Multihop Scatternet Formation and Routing for Large Scale Bluetooth Networks

Abstract: This paper addresses the scatternet formation for large scale multi-hop Bluetooth networks. We first describe a novel communication efficient method to build a connected dominating set (CDS) as the backbone of multi-hop Bluetooth network, then propose new algorithms to form the *dBBlue* scatternets [1] in each cluster. Notice that our methods is not a simple combination of existing CDS methods and dBBlue algorithm. We propose a new algorithm for finding the connectors in CDS that are more suitable for multihop Bluetooth networks and several new algorithms to enable formation of dBBlue scatternets inside each cluster. The final scatternet, *M-dBBlue*, guarantees the connectivity. Our experiment shows the majority of nodes have degree smaller or equal to 7 which means our scatternet seldom parks any node. Our scatternet also enjoys efficient updating, since both the backbone and the dBBlue structure of each cluster can be maintained efficiently in a dynamic environment. We then propose a complete set of hierarchical routing methods for M-dBBlue which enables the self-routing inside each cluster. Moreover, our scatternet formation and routing algorithm do not require any position information at all.

Keywords: Bluetooth, scatternet formation, multi-hop, dBBlue, connected dominating set

1 Introduction

Wireless ad hoc networking has gathered significant research interests in the past years. Bluetooth [2] is a promising low cost and low power wireless technology, which enables portable devices to form short-range wireless ad hoc networks based on a frequency hopping physical layer. Bluetooth operates in the unlicensed 2.4GHz ISM band, with the frequency hopping technique to alleviate the effects of the interference. The nominal bit rate of transmission is 1Mbps. It has been widely predicted that Bluetooth will be the major technology for short range wireless networks and wireless personal area networks. This paper deals with the problem of building multi-hop ad hoc networks using Bluetooth technology.

Bluetooth ad-hoc networking presents new technical challenges, such as scheduling, network forming and routing. According to the Bluetooth specification 2.0 [2], when two Bluetooth devices come into each other's communication range, one of them assumes the role of *master* of the communication and the other becomes the *slave*. This simple one hop network is called a *piconet*, and may include more slaves. The network topology produced by the connection of piconets is called a *scatternet*. There is no limit on the maximum number of slaves connected to one master, although the number of active slaves at one time cannot exceed 7. If a master node has more than 7 slaves, some slaves must be parked. To communicate with a parked slave, a master has to *unpark* it, thus possibly parking another active slave instead. The standard also allows multiple roles for the same device: a node can be master in one piconet and a slave in one or more other piconets. However, one node can be active only in one piconet. To operate as a member of another piconet, a node has to switch to the hopping frequency sequence of the other piconet. Since each switch causes delay (e.g., scheduling and synchronization time), an efficient scatternet formation protocol can be one that minimizes the roles assigned to the nodes, without losing network connectivity.

While several solutions and commercial products have been introduced for one-hop Bluetooth communication, the Bluetooth specification [2] does not indicate any method for multihop scatternet formation. The problem of scatternet formation has attracted many attention with in past several years. Several criteria could be set as the objectives in forming scatternet. For example, the node degree of the scatternet should be small, if degrees of all nodes are smaller than 8, the scatternet can avoid parking any node. The formation and maintenance of scatternet should have small communication overhead so that it can be efficiently updated in dynamic networks. In this paper, we focus on scatternet formation for *large* multi-hop ad hoc networks.

Previous literature on scatternet formation assumed that devices are not able to communicate unless they have previously discovered each other by synchronizing their frequency hopping patterns. Thus, even if all nodes are within direct communication range of each other, only those nodes, which are synchronized with the transmitter, can hear the transmission. Synchronizing the frequency hopping patterns is apparently a time consuming and pseudo-random process [3]. In this paper we assume that the problem of discovering all neighbors within transmission radius of a device is resolved by separate Bluetooth protocols [3; 4]. These protocols are applicable as the pre-phase of our scheme.

This paper addresses the scatternet formation solutions in large multi-hop Bluetooth networks. Our method is based on a hierarchical structure to guarantee the network connectivity and provide efficient routing. We first propose a novel communication efficient method to cluster the nodes, build a connected dominating set (CDS) as the backbone of multi-hop Bluetooth

network, and then construct the *dBBlue* scatternets [1] for each cluster, which makes self-routing within the cluster possible. A cluster is defined by a dominator node and all its dominatee nodes. We propose several methods to construct the piconets on top of the backbone to provide high performance. The final scatternet, hereafter called *M-dBBlue*, guarantees the connectivity, and its node degree is small in most cases. Later our experiment shows most nodes have degree smaller or equal to 7. Our scatternet can be quickly updated in mobile networks since both CDS and dBBlue structure can be maintained efficiently. Moverover, all of our methods do not require any position information of the nodes.

We then discuss how to route packets efficiently in our scatternet using a hierarchical routing method. When two nodes want to communicate, the source node first sends data packets to the dominator node within its cluster using a self-routing method supported by the dBBlue structure; the dominator node then forwards the data packets to the dominator node whose cluster contains the target node; the target dominator node then forwards the data to the target node using self-routing method provided by dBBlue scatternet. Here the routing along the backbone could be any greedy methods [5; 6] if position information is known, or source-initiated on-demand routing protocols such as AODV [7], DSR [8], or table driven protocols such as DSDV [9].

In summary, the contributions of this paper are as follows:

- We propose a new multihop scatternet formation algorithms which first builds a CDS as the backbone then forms the dBBlue scatternets inside each cluster.
- For the CDS formation algorithm, we use the method in [13] to select the dominators and form clusters, and then propose a new efficient algorithm (Algorithm 3) to find connectors to form the connected backbone. Our new connector algorithm generates more connections than the spanning tree so that it enhances efficiency of inter-cluster routing. Different from all previous methods, we do not use local broadcast in our method, since local broadcast cannot be performed efficiently in practice due to the constraint in MAC layer.
- When we form the dBBlue scatternets inside each cluster, the method in [1] *cannot* be applied directly since it only works for single-hop networks and the nodes in each cluster may not be single-hop. We propose two novel methods to solve the problems: (1) decreasing the transmission radius to half so that all dominatees of a dominator can communicate directly; (2) partition the dominattees into cliques such that each cliques is single-hop. In addition, we also propose two methods to build the one-hop scatternet dBBlue inside each clique. They applies different role assignments for the dominators and connectors in the backbone to achieve some nice properties. The final scatternet, *M-dBBlue*, guarantees the connectivity, and its node degree is small in most cases.
- We also describe how to perform IP-based routing in the M-dBBlue scatternet. The inter-cluster routing is handled by a modified Bluetooth based RIP protocol on the backbone, and the intra-cluster routing is derived from the self-routing mechanism of dBBlue.

The rest of the paper is organized as follows. In Section 2, we introduce our network model and review two algorithms which will be used in the proposed scatternet formation method. In Section 3, we describe a novel multihop scatternet formation algorithm, which integrates the CDS-based backbone and dBBlue structure together seamlessly, and hence enjoys many nice properties. In Section 4, we discuss in detail the IP-based routing solution for M-dBBlue scatternet. We evaluate our structures by simulation in Section 5. Some related works are reviewed in Section 6. Finally, section 7concludes the paper.

2 Preliminary

In Bluetooth specification 2.0 [2], piconet formation is performed in two phases: neighbor discovery and link establishment. The neighbor discovery phase is accomplished by the inquiry handshake procedures, implemented by the *inquiry* and *inquiry scan* command. Once two neighboring devices complete an inquiry handshake, only the node in *inquiry* mode knows the ID and weight information of the node in *inquiry scan* mode, not vice versa. To get the mutual information of each other, two neighboring nodes may set up a temporary piconet that lasts only the time necessary to exchange their ID and other useful information [10]. In the link establishment phase, each piconet is formed by one master and limited number of slaves, and each node decides its role locally based on the neighboring information gathered in the neighbor discovery phase. The link establishment phase is achieved by the page handshake procedure, performed by the *page* and *page scan* commands. Once two neighboring nodes complete a page handshake successfully, the node in *page* mode assumes *master* role, while the other node in *page scan* mode assumes *slave* role.

In this paper, for simplicity, we will not use these engineering terms to describe our scatternet formation algorithms in detail. Please refer to [10; 4] for more details of engineering aspects. We assume that the problem of discovering all neighbors within transmission radius of a device has been resolved by separate Bluetooth protocols, and any two neighboring nodes can communicate directly to exchange information. Our goal is focusing on the topology formation with all preferred properties in Bluetooth specification [2], such as bounded node degree and zero role switch. In this section, we will discuss the network model and review two algorithms, which form the ground for further discussion of the proposed approach.

2.1 Network Model - a Graph Model

We assume that all devices have the same maximum transmission range r. Thus, the set of Bluetooth devices V define a unit disk graph (UDG) $G(V, r)$, in which two nodes are connected if and only if their distance is no more than the maximum transmission range r. Notice that we only assume that the underlying global communication graph is a UDG. The position information of the node is never used in our method. Given a subset U

of V, graph $G_k(U)$ has an edge xy if and only if nodes x and y are at most k-hops away in the original communication graph G.

2.2 Connected Dominating Set - the Virtual Backbone

Connected dominating set (CDS) has been used as the virtual backbone for wireless networks. In our proposed scatternet formation method, we will first form the clusters and build the connected dominating set as the backbone of M-dBBlue scatternet. Assume V is the set of all nodes (Bluetooth devices). A subset S of V is a *dominating set* if any node u in V is either in S or is adjacent to some node in S. Nodes in S are called dominators, while nodes in $V - S$ are called dominatees. A cluster is defined by a dominator node and all its dominatee nodes. Once some nodes, hereafter called *connectors*, are selected to form a connected graph together with dominators, the final structure is called *connected dominating set*. Two dominators are said to be *adjacent* if they are within *two* hops of each other.

Several efficient methods [11; 12; 13; 14] have been proposed to construct a connected dominating set, i.e., the backbone, of the network modelled by UDG. In this paper, we first use the method given in [13] to find a dominating set, then we design a novel communication efficient method to find a connector for each pair of dominators within 2 hops based on unicast communication only (unlike the previous methods that use broadcast communication model). The generated CDS backbone is guaranteed to be connected.

For completeness, we briefly review the method in [13] here with our own interpretation. Their method uses a carefully chosen rank definition. The ranking of nodes is induced by an arbitrary spanning tree T rooted at a leader. The message complexity of their method is $O(n)$ if a leader is already known and $O(n \log n)$ if leader election is needed. Given a rooted spanning tree, the *level* of a node is the number of hops in T between itself and the root of T. The *rank* of a node is then given by the ordered pair *(level, ID)*, and such ranking gives rise to a total ordering of the nodes in the lexicographic order. After each node knows the rank of its own and all its neighbors, the algorithm [13] first finds dominators by a color-marking process. All nodes are initially marked with *white* color and will be marked with either *gray* or *black* eventually. A node marked with *black* will become dominator eventually. Each node maintains two local variables: *pendingChildrenNum* and *pendingLowNbNum*. Variable *pendingChildrenNum* counts the number of children who have not reported the completion and is thus initialized to the number of children in the tree. Variable *pendingLowNbNum* stores the number of lower-ranked neighbors who have not reported the status. Each node also maintains a *blackList* that records the IDs of its black neighbors. The detailed method is given in Algorithm 1.

Let U be the set of dominators constructed by Algorithm 1. It was shown that $G_2(U)$ is a connected graph [13; 11]. Based on this, they gave a communication efficient method to build a tree spanning all dominators as the final connected dominating set. They also showed that the number of dominators within two hops of a dominator is at most 24. Since the graph $G_2(U)$ is connected, we only need to select one dominatee node as connector (or gateway) to connect two dominators separated by two hops. Consequently, every connector node is connected only to dominators, which implies that each connector node has degree at most 5. To minimize the number of connectors, they build a spanning tree of dominators as the connected dominating set. Thus, in their method, two adjacent dominators are not necessarily connected by a connector. In the worst case, two adjacent dominators may be connected by a path with $O(k)$ hops, where k is the number of dominators found. However, in our multi-hop scatternet formation method, we prefer that the backbone keeps more connections than the spanning tree so that this can enhance the efficiency of inter-cluster routing. Thus, we design a new communication efficient method (in Section 3.1) to find a connector for each pair of dominators within 2 hops, instead of using the spanning tree. Figure 1 illustrates a backbone topology formed by our algorithm, in which each adjacent dominator pair is connected by exactly one connector.

Algorithm 1 Finding Dominators

Given the ranks of nodes defined by an arbitrary spanning tree T rooted at the access point, all nodes are initially marked with *white* color.

1: The root first marks itself black and broadcasts a BLACK message.
2: Upon receiving a BLACK message, a white node adds the sender's ID to *blackList*, then it marks itself gray and broadcasts a GRAY message which contains its *rank*.
3: Upon receiving a GRAY message, if the rank of the sender is lower than its own, a white node decreases *pendingLowNbNum* by one. If *pendingLowNbNum* becomes 0 after the update, it marks itself black and broadcasts a BLACK message. When a leaf node is marked with either gray or black, it transmits a COMPLETE message to its parent.
4: Upon receiving a COMPLETE message toward itself, a node decreases *pendingChildrenNum* by one. If *pendingChildrenNum* is 0 and it is not the root, it transmits a COMPLETE message to its parent.

By the time when local variable *pendingChildrenNum* at the root equals to 0, all nodes has been marked with either gray or black. All black nodes become the *dominators*.

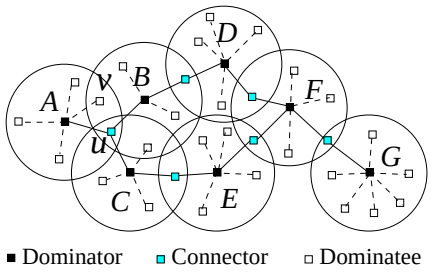


Figure 1: The disks represent the clusters. The solid lines connecting dominators, connectors; the dashed lines connect the dominatees and their dominators. The dominators (denoted by black nodes) and connectors (denoted by gray nodes) form the network backbone.

2.3 dBBlue Scatternet for Single-Hop Networks

Single-hop Bluetooth scatternet formation has been well studied in [15; 1; 3; 16]. In a single-hop bluetooth scatternet, all wire-

less devices are within the radio vicinity of each other. In [1], Song *et. al* adopt the well-known de Bruijn graph to build a self-routing scatternet, called dBBlue, with low-diameter $O(\log n)$ and bounded node-degrees. Each master has at most seven slaves and each slave node exists in at most two piconets, and no node assumes both master and slave roles. They also presented a scalable MAC assignment mechanism and a vigorous method to *locally* update the *dBBlue* structure using at most $O(\log n)$ communications when a node joins or leaves the network. The computation cost is $O(n)$ for static construction. Moreover, the congestion of every node is at most $O(\log n/n)$, assuming that a unit of total traffic demand is equally distributed among all pair of nodes. Since we will use dBBlue as the intra-cluster structure in the proposed mutli-hop scatternet formation method, for completeness of presentation, we now briefly review the dBBlue scatternet formation algorithm from [1].

The dBBlue scatternet construction is based on the well-known de Bruijn graph [17]. The de Bruijn graph, denoted by $B(d,k)$, is a directed graph with d^k nodes. Assume that each node is assigned a unique label of length k on the alphabet $\{0,\cdots,d-1\}$. There is an edge in $B(d,k)$ from a node with label $x_1x_2\cdots x_k$ to any node with label $x_2\cdots x_ky$, where $y\in\{0,\cdots,d-1\}$. It is well-known that the de Bruijn graph enables self-routing intrinsically. The self-routing path from the source with label $x_1x_2\cdots x_k$ to the target with label $y_1y_2\cdots y_k$ is $x_1x_2\cdots x_k\rightarrow x_2\cdots x_ky_1\rightarrow x_3\cdots x_ky_1y_2\rightarrow\cdots\rightarrow x_ky_1\cdots y_{k-1}\rightarrow y_1\cdots y_k$. Observe that, we could find a shorter route by looking for the longest sequence that is both a suffix of $x_1x_2\cdots x_k$ and a prefix of $y_1y_2\cdots y_k$. Suppose that $x_i\cdots x_k=y_1\cdots y_{k-i+1}$ is such a longest sequence. The shortest path between the source and the target is $x_1\cdots x_k\rightarrow x_2\cdots x_ky_{k-i+2}\rightarrow\cdots\rightarrow x_{i-1}\cdots x_ky_{k-i+2}\cdots y_{k-1}\rightarrow y_1\cdots y_k$. Clearly, the route between any two nodes is at most k hops, i.e., $B(d,k)$ has diameter $k=\log_d n$, where $n=d^k$ is the number of nodes of the graph.

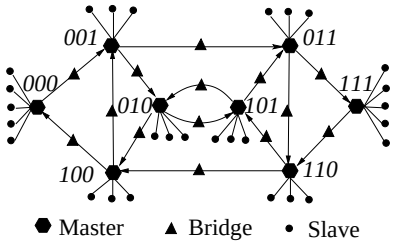


Figure 2: dBBlue scatternet with 48 nodes based on $B(2,3)$.

The classical de Bruijn graph is *balanced* in the sense that the labels of all nodes have the same length. The de Bruijn graph can be generalized to any set of vertices whose labels form a universal prefix set. In [18], Fraigniaud and Gauron proposed a novel method to construct an efficient topology for P2P network based on the generalized de Bruijn graph defined on a universal prefix set. *"A universal prefix set is a set S of labels on an alphabet Σ such that, for any infinite word $w\in\Sigma^\star$, there is a* unique *word in S, which is a prefix of w. The empty set is also a universal prefix set."*[18] For instance, $\{00,01,100,101,110,111\}$ is a universal prefix set on alphabet $\Sigma=\{0,1\}$, but $\{00,01,10\}$ and $\{00,01,100,1000,101,110,111\}$ are not. There is a directed edge from node $u=x_1x_2\cdots x_k$ to another node v in the generalized de Bruijn graph if $x_2\cdots x_k$ is the prefix of the label of node v. A generalized de Bruijn graph is *pseudo-balanced* if the lengths of the labels are different by at most one. For simplicity, we still denote a pseudo-balanced de Bruijn graph on alphabet $\{0,1\}$ by $B(2,k)$ if the node labels have length at least k bits and at most $k+1$ bits. We also say that a node from $B(2,k)$ is at level k if its label has k bits. In this paper, we only consider the balanced or pseudo-balanced binary de Bruin graph $B(2,m)$.

Algorithm 2 Static DeBruijn-Based Scatternet

1: Assume that there is a leader already among these n nodes. The leader could be the node with smallest ID. We give the *token* to the leader and call it token node. Token node randomly selects 2^m nodes (including itself) into the master set M which assumes the *master* role in final scatternet topology, where $2^{m-1}<\lceil\frac{n}{6}\rceil\leq 2^m$. Let $r=n-3\cdot 2^m$, which is the total number of nodes that can be assigned as pure slaves. Token node assigns itself with label 0^m, and each node in M with a unique m bits label in the range from $0\cdots01$ to $1\cdots11$. M forms a de Bruijn graph $B(2,m)$ as the scatternet backbone.
2: **repeat**
3: Token node, with label $x_1\cdots x_m$, selects 2 nodes from the remaining as its bridge slaves, and assigns them labels $(x_1\cdots x_m,010)$ and $(x_1\cdots x_m,101)$ respectively. Here 010, 101 will also serve as the Medium Access Code (MAC) for these two slaves in the piconet mastered by this token node. Token node uses its bridge slave node $(x_1\cdots x_m,010)$ to connect with its out-neighbor $x_2x_3\cdots x_m0$ and the bridge slave node $(x_1\cdots x_m,101)$ to connect the out-neighbor node $x_2x_3\cdots x_m1$. {There are two special nodes 0^m and 1^m, which only have 1 out-neighbor, we then just use one bridge slave node to connect with its out-neighbor.}
4: The token node selects $t=\min\{3,r\}$ nodes from the remaining as its slaves and assigns them with labels $(x_1\cdots x_m,001)$, $(x_1\cdots x_m,100)$ and $(x_1\cdots x_m,110)$ in the order if they exist. Let $r_i=r_{i-1}-t$. {Node 0^m and 1^m may choose 5 nodes as its pure slaves since they only have one in-neighbor and one out-neighbor.}
5: The token node passes the *token* to its successor.
6: **until** all nodes in M are processed
7: After all nodes have been processed, the token will be passed back to node 0^m again.

The method [1] constructs a balanced de Bruijn graph $B(2,m)$ as the initial backbone of the network. It chooses integer m such that $2^{m-1}<\lceil\frac{n}{6}\rceil\leq 2^m$. The choosing of m guarantees that there are enough bridge slave nodes, which implies that no master node serves as bridge slave. The detailed method is given in Algorithm 2. Once the initial topology construction is finished, the token node t will be responsible for following node joining and leaving issues. Master nodes and bridges form the backbone of Bluetooth scatternet, and a piconet works like a node in de Bruijn graph. Figure 2 illustrates a dBBlue scatternet containing 48 nodes based on $B(2,3)$ graph. Notice that dBBlue scatternet does not work for multihop networks, since it requires every node can be connected to other nodes.

3 M-dBBlue Scatternet Formation

Our M-dBBlue scatternet formation algorithms for multihop Bluetooth networks first builds a connected dominating set (CDS) as the backbone of multi-hop Bluetooth networks, then constructs the *dBBlue* scatternets in each cluster.

3.1 Backbone Construction for M-dBBlue Scatternet

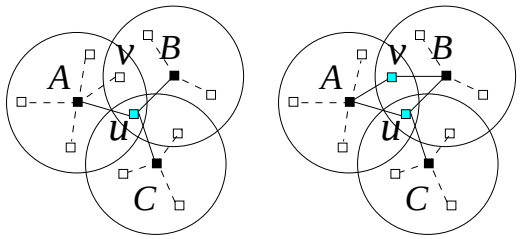
In the proposed method, we first use the method given in [13] to find a dominating set, then adopts a new communication efficient method to find a connector for each pair of dominators within 2 hops based on unicast communication only (unlike the previous methods that use broadcast communication model). The method in [13] has been briefly introduced in the section 2. The CDS backbone is guaranteed to be connected. Similar to [11; 13], we can show that the node degree of the connected dominating sets is bounded by a constant, and the hops and length stretch factor are bounded by small constants. In addition, the number of dominators is at most a small constant factor of the minimum number of dominators.

In [13], they used the spanning tree of dominators to select the connector and form the connected dominating set. However, in our M-dBBlue network, we prefer that the backbone keeps more connections than the spanning tree to enhance the efficiency of inter-cluster routing. Notice that Algorithm 1 finds a set of dominators with the following property: the backbone by connecting each pair of dominators separated by two hops is connected. Thus, we only need to find connector to connect *any* pair of 2-hop adjacent dominators to form the connected backbone. We try to minimize the number of connectors used while keeping at least one connector for each pair of adjacent dominators. Therefore, a connector could be used to connect many pairs of dominators in our method. To find a connector for each pair of dominators is not our innovation, several methods [19; 11] have been proposed. However, in all previous methods [13; 11; 19], they adopt the broadcast communication model to build CDS graph. It is well known that local broadcast cannot be performed efficiently in practice, due to the constraint in MAC layer that simultaneous broadcast by dominatees could cause massive signal interference which causes large latency. In Algorithm 3, we actually reduce the communication cost significantly by using unicast instead of broadcast.

In our algorithm, each dominator maintains two lists: *adjacentDominatorList* and *dominateeList*, which are initially empty. Here *adjacentDominatorList* records all adjacent dominators of this node, in addition, the connection flag is set for each pair of adjacent dominators acknowledged by a connector; *dominateeList* records all dominatees dominated by this node, which is reserved for dBBlue scatternet construction as will be seen later. Notice that, our method not only generates a CDS-based backbone, but also splits all nodes into separated clusters. Each dominatee also maintains two lists: *blackList* and *neighborDominatorList*. The *blackList* is generated in Algorithm 1, which records the known dominator neighbors of the dominatee. The *neighborDominatorList* stores the dominator neighbors which need be connected by itself, if this node is a connector. The detailed algorithm is given by Algorithm 3.

Algorithm 3 Finding Connectors

1: Each dominatee selects the neighboring dominator with the smallest ID in *blackList* and sends it a TRYCONNECTOR message, which includes *blackList*.
2: Once a dominator gets the TRYCONNECTOR message from a dominatee node, it first adds the sender to its *dominateeList*, then performs the following two steps:
(a) Adds those unknown adjacent dominators (if exist) in *blackList* from the message into *adjacentDominatorList*.
(b) Sets connection flag for each new pair of adjacent dominators (if exist) acknowledged by the sender. Simultaneously, generates a *confirmList* including all new pairs of adjacent dominators that will require this dominatee node to connect.
Finally, if *confirmList* is non-empty, it confirms the sender with a CONFIRMCONNECTOR message, which includes the *confirmList*.
3: Once a dominatee gets the CONFIRMCONNECTOR message, it will copy *confirmList* to its *neighborDominatorList*, and announces itself as connector by sending all dominators (except the sender) in the *confirmList* a IAMCONNECTOR message including the *neighborDominatorList*.
4: Once a dominator gets the IAMCONNECTOR message from a connector, it simply adds all unknown adjacent dominators in the *neighborDominatorList* of the message to its *adjacentDominatorList* and sets flag for each unknown adjacent pairs if necessary.



(a) u first reaches A. (b) v first reaches A.

Figure 3: Finding connectors with minimal communication.

It is easy to show that each 2-hop adjacent dominator pair is connected, since the connection will be acknowledged by some dominatee for sure in the algorithm. Figure 1 illustrates a backbone topology formed by our algorithm, in which each adjacent dominator pair is connected by exactly one connector. One connector could be used to connect several dominators. For instance, node u serves connector role among three dominators A, B and C, hence node v will not be selected as dominators.

Figure 3 illustrates the procedure of finding connectors. Node A, B and C are three dominators, we assume that $ID(A) < ID(B) < ID(C)$. Both node u and v are dominatees which have $blackList_u = \{A, B, C\}$ and $blackList_v = \{A, B\}$ respectively. Since node A has the smallest ID in their *blackList*, both u and v send a TRYCONNECTOR message only to dominator A. There are two cases here:

1. In Figure 3(a), the message from u first reaches A. Dominator A will select u as connector to connect B and C, then the message from v will be discarded since it is redundant.

2. In Figure 3(b), the message from v first reaches A. Domi-

nator A will select v as connector to connect B. When the message from u arrives, node u will be asked to connect with node C, in addition, it will also be asked to connect B and C because their connection flag has not been set.

The following lemma which bounds the number of dominators within k units from a node v is proved in [13; 11; 19] by using a simple area argument.

Lemma 1. *For every node v, the number of dominator nodes inside the disk centered at v with radius k-units is bounded by a constant $\ell_k = (2k+1)^2$ for $k > 1$ and $\ell_1 = 5$.*

The bounds on ℓ_k can be improved by a tighter analysis. Therefore, as [13; 11; 19], we can prove the following theorem.

Theorem 2. *In the CDS built by Algorithm 1 and Algorithm 3, node degree is bounded by a constant.*

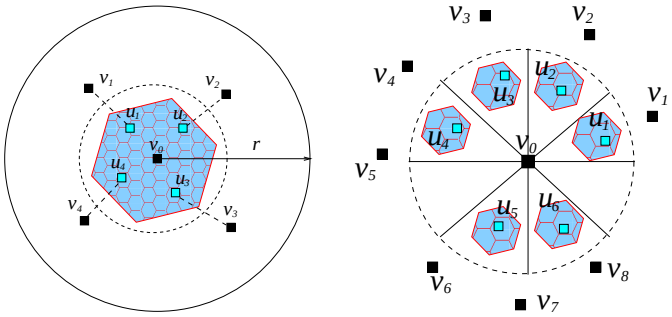
Proof. For a dominator u, the number of neighboring connectors in the CDS is bounded by 24, since the number of dominators within two hops of u is bounded by $\ell_2 - 1 = 24$ (remember that u is a dominator). For a connector v, the number of its neighboring dominator is at most 5. □

However, the node degrees of domiatees who are not in the CDS are not bounded. In other words, in a cluster the number of dominatees dominated by a dominator could be very large. Thus in the next section, we will use dBBlue structure to further form scatternet inside each cluster.

3.2 dBBlue Scatternet Formation in Clusters

Forming the CDS and clusters provides the backbone and the globe topology for our M-dBBlue scatternet, in this section, we will study how to form the scatternet inside each cluster, i.e, how to assign the roles for each node and form the piconets. After building the CDS, a cluster may have many nodes (dominatees), and it is inefficient to connect all dominatees to the dominator. The node degree is preferred to be bounded by a constant number, where *seven* is the best match to Bluetooth specifications. In this section, we will continue to describe our scatternet formation approach in clusters based on the dBBlue protocol. The dBBlue protocol only works for single-hop Bluetooth network, where each device can directly communicate to all other devices. Unfortunately, each cluster (composed of a dominator node and all its dominatees) may*not* be a single-hop Bluetooth network, i.e., some dominatee pairs could not communicate directly at all. Thus, the single-hop scatternet formation algorithm can not be applied here directly. There are two possible solutions here: (i) make sure that all dominatees of a dominator node can communicate directly. (ii) partition the dominatee nodes into groups (cliques) such that the dominatee nodes of each group can communicate directly. For all nodes in a cluster, we propose two methods to build a dBBlue scatternet, which will be discussed in detail later. Figure 4 illustrates the idea of applying the dBBlue scatternet formation protocol directly to each cluster.

The first approach (Figure 4 (a)) is based on the following observation: if all dominatee nodes lie inside the disk centered at the dominator with radius equal to half of the transmission



(a) Half the transmission range (b) Divide nodes into cliques

Figure 4: Two approaches to partition the network to clusters, such that any two nodes in each cluster can talk with each other directly. (a) All nodes use half-radius to build CDS; (b) Divide the dominatee nodes into cliques.

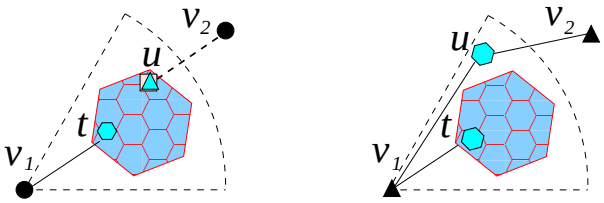
range, then all dominatee nodes are guaranteed to be able to communicate directly, i.e., being one-hop. Then, in the backbone construction phase, we may set every Bluetooth node in power-saving mode so that each node decreases its transmission radius to half of the maximum. To make sure that the backbone constructed this way is still connected, we need that the communication graph $G(V, r/2)$ be connected. In other words, if each node in the wireless ad hoc network decreases its transmission radius to $r/2$, the network is still connected. Once the modified backbone is built, every node returns to normal operation mode with transmission radius equal to r. Consequently, each cluster is a complete graph.

This approach is straightforward but need strong requirement of the network to ensure the connectivity of the backbone. Practically, the condition that $G(V, r/2)$ is connected could be satisfied frequently. Nonetheless, the second solution can be applied in general. In the remaining of the paper, we always assume the scatternet formation and routing is based on the second approach. All following mechanisms can be easily transformed to support the first approach.

The second approach (Figure 4 (b)) is using some algorithms such as those in [20; 21; 22; 23] to partition the dominatee nodes into cliques (groups such that the dominatee nodes of each group can communicate directly). If the position information is available for each node, a simple and efficient method to divide into cliques is as follows. We can divide the transmission region of the dominator node into six equal-sized cones (with degree $\pi/3$) and notice that all the dominatee nodes from a cone are guaranteed to form a clique. By this way, we may divide a cluster into at most 6 cliques.

We now continue the one-hop scatternet formation in each clique and present two algorithms to build a dBBlue scatternet for each clique. In both algorithms, we will let the dominator nodes be slaves of piconets. Notice that this approach is different from all previous scatternet formation methods based on the connected dominating set, in which the dominator is naturally assigned master role instead. We will show that assigning dominator slave roles actually produces scatternet with several nice properties.

In our first method, we build a dBBlue scatternet using all



(a) Illustration of Algorithm 4 (b) Illustration of Algorithm 5

Figure 5: (a) All nodes in each cone of unit-radius cluster form a dBBlue scatternet (b) All dominatees and the dominator in each clique of unit-radius cluster form a dBBlue scatternet

nodes in a clique. Notice that a connector node appears in several cliques (at least two since it is connector). If the role assignment is not treated carefully, a connector node (as connector u in Figure 5(a) in two cliques: one for dominator v_2, the other for dominator v_1) may be assigned master role in one dBBlue scatternet and slave role in another dBBlue scatternet. This multi-role assignment could cause some delay in scheduling packets. In an ideal situation, we would like the connector to be only slave of a couple of piconets without being master node in any piconet.

In our second method, we will first build a piconet for each connector: the connector being the master node of the piconet and all its dominator nodes (at most five) being the slave nodes of the piconet. For each clique partitioned from the dominatee nodes of a dominator, we build a dBBlue scatternet using all dominatee nodes and the dominator node, excluding the connectors. In other words, we build two level piconets: the top level piconets are built for the backbone nodes, and the low level piconets are built for all dominatees and dominators. Here the dominator node is used to connect each low level dBBlue scatternet to the top level backbone, serving as the gateway (as node v_1 in Figure 5(b) which connects u's piconet and t's piconet from two levels).

Algorithm 4 dBBlue Scatternet Construction in Clique Including Connectors

1: The dominator randomly selects a *dominatee* as leader t, which initiates the dBBlue scatternet formation on all nodes in the clique including the *dominator* and *connectors*, using Algorithm 2.
2: The dominator always assumes the pure slave of the leader with MAC 100.
3: All connectors have *higher priority* to be pure slaves or bridge slaves than other dominatees. Notice that the average number of connectors in each cone is $24/6 = 4$ and the leader could have up to 7 slave nodes, if we use the position based method. Thus, a connector will assume the slave role of the leader with high probability.

Figure 5(a) illustrates the first method which described in detail in Algorithm 4. Notice that we made some special treatments with the dominator and connectors in the algorithm, because we need diminish the probability to assign them *master* roles in the dBBlue structure. Though we can not avoid the role switch between *bridge* and *master* absolutely, it is not difficult to show that, at most one connector need assume dual roles, even in the worst situation that there are no dominatees in the clique, which rarely happens in practice as we will see in our simulation experiments.

There are three kinds of nodes in the M-dBBlue scatternet built by Algorithm 4: dominator, dominatee and connector. For their node degrees in the M-dBBlue scatternet, we can analyze them case by case. Case 1: for a dominator node, according to Algorithm 4, the dominator assumes pure slave in the one-hop dBBlue scatternet. So the degree of a dominator node is bounded by the number of cliques it has inside its cluster. If we use the position-based method, there are at most 6 dBBlue structures in a cluster, so its degree is bounded by 6. However, if we do not use position-based method, the number of cliques may be very large. Case 2: for a connector node, it could exist in at most 5 clusters, as long as there is at least one dominatee node in a clique. We can let that dominatee assume *master* role and the connector be its pure slave. Hence its degree is bounded by 5. Case 3: for a dominatee node, its node degree is obviously bounded by 7 according to the property of dBBlue scatternet since it only appears in one dBBlue scatternet. Therefore, in the M-dBBlue scatternet built by Algorithm 4, node degree is less or equal to 7 with high probability.

Another approach is to exclude the connectors from participating in the one-hop dBBlue scatternet formation in each clique, so that *no* nodes need assume both master and slave roles. Figure 5(b) illustrates the second method which is described in detail in Algorithm 5. Notice that the connector u will be a master node who forms a piconet with dominators v_1 and v_2 and assigns pure slaves to them. The leader t will become a master node and v_1 become its pure slave.

Algorithm 5 dBBlue Scatternet Construction in Clique Excluding Connectors

1: The dominator assumes *slave* role and each connector in the clique assumes *master* role in the CDS-based backbone.
2: The dominator randomly selects a *dominatee* as leader t, which initiates the dBBlue scatternet formation on all nodes in the clique excluding the *dominator* and *connectors*, using Algorithm 2.
3: The dominator always assumes the pure slave of the leader with MAC 100.

Similarly, we can analyze the node degree of the M-dBBlue scatternet built by Algorithm 5 case by case. Case 1: for a connector (as node u in Figure 5(b)), it does not participate in one-hop dBBlue construction, so all its neighbors must be dominators, which is at most 5. Case 2: for a dominatee, it only participates in one-hop dBBlue formation, so its degree is always bounded by 7. Case 3: for a dominator (as node v_1 in Figure 5(b)), it assumes the bridge slave role for the backbone. It is in at most $\ell_2 - 1$ piconets for the backbone since it has at most $\ell_2 - 1 = 24$ neighboring connectors. Additionally, it could be in several cliques, assume c cliques. Then the degree of dominator is at most $\ell_2 + c - 1$ under pessimistic estimation. If the position-based method is used, there are at most 6 cliques, thus the degree is at most $\ell_2 + 6 - 1 \leq 30$. As will see later in simulation results, the average node degree of dominators is much lower than these bounds.

In summary, Algorithm 4 could build a degree-7 Bluetooth

scatternet as long as there is one dominatee in each clique and position information is available. While the structure built by Algorithm 5 is easier to be dynamically updated since CDS-based backbone and clusters are independent of each other. To evaluate the performance of the two algorithms, simulation is conducted in the section 5.

4 Routing in M-dBBlue Scatternet

Position-based routing for wireless ad hoc networks has drawn considerable attention recently. However, it is not suitable for Bluetooth based personnel area networks since it requires additional GPS equipments which will increase the cost of Bluetooth devices. Moreover, Bluetooth networks are usually regarded as the extension of Internet, where IP-based routing is dominating. In this section, we propose a complete IP-based routing mechanism to integrate M-dBBlue scatternet with Internet seamlessly.

Internet is comprised of many separate administrative domains or *Autonomous Systems* (AS). There are two levels routing protocol, intra-domain and inter-domain, in Internet. Intradomain routing protocols, such as RIP and OSPF, route packets within a AS; while interdomain routing, currently handled by the *Border Gateway Protocol* (BGP), routes packets between ASes. We propose a similar hierarchical way to implement the IP-based routing in M-dBBlue scatternet. The inter-cluster routing is handled by a modified Bluetooth based *RIP* protocol on the M-dBBlue backbone. The intra-cluster routing is derived from the self-routing mechanism described in [1].

Inter-cluster Routing: The backbone of M-dBBlue scatternet works in a similar way as Internet, so we may apply any IP-based routing protocol here directly without much modification, such as the modified RIP protocol. In M-dBBlue scatternet, each cluster is assigned a network number, and every node in the cluster is dynamically assigned an IP address with same network number. See Figure 6 for an illustration of possible network number assignment of the clusters derived from Figure 1. The routing along the backbone could also be any greedy method [5; 6] if geometry information is known, or source-initiated on-demand routing protocols such as AODV [7], DSR [8], or table driven protocols such as DSDV [9]. Since such kind of routings are well-studied, we omit the details of routing along the backbone here.

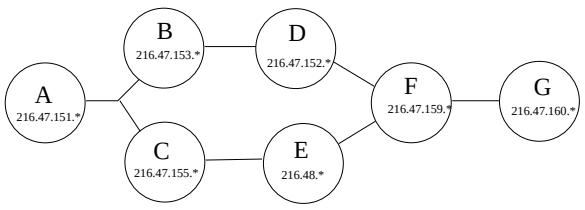


Figure 6: Intercluster routing in the backbone of M-dBBlue scatternet.

Intra-cluster Routing: In each clique of a cluster, dBBlue structure [1] intrinsically provides the self-routing mechanism based on the labels derived from a pseudo-balanced de Bruijn graph. To enable the IP-based routing in a clique, we need map the IP address to the corresponding label. Given the IP address of the target node, the source node need know the label of the target node if self-routing based on de Bruijn graph is used. One possible approach to solve this is to store all pairs of (IP, label) for all nodes in a node, e.g., the dominator node of the cluster. The source node always queries the dominator node for the label of the target node. Notice that such queries can be conducted using self-routing since the label of the dominator node is always fixed in our dBBlue structure. This centralized approach is simple, however, it suffers several disadvantages: the traffic storm problem to the dominator node, the single failure of the dominator node breaks the network, and so on.

We propose to use a distributed storage of the (IP, label) pairs. Each master node u in the dBBlue structure manages a lookup table, which stores the (IP, Label) pairs of those nodes whose key has u's piconet ID as prefix. Notice that the label is generated when we construct the dBBlue scatternet for each clique. The key of a node is some value computed from its IP, e.g., the hash value of its IP. Table 1 illustrates a possible lookup table stored in a master node with label 1011. Notice that, we represent the label of a master node by its piconet ID. It is assumed that the key value of any pair stored in this lookup table has prefix 1011.

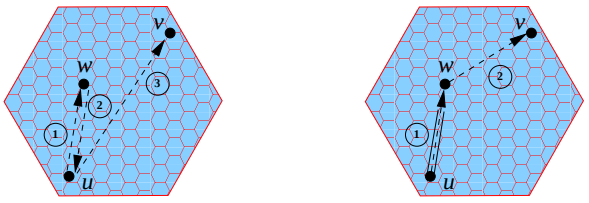
Table 1: The lookup table in the master node $u = 1011$.

IP-Address	Piconet ID	MAC
216.47.152.22	00100	000
216.47.152.44	01001	100
⋮	⋮	⋮
216.47.152.90	1100	010

Assume that the length of every piconet ID in the dBBlue scatternet is between m and $m+1$. Each node first maps its host address, the suffix of its IP address, to a binary *key* with length $m+1$. The mapping technique could adopt any hashing function or simply translate its host address to binary format which is then abbreviated or extended to $(m+1)$-bits key. The node then forwards its key and (IP, label) pair to the target master node through the label-based routing in dBBlue scatternet. Notice that the target master node in which the pair will be stored has a label being a prefix of the key. Since the labels of the nodes are universal prefix free, the target master node is unique.

Notice the simple abbreviation/extension is efficient but could cause unbalanced overload between master nodes with small and large labels. In contrast, the hash based mapping is believed to achieve uniformly distribution with high probability. In this paper, for simplification of our presentation, we adopt the latter to map host IP address to a key. For example, in a cone, suppose $m = 4$ and the network number of its cluster is $216.47.152.*$ with mask $255.255.255.0$. The node with IP address $216.47.152.90$ first translates its host address 90 to the binary format 1011010, then abbreviates its suffix to a $(m+1)$-bits key 10110. The node $216.47.152.90$ forwards its (IP, label) pair to the master node with label 1011 since the key 10110 has 1011 as prefix. Another node with IP address $216.47.152.12$

first translates its host address 12 to the binary format 1100, then extends it to a $(m+1)$-bits key 11000 by simply appending 0. Similarly, the node 216.47.152.12 forwards its (IP, label) pair to the master node 1100 or 11000, whichever exists.



(a) Search and Forwarding (b) Packet-in-tunnel Forwarding

Figure 7: Label based intradomain routing.

Consider the case that a node u wants to send packets to target node v in the same dBBlue scatternet while only IP address of node v is known. W.l.o.g., suppose the master node w holds the (IP, label) pair of node v. Node u first maps the IP address of node v to a key, say k. Two options, which are illustrated by Figure 7, could be used to send out the packet:

1. **Search and forwarding**. Node u queries the dBBlue backbone based on key k and gets the label of node v from the master node w. Node u then forwards the packet targeting v through dBBlue routing protocol. Figure 7(a) illustrates the mechanism.

2. **Packet-in-tunnel forwarding**. Node u adds an additional header with the key k to the packet then sends the packet out. The routing of the packet is based on the label of k. Once the master node w gets packet, it strips out the header and relays the packet to node v according to node v's label in its lookup table. Figure 7(b) illustrates the mechanism.

Remember the path between any two nodes in dBBlue structure is at most $2m+2$ hops. The former approach can reduce the overall workload of dBBlue structure since the data packet travels through the network at most $2m+2$ hops, while the data packet travels at most $4m+4$ hops in the latter case. But the latter approach does not need keep the packet before getting the target label as in the former approach, and the packet can reach the target faster if the time difference between transmitting different size packets is negligible, because the total communication path is at most $6m+6$ hops in the former while at most $4m+4$ hops in the latter. On the other hand, the latter approach can keep the anonymity of node w hence increase security.

We continue to describe the IP-based routing for the node pairs within different cliques of one cluster. Suppose that the dominator keeps an IP address range table for each cone. For the packets targeting a node in other cones in the cluster, the dBBlue protocol will first forward them to the cluster dominator, which then forwards the packets to the target clique. Eventually the packet will reach the target through the intra-clique routing as described above.

5 Performance Evaluation

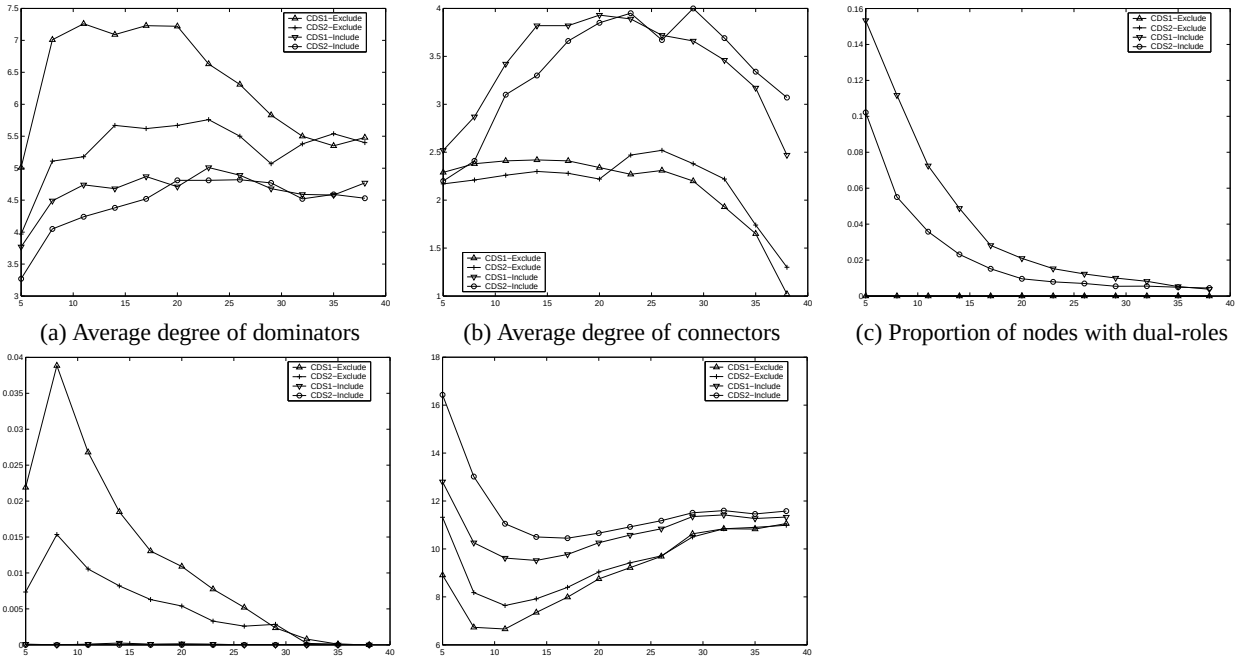
We have conducted extensive simulation to study the performance (topology properties) of different multi-hop scatternet structures proposed in this paper. In our experiment, we generate n wireless nodes, with uniform transmission range, randomly distributed in a square area with side length 40 units. To get a stable result, we randomly generate 100 samples of the network for each case and construct the scatternet using different methods for each sample. When running algorithm 4 and 5, we divide each cluster into 6 equal-size cones, i.e., 6 cliques. The average communication hops between all pairs of nodes is obtained by actually running message passing mechanism in the scatternet. For inter-cluster routing, we first find the shortest path between each pair of dominators/connectors. To follow the shortest path during routing, every node only needs to record the next hop neighbor to any other nodes. For intra-cluster routing, we suppose that the label of the target node is known. In other words we assume that the IP-address has already been mapped to a label in a distributed way as described in Section 4.

All experiment results are shown in Figure 8 and Figure 9. In the figures, we use *CDS1* to denote the CDS constructed by Algorithm 3, and *CDS2* to denote the CDS constructed by the algorithm from Wan *et., al* [13]. To distinguish the two proposed different scatternet formation methods for clusters, we use *Exclude* to denote Algorithm 5, and *Include* to denote Algorithm 4.

As we expect, M-dBBlue scatternet formed based on the backbone *CDS1* does provide smaller average hops between any pair of nodes than *CDS2* structure. Thus, more energy is saved. The tradeoff is that the average degree of dominators and connectors in *CDS1* is a little bit higher than the scatternet based on *CDS2*. However, this is tolerable since the average degree of connectors is still less than 4, and the average degree of dominator is less than 8 if CDS1 is used. Notice that only connector could be assigned *master* role. Thus, every master on the backbone will have at most 7 slaves with high probability, while the master in each cluster is guaranteed to have at most 7 slaves.

On the other hand, the M-dBBlue scatternet, when *Exclude* method is used to form scatternet for each cluster, has lower diameter than that produced by *Include* method. In addition, no nodes assume dual role in the scatternet formed by the *Exclude* method. The *Include* method has its own advantage: the average degree of dominators and connectors is smaller, and almost no node has degree more than 7 (See Figure 8 (d) and Figure 9 (d)).

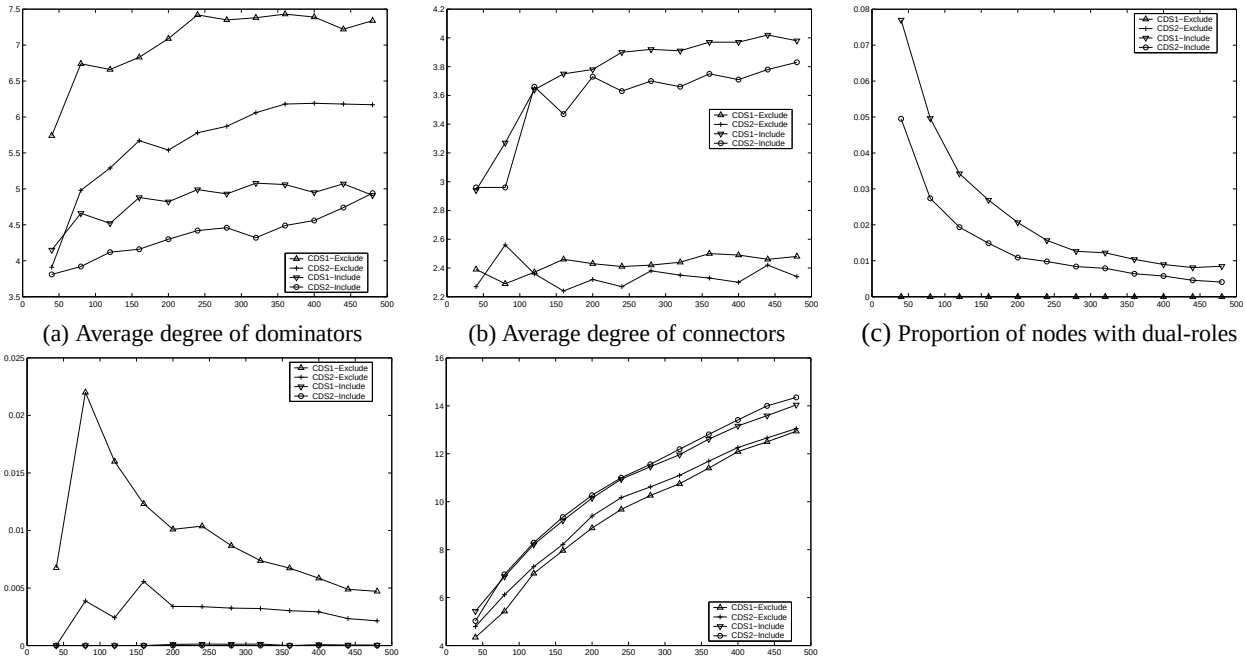
Figure 8 illustrates the performance variation when the uniform transmission range of nodes varies in $[5, 38]$, while the total number of nodes is fixed at 200. The first observation is that the average hops of shortest path between all pair of nodes first drop then rise to a number around $O(\log n)$ when the transmission range increases. This is because when the transmission range is small, the communication graph is sparse and the backbone has large diameter, which consequently increases the communication hops for nodes from different clusters; when the transmission range becomes larger, the diameter of the backbone becomes smaller and the scatternet degenerate to an one-hop dBBlue scatternet in the extreme case, which gives a $O(\log n)$ bound on communication hops. Figure 8 (a)



(a) Average degree of dominators (b) Average degree of connectors (c) Proportion of nodes with dual-roles

(d) Proportion of nodes with degree > 7 (e) Average communication hops

Figure 8: Performance evaluation of our different scatternet construction methods when node transmission range varies in the range $[5, 38]$ and the number of nodes is fixed at 200.



(a) Average degree of dominators (b) Average degree of connectors (c) Proportion of nodes with dual-roles

(d) Proportion of nodes with degree > 7 (e) Average communication hops

Figure 9: Performance evaluation of our different scatternet construction methods when the number of nodes varies in the range from 40 to 480 and the node transmission range is fixed at 8.

and (b) show that the average degree in backbone reaches the peak when the transmission range is around half of the simulation field width. It drops down after the peak because the size of backbone shrinks.

Figure 9 illustrates the performance variation when the number of wireless nodes varies in $[40, 480]$, while the transmission range of each node is fixed at 8 units. The average degree in backbone keeps almost constant after the density reaches some extent. In both Figure 8 and Figure 9, the proportion of nodes with dual roles, and the proportion of nodes with degree exceeding 7 drop when the network density increases. Notice we need not test the degree of dominatees because Algorithm 2 can guarantee the perfect degree bound as discussed in [1].

Notice that in our simulation, we only study the static networks. However, our scatternet also enjoys efficient updating in a dynamic environment, since both the backbone and the dB-Blue structure of each cluster can be maintained efficiently. For the backbone built by the method in [13], it may need many massages to update or rebuild since a spanning tree is involved, but in [19; 11], they gave a more message efficient backbone formation method. In [11], they described the detail about how to maintain their backbone in mobile environment. In [1], they described a vigorous method to *locally* update the structure dB-Blue using at most $O(\log n)$ messages when a node joins or leaves the network. In most cases, the cost of updating the dB-Blue is actually $O(1)$ since a node can join or leave without affecting the remaining scatternet. The number of affected nodes is always bounded from above by a constant when a node joins or leaves the network. We leave the simulation study of the mobile cases as one of our future work.

6 Related Work

In this section we review the solutions of scatternet formation for multi-hop networks by dividing them into five categories.

Tree Based Methods: Zaruba, Basagni and Chlamtac [24] proposed two distributed tree-based methods for forming connected scatternet. In both methods, the resulting topology is termed as *BlueTree*. The first method is initiated by a single node (*blueroot*). A rooted spanning tree is built from the blueroot. Each node is a slave of its parent and a master of its children in the tree. In the second method [24], several roots are initially selected. Each of them then creates its own scatternet as in the first method. After that, sub-tree scatternets are connected into one scatternet spanning the entire network. Remember that the tree topology suffers from a major drawback: the root is a communication bottleneck. In addition, dynamic updating that preserves correct routing is not discussed in these protocols. There are several modified versions of BlueTree, such as the methods by Dong and Wu [25] and Huang *et al.* [26], to reduce the communication overhead or increase the connectivity. Cuomo *et al.* [27] also proposed a tree-based scatternet formation algorithm *SHAPER* for multi-hop networks, which focuses on the self-healing behavior of the tree structure: i.e., it is able to dynamically reconfigure the scatternet after topological variations due to mobility or failure of nodes. Guerin *et al.* [28] proposed depth/breath first search and MST-based scatternet formation schemes for unit disk graphs in two and three dimensions, but their schemes are not localized.

Cluster Based Methods: Basagni, Petrioli and Chlamtac [29; 30] described a multihop scatternet formation scheme based on clustering scheme [31]. The constructed scatternet is called *BlueStars*. The protocol proceeds in three phases: device discovery, partitioning of the network into piconets (stars) by clustering, and interconnection of the piconets to a connected scatternet. All clusterhead nodes are declared master nodes in a piconet, with all nodes belonging to their clusters as their slaves. In the third phase, *BlueConstellation*, some of the slaves become masters of additional piconets, i.e. become master-slave bridges, to assure the connectivity of the scatternet. However, piconets in the scatternet may have more than seven slaves. This may result in performance degradation, as slaves need to be parked and unparked in order to communicate with their master. A performance evaluation of the clustering-based scatternet formation scheme [30] is given by [32; 29]. To fix the unbound slave number, Basagni, Petrioli and Chlamtac [33; 34] modified their protocol [29; 30] and proposed a new scatternet called *BlueMesh*. The idea of bounding the slave number in BlueMesh is again based on the observation that if a node in unit disk graph has more than five neighbors then at least two of them must be connected. Same with *BlueStars*, the selection of the masters is based on the node weights. However, the selection of slaves is performed in such a way that if a master has more than seven neighbors, it only chooses seven slaves among them so that via them it can reach all the others. Variants of clustering-based scatternet formation schemes [35; 36] were also proposed. Both clustering processes [35; 36] follow a random fashion. Initial connections are made by nodes entering scan or inquiry scan phases at random. Already existing master nodes have priority in attracting more slaves up to the limit. After each node is assigned master or slave role, or is unable to join any piconet or attract any neighbor as its slave to create its own piconet, some bridge piconets are added to connect the scatternet. However, both methods [35; 36] do not always lead to a connected structure.

Position Based Methods: Li, Stojmenovic and Wang [37] proposed the first position-based schemes that construct degree limited and connected piconets in multihop networks without parking any node. Notice that the schemes in [33; 34] can also achieve bounded degree scatternet. Their neat scheme does not require position information, but instead the local information is extended to two hop information, with a two round device discovery phase for obtaining necessary information. In Li *et al.*'s solution, nodes know their positions and are able to establish connections with other nodes within their transmission radius in the neighbor discovery phase. The degree of each node is limited to seven by applying Yao structure [38], and the master-slave relations are formed in created subgraphs. This phase follows clustering based approach, and consists of several iterations. In each iteration, undecided nodes with higher keys than any of their undecided neighbors apply Yao structure to bound the degree, decide master-slave relations on the remaining edges, and inform all neighbors about either edge deletion or master-slave decision. The experiments confirmed good functionality of created Bluetooth networks in addition to

their fast creation and straightforward maintenance. Basagni *et al.* [39; 40] described the results of an ns2-based comparative performance evaluation among four major solutions for forming multihop scatternet [37; 29; 24; 35]. They found that device discovery is the most time-consuming operation, independently of the particular protocol to which it is applied. The comparative performance evaluation showed that due to the simplicity of its operations BlueStars is by far the fastest protocol for scatternet formation. However, BlueStars produces scatternets with an unbounded, possibly large number of slaves per piconet, which imposes the use of potentially inefficient Bluetooth operations. They proposed a combined solution by applying a Yao structure on each piconet, to limit the degree of each master node to seven. This is a variant of the clustering-based scheme [37].

On-demand Methods: Most above scatternet formation protocols tend to interconnect all Bluetooth devices at the initial network startup stage and maintain all Bluetooth links thereafter. The master or bridge nodes in the resulting scatternet may become the traffic bottleneck and reduce network throughput. To make the scatternet structure more suitable to serve in mobile ad hoc networks, several on-demand methods [41; 42; 43; 44] (to build scatternets only along the multihop routes with traffic demands and eliminate unnecessary link and route maintenances) are proposed recently.

QoS Based Methods: Marsan *et al.* [45] studied how to construct the optimal topology that provides full network connectivity, fulfills the traffic requirements and the constraints posed by the system specification, and minimizes the traffic load of the most congested node in the network, or equivalently its energy consumption. By using a min-max formulation, they provided a centralized solution based on integer linear programming. Chiasserini *et al.* [46] extended the work of Marsan *et al.* and enhanced the optimization problem by adding the constraints on the network capacity. Augel and Knoor [47] proposed an approach of scatternet formation in which the formation is dependent on the QoS requirements of the applications. In their solution, to avoid larger degree which may cause negative influence on throughput, nodes with high degree stop paging and instruct a neighbor with a low degree to start paging instead. Each device may try to influence the topology depending on the QoS requirements. They described a general scatternet formation design guidelines for QoS applications, but did not present any particular scatternet formation protocol. Melodia and Cuomo [48; 49; 50] discussed the scatternet formation issue in Bluetooth by setting a framework for scatternet analysis based on a matrix representation, which allows developing and applying different metrics. They identified several metrics (capacity, average load, or path length) both in a traffic independent and in a traffic dependent context, and showed the relevant numerical results. Then, a distributed algorithm for scatternet topology optimization was introduced, that supports the formation of a locally optimal scatternet based on a selected metric. Numerical results obtained by adopting this distributed approach to optimize the network topology were shown to be close to the global optimum. Cuomo *et al.* [51; 52] extended their work [48; 49; 50] and provided an integrated approach for scatternet formation and quality-of-service support (called SHAPER-OPT) by combining the tree-based scatternet formation algorithm *SHAPER* [27] and the distributed scatternet optimization algorithm (DSOA) [48; 49; 50].

7 Conclusion

In this paper, we proposed a practical solution for large multihop Bluetooth scatternet formation and IP-based routing mechanism according to Bluetooth specification. We proposed a novel communication efficient method to build a connected dominating set (CDS) as the backbone of multi-hop Bluetooth network. Then *dBBlue* scatternet is formed for each cluster. The final scatternet, *M-dBBlue*, guarantees the connectivity and each cluster has self-routing property. Our experiment shows the majority of nodes have degree smaller or equal to 7 which means our scatternet seldom parks any node. Our scatternet also enjoys efficient updating, since both the backbone and the dB-Blue structure of each cluster can be maintained efficiently in a dynamic environment. Our method does not need any position information of Bluetooth devices for scatternet construction and packets routing. It is interesting to notice that M-dBBlue Bluetooth network intrinsically supports the future P2P applications, since each cluster supports the content based routing through pseudo-balanced de Bruijn structure.

REFERENCES

[1] Wen-Zhan Song, Xiang-Yang Li, Yu Wang, and Weizhao Wang, "dBBlue: Low diameter and self-routing Bluetooth scatternet," *Journal of Parallel and Distributed Computing*, vol. 65, no. 2, pp. 178–190, 2005.

[2] Bluetooth SIG, "Specification 2.0 of the Bluetooth system," http://www.bluetooth.com/.

[3] T. Salonidis, P. Bhagwat, L. Tassiulas, and R. LaMaire, "Distributed topology construction of bluetooth personal area networks," in *Proc. IEEE INFOCOM*, 2001.

[4] S. Basagni, R. Bruno, and C. Petrioli, "Device discovery in bluetooth networks: A scatternet perspective," in *Proc. IFIP-TC6 Networking Conference, Networking 2002*, 2002.

[5] Brad Karp and H.T. Kung, "Gpsr: Greedy perimeter stateless routing for wireless networks," in *Proc. of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, 2000.

[6] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger, "Worst-Case Optimal and Average-Case Efficient Geometric Ad-Hoc Routing," in *Proc. 4^{th} ACM Int. Symposium on Mobile Ad-Hoc Networking and Computing (MobiHoc)*, 2003.

[7] C.E. Perkins and E.M.Royer, "Ad-hoc on demand distance vector routing," in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA*, February 1999, pp. 90–100.

[8] J. Broch, D. Johnson, and D. Maltz, "The dynamic source routing protocol for mobile ad hoc networks," 1998.

[9] C.E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers," *Computer Communications Review*, pp. 234–244, October 1994.

[10] C. Petrioli, S. Basagni, and I. Chlamtac, "Configuring bluestars:multihop scatternet formation for bluetooth networks," *IEEE Transactions on Computers*, vol. 52, no. 6, pp. 779–790, 2003.

[11] Khaled Alzoubi, Peng-Jun Wan, and Ophir Frieder, "Message-Optimal Connected-Dominating-Set Construction for Routing in Mobile Ad Hoc Networks," in *ACM MobiHoc*, 2002.

[12] I. Chlamtac and A. Farago, "A new approach to design and analysis of peer to peer mobile networks," *Wireless Networks*, vol. 5, pp. 149–156, 1999.

[13] Peng-Jun Wan, Khaled M. Alzoubi, and Ophir Frieder, "Distributed construction of connected dominating set in wireless ad hoc networks," in *IEEE INFOCOM*, 2002.

[14] J.Wu and H.L. Li, "On calculating connected dominating set for efficient routing in ad hoc wireless network," in *3rd ACM international workshop on Discrete algorithms and methods for mobile computing and communications*, 1999, pp. 7–14.

[15] L. Barriere, P Fraigniaud, L. Narajanan, and J. Opatrny, "Dynamic construction of bluetooth scatternets of fixed degree and low diameter," in *14th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, 2003, pp. 781–790.

[16] C. Law, A.K. Mehta, and K.Y. Siu, "Performance of a new bluetooth scatternet formation protocol," in *Proc. ACM Symposium on Mobile Ad Hoc Networking and Computing MobiHoc*, 2001, pp. 183–192.

[17] N. de Bruijn, "A combinatorial problem," in *Koninklijke Nederlandse Academie van Wetenschappen*, 1946, 49, pp. 758–764.

[18] Pierre Fraigniaud and Philippe Gauron, "The content-addressable network d2b," Tech. Rep. Technical Report TR-LRI-1349 (also appeared in 22nd ACM Symp. on Principles of Distributed Computing (PODC)), 2003.

[19] Khaled Alzoubi, Xiang-Yang Li, Yu Wang, Peng-Jun Wan, and Ophir Frieder, "Geometric spanners for wireless ad hoc networks," *IEEE Transactions on Parallel and Distributed Processing*, vol. 14, no. 4, pp. 408–421, 2003, Short version in IEEE ICDCS 2002.

[20] H. Ishii and H. Kakugawa. A self-stabilizing algorithm for finding cliques in distributed systems. In *Proc. of the 21st IEEE Symposium on Reliable Distributed Systems (SRDS'02)*, 2002.

[21] E. Balas and C.S. Yu. Finding a maximum clique in an arbitrary graph. *SIAM J. on Computing*, vol. 15, pp.1054–1068, 1986.

[22] R. L. Wang, Z. Tang, and Q. P. Cao. An efficient approximation algorithm for finding a maximum clique using hopfield network learning. *Neural Computation*, vol. 15, pp.1605-1619, 2003.

[23] R. Carter and K. Park. How good are genetic algorithms at finding large cliques: An experimental study. Technical Report BU-CS-93015, Computer Science Dept., Boston University, 1993.

[24] G.V. Zaruba, S. Basagni, and I. Chlamtac, "Bluetrees - scatternet formation to enable bluetooth based ad hoc networks," in *Proc. IEEE ICC*, 2001.

[25] Yuhong Dong and Jie Wu, "Three Bluetree formations for constructing efficient scatternets in Bluetooth," in *Proc. of the 7th Joint Conference on Information Sciences*, 2003.

[26] Tsung-Chuan Huang, Chu-Sing Yang, Chao-Chieh Huang, and Shen-Wen Bai, "Hierarchical grown Bluetrees (HGB) - an effective topology for Bluetooth scatternets," in *Proc. of International Symposium on Parallel and Distributed Processing and Applications (ISPA 2003), LNCS 2745*, Aizu, Japan, 2003.

[27] F. Cuomo, G. di Bacco, and T. Melodia, "Shaper: a self-healing algorithm producing multi-hop Bluetooth scatternets," in *Proceedings of IEEE Globecom 2003*, San Francisco, CA, USA, December 2003.

[28] R. Guerin, J. Rank, S. Sarkar, and E. Vergetis, "Forming connected topologies in Bluetooth adhoc networks," in *Proceedings of ITC'18*, Berlin, 2003.

[29] C. Petrioli, S. Basagni, and I. Chlamtac, "Configuring Bluestars: Multihop scatternet formation for Bluetooth networks," *IEEE Transactions on Computers*, vol. 52, no. 6, pp. 779–790, 2003.

[30] S. Basagni and C. Petrioli, "A scatternet formation protocol for ad hoc networks of Bluetooth devices," in *Proceedings of the IEEE Semiannual Vehicular Technology Conference, VTC Spring 2002*, Birmingham, AL, May 6–9 2002.

[31] Chunhung Richard Lin and Mario Gerla, "Adaptive clustering for mobile wireless networks," *IEEE Journal of Selected Areas in Communications*, vol. 15, no. 7, pp. 1265–1275, 1997.

[32] S. Basagni, R. Bruno, and C. Petrioli, "Performance evaluation of a new scatternet formation protocol for multihop Bluetooth networks," in *Proceedings of the* 5*th International Symposium on Personal Wireless Multimedia Communications, WPMC 2002*, Honolulu, Hawaii, October 27–30 2002, pp. 208–212.

[33] C. Petrioli, S. Basagni, and I. Chlamtac, "Bluemesh: degree-constrained multi-hop scatternet formation for Bluetooth networks," *Mobile Networks and Aplications*, vol. 9, no. 1, pp. 33–47, 2004.

[34] C. Petrioli and S. Basagni, "Degree-constrained multi-hop scatternet formation for bluetooth networks," in *Proc. IEEE GLOBECOM*, 2002.

[35] 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, 2002.

[36] R. Guerin, E. Kim, and S. Sarkar, "Bluetooth technology key challenges and initial research," in *Proc. SCS Communication Networks and Distributed Systems Modeling and Simulation CNDS*, 2002, pp. 157–163.

[37] Xiang-Yang Li, Ivan Stojmenovic, and Yu Wang, "Partial delaunay triangulation and degree limited localized Bluetooth multihop scatternet formation," *IEEE Transaction on Parallel and Distributed Systems*, vol. 15, no. 4, pp. 350–361, 2004, The short version appeared at AdHoc-Now 2002.

[38] A. C.-C. Yao, "On constructing minimum spanning trees in k-dimensional spaces and related problems," *SIAM J. Computing*, vol. 11, pp. 721–736, 1982.

[39] S. Basagni, R. Bruno, and C. Petrioli, "A performance comparison of scatternet formation protocols for networks of bluetooth devices," in *Proc. IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2003.

[40] Stefano Basagni, Raffaele Bruno, Gabriele Mambrini, and Chiara Petrioli, "Comparative performance evaluation of scatternet formation protocols for networks of Bluetooth devices," *Wirel. Netw.*, vol. 10, no. 2, pp. 197–213, 2004.

[41] Yong Liu, M.J. Lee, and T.N. Saadawi, "A Bluetooth scatternet-route structure for multihop ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 2, pp. 229– 239, 2003.

[42] Yoji Kawamoto, Vincent W.S. Wong, and Victor C.M. Leung, "A two-phase scatternet formation protocol for Bluetooth wireless personal area networks," in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC03)*, New Orleans, LA, 2003.

[43] Elena Pagani, Gian Paolo Rossi, and Stefano Tebaldi, "An on-demand Bluetooth scatternet formation algorithm," in *Proceedings of First Working Conference on Wireless On-demand Network Systems (WONS 2004)*, Madonna di Campiglio, Italy, January 2004.

[44] Ming-Te Chou and Ruay-Shiung Chang, "Blueline: A distributed Bluetooth scatternet formation and routing algorithm," in *IASTED International Conference on Parallel and Distributed Computing and Networks(PDCN)*, 2004.

[45] M.A. Marsan, C.F. Chiasserini, A. Nucci, G. Carello, and L. de Giovanni, "Optimizing the topology of bluetooth wireless personal area networks," in *INFOCOM*, 2002.

[46] Carla-Fabiana Chiasserini, Marco Ajmone Marsan, Elena Baralis, and Paolo Garza, "Towards feasible distributed topology formation algorithms for Bluetooth-based wpans," in *Proc. of 36th Hawaii International Conference on System Science (HICSS-36)*, Big Island, Hawaii, 2003.

[47] Markus Augel and Rudi Knorr, "Bluetooth scatternet formation - state of the art and a new approach," in *Proc. of the 17th International Conference on Architecture of Computing Systems (ARCS), LNCS 2981*, Augsburg, Germany, 2004.

[48] T. Melodia and F. Cuomo, "Ad hoc networking with Bluetooth: Key metrics and distributed protocols for scatternet formation," *Ad Hoc Networks (Elsevier)*, vol. 2, no. 2, pp. 185–202, April 2004.

[49] F. Cuomo and T. Melodia, "A general methodology and key metrics for scatternet formation in Bluetooth," in *Proceedings of IEEE Globecom 2002*, Taipei, Taiwan, November 2002.

[50] T. Melodia and F. Cuomo, "Locally optimal scatternet topologies for Bluetooth ad hoc networks," in *Proceedings of First Working Conference on Wireless On-demand Network Systems (WONS 2004)*, Madonna di Campiglio, Italy, January 2004.

[51] F. Cuomo, T. Melodia, and I. F. Akyildiz, "Distributed self-healing and variable topology optimization algorithms for qos provisioning in scatternets," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 7, pp. 1220–1236, 2004.

[52] F. Cuomo, G. di Bacco, and T. Melodia, "Optimized scatternet topologies for personal area networking in dynamic environments," in *Proceedings of IEEE IEEE International Conference on Communications (ICC 2004)*, Paris, France, June 2004.