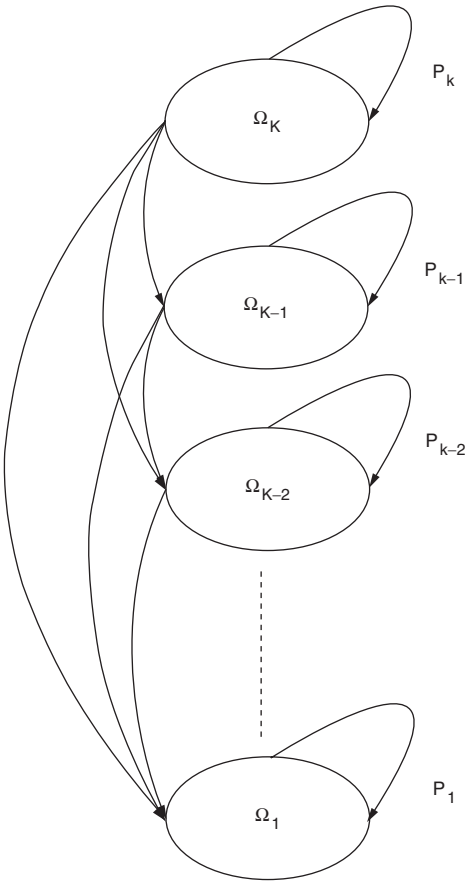

Fig. 5 *State transitions in a system of k unsynchronised piconets*

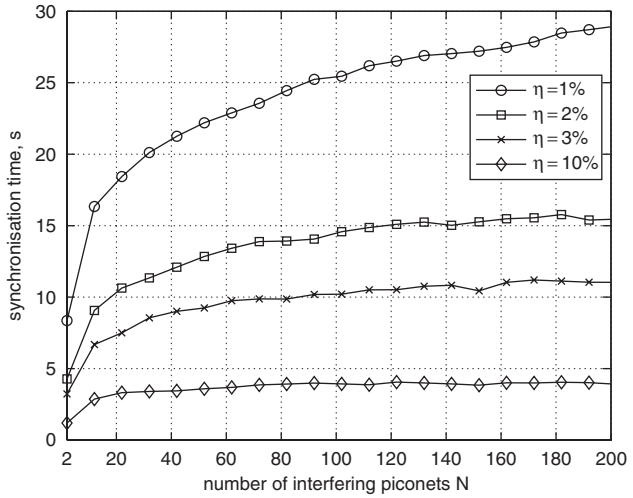

Fig. 6 *Synchronisation time against N*

(PER), individual piconet throughput (S) and aggregate network throughput (Θ). The relevance of each performance metric is dependent on the specific network requirements. In this paper, the above-mentioned three measures of performance are considered for accentuating the effect of synchronisation in collocated piconets. The respective equations for PER, S and Θ are derived using a probabilistic treatment entailing different Bluetooth baseband parameters. The approach is similar to the one followed by the authors in [6, 7, 14].

We begin by considering two co-existing piconets X and Y. Each of these piconets can transmit a 1-, 3- or 5-slot (DH1, DH3 or DH5) packet with respective probabilities ρ_1, $3\rho_3$ or $5\rho_5$. We focus our attention on piconet X and derive the success probabilities $P_s(i)$, for each of the

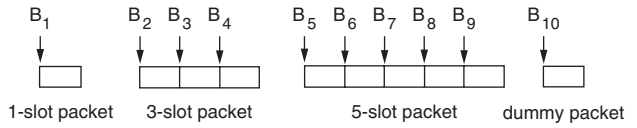

Fig. 7 *Slot beginnings for different packet types*

three packet types i.e. $i=1,3$ and 5, in the presence of interference from piconet Y. For an ongoing transmission in X, the probability that piconet Y chooses another frequency (no collision) as the one chosen by piconet X is $P_0 = 1 - 1/79 = 78/79$.

We denote the starting slot boundary in each packet type as shown in Fig. 7, such that:

- B_1, B_2, and B_5 mark the beginning of a 1-, 3-, and 5-slot packet, respectively.
- B_3 and B_4 represent the beginnings of the second and third slots of a 3-slot packet, respectively.
- B_6, B_7, B_8, and B_9 denote the beginnings of the second, third, fourth, and fifth slots of a 5-slot packet, respectively.
- B_{10} corresponds to the beginning slot boundary of an empty or dummy packet.

All slot beginnings B_j, $j=1,\ldots,10$, have arrival probabilities associated with them. It is clear that the probability of occurrence of B_1 is ρ_1 per slot; the probability of occurrence for each of B_2, B_3, and B_4 is ρ_3; the arrival probability for each of B_5, B_6, B_7, B_8, and B_9 is ρ_5; and the probability of occurrence of B_{10} is ρ_0. To emphasise on the occurrence of a particular slot, we denote the arrival probability of B_j by $\xi(B_j)$, $j=1,\ldots,10$. We also define $g(j)$ to be the number of slots that follow the slot beginning B_j and belongs to the same packet. As an example, $g(1)=1, g(3)=2, g(8)=2,$ and $g(10)=1$.

Before we formulate the success probability $P_s(i)$ of an i-slot packet, we present an example of packet collision for a 3-slot ($i=3$) packet in piconet X as illustrated in Fig. 8. Since the two piconets X and Y are time-synchronised, the slot beginnings of transmitted packets in the two interfering piconets start at the same time instant. In Fig. 8, piconet Y transmits a 1-slot packet containing the slot beginning B_1, with probability ρ_1. The success probability for the first slot of the 3-slot packet would therefore be $P_0=78/79$. We denote this probability function by $\tau(j)$, where j represents the jth slot beginning in Y's packet. Intuitively, if Y transmits the dummy packet (B_{10} with probability ρ_0), the success probability is simply 1. We can move on to

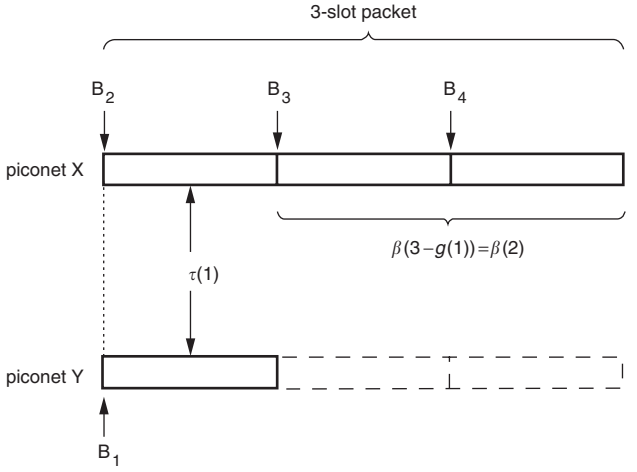

Fig. 8 *Packet collisions in synchronised piconets*

consider the success probability of the remaining part ($i-g(1)$) of X's 3-slot packet. In Bluetooth, two consecutive packets are transmitted on two different frequencies. The next packet in Y could be either a 1-, 3-, 5-slot or a dummy packet, but is transmitted on a different frequency with respect to the first one. In any case, Y transmits a 1-slot packet next, the success probability of the second slot of X's packet will be $\widetilde{P}_0 = 1 - 1/78 = 77/78$. Else, if Y transmits a dummy packet next, the success probability will be 1 for the second slot of the X's packet. In both cases, we will then need to consider the success probability of the remaining last slot of X's packet. Thus, the success probability of the remaining portion of X's packet is solved recursively by defining a probability function $\beta(m)$. Intuitively, $\beta(m)$ is the success probability of the last m slots of X's packet, excluding the first slot.

Based on the aforementioned probability functions, we formulate $P_s(i)$ as follows

$$P_s(i) = \sum_{j=1}^{10} \xi(B_j) \cdot \tau(j) \cdot \beta(i - g(j)) \tag{2}$$

where

$$\tau(j) = \begin{cases} 1 & \text{if } j = 10, \\ P_0 & \text{otherwise} \end{cases} \tag{3}$$

and $\beta(m)$ is defined for $m>0$ as follows

$$\begin{aligned} \beta(m) = & \frac{\rho_0}{\rho_0+\rho_1+\rho_3+\rho_5} \cdot \beta(m - g(10)) \\ & + \frac{\rho_1}{\rho_0+\rho_1+\rho_3+\rho_5} \cdot \widetilde{P}_0 \cdot \beta(m - g(1)) \\ & + \frac{\rho_3}{\rho_0+\rho_1+\rho_3+\rho_5} \cdot \widetilde{P}_0 \cdot \beta(m - g(2)) \\ & + \frac{\rho_5}{\rho_0+\rho_1+\rho_3+\rho_5} \cdot \widetilde{P}_0 \cdot \beta(m - g(5)) \end{aligned} \tag{4}$$

where $\beta(m)=1$, for $m \le 0$. In (2), we consider each type of slot beginning (B_j), $j=1,\ldots,10$, to appear in the first slot. The corresponding probability is $\xi(B_j)$. The function $\tau(j)$ gives the success probability for the first slot of X's packet and the function $\beta(m)$ accommodates the success of the remaining part of X's packet.

Equation (2) gives the probability of success for an i-slot packet in piconet X in the presence of the interfering piconet Y. X experiences interference from $N-1$ independent sources when there are N piconets co-existing. The PER_i for an i-slot packet in piconet X is thus given as

$$PER_i = 1 - P_s(i)^{N-1} \tag{5}$$

The throughput, S, for piconet X in an N-piconet environment is given as

$$\begin{aligned} S = & \rho_1 \cdot P_s(1)^{N-1} \cdot R_1 + 3 \cdot \rho_3 \cdot P_s(3)^{N-1} \cdot R_3 \\ & + 5 \cdot \rho_5 \cdot P_s(5)^{N-1} \cdot R_5 \end{aligned} \tag{6}$$

where $R_1(345.6)$, $R_3(780.8)$ and $R_5(867.9)$ are the data rates (kbps) for DH1, DH3, and DH5 packets, respectively (for example, 1464 data bits are contained in a DH3 packet of time duration 1875 μs, therefore $R_3 = 1464/1875 = 780.8$ kbps). The aggregate network throughput, Θ, of successfully transmitted packets in all the piconets is thus given as $\Theta = N \times S$.

5 Numerical results

Based on the analysis presented in the previous section, several numerical results are shown, and the performance of synchronised transmissions is compared against unsynchronised transmissions in this section. Note that apart from the idle time in the piconet, ρ_0 also takes into account the overhead caused by the proposed scheme.

In Fig. 9, the PER performance of the new synchronisation scheme is plotted against unsynchronised transmissions, as a function of number of piconets co-existing in a certain environment. DH1/3/5 packets are considered with equal arrival probabilities ($\rho_1 = \rho_3 = \rho_5$). In Fig. 9*a*, a lighter uniform traffic load of 30% ($\rho_0 = 0.7$) is considered in all collocated piconets. There are a couple of common points to be considered here. First, the PER increases as the number of piconets increases, reflecting an increased mutual interference. Second, smaller packets (DH1) have lower values for PER with respect to the longer packets (DH5). This is because they suffer fewer collisions owing to their shorter time durations. As shown in the figure, the new scheme delivers significant PER improvements for all the three packet types. For example, it offers approximately 20% PER improvement for DH1 type packets when 150 piconets are existing together. It is worth considering here that improvement gaps increase as the number of piconets increases. The new scheme, hence, yields even better results with a large number of co-existing piconets.

Figure 9*b* depicts the PER performance of the new scheme for a higher traffic load of 70% ($\rho_0 = 0.3$). The PER is higher for all the three packet types with respect to the lighter traffic load of 30%. Even at heavier loads, the synchronisation scheme outperforms unsynchronised transmissions.

Figure 10 reports aggregate network throughput as a function of the number of interfering piconets. In Fig. 10*a*, the traffic load is chosen as 30% ($\rho_0 = 0.7$). Two different arrival models are considered here. The ratio $\rho_1 : \rho_3 : \rho_5$ is set to 3 : 2 : 1 to reflect the case of more shorter packets in each piconet. Similarly, the ratio $\rho_1 : \rho_3 : \rho_5 = 1 : 2 : 3$ depicts the case of more longer packets. The graph shows that higher throughput is achieved with more longer packets. This is because, though longer packets are more vulnerable to collisions, they carry more data bits per slot ($R_5 >> R_1$). Also, the aggregate throughput reaches a maximum for a certain value of N and then drops as N increases. Synchronised transmissions generate approximately 25% more aggregate throughput for 100 interfering piconets. This improvement increases with increasing value of N. Also, the peak aggregate throughput with the new scheme is reached with a greater number of interfering piconets as compared to the unsynchronised case.

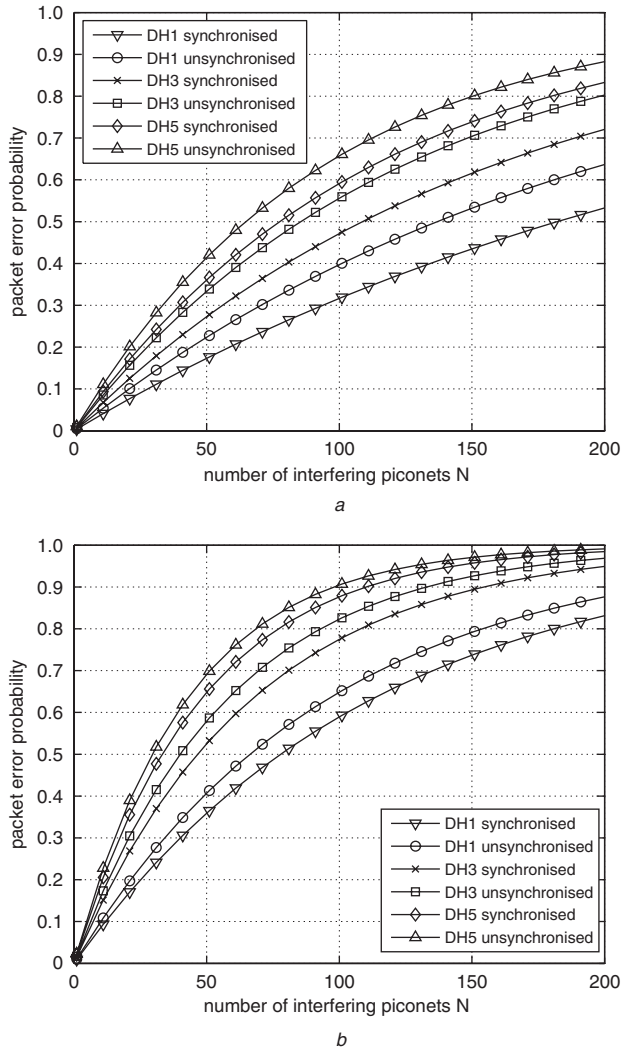

Fig. 9 *PER comparison of synchronised and unsynchronised transmissions*
a 30% traffic load
b 70% traffic load

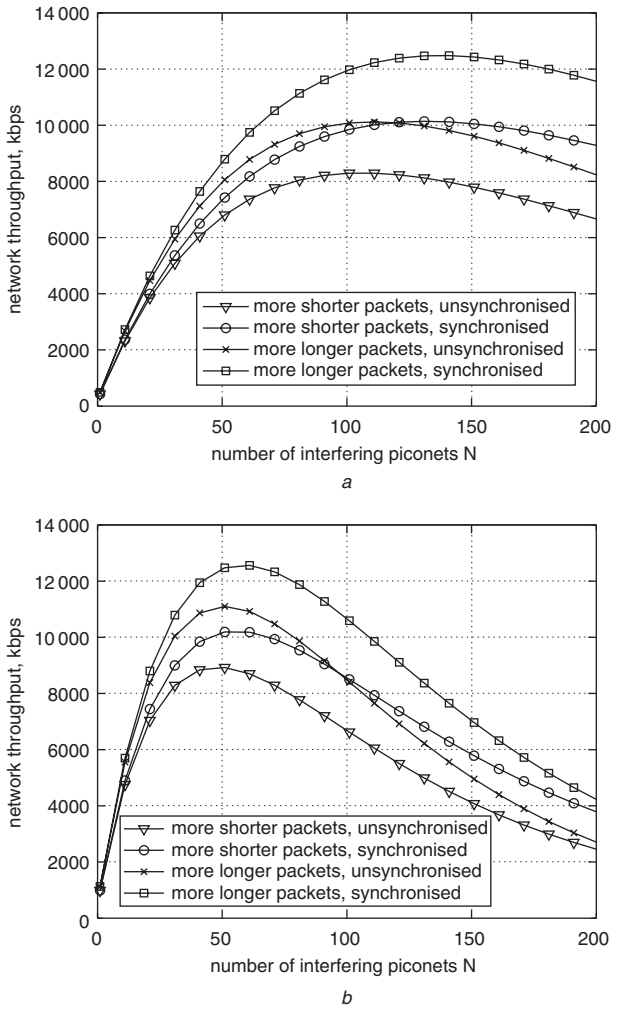

Fig. 10 *Aggregate network throughput comparison of synchronised and unsynchronised transmissions*
a 30% traffic load
b 70% traffic load

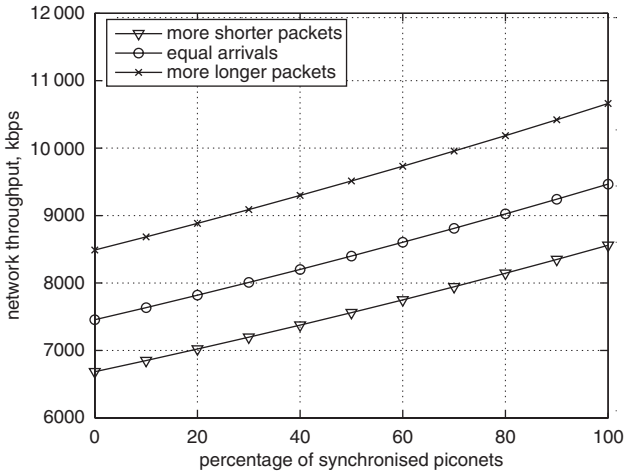

Fig. 11 *Throughput improvement with the new synchronisation scheme ($N = 100$, 70% uniform traffic)*

In Fig. 6, we have shown the synchronisation time as a function of N. However, at a certain time instant, the whole network may still not be completely synchronised. The effect of this partial synchronisation is illustrated in Fig. 11, where the network throughput is plotted in dependency of percentage of synchronised piconets, with N and ρ_0 fixed at values of 100 and 0.3, respectively. It shows that 100% synchronisation in the network can offer a throughput improvement of approximately 26%.

6 Conclusions

A new synchronisation scheme for multi-piconet Bluetooth environments has been presented. Details of the synchronisation process are described and an analytical model of the scheme is also presented. The objective of the synchronisation architecture is to increase both individual piconet's and overall network performance, yet at the same time induce minimal overhead on the collocated piconets. It is shown that the new scheme is resilient to the dynamics of the network topology where users are entering and exiting the system constantly. Furthermore, performance of the scheme is measured by formulating quantitative metrics such as PER, individual piconet throughput and aggregate network throughput. The analytical model presented is a general approach and includes different standard Bluetooth packet types. The results indicate that the scheme outperforms the current unsynchronised Bluetooth transmissions, under varying traffic load conditions. As an example, the scheme offers 26% network throughput improvement in the presence of 100 co-existing piconets.

7 Acknowledgment

The work reported in this paper has formed part of the Wireless Enablers area of the Core 3 Research Programme of the Virtual Centre of Excellence in Mobile & Personal Communications, Mobile VCE, http://www.mobilevce.com, whose funding support is gratefully acknowledged.

8 References

1 'Bluetooth Core Specification v1.1', http://www.bluetooth.com
2 'Bluetooth Core Specification v1.2', http://www.bluetooth.com
3 Zürbes, S., Stahl, W., Matheus, K., and Haartsen, J.: 'Radio network performance of Bluetooth'. Proc. IEEE ICC, New Orleans, LA, USA, Jun 2000, pp. 1563–1567
4 Zürbes, S.: 'Considerations on link and systems throughput of Bluetooth networks'. Proc. IEEE PIMRC, London, UK, Sept. 2000, pp. 1315–1319
5 El-Hoiydi, A.: 'Interference between Bluetooth networks – upper bound on the packet error rate', *IEEE Commun. Lett.*, 2001, **5**, pp. 245–247
6 Lin, T.-Y., and Tseng, Y.-C.: 'Collision analysis for a multi-Bluetooth picocells environment', *IEEE Commun. Lett.*, 2003, **7**, pp. 475–477
7 Lin, T.-Y., Liu, Y.-K., and Tseng, Y.-C.: 'An improved packet collision analysis for multi-Bluetooth piconets considering frequency hopping guard time effect', *IEEE J. Sel. Areas Commun.*, 2004, **22**, pp. 2087–2094
8 Howitt, I.: 'WLAN and WPAN coexistence in UL band', *IEEE Trans. Veh. Technol.*, 2001, **50**, pp. 1114–1124
9 Howitt, I.: 'Bluetooth performance in the presence of 802.11b WLAN', *IEEE Trans. Veh. Technol.*, 2002, **51**, pp. 1640–1651
10 Conti, A., Dardari, D., and Andrisano, G.: 'Bluetooth and IEEE 802.11b coexistence: analytical perfomance evaluation in fading channels', *IEEE J. Sel. Areas Commun.*, 2003, **21**, pp. 259–269
11 Ophir, L., Bitran, Y., and Sherman, I.: 'Wi-Fi (IEEE 802.11) and Bluetooth co-existence: issues and solutions'. Proc. IEEE PIMRC, Barcelona, Spain, Sep 2004
12 Treister, B., Chen, K.C., and Batra, A.: 'Clause 14.3 adaptive frequency hopping'. Technical Report IEEE P802.15-TG2-366rl, Mar 2003
13 'Bluetooth Core Specification v2.0', http://www.bluetooth.com
14 Lin, T.-Y., Liu, Y.-K., and Tseng, Y.-C.: 'An improved packet collision analysis for multi-Bluetooth piconets considering frequency hopping guard time effect'. Proc. IEEE VTC Fall, Florida, USA, Oct 2003