Interconnecting Bluetooth-like Personal Area Networks

Godfrey Tan GODFREYT@MIT.EDU
MIT Laboratory of Computer Science, 200 Technology Square, Cambridge MA, 02139 USA

1. Introduction

The ubiquitous use of information intensive consumer devices such as cell phones, personal digital assistants (PDAs), and laptop computers calls for a new networking paradigm for interconnecting them. The goal is to create a personal area network (PAN) that accommodates seamless information transfer between different devices with varying capacity in an *ad hoc* manner without the need for manual configuration, cables, or wired infrastructure.

An industry consortium has recently standardized Version 1 of a short-range, low-power radio frequency (RF) technology called *Bluetooth*, motivated by the need for suitable link-layer PAN technologies (Haartsen, 2000). The design of Bluetooth's MAC protocol is based not on distributed contention resolution as in traditional wireless LANs, but on a centralized master-slave mechanism. Bluetooth, operating in the 2.4 GHz frequency band, employs a pseudo-random frequency-hopping scheme, where a device transmits on a single frequency for 625μs before hopping to the next frequency. This use of frequency hopping allows multiple concurrent Bluetooth communications within radio range of each other, without adverse effects due to interference.

A Bluetooth *piconet* consists of one master and up to seven slaves. The master allocates transmission time slots (and therefore, channel bandwidth) to the slaves in the piconet. A slave transmits data only if the previous time slot contained a message from the master destined for it. This scheme, called *time-division duplex (TDD)*, is at the heart of Bluetooth's MAC protocol.

Because frequency-hopping facilitates high densities of communicating devices, it is possible for dozens of piconets to co-exist and independently communicate in close proximity without significant performance degradation. The Bluetooth specification alludes to the possibility of internetworking multiple piconets, calling it a *scatternet*, but does not specify how it is to be done.

We identify the three main challenges in interconnecting multiple Bluetooth-like PANs: i) scatternet topology formation, ii) packet routing, and iii) channel or link scheduling. The need for an explicit topology formation process stems from the fact that devices need to discover each other and explicitly establish a point-to-point link to synchronize the frequency hopping sequence and exchange signaling information. Once the scatternet is formed, some mechanism is required to efficiently route packets across multiple PANs. The scheduling problem arises because we would like to use channel bandwidth efficiently, and the TDD nature introduces problems not seen in traditional wireless channel scheduling. In the later sections, we present novel approaches to solve each problem.

2. Scatternet Formation

Our topology formation algorithm assigns unique addresses to nodes while connecting them in a tree structure that minimizes the number of links or channels required to form a connected scatternet. Our algorithm is both decentralized and self-healing, in that nodes can join and leave at any time without causing long disruptions in connectivity. It does not use broadcasts, which are an expensive primitive in many wireless PANs because they consume significant amounts of energy and waste useful bandwidth.

Our algorithm combines the routing strategy with the address allocation mechanism in such a way that packets are routed in a loop-free manner that does not incur any per-packet overhead (in contrast to source-routing approaches), per-node state (in contrast to on-demand protocols), or periodic routing messages (as in most other routing protocols). These attributes make our approach a good match for all PAN networks, where simplicity and energy-efficiency are important considerations.

Our algorithm constructs a tree incrementally, one node at a time. When a node wishes to join the network, it sends out frequent *search* announcements. Nodes that already belong to the scatternet periodically listen on a pre-defined channel for these announcements and respond if they are willing to accept a new neighbor. When there are more than one nodes responding to the new node, a decision must be made on where a new node should join. This decision can be made by the new node based on the responses it hears or by the root. The root can gather the information from all the child nodes which hear the *search* messages and choose

which one to respond to the searching node. We use this option due to Bluetooth specific limitations.

When a new node connects to a node in the scatternet, the latter becomes a *parent* and the former its *child*. Each connected node has an N-bit address and a portion of the entire 2^N-bit address space that it can delegate to children that join it. Our design uses $N = 16$ bits for the address space allowing the largest scatternet of $65,536$ nodes. The parent splits its current address space in two, delegates one part of it to the new child, and updates its current address space to correspond to the other half it is still in charge of. The new child and parent form a communication link, and the child is now a member of the scatternet. Like other nodes in the scatternet, the child now periodically listens for solicitation from other new nodes, responding (if appropriate) by accepting children of its own.

An example of the evolution of the scatternet topology using this method as new nodes join is shown in Figure 1. Each node in the figure is labeled with two quantities: its address (in italics), and the portion of the address space that corresponds to the subtree rooted at the node (denoted by a prefix). In the figure, the notation b^k denotes a string of k b's, where $b = 0$ or 1. Although not shown, each node maintains a small amount of per-child-link state corresponding to the portion of the address space allocated to each child.

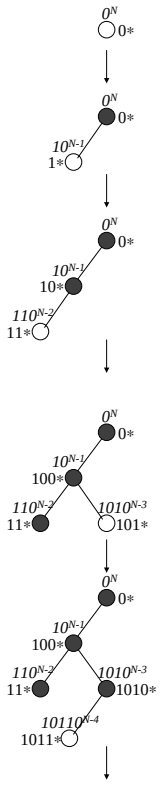

Figure 1. Evolution of the scatternet topology. The italicized label is the node's address in the scatternet, while the other one is its allocatable address space. The most recent node is unshaded.

3. Packets Relaying

By carefully assigning addresses, our topology construction mechanism has simplified packet forwarding and avoids the need for a routing protocol. The reason for this can be seen by considering an example from the last tree (at the bottom) in Figure 1. When a packet from node 10110^{N-4} is destined for node 110^{N-2}, a longest-prefix match is done at the node, similar to an IP forwarding table lookup. Since the destination address is not within any child's address space, the packet is simply pushed upwards to the parent, node 1010^{N-3}, where the same operations are performed. Once again, this causes the packet to be pushed upwards to node 10^{N-1} where the destination is found as one of its children. It is easy to see that our addressing strategy provides the guarantee that any packet will reach its destination in the scatternet.

4. Channel Scheduling

An efficient scatternet-wide scheduling mechanism is necessary since relay nodes such as node 10^{N-1} communicate in several channels on a Time Division Multiplex basis. This problem is similar to the maximal matching problem for bi-partite graphs. Our choice of tree topology simplifies the problem. We have developed a simple distributed scheduling scheme to reduce the message latency assuming uniform workload distribution. Our algorithm achieves an upper bound of $\lceil \frac{d}{2} \rceil \times MaxDegree$ on the latency (in time slots) between any two nodes, where $MaxDegree$ is the maximum degree in the entire tree, and d is the distance in hops between source and destination.

5. Discussion

We have presented a set of online algorithms for interconnecting multiple piconets. At the core of our algorithms is the distributed construction of scatternet as a tree. We then build efficient schemes for packet relaying and channel scheduling. We have implemented our algorithms in the network simulator *ns-2* and the simulations results show good performance. We soon plan to build a complete system using real hardware.

References

Haartsen, J. (2000). The Bluetooth Radio System. *IEEE Personal Communications Magazine*, 28–36.