Wireless Networks 10, 197–213, 2004


# Comparative Performance Evaluation of Scatternet Formation Protocols for Networks of Bluetooth Devices

STEFANO BASAGNI *
*Department of Electrical and Computer Engineering, Northeastern University, USA*

RAFFAELE BRUNO
*Institute for Informatic and Telematic, C.N.R. Pisa, Italy*

GABRIELE MAMBRINI
*CASPUR, Roma, Italy*

CHIARA PETRIOLI
*Dipartimento di Informatica, Università di Roma "La Sapienza", Roma, Italy*

**Abstract.** This paper describes the results of the first ns2-based comparative performance evaluation among four major solutions presented in the literature for forming multi-hop networks of Bluetooth devices (*scatternet formation*). The four protocols considered in this paper are *BlueTrees* [1], *BlueStars* [2], *BlueNet* [3] and the protocol presented in [4] which proposes geometric techniques for topology reduction combined with cluster-based scatternet formation. We implemented the operations of the four protocols from device discovery to scatternet formation. By means of a thorough performance evaluation we have identified protocol parameters and Bluetooth technology features that affect the duration of the formation process and the properties of the produced scatternet. We have investigated how possible modifications of the BT technology (e.g., backoff duration, possibility for a BT inquirer to identify itself) make device discovery more efficient for scatternet formation in multi-hop networks. We have then discussed implementation concerns for each of the selected protocols. Finally, we have analyzed the protocols overhead as well as the effect of the different protocols operations on key metrics of the generated scatternets, which includes the time needed for forming a scatternet, the number of its piconets, the number of slaves per piconet, the number of roles assumed by each node and the scatternet route lengths.

**Keywords:** wireless networks, Bluetooth, scatternet formation, performance evaluation

## 1. Introduction

The Bluetooth (BT) technology, as described in the Specifications of the Bluetooth System Version 1.1 [5], is expected to be one of the most promising enabling technologies for ad hoc networks and pervasive computing. Originally introduced as short-range cable replacement, the BT specifications define ways for which each BT device can set up multiple connections with neighboring devices so that communication can be established in a multi-hop fashion. Therefore, Bluetooth devices spread in a geographic area can provide the missing wireless extension to the various heterogeneous network infrastructures, allowing a more pervasive wireless access.

This paper addresses the fundamental problem of *scatternet formation*, i.e., the problem of the self organization of BT devices into a multi-hop network. In particular, this paper describes and compares the performance of four among the most promising solutions proposed so far to this problem. Beyond providing the first performance comparison among available solutions, this paper also provides insights on the Bluetooth technology and on some limitations of the current specifications with respect to scatternet formation.

According to the current BT specifications, the way in which BT allows multi-hop communication is summarized as follows. When two BT nodes that are into each others communication range want to set up a communication link, one of them must assume the role of *master* of the communication while the other becomes its *slave*. This simple "one-hop" network is called a *piconet* and may include several slaves no more than 7 of which can be actively communicating with the master at the same time. If a master has more than 7 slaves, some slaves have to be *parked*. To communicate with a parked slave a master has to *unpark* it, while possibly parking another slave.

The specifications allow each node to assume multiple roles. A node can be a master in one piconet and a slave in one or more other piconets, or a slave in multiple piconets. Devices with multiple roles act as *gateways* to adjacent piconets thus *forming* a multi-hop ad hoc network called a *scatternet*.

Figure 1(a) shows the case where 13 BT devices have been divided into four piconets (A, B, C and D). Masters are represented by pentagons (surrounded by a large circle that represents their transmission radius), while slaves are depicted

* Corresponding author.
E-mail: basagni@ece.neu.edu

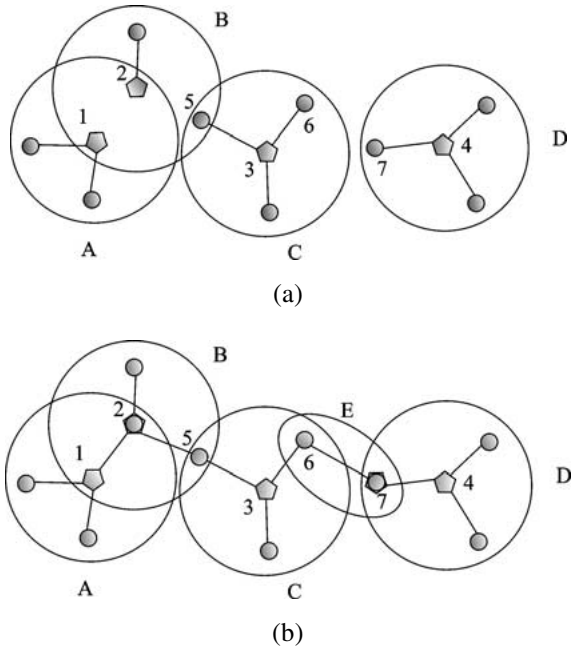



Figure 1. (a) Four disjoint piconets, and (b) the corresponding scatternet with five piconets.

as small circles. Adjacent piconets can be interconnected in different ways. Piconets A and B depict the *master–master* case, when two masters are neighbors and interconnection is achieved by having one of the two masters joining the piconet of the other as slave (in the figure, node 2 became the slave of node 1). Two piconets can be joined by a common slave, termed a *gateway slave*. This is the case of piconets B and C which are joined by node 5. The third case is when piconets are interconnected through a pair of neighboring slaves, called in the following *intermediate gateways*, as in the case of piconets C and D, joined by nodes 6 and 7. In the latter case, interconnection requires that one of the two intermediate gateways becomes the master of a new piconet that includes the other intermediate gateway as slave (in the figure, node 7 becomes the master of the extra piconet). With the creation of piconet E, the five piconets of figure 1(b) form a connected scatternet.

A first broader classification of the solutions proposed so far in the literature distinguishes between scatternet formation protocols that require the radio vicinity of *all* nodes (*single-hop* topologies) and protocols that work in the more general *multi-hop* scenario. All the solutions are *localized*, in the sense that the protocols are executed at each node with the sole knowledge of its immediate neighbors (nodes in its transmission range).

Solutions of the first kind are presented in [6], [7] and [8]. The solution proposed in [6] is based on a leader election process to collect topology information at the leader. Then, a centralized algorithm is run at the leader to assign the roles to the network nodes. In order to achieve desirable scatternet properties, the centralized scheme executed by the leader requires that the number of network nodes is $\leqslant 36$. When $n > 36$ other centralized schemes could be used such as the one proposed in [9] and [10]. The protocols presented in [7] and [8] run over single-hop topologies with no limitations on the number of nodes. However, the resulting scatternet is a tree, which limits efficiency and robustness. Given the restrictive assumption that characterizes these solutions, and the impossibility of fair comparison between single-hop and multi-hop protocols, we have not considered this first kind of solutions in the performance comparison described in this paper.

Among the solutions that apply to the more general case of multi-hop topologies, the scatternet formation protocol described in [1] requires that the protocol is initiated by a designated node (the *blueroot*) and generates a tree-like scatternet. The blueroot starts the formation procedure by acquiring as slaves its one hop neighbors. These, in turn, start contacting their own neighbors (those nodes that are two hops from the root) trying to acquire them as their slaves and so on, in a "wave expansion" fashion, till the whole tree is constructed. The obtained scatternet has piconets with an unbounded number of slaves. A procedure based on geometric properties of networks of devices scattered in the plane is described to re-configure the tree so that each master has no more than seven slaves. This allows the master to avoid the time and bandwidth consuming operation of parking and unparking of slaves.

To the best of the authors' knowledge, the only presented solutions for scatternet formation in multi-hop BT networks that produce topologies different from a tree are those introduced in [4], [2], [3] and [11].

The main aim of the protocol proposed in [4] is to build up a connected scatternet in which each piconet has no more than 7 slaves. To this purpose, degree reduction techniques are initially applied to the network topology graph to reduce the number of wireless links at each node to less than 7 without affecting the connectivity of the resulting topology. A scatternet formation protocol (which is left unspecified) is then executed on the resulting topology. These techniques require each node to be equipped with additional hardware that provides to the node its current (geographic) location (e.g., a GPS receiver). Beyond being potentially expensive, this solution is not feasible when such extra hardware is not available.

The protocol described in [2], termed *BlueStars*, forms connected scatternets without using any extra hardware. Relying on the sole knowledge of a node's one-hop neighbors, BlueStars selects the piconets' masters based on how "fit" a node is to serve as a master. Once piconets are formed, gateway nodes are selected so that there is an inter-piconet route between all masters that are at most tree hops away (i.e., all adjacent piconets are interconnected). This condition ensures the connectivity of the BlueStars scatternet [12]. However, BlueStars may produce scatternets whose piconets have more than 7 slaves. Whenever positioning information is available at the nodes, BlueStars can be combined with the protocols described in [4] to form scatternets in which each piconet has no more than 7 slaves.

The scatternet formation scheme proposed in [3], *BlueNet*, produces a scatternet whose piconets have a bounded num-

ber of slaves. After an (unspecified) device discovery phase, some of the nodes enter either page or page scan randomly. Nodes in page mode will be masters and try to invite a bounded number of neighbors to join their piconets as slaves. Nodes that are unable to acquire slaves or to join a piconet within a given amount of time re-execute this process. This provides a statistical guarantee that either a node becomes a member of a piconet (being either the master or a slave of the piconet) or it is isolated (i.e., all its neighbors made a decision on their role but none of its neighboring masters invited it in its piconet). Isolated nodes become masters and try to connect to some already formed piconets. Finally, the master of each piconet instructs its slaves to set outgoing links to neighboring piconets to form a scatternet. The connectivity of the resulting scatternet is not guaranteed (i.e., not all the BlueNets are connected, even when the initial topologies are).

Recently, a new protocol that produces connected scatternets with no more than 7 slaves per piconet have been introduced in [11], termed *BlueMesh*. Differently from [4], no location information is needed. In order to form a connected scatternet, the protocol proceeds in successive iterations through which connectivity is achieved progressively. In each iteration, the selection of the slaves is performed in such a way that if a master has more than seven neighbors, it chooses seven slaves among them so that via them it can reach all the others. Once masters and slaves are selected, gateways are chosen to join adjacent piconets. Intermediate gateways proceed onto the following iteration to be interconnected, and the whole process is repeated until a connected scatternet is formed.

All the mentioned solutions refer to the problem of *setting up* scatternets from a set of nodes spread in a geographic area. The problem of *maintaining* scatternets in presence of dynamic addition/removal of nodes, and of nodes mobility has just started to be addressed for multi-hop topologies and it is not addressed in this paper. (A first solution providing a preliminary account on how to deal with changing network topology has been recently presented in [13].)

In this paper we compare the performance of four scatternet formation protocols for multi-hop topologies among those outlined above, namely, BlueTrees, BlueNet, BlueStars and the protocol presented in [4] which makes use of location information. Among the several geometric construction techniques proposed in [4], here we implement the one that the authors of [4] consider the most promising for forming scatternets with desirable properties, namely, the *Yao construction* based on the geometric methods presented in [14]. In [4] Stojmenovic and Li suggest that the Yao construction should be combined with a clustering-based solution for the actual scatternet formation. Therefore, we have combined the Yao construction with BlueStars to obtain a protocol that forms scatternets with a bounded number of slaves per piconet. We have termed this protocol the *Li–Stojmenovic/BlueStars*, or LSBS, protocol.

All operations of the chosen protocols require each node to be aware of its neighbors. To this purpose, *device discovery* has to be performed before the actual scatternet formation process takes place.

A comparison of the four protocols has been performed by means of ns2-based simulations [15] that investigate the impact of device discovery on the performance of the protocols, and compare the protocols with respect to key metrics considered crucial for scatternet formation. These metrics are: the time needed for scatternet formation (including the phase of device discovery); the average number of piconets; the number of slaves per piconet; the number of roles assigned to each node, the average length of the routes between any two BT devices in the scatternet, and the overhead associated to the scatternet formation process.

The paper is organized as follows. In the following section we briefly describe the way we solve the problem of device discovery, which is common to all solutions described and compared in this paper. Section 3 describes the four protocols and the problems we encountered in implementing them. In section 4 we describe in details our simulation environment and we evaluate and compare the performance of the four protocols with respect to the selected metrics of interest. Finally, section 5 concludes the paper.

## 2. Device discovery in multi-hop bluetooth networks

The device discovery phase should lead each of the network nodes to become aware of all the nodes within its transmission range. This knowledge should be "symmetric", which means that if node $v$ knows node $u$, $u$ must also know $v$ (this is an assumption upon which all protocols for multi-hop scatternet formation rely).

The mechanisms provided by the BT specifications for device discovery, i.e., the *inquiry* procedures, do not lead to the needed symmetric neighbor knowledge: An *inquirer* that is trying to discover neighboring nodes does not transmit its unique BT identifier, thus remaining unknown to the node that receives the inquiry packet. Furthermore, the BT discovery mechanisms require nodes to be in opposite inquiry modes (called *inquiry* and *inquiry scan* modes) in order to be able to communicate. However, no method is described in the specifications on how to (even statistically) guarantee that two neighboring devices are in opposite modes. Therefore, specifications compliant mechanisms must be defined to ensure that, for each pair of neighboring nodes $v$ and $u$, they are eventually in opposite modes and that, when node $v$ discovers node $u$, $u$ is also made aware of $v$.

The only solution for device discovery in multi-hop networks proposed so far is derived by the mechanism first described in [6] (see also [2] for a detailed description of device discovery operations in a multi-hop scenario). Each device is allowed to alternate between inquiry mode and inquiry scan mode, remaining in each mode for a time selected randomly and uniformly in a given time range. The whole process is performed for a predefined device discovery time length. The operations while in each of the two modes are those described in the specifications.

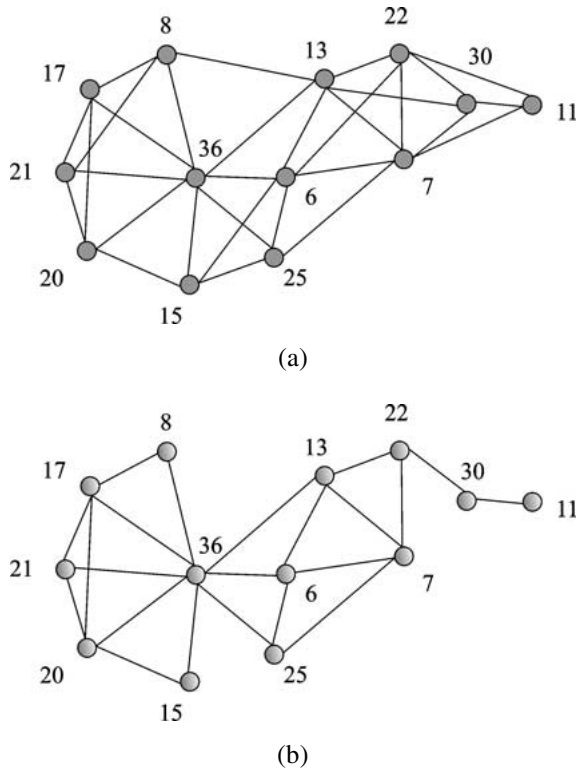



Figure 2. (a) The visibility graph of a BT network with 13 nodes, and (b) a possible discovered BT topology.

When two nodes in opposite inquiry modes handshake, they set up a temporary piconet that lasts only the time necessary to exchange their ID and possibly other information necessary for the following phases of the protocol. The formation of temporary piconets and the exchange of information achieve the required mutual (i.e., symmetric) knowledge. Piconet formation is performed by executing the BT *page* procedures. Specifically, the inquirer goes in *page* mode and becomes the master, while the inquired node goes in *page scan* mode, becoming a slave. As soon as the information has been successfully communicated the piconet is disrupted.

The effectiveness of the described mechanism in providing the needed mutual knowledge to pairs of neighboring devices relies on the idea that, by alternating between inquiry and inquiry scan mode and randomly selecting the length of each inquiry (inquiry scan) phase, there is high probability that any pair of neighboring devices will be in opposite mode for a sufficiently long time, thus allowing the devices to discover each other. However, this process can be extremely time consuming [16]. As shown in section 4.2, we have observed that in networks with high density, after 10 s of device discovery, only a fraction of a node's one-hop neighbors have been discovered.[1]

[1] The rate of discovered neighbors decreases with time due to the fact that an increasing number of nodes which handshake have already discovered each other. This de facto prevents to discover all the neighbors of a node in a reasonable amount of time. In section 4.2 it is shown that by increasing the time from 10 s to 20 s, only an increase in the fraction of discovered neighbors from 67% to 88% is achieved at $n = 110$.

However, we have verified that we can keep device discovery reasonably short while still obtaining that the resulting discovered topologies are connected, which is the necessary requirement for generating connected scatternets. The price to pay is that in the discovered topology it is not guaranteed that two nodes that are in each other transmission range discover each other. Given a set of BT nodes, we call *visibility graph* the network topology where there is a link between any two nodes whose Euclidean distance is less than or equal to the nodes transmission radius. (When the BT nodes are scattered in the plane, as it is for the solutions compared in this paper, these topologies are also often referred to as *unit disk graphs* (UDGs) [17].) Figure 2(a) shows the visibility graph network with 13 BT nodes. After the device discovery not all nodes that are in communication range of each other have been discovered. For instance, node 8 and node 13, although within communication range did not discover each other while running the device discovery. A possible *BT topology* resulting from the device discovery phase is shown in figure 2(b).

As detailed in the next section, this can be a problem for those protocols that use the assumption of complete neighbor knowledge to form connected scatternets with a bounded number of slaves per piconet.

## 3. Four scatternet formation protocols

We describe here the four protocols for scatternet formation in multi-hop networks and the solutions to the problems we encountered in implementing them. We start by describing a simple protocol for exchanging information among neighboring nodes throughout the network, which is needed in the implementation of some of the selected protocols. In the protocol descriptions, the term neighbors refers to the nodes that a node has discovered during the discovery phase.

### *3.1. The pecking protocol*

The mechanism we use for information exchange among neighboring nodes is the temporary set up of a piconet between every pair of neighboring nodes. For the formation of a piconet to be possible, we have to guarantee that every pair of neighboring nodes that need to set up a piconet among them are in opposite page modes. This is obtained in the following way, which makes use of a generic method to establish an ordering among the nodes according to their "weight", i.e., according to a number $\geqslant 0$ locally computed at each node. (For the purpose of protocol description, here we can consider the weight the node's unique identifier.) A node $v$ checks whether it has the bigger weight among its neighbors $N(v)$. If this is the case, that node, called in the rest of the paper an *init* node, executes the following procedure.

```
PECKORDER v
1  PAGEMODE
2  for each smaller u in N(v)
3     do PAGE(u, v)
4  EXIT
```

An init node goes to page mode and starts paging all its neighbors (which, by definition, are "smaller neighbors", i.e., nodes with a smaller weight) setting up temporary piconets with each one of them. While the piconet is up, the nodes exchange the information needed for the protocol operation. Then, the piconet is disrupted.

A non-init node $u$ executes the following procedure.

```
SUBNODEu
1  PAGESCANMODE
2  for each bigger v in N(u)
3     do WAITPAGE(u, v)
4  PECKORDER(u)
```

Node $u$ goes to page scan mode and waits for a page from all its neighbors with bigger weight ("bigger neighbors"). As soon as node $u$ has received information from the bigger neighbors (i.e., it has been paged by all of them), it becomes the "bottom of the pecking order", i.e., being now the bigger node among those with which it has to exchange the information, it switches to page mode, and starts setting up temporary piconets with all its smaller neighbors (if any).

### 3.2. BlueTrees

The scatternet formation protocol presented in [1] is the first to solve this problem in a multi-hop topology. The protocol is initiated by a designated node (the *blueroot*) and generates a tree-like scatternet.

The blueroot starts the formation procedure by acquiring as slaves its one hop neighbors. These, in turn, start paging their own neighbors (those nodes that are at most two hops from the root) and so on, in a "wave expansion" fashion, till the whole tree is constructed. In order to limit the number of slaves per piconet, it is observed that *if a node in a unit disk graph has more than five neighbors, then there are at least two of them which are in each other transmission range*. This observation is used to re-configure the tree so that each master node has no more than seven slaves. If a master $v$ has more than seven slaves, it selects two of them which are necessarily in each other transmission range, and instructs one of the two to be the master of the other, which is then disconnected from $v$'s piconet. Such "branch reorganization" is carried throughout the network leading to a scatternet where each piconet has no more than seven slaves.

Beyond producing a tree-like topology, which limits the robustness of the obtained scatternet, BlueTrees depends on a selected node to start the formation procedure so that this solution does not work in networks whose topology after the discovery phase is not connected. The original paper does not discuss the selection or the positioning of the blueroot.

We encountered the following two problems in implementing BlueTrees.

In order to limit the number of slaves per piconet to $\leqslant 7$, the protocol assumes that the topology resulting from the device discovery phase is such that, if two neighboring nodes $u$ and $v$ have discovered each other and they both have a common neighbor $z$, then either both discovered $z$ or neither of them did. This property, which is necessary for a reconfiguration of the BlueTree into a scatternet with no more than 7 slaves per piconet, cannot be guaranteed by the discovery phase as described earlier.

Consider, for example, a star-like topology with 9 nodes in which the central node $v$ becomes master and chooses 7 among its 8 neighbors. Assume neighbor $z$ is the one that has been left out of $v$'s piconet. For the geometric property mentioned above, at least one of $v$'s neighbors, say $u$, is in the transmission range of $z$. If $u$ and $z$ did not discover each other, chances are that the formed scatternet is not connected ($v$ cannot reach its neighbor $z$).

Therefore, for the protocol to correctly work we need to perform extra operations to achieve a consistent knowledge of each node's neighborhood. We call these operations, which aim at "bringing back" some links that were not discovered, the *replenish phase* of the protocol. This phase can be implemented in the following way.

At the end of a discovery phase all neighboring nodes that have discovered each other exchange the list of the nodes they just discovered. This list exchange can be performed by executing the pecking protocol as described above and leads to the construction at each node $v$ of a set $A_v$ of all nodes discovered by $v$'s neighbors that $v$ did not discover.

Once the list exchange is finished, node $v$ starts contacting the nodes in $A_v$ to see whether they are nodes within its transmission range (i.e., undiscovered neighbors). To this purpose, a node $v$ alternates for a predefined amount of time between page and page scan modes attempting to discover the nodes in its $A_v$. More specifically, when in page mode $v$ attempts to page one after another (round robin fashion) all the nodes in $A_v$. Each time two nodes $u$ and $v$ discover each other, they remove each other from their set $A_u$ and $A_v$ and exchange their lists of neighbors. This may lead to new nodes for $u$ and $v$ to be added to their sets $A_u$ and $A_v$, i.e., to new nodes to page. The length of this phase has to be carefully chosen so to discover all the nodes in $A_v$ that are actually in $v$'s transmission range.

The second problem concerns the way the BlueTrees protocol has been defined. According to [1], once a node has been acquired as a slave by a master, it becomes a master itself, and starts contacting (i.e., paging) all its neighbors (except its own master). Assume that a master $v$ has acquired two nodes $u$ and $z$ as slaves, and assume that $u$ and $z$ are nodes that discovered each other during the discovery phase. Then a deadlock situation may arise due to the fact that $u$ starts paging $z$ and vice versa. To solve this problem we have associated to the paging of neighboring devices a time-out. If a discovered neighbor does not reply to a page within a certain amount of time, it is assumed that it is in page mode itself and thus that it belongs to a piconet already.

### 3.3. BlueNet

The scatternet formation scheme proposed in [3], *BlueNet*, produces a scatternet whose piconets have a bounded number

$k$ of slaves. The following description is based on [3] and on personal communications with the authors.

After the device discovery phase, each node randomly enters the page or the page scan mode with probability $p_0$ (phase 0).

When a node succeeds in getting at least one slave, it proceeds to phase 1 and tries to acquire up to $k$ neighboring nodes as slaves. The nodes that are acquired as slaves also proceed to phase 1 along with their master. Slaves that move to phase 1 keep being in page scan mode. Masters that move on to phase 1 perform the twofold task of (1) getting to know neighboring masters and the masters of those neighboring slaves that affiliated with other piconets, and (2) finding nodes in phase 0 that are in page scan and acquiring them as slaves. To perform these tasks a master pages all its neighbors in a round-robin fashion, and then decides whether to go to page mode or page scan mode randomly with probability $p$. By alternating in this way between page and page scan, every master acquires up to $k$ among its neighbors as slaves and gets to know the needed information about the other neighbors.

If a node is unsuccessful in either acquiring at least a slave or in being invited to join some other node's piconet, it keeps executing phase 0, deciding again whether to be in page mode or page scan mode with probability $p_0$, until it becomes aware that all neighboring nodes are part of some other piconet. (The fact that a node in phase 0 can actually contact all its neighbors can be guaranteed only statistically.)

In case a node in phase 0 remains isolated it enters phase 2, i.e., it goes to page mode and tries to interconnect to neighboring piconets by acquiring as slave one node from each such piconets (up to $k$). After having accomplished this task, a node in phase 2 exits the protocol.

A master in phase 1 that has contacted all its neighbors and acquired at most $k$ of them in its piconet, proceeds to phase 3, the piconet interconnection phase. In this phase, the slaves of the piconets formed in phase 1, by alternating between page and page scan mode, attempt to set up links with neighboring slaves of other phase 1 piconets as instructed by their masters. Masters select one of their slaves to go to page mode and instruct all the other slaves to go to page scan mode. The selected slave pages its neighbors in a round-robin fashion, each for at most a given amount of time $t_{\text{page}}$. For each successful page the slave asks the master whether the corresponding link has to be set up or not. When all the attempts to paging the needed neighbors has been made, the master instructs the slave to go to page scan mode and selects another slave to go to page mode. When only one slave has remained with some neighboring slaves to contact, this slave is instructed to alternate between page and page scan mode in order to set up the remaining links. The whole process terminates when all needed links to adjacent piconets have been established.

The connectivity of the resulting scatternet is not guaranteed, i.e., not all the BlueNets are connected, even when the topologies resulting from the discovery phase are. A simple counter-example is depicted in figure 3.

Assume that nodes M1 and M2 discovered each other (dotted line) and that they are the only two nodes that go to page mode in phase 0. Assume also that they successfully invite to join their piconet the 7 leftmost neighbors and the 7 rightmost neighbors, respectively. In this case, the nodes in both piconets proceed to phase 3. However, there is no way to interconnect the piconets either via intermediate gateways (no slaves in piconet M1 are neighbors of any of the slaves in piconet M2, and vice versa) or via interconnecting directly M1 and M2, since they have already 7 slaves each.

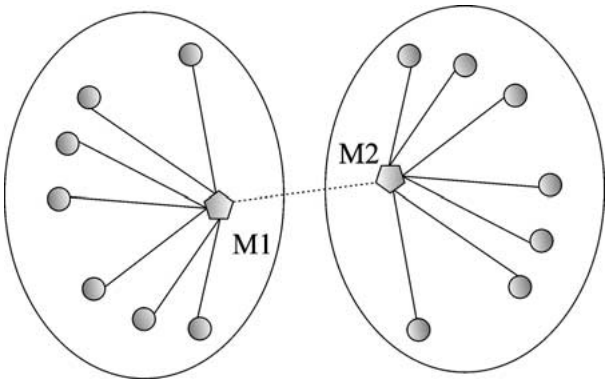



Figure 3. A disconnected BlueNet.

According to the description of the protocol [3], slaves in a piconet must be prevented from establishing connections with nodes belonging to their own piconet. Furthermore, multiple interconnections between the same pair of adjacent piconets have to be avoided. The former requirement is achieved by having the master informing each of its slaves about the identity of any new slave joining the piconet. Our choice for performing this is by using LMP packets from the master to its slaves. For the master (slaves) to successfully transmit (receive) such packets, slaves that have joined a piconet enter *sniff mode*, periodically stopping page, page scan activities or connection activities and tuning on the piconet frequency hopping sequence to listen to their master's messages. (For details on the BT LMP layer and sniff mode the reader is referred to [5].) In order to avoid multiple connections between the same pair of piconets, every time a slave creates or joins a temporary piconet with a slave of a different piconet, it contacts its master to check whether a connection to this piconet exists already [3]. If this is not the case, the master informs its other slaves of the new inter-piconet connection in the first upcoming sniff window.

BlueNet is described by the following example. We start again from the network depicted in figure 2(b) as the network resulting from the device discovery phase. (In this case, there is no need for the replenish phase.) We assume that the number associated to each node represents its unique ID. In the initial phase 0, nodes 6, 11, 17, 21 and 22 randomly decide to start paging their neighbors. All other nodes go to page scan. As soon as node 6 is successful in acquiring node 13 as slave, it moves on to phase 1, along with node 13. The same happens to node 11 and 30, that move to phase 1 (node 11 is the master and node 30 is its slave). Similarly, nodes 17 and 21 acquire nodes 36 and 20 as slaves, respectively. The four nodes all proceed to phase 1. Node 6 enlarges its piconet by acquiring nodes 7 and 25 as slaves, and node 8 joins the

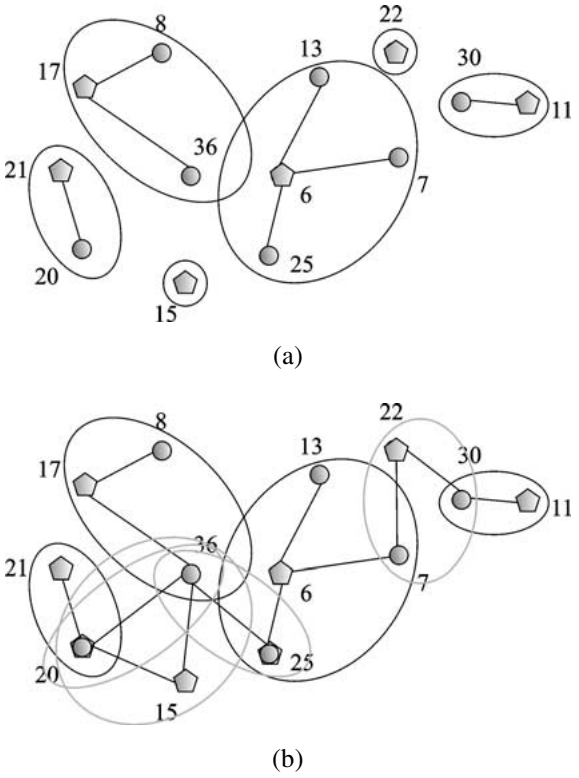



Figure 4. (a) BlueNet's "original piconets", and (b) the final BlueNet.

piconet of node 17. When node 22 succeeds in paging nodes 7, 13 and 30, it realizes that these nodes have already joined some other piconet. Being an isolated master, node 22 moves on to phase 2. Node 15 is waiting for a page from a neighboring node. Since no such page arrives, after a certain time it randomly decides whether to keep staying in page scan mode or to alternate to page mode. When it finally switches to page mode and succeeds in contacting its two neighbors 20 and 36, finding them already enslaved to some other masters, it moves on to phase 2. Figure 4(a) shows the "original piconets" [3], i.e., the piconets formed by the nodes in phase 1 and phase 2.

Nodes 15 and 22, which are in phase 2, page all their neighbors in order to connect to adjacent piconets. Node 22, for instance, pages its neighbors 7, 13 and 30, until it has contacted them all. In communicating to node 30, node 22 asks it to set up a link between piconet 22 and piconet 11. Node 30 relays this request to its master, node 11, which agrees to share node 30 as gateway slave. Similarly, 22 makes the same request to node 7, and node 6 agrees to share node 7 as gateway slave. When node 22 does the same with its neighbor node 13, node 13 asks its master node 6 for the permission of joining piconet 22, but in this case, node 6 does not agree to share node 13 since piconet 6 is already interconnected to piconet 22, and the protocol specifications demands no multiple links between adjacent piconets. Similarly, node 15 enlarges its piconet to include nodes 20 and 36 as slaves, thus establishing links to piconets 21 and 17, respectively. Nodes 15 and 22 exit the execution of the protocol at this time.

Once they have contacted all their neighbors, masters in phase 1 proceed to phase 3, along with all their slaves. In this phase adjacent piconets are interconnected via intermediate gateways. Node 21 instructs its only slave, node 20, to alternate between page and page scan in order to contact all slaves of adjacent piconets. Node 36 and 20 will eventually set up a link to interconnect piconets 17 and 21, respectively. In the case depicted in figure 4(a), node 20 becomes the master of an extra piconet including node 36 as slave. Piconet 25, which also includes node 36 as slave, is similarly formed to interconnect piconets 6 and 17. Node 13 receives notice that node 36 has been linked to piconet 6 via node 25. Therefore, it does not contact node 36. The resulting BlueNet is depicted in figure 4(b) (the extra piconets are depicted as light gray ovals).

### *3.4. BlueStars*

The protocol presented in [2], *BlueStars*, proceeds from the device discovery phase into the following two phases of piconet formation and of the interconnection of the piconets into a connected scatternet. Based on a locally and dynamically computed weight (a number that expresses how suitable that node is for becoming a master) and on the knowledge of the weight of its neighbors (obtained during the discovery phase) each node decides whether it is going to be a master or a slave. This decision is taken at a node depending on the decision of the bigger neighbors, and then communicated to the smaller neighbors. The mechanism through which this is implemented is similar to the pecking protocol described above. In particular, a node that decides to be master is either an init node or a node whose bigger neighbors decided all to be slaves. A node that has been told (via paging) by one or more of its bigger neighbors that they are masters, becomes the slave of the first master who paged it. This phase of the protocol leads to the partition of the topology resulting from the discovery phase into piconets with one master and a number of slaves which is not necessarily bounded (if not by the number of the neighbors of the master). Notice that no two masters can be neighbors.

After this phase, each master proceeds to the the selection of *gateway devices* to connect multiple piconets so that the resulting scatternet is connected. In order to achieve connectivity it is necessary (and sufficient) that each master establishes a path with (i.e., chooses gateways to) all the masters that are two or three hops away [12]. More precisely, masters are two hops away if they can be interconnected by gateway slaves (this is the preferred way of interconnecting adjacent piconets). Masters are three hops away, if they are not two hops away and there is a pair of intermediate gateways through which they can be joined. The knowledge about which nodes are the masters two and three hops apart is achieved during the piconet formation phase. Specifically, each node $v$ communicates its role (and possibly the identity and weight of its master) to all its smaller neighbors and to the bigger neighbors that are slaves. If a node is a slave, it waits for the smaller neighbors to communicate the same information. In this way, at the end of the piconet formation phase each node is aware of all its neighbors ID, and of the ID and

weight of their masters, which are the information needed in the piconet interconnection phase.

The process of piconets interconnection is based again on a mechanism similar to the pecking protocol, this time executed only among the masters. The details of the rule adopted for consistent gateway selection and for piconets interconnection, as well as a detailed example of BlueStars operations, can be found in [2].

### *3.5. LSBS protocol*

The main aim of the *Li–Stojmenovic/BlueStars protocol* (LSBS), obtained by combining the Yao construction proposed in [4] and BlueStars, is to build up a connected scatternet in which each piconet has no more than seven slaves.

The protocol assumes that each node knows its own identity, a dynamically computed weight that indicates how much that node is suitable for serving as a master (as in BlueStars), and its own location in the plane (usually provided by an onboard GPS device, or by any suitable inertial positioning system device). It is assumed that, as the outcome of the device discovery phase, a node also knows the identity of its neighbors, their weight and their location. By using the pecking protocol, the discovered devices also exchange information about their neighbors (achieving two-hop neighborhood knowledge).

In the description of the algorithm given in [4] it is assumed that nodes are scattered in the plane and that the network graph resulting from the device discovery phase is a connected unit disk graph.

Given that the discovery phase does not produce an UDG, in the simulation experiments that we performed, we run the extra phase described for BlueTrees in order for this protocol to correctly work (replenish). Note that in this case, since node $v$ is aware of the location information of its two-hop neighbors, only the nodes expectedly in $v$'s transmission range need to be included in $A_v$. This reduces consistently the number of nodes to be paged compared to the number of nodes to be paged by a node executing the replenish phase of BlueTrees.

The knowledge of the location is then exploited for applying to the "replenished topology" geometric-based techniques to reduce the degree of the network to at most $k \leqslant 7$. The Yao construction is executed at each node $v$ and proceeds as follows. Node $v$ divides the surrounding plane into $k$ equal angles. In each angle, node $v$ chooses the closest neighbor $u$, if any. (Ties are broken arbitrarily.)

A link between nodes $v$ and $u$ survives the Yao construction phase if and only if $v$ has chosen $u$ and vice versa. All other links are deleted. To make such decision nodes need to exchange with their neighbors the information on the nodes they selected. This is performed by using again the pecking protocol. Despite some links being deleted, the Yao construction guarantees the connectivity of the resulting topology [4,14].

Once a connected topology with such a bounded degree has been obtained, the BlueStars algorithm for scatternet formation outlined above uses the nodes' weight for selecting the masters, the slaves and the gateways necessary to form a degree-bounded connected scatternet.

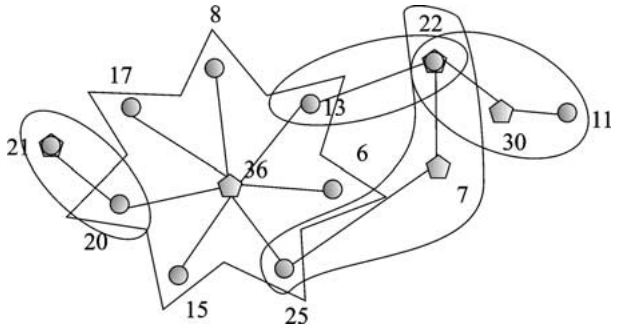



Figure 5. The scatternet resulting from the LSBS protocol.

Table 1
The four protocols for multi-hop scatternet formation and the properties of the resulting scatternet.

| Protocols | Properties | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| BlueTrees | ⋆ | | | ⋆ | |
| BlueNet | | ⋆ | ⋆ | ⋆ | |
| BlueStars | ⋆ | ⋆ | ⋆ | | ⋆ |
| LSBS protocol | ⋆ | ⋆ | ⋆ | ⋆ | ⋆ |

Let us consider again the network of figure 2(b) as the network resulting from the device discovery and the replenish phases. The only node with more than 7 neighbors is node 36. Therefore, it executes the Yao construction procedure to discard one of its 8 neighbors. Assuming that nodes 20 and 21 fall in the same of the 7 angles in which node 36 has partitioned the plane around itself, the result of the Yao construction phase is the cancellation of the link between node 21 and node 36. The following execution of BlueStars over the Yao topology leads to the same scatternet we would obtain with BlueStars except for the fact that now node 36 and 21 are no longer neighbors. Therefore, being node 21 the bigger in its neighborhood, it becomes a master and acquires node 20 as its slave. In the final piconet interconnection phase node 20 is chosen as gateway slave between master 21 and master 36.

We conclude this section by summarizing, for each of the four protocols compared in this paper, the desirable properties that the formed scatternet satisfies. These properties are: (1) formation of connected scatternets; (2) ability of functioning in disconnected networks; (3) formation of scatternets with multiple routes; (4) formation of scatternets with a bounded number of slaves per piconets; and (5) resource-based (weight-based) master selection.

We notice that the fact that the LSBS protocol is a "five stars" protocol relies on the fact the property 5 is brought in by BlueStars, which implements the LSBS protocol phases of piconet selection and interconnection. As explained, the trade-off here is the need for extra hardware at each node (positioning devices).

All the considered protocols rely on a phase of device discovery which precedes the phases of piconet connection and

interconnection. BlueTrees, BlueStars and the LSBS protocol guarantee the connectivity of the produced scatternet, give that the BT topology is connected (as proven in the corresponding papers). As shown above, BlueNet does not provide such deterministic guarantee.

## 4. Comparative performance evaluation

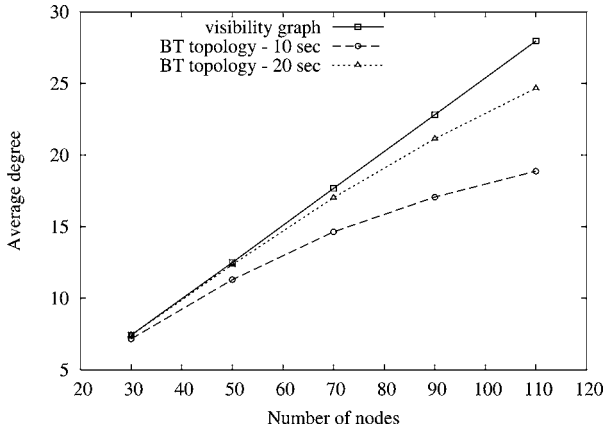
### *4.1. Simulation environment*

To evaluate the performance of the four scatternet formation protocols we have developed a Bluetooth extension to the VINT project network simulator ("ns2") [15]. We based our extension on BlueHoc, the ns2-based simulator released by IBM [18]. In particular, in order to implement the operations of the solutions for device discovery and scatternet formation, we have enriched BlueHoc with mechanisms for (a) giving to each node the possibility to dynamically assume either the role of master or the role of slave, (b) handling collisions that might arise during the establishment of a link, (c) parking/unparking slaves, (d) entering sniff and hold modes, and (e) having a node alternating between inquiry and inquiry scan. (These functions are not available in BlueHoc.) Our simulator strictly follows all the specifications of the BT protocol stack concerning scatternet formation.

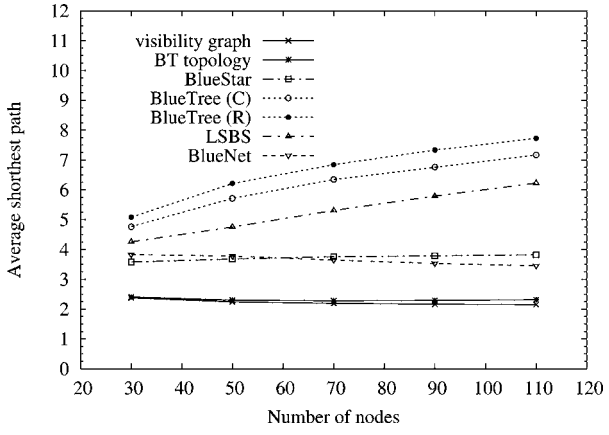
In the simulated scenarios, $n$ Power Class 3 BT nodes (i.e., nodes with maximum transmission radius of 10 meters) are randomly and uniformly scattered in a geographic area which is a square of side $L$. We make the assumption that two nodes are in each other transmission range if and only if their Euclidean distance is $\leqslant$ 10 m. As mentioned, we call *visibility graphs* the topologies generated by drawing an edge between each pair of BT devices that are in each other transmission range (radio visibility).

In the simulation results presented here, the number of BT nodes $n$ has been assigned the values 30, 50, 70, 90 and 110, while $L$ has been set to 30 m. This allowed us to test our protocol on increasingly dense networks, from (moderately) sparse networks, where only 95% of the visibility graphs are connected ($n = 30$), to highly dense networks. The average degree ranges from 27.9 for $n = 110$ down to 7.4 when $n = 30$ (see figure 6(a)). As the density increases, the average shortest path length in the visibility graph slightly decreases (10%) from 2.37 to 2.14, as shown in figure 6(b).

The simulated scenarios refer to the case when all the nodes enter the network within a time $T_{\text{acc}}$ (set to 100 ms) and run the device discovery and scatternet formation protocols to self-organize themselves in a Bluetooth scatternet. The value assigned to every node's clock at simulation start is drawn randomly and uniformly in the range 0 and $2^{27} - 1$. This allows us to evaluate the performance of device discovery in the worst case in which nodes in inquiry use frequencies from both trains A and B. All results presented in this section were obtained by running the four protocols on 300 connected visibility graphs.




(a)




(b)

Figure 6. (a) Average degree and (b) average shortest path length in the various obtained topologies ($T_{\text{disc}} = 10$ s).

### *4.2. Simulation results*

Our simulations concern the two main aspects of scatternet formation, namely, device discovery, which is common to all protocols, and piconet formation and interconnection. In particular, simulations have been conducted to compare the performance of the different solutions which include (1) measuring the time needed for scatternet formation (including the phase of device discovery); (2) assessing the effect of the duration of the device discovery phase on the entire scatternet formation process; (3) counting the average number of piconets; (4) counting the number of slaves per piconet (average and 95th percentile); (5) counting the number of roles (either master or slave) assigned to each node (average and nature of roles); (6) comparing the average length of the routes between any two BT devices in the scatternet to the average shortest path length between any pairs of nodes in the visibility graph; and (7) comparing the overhead (number of transmitted packets) associated with running the different scatternet formation protocols.

#### *4.2.1. Device discovery in multi-hop networks*

We have run the device discovery phase for a predefined time $T_{\text{disc}} \leqslant 20$ s over each visibility graph. Nodes alternate be-

tween inquiry and inquiry scan mode, spending a variable time, uniformly and randomly selected in the interval (0.01 s, $T_{\text{inq}}$), in each mode. Unless otherwise specified $T_{\text{inq}}$ has been set to 0.5 s. In the case a node is in backoff when the inquiry scan interval expires, we choose to let the node finish up the backoff and try to perform the inquiry handshake for 256 clock ticks (inquiry response) before switching to inquiry mode. Nodes in inquiry mode randomly select the frequency train they use upon entering the inquiry phase. The topology resulting from the device discovery phase, which we call a *BT topology*, has links only between those pairs of BT nodes that were able to discover each other during the device discovery phase.

The effect of different discovery phase durations are shown in figure 7. We notice that within 20 s not all neighbors are discovered (on average) by each node. The higher the density (and the nodes average degree), the longer the time needed to discover all the neighbors. When $T_{\text{disc}} = 6$ s, the percentage of discovered neighbors ranges from 87% ($n = 30$) down to 49% ($n = 110$) (part (a) of the figure). When $T_{\text{disc}} = 10$ s and $= 20$ s this percentage increases, ranging from 96% down to 67% and from 99% down to 88%, respectively. The number of discovered neighbors increases with the length of the device discovery phase. However, the rate at which new neighbors are discovered decreases with time, as there is a higher chance to handshake again with nodes already discovered. The results obtained for $T_{\text{disc}} = 20$ s show that it is extremely time-consuming to require that each node becomes aware of all its neighbors.

A necessary condition for a protocol to form a connected scatternet is to start from a connected BT topology. Therefore, we have investigated how long the device discovery must run in order for each node to discover enough neighbors to obtain a connected BT topology. As shown in figure 7(b), we can provide statistical guarantee that the BT topologies are connected in case of moderately dense to heavily dense visibility graphs provided that the device discovery runs for at least 6 s.

A short device discovery reduces the overall time needed for scatternet formation as well as corresponds to a significant decrease of the average degree of the BT topologies with respect to the degree of the corresponding visibility graphs. This can be beneficial for protocols like BlueStars that do not form scatternets with limited number of slaves per piconet. When $T_{\text{disc}} = 10$ s, the average nodal degree decreases from 3.5% ($n = 30$) to the more significant 32.6% ($n = 110$). (The price to pay in this case is only a small increase in the average shortest path length which ranges from 1% when $n = 30$ to 6% when $n = 110$ as shown in figure 6(b).) When $T_{\text{disc}} = 20$ s the decrease of the average nodal degree with respect to the visibility graph is much more limited, ranging from a mere 0.3% ($n = 30$) to a 12% ($n = 110$) decrease, as shown in figure 6(a).

To evaluate the impact of the parameter $T_{\text{inq}}$, we have performed simulations varying $T_{\text{inq}}$ in the set {0.2 s, 0.5 s, 1 s, 2 s}. We observed that when decreasing $T_{\text{inq}}$ from 2 s down to 0.5 s the percentage of discovered neighbors increases independently of the network density. This is due to the fact that

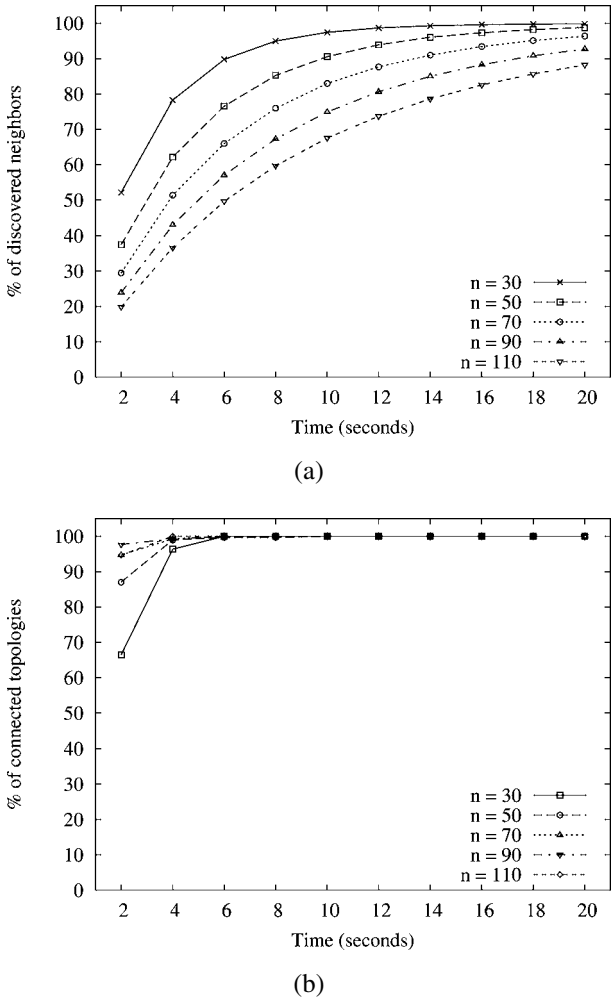



Figure 7. (a) Average percentage of discovered neighbors, and (b) corresponding average percentage of connected BT topologies.

a shorter $T_{\text{inq}}$ allows the nodes to alternate more often, and to switch faster between train A and train B when in inquiry mode. For smaller values ($T_{\text{inq}} = 0.2$ s) we have observed that in sparse topologies (e.g., when $n = 30$) the percentage of discovered neighbors decreases with respect to the case $T_{\text{inq}} = 0.5$ s. In this case, it may happen that nodes in inquiry scan either end this phase without starting a backoff at all or when coming out of a backoff receive no ID packet from their neighbors (that are likely to be in inquiry scan as well), leading to a degraded performance. This is why we selected $T_{\text{inq}} = 0.5$ s as the value we used in all our simulations.

We have identified three features of the BT technology which are responsible for the device discovery for multi-hop BT networks being an extremely time consuming operation: (a) the need to adopt (stochastic) mechanisms to have neighboring nodes in opposite inquiry modes, so they can discover each other; (b) the overly long duration of the backoff interval as stipulated in the BT specifications (2048 clock ticks); and (c) the impossibility of identifying the inquirer, which demands the construction of a temporary piconet (and the amount of time needed for an handshake) between neighbors that discovered each other already.

In our performance evaluation we have attempted to quantify the impact of each of the above effects. First we have run the device discovery using a shorter backoff interval length (one fourth of what specified in the Bluetooth specifications). As shown in figure 8 by reducing the backoff interval length the percentage of discovered neighbors improves remarkably. After 6 s of device discovery the whole visibility graph is discovered in moderately dense networks ($n = 30$), while at $n = 110$ the improvement ranges from 110.5% after 2 s of device discovery to 34.8% after 10 s. In case of reduced backoff length, after 20 s all neighbors are discovered (an increase of 12.4%).

We have then made a major modification of the ID packet format, to allow it to carry the identity of the inquirer. Upon receiving an ID packet, nodes in inquiry scan mode check whether the inquirer is an unknown neighbor. Only in this case they proceed according to the usual procedures (either backing off or answering the packet with an FHS packet that initiates the set up of a temporary piconet). Otherwise, the packet is ignored. The trade-off here is that we had to "slow down" the scan of the inquiry frequencies. Instead of an ID packet every half a slot, since the ID packet carries now the inquirer's address (an additional 48 bits), it takes a whole slot to send the enlarged ID packet.

As shown in figure 8 we have verified that the benefits obtained from these modifications overcome the possible performance degradation due to the extra time needed for two neighbors to start handshaking except for sparse topologies ($n = 30$). In this case, the smaller average degree ($< 8$) and the high number of nodes in inquiry scan (due to the short $T_{\text{inq}}$) makes it difficult for a node exiting backoff to find neighbors that are in inquiry mode, on the same train and that can send an ID packet within the inquiry response time. This nullifies the advantage of the modified inquiry which allows a node exiting the backoff to wait for an undiscovered neighbor before actually handshaking (paging). What we actually notice here is a slight decrease in the percentage of neighbors discovered in the first 8 s over the specification compliant case which is due to the slower scanning of the inquiry frequencies. As the nodes density increases ($n \geqslant 50$), the curve representing the performance of device discovery with modified inquiry procedures always outperforms the performance of device discovery with the standard parameters. Figure 8(b) shows the case with $n = 110$. We notice that as the time spent in the discovery phase (and thus the number of discovered neighbors) increases, the improvement achieved by the modified inquiry steadily increases till 10 s (from 16% after 6 s to 23.7% after 10 s at $n = 110$), since it is more and more likely that two neighbors that already discovered each other would do it again. After 10 s the knowledge of all neighbors gets closer and closer to 100% which is reached at 20 s.

Finally, we have considered the combined cases of shorter backoff and ID packet carrying the inquirer identifier. This leads to very good performance, as all neighbors are discovered within 6 s at $n = 30$, and within 8 s in the heavily dense case of networks with 110 nodes. The percentage of discovered neighbors improves from 148% after 2 s down to 12% after 20 s over the percentage of discovered neighbors when specifications mandated values are used ($n = 110$).

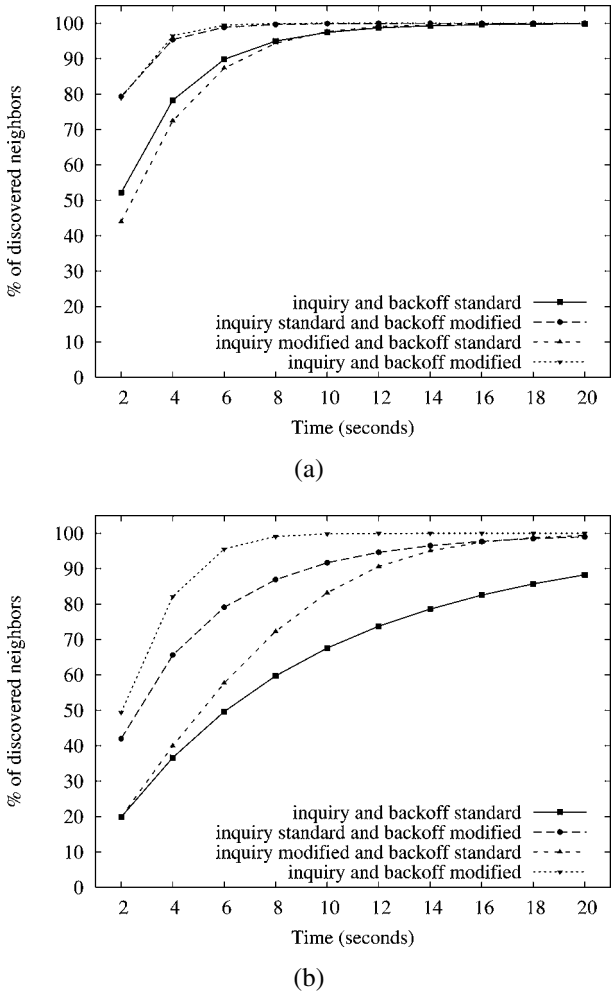



Figure 8. Average percentage of discovered neighbors in networks with (a) 30 nodes, and (b) 110 nodes with modified backoff and modified inquiry procedure.

### *4.2.2. Performance evaluation comparison of the four scatternet formation protocols*

The comparative performance evaluation aimed at investigating the time needed to complete the four protocols as well as metrics identified in the literature as measures of the "quality" of the generated scatternets. The BlueTrees performance results refer to the case in which a central node is designated as blueroot as well as to the case when the blueroot is placed randomly within the geographic area. In implementing BlueNet we have followed the suggestions given by the authors to speed up the protocol operations. Specifically, whenever a link is set up between nodes belonging to different piconets, the piconets composition is exchanged. The gathered information is then communicated to all the other members of the piconet. This significantly reduces the number of neighbors that each node has to contact during BlueNet operations, and thus the overall protocol duration.

The parameters used for the simulations of BlueNet are summarized in table 2. In the table, $p_0$ and $p$ are the proba-

Table 2
BlueNet parameters.

| $p_0$ | $p$ | $t_{\text{page}}$ (slots) | $I_{\text{sniff}}$ (slots) | $W_{\text{sniff}}$ (slots) |
|---|---|---|---|---|
| 0.2 | 0.5 | 128 | 300 | 32 |

bilities of entering page mode when in phase 0 and in phases 1 and 3, respectively. The parameter $t_{\text{page}}$ denotes the maximum number of slots for which a node attempts to page one of its neighbors; $I_{\text{sniff}}$ and $W_{\text{sniff}}$ denote the interval between sniff windows and the sniff window length. The first two parameters have been set according to the authors specifications, while the latter three have been tuned by means of simulations.

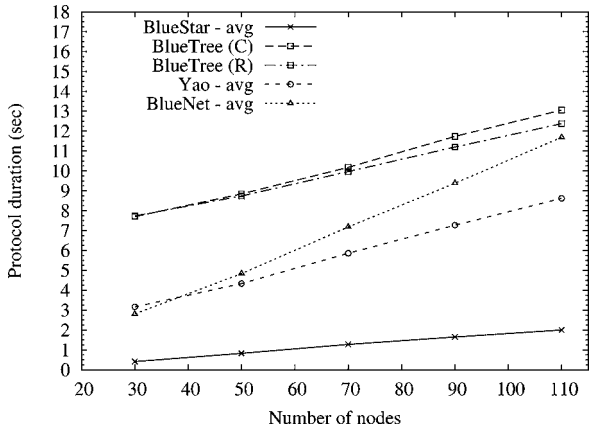
*Protocol duration*

Figures 9(a) and (b) depict the average time needed by BlueStars, BlueTrees, BlueNet and by the LSBS protocol to form a scatternet starting from the BT topologies obtained with device discovery that lasts 10 s and 20 s, respectively.

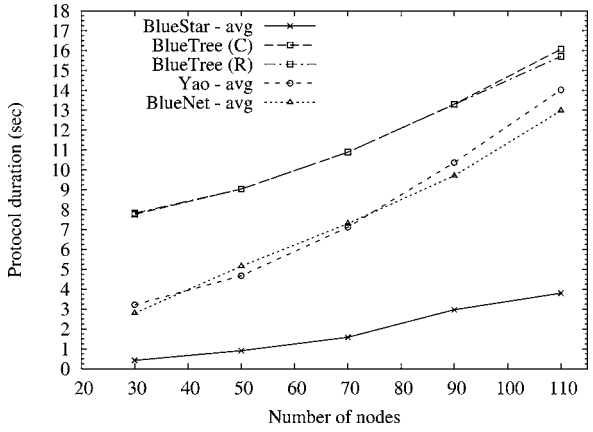
BlueStars is the fastest protocol. After a 10 s long device discovery, it requires an average of 2 s to build a connected scatternet in networks with 110 nodes, while the other three protocols require an average of 8.65 s (LSBS), 12.32 s (BlueTrees) and 11.7 s (BlueNet). The 99th percentiles of the considered protocols when $n = 110$ are 2.52 s (BlueStars), 10.2 s (LSBS), around 13.6 s (BlueTree), and 20.22 s (BlueNet).

The major reason for BlueStars being faster than the LSBS protocol is that the latter needs to run the extra phases with which every pair of nodes $u$ and $v$ that have discovered each other exchange their neighbor lists (via the pecking protocol), start paging all nodes in their sets $A_u$ and $A_v$ (replenish) and exchange information (again via the pecking protocol) on the links selected to be part of the Yao topology (Yao construction phase). For instance, the neighbors lists exchange that immediately follows the device discovery phase lasts up to 3.2 s at $n = 110$. The replenish phase needs to be performed for at least 2 s to have the statistical guarantee that all the nodes in the set $A_v$ have been contacted by node $v$ and that all the needed links are set up. Since almost all links in the visibility graph are discovered during this phase, a very high number of neighbors have to be paged in the Yao construction phase, making the pecking-like exchange of information a phase that further adds time to the overall duration (a maximum of 2.3 s on average, in networks with 110 nodes). The last part of the LSBS protocol is faster, since it is implemented by BlueStars now applied to sparser Yao topologies. The time needed for the LSBS protocol to complete its operations significantly increases with the number of nodes as this results in a higher number of neighbors to contact.

BlueTrees also needs to run the replenish phase before the actual tree construction can be started. Since no location information are available, no node can be automatically deleted by the set of potential neighbors $A_v$ as in the case of the LSBS protocol. This imposes a longer time to successfully complete the replenish process. Based on our simulation results we had to set the length of such phase to 4 s to have the statistical




(a)




(b)

Figure 9. Scatternet formation duration: (a) 10 s long device discovery, and (b) 20 s long device discovery.

guarantee of setting up all the links needed for BlueTrees to operate correctly. Furthermore, the protocol duration reflects the need to adopt time-outs to solve possible deadlocks in the formation of the BlueTree. (Such time-outs have been set to 3 s.)

The time needed to run BlueNet is higher than the time required by BlueStars and LSBS, and increases significantly with the network density, approaching the worst performing protocol, BlueTree, when $n = 110$. BlueNet performance is highly variable. This is shown by the increase of the 99th percentiles over the average (20.22 s over 11.7 s, when $n = 110$). The 99th percentile of BlueNet ranges from 7.25 s ($n = 30$) up to 20.22 s ($n = 110$), significantly higher than all the other protocols.

The first phases of the protocol are quite fast: the duration of phase 1 varies from 0.3 s to 0.49 s as the number of BT devices (and thus the number of neighbors to contact) increases from 30 to 110. The duration of phase 1 increases with $n$, from 0.7 s to 3.3 s, as well as the duration of phase 2, which increases from 0.8 s to 1.9 s. The longest phase is phase 3, which ranges, on average, from 1.8 s when $n = 30$ to 7.9 s when $n = 110$. The length of this phase is due to several factors: The need to ask for the master approval every time

a slave attempts to set up a link with a neighboring slave; the high number of nodes to be contacted (the slaves's neighbors); the need for each pair of neighboring slaves to wait to be both in phase 3 before interpiconet links between them can be established. The delay imposed by nodes entering phase 3 at different times depends on the fact that BlueNet relies on potentially time consuming mechanisms, e.g., alternating between page and page scan modes and waiting for sniff windows for communication exchange. The duration of these operations may lead to significant differences in the time needed by a node to complete a phase depending on whether information exchange can be completed successfully or not, on how many times a node has to alternate, etc. As shown earlier, differently from the other protocols, BlueNet may result in disconnected scatternets. The percentage of such scatternets is significant in case of moderately dense networks (8.7% at $n = 30$) but quickly decreases (to 0.6% or less at $n = 50$ and $n = 70$) when the density increases. For highly dense networks ($n \geqslant 90$) all the scatternets generated by BlueNet were found to be connected.

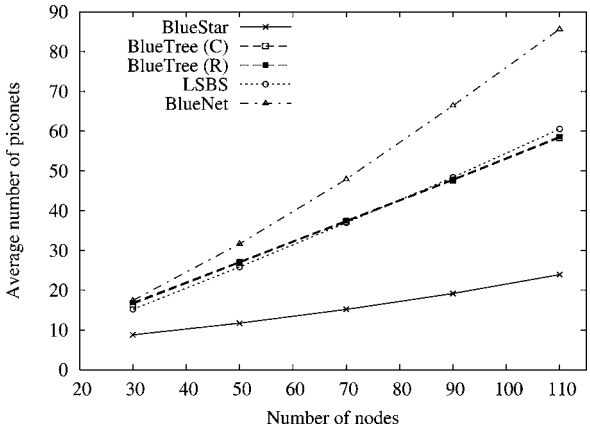
Figure 9(b) depicts the case when scatternet formation is executed over BT topologies obtained after 20 s of device discovery. Being the topologies denser, the duration of the scatternet formation phase increases. This is clearly shown in the case of BlueStars, which is naturally sensitive to the higher network degree of the BT topologies obtained after 20 s of device discovery. The LSBS and the BlueTree protocols also suffer an increase in protocol duration, which is particularly evident at high densities. This is due to the increased number of nodes to contact and to the increased amount of information to be exchanged. The time needed for the LSBS protocol to produce a connected scatternet is affected by longer neighbors lists to be exchanged (6.52 s when $n = 110$) and by a longer Yao construction phase (4.3 s when $n = 110$), leading to an overall algorithm duration which is close to BlueNet.

We notice that as for scatternet formation duration, the placement of the blueroot in BlueTree does not affect the performance of the protocol.

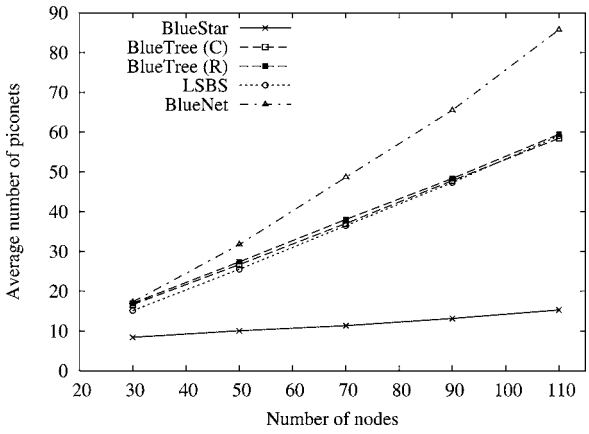
*Piconet number and number of slaves per piconet*

The number of piconets and their size affect the performance of BT communications. The magic number for piconet size is 7, this being the maximum number of slaves that can actively communicate with the piconet master. Scatternet formation protocols that produce scatternet with more than 7 slaves will incur the overhead of parking and unparking to manage all the slaves (this is the case of BlueStars). Even for protocols that guarantee piconets with less than 8 slaves, there is a tradeoff among small piconets resulting in a higher data rate per slave and increased complexity (e.g., for scheduling and routing) and interference associated to a higher number of piconets.

Figure 10(a) shows the average number of piconets in the scatternets formed by the four protocols in the case of BT topologies generated after a 10 s device discovery phase. In a BlueTree, all nodes but the leaves are masters, resulting in a high number of piconets which varies from 53% to 56% of the



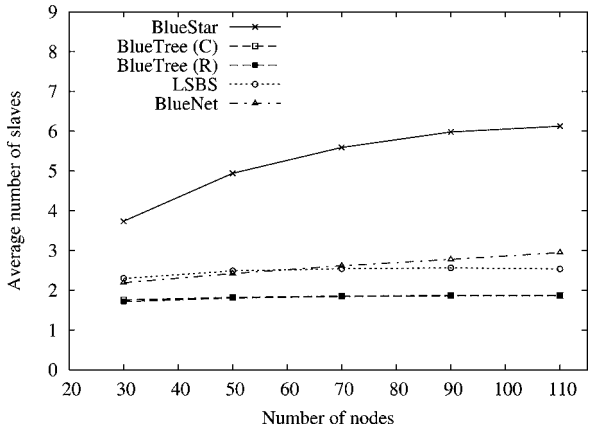

(a)



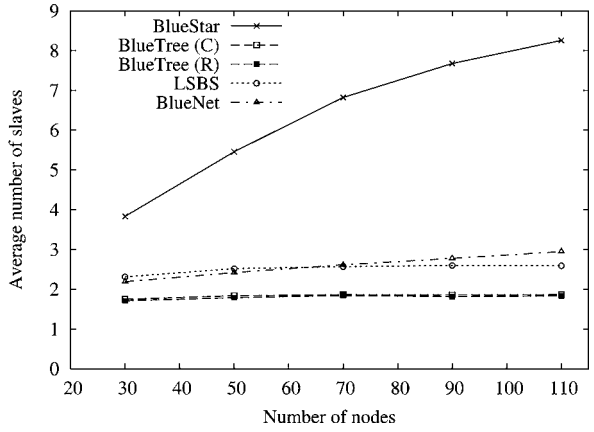

(b)

Figure 10. Average number of piconets: (a) 10 s long device discovery, and (b) 20 s long device discovery.

total number of nodes (independently of the placement of the blueroot). The number of piconets is much lower in BlueStars, ranging from 22% to 29% of the number of nodes. As the number of nodes increases, a higher number of piconets are obtained, as expected, to partition the nodes into piconets. When a bound on the maximum number of slaves per piconet is enforced, like in the LSBS case, the number of piconets in the scatternet further increases. This, in turn, results in a higher number of extra piconets needed for interconnecting adjacent piconets (like piconet E in figure 1), motivating the overall 60–153% increase in the number of piconets formed by LSBS with respect to the number of BlueStars piconets. Even more "interconnection piconets" are needed in BlueNet, which partitions the network in a higher number of small piconets. This is due to several factors, which include the enforcement of a bound on the number of slaves per piconet, and the fact that a node in phase 0 becomes a master probabilistically, and this probability increases with time. This generates more masters, and more inter-master competition to acquire slaves. Adjacent piconets are then always interconnected via intermediate gateways (which form an extra piconet), leading to an overall percentage of piconets ranging from 58% to 78.8% of the network nodes.




(a)




(b)

Figure 11. Average number of slaves per piconets: (a) 10 s long device discovery, and (b) 20 s long device discovery.

In the case of scatternet formation run on BT topologies from a 20 s device discovery, the performance of the four protocol is very similar. We note a decrease in the number of piconets generated by BlueStars. This is due to the higher density of the BT topologies and to the lack of the bound on the number of slaves per piconets which leads to forming a smaller number of bigger piconets.

The average number of slaves per piconet is depicted in figure 11. BlueTree, BlueNet and LSBS result in piconets with a very limited number of slaves (the 95% of the piconets has less than 4 slaves in BlueNet and BlueTree and less than 5 in LSBS). All the three protocols guarantee that no piconet has more than 7 slaves. The size of the "bigger" piconets instead exceeds the threshold of 7 in BlueStars. As mentioned, we use the duration of the device discovery phase as a "tuning knob" for obtaining discovered topologies with a decreased nodal degree. Doing so we could limit the 95th percentile of the number of slaves per piconet to 13 after 10 s of device discovery (instead of 24 after 20 s of device discovery). This, however, still requires parking and unparking of slaves, which may lead to time and bandwidth inefficiencies.

*Average number of roles per node*

Critical performance measures for Bluetooth scatternets are the (average) number of roles assigned to each node and the number of nodes with master–slave roles. A high number of roles per node translates into reduced throughput performance due to the overhead (an average of 2 slots) associated to piconet switching. Master–slave roles (e.g., the roles assumed by a master–master gateway) are inefficient since all the communications in the piconet of the master that switches to be slave have to be "frozen".

In figure 12 we show the average number of roles assumed by BT devices in the scatternets produced by BlueTrees, BlueStars, BlueNet and by the LSBS protocol.

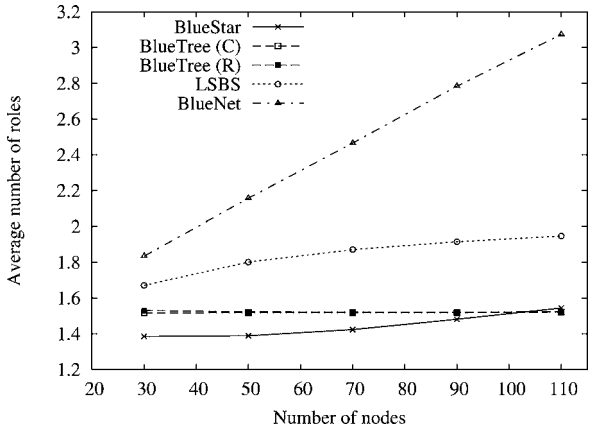
The average number of roles per node in the BlueStars, LSBS and BlueNet protocols on scatternets obtained on BT topologies from a 10 s device discovery increases with $n$ (from 1.2 at $n = 30$ to 1.5 at $n = 110$ in BlueStars, from 1.6 at $n = 30$ to 1.9 at $n = 110$ in LSBS, from 1.8 at $n = 30$ to 3.1 at $n = 110$ in BlueNet) to take into account the higher number of adjacent piconets that need to be joined (and thus the number of gateways). In BlueTrees all internal nodes have two roles, while the leaves only assume the role of slave. Since the percentage of nodes which are masters do not significantly change with $n$, the average number of roles per node remains steadily around 1.5, independently of $n$.

A major difference among the protocols is that BlueTrees adopts master–master gateways for piconet interconnection, BlueNet may use master–master gateways (phase 2 masters could set up links with neighboring masters) and uses intermediate gateways for interconnecting piconets. BlueStars and the LSBS protocol, instead, use gateway slaves whenever possible and intermediate gateways only when gateway slaves are not available. The percentage of nodes with master–slave roles is thus much more limited (from over 50% of the nodes in BlueTrees and 60–78% of the nodes in BlueNet to only 8–22% and 14–21% of the nodes in BlueStars and LSBS, respectively). BlueStars, the LSBS protocol and BlueNet have the further advantage of forming scatternets with higher degree of piconet interconnection (a mesh-like topology instead of a tree).

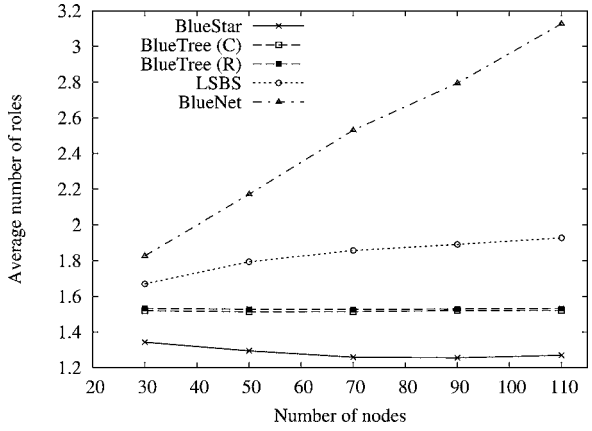
While comparing BlueStars and LSBS's performances, we noticed that the percentage of nodes with only one role is much higher in BlueStars (67–71%, at $n = 110 - 30$) than for the LSBS protocol (44–53%). This is due to the higher probability for slaves in a LSBS piconet to be selected as gateways to a bigger number of adjacent piconets, and, partially, to the less efficient way of selecting gateways in a sparser topology.

In BlueNet, almost all the nodes are gateways to a possibly high number of piconets. The percentage of nodes with only one role is very limited, ranging from 21.6% at $n = 30$ downto 5.6% at $n = 110$. The percentage of nodes with an extremely high number of roles quickly increases with $n$ to take into account the increased number of extra piconets to which nodes affiliate.

At $n = 90$, the 27.8% (the 34.2% when $n = 110$) of the nodes have four roles or more.




(a)




(b)

Figure 12. Average number of roles per node: (a) 10 s long device discovery, and (b) 20 s long device discovery.

In case of scatternet formed from BT topologies obtained from a 20 s device discovery, the results are similar to those of scatternet from BT topologies at 10 s. The only noticeable variation occurs for BlueStars, that given the higher density of the BT topologies and the corresponding lower number of (bigger) piconets, requires a lower number of gateways which are likely to be gateway slaves.

*Average shortest path length*

Independently of the particular routing protocol to be finally used in the formed scatternet, the length of the shortest route is a metric that gives already a measure of how well a routing protocol can perform. Figure 6(b) shows the increase of the average length of shortest paths in the formed scatternets with respect to the same metric in the BT topologies and in the visibility graphs. We first observe that there is little difference between the shortest path lengths in the visibility graph and the lengths of the same paths in the discovered topologies. The highest increase is for networks with 110 nodes, and it is around the 8% after 10 s of device discovery. Among the four protocols, BlueStars and BlueNet show similar performance and are the ones with the shortest routes between any two nodes. The average increase of route length in the mesh-like scatternets formed by BlueStars and BlueNet with respect to the length of the corresponding routes in the BT topologies is 65%, independently of the increasing number of nodes. This is essentially due to the piconet-based network organization, which may force nodes that are neighbors in the visibility graph to communicate through their common master (if they belong to the same piconet), or through a possibly long inter-piconets route in case they belong to different piconets.

Routes in scatternets formed by the LSBS protocol are from 19% ($n = 30$) to 63% ($n = 110$) longer than routes in BlueStars scatternets, to reflect the higher number of piconets that have to be crossed. A higher number of piconets is formed by the LSBS protocol because of the limit imposed on the number of slaves per piconet. This is the case of BlueNet as well, since BlueNet piconets have a bounded number of slaves too. In the case of BlueNet, however, the length of the scatternet routes is shorter than LSBS' because of the higher number of roles per node. We observe that BlueNet gateways interconnect more piconets than LSBS', thus providing shorter interpiconet routes.

Finally, as expected, in the tree-like scatternets formed by BlueTrees routes are longer than those in BlueStars and BlueNet scatternets. In the case of the blueroot placed in a central position the increase ranges from 33% ($n = 30$) to 87.7% ($n = 110$). When the blueroot is randomly positioned, we obtained longer routes: The increase over BlueStars routes ranges from 42% ($n = 30$) to 102% ($n = 110$). In both cases two nodes that are possibly close to each other have to communicate through the only one route that passes through the first common ancestor.

We observed a similar trend for scatternet obtained from BT topologies generated after a 20 s long device discovery phase.

*Packet overhead*

As a measure of the control message overhead of the scatternet formation protocols, we have counted the average number of LMP packets exchanged by the nodes after device discovery phases of 10 and 20 seconds (figures 13(a) and (b)).

As expected, BlueStars experiences little LMP traffic, due to its simple operations and the fact that a node needs to exchange very little information with its one-hop neighbors. BlueNet requires a higher number of messages, since interpiconet link set up requires the exchange of a piconet composition among neighboring nodes.

We notice a remarkable increase in the packet overhead of BlueTree and LSBS with respect to the overhead of BlueStars and BlueNet. The increased LMP traffic is due to the neighbor list exchange protocol that all nodes that have discovered each other must execute. Additional LMP packets are needed for the replenish phase, on which BlueTree and LSBS rely to operate correctly. As mentioned, the information exchanged in these two phases of the protocols can be non-negligible, due to the possibly high number of neighbors that a node has to contact, and to the need of exchanging nodes' ID, weight, clock and also location information (only in LSBS).

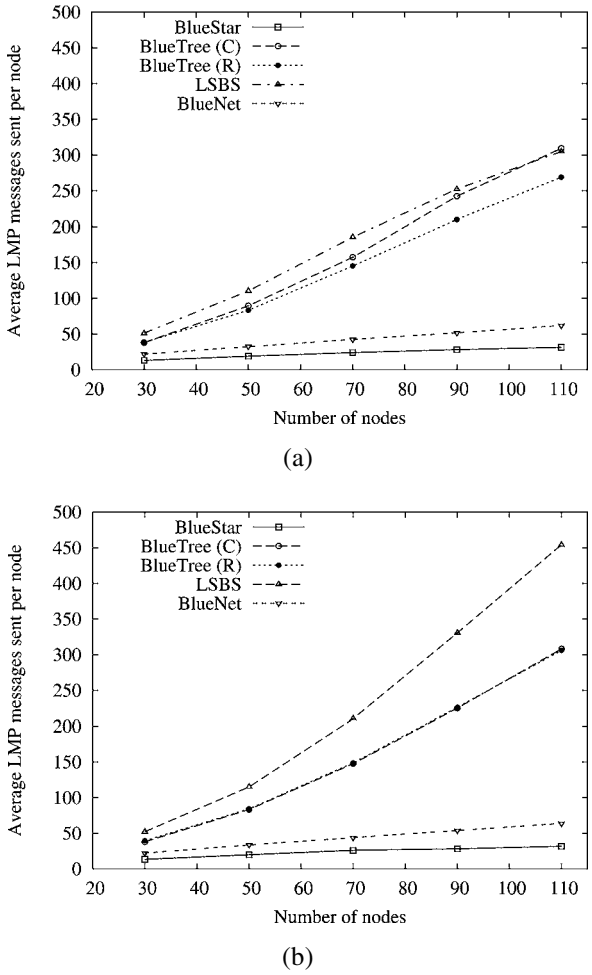



Figure 13. Average number of LMP packets per node: (a) 10 s long device discovery, and (b) 20 s long device discovery.

In the case of LSBS, we notice that even if the knowledge of the two-hop neighbors location may reduce the number of nodes that need to contact each other in the replenish phase, the additional amount of data needed by this protocol (mainly, the bytes that carry the location information) decreases the advantage given by the knowledge of the location. Also, LSBS relies on an extra phase in which each node informs its neighbors of the links selected during the Yao construction, resulting in an overall increase up to 43% (at $n = 110$ after 20 s device discovery) in the number of exchanged LMP packets over BlueTree.

## 5. Conclusions

This paper described solutions to the problem of scatternet formation, namely, to the problem of setting up multi-hop networks of Bluetooth devices. The performance of four scatternet formation protocols – BlueTrees, BlueStars, BlueNet and the LSBS protocol – have been compared by means of thorough simulations. We observed that device discovery is the most time-consuming operation, independently of the particular protocol to which it is applied. We also identified protocol parameters and Bluetooth technology features that affect the duration of device discovery, showing how modification to the BT specifications could lead to considerable improvements and the possibility to discover all the neighbors of a node within a few seconds. Finally, we have analyzed the effect of the different protocols operations on key metrics of the generated scatternets. The comparative performance evaluation showed that due to the simplicity of its operations and to its basic working requirements BlueStars is by far the fastest protocol for scatternet formation which also yields to scatternets with a lower number of piconets, average route length and number of roles per node. However, BlueStars produces scatternets with an unbounded, possibly large number of slaves per piconet, which imposes the use of inefficient Bluetooth operations (parking and unparking of slaves). In order to solve this problem we have shown how a good compromise is obtained by combining BlueStars and the LSBS protocol.

Overall, our performance study reveals that, although the BT specifications enable the formation of multi-hop networks, forming scatternets is still a formidable task. This is particularly due to the inefficiencies of device discovery, and to the extra complexity imposed by the BT technology on the implementation of distributed algorithms.

## References

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

[2] C. Petrioli, S. Basagni and I. Chlamtac, Configuring BlueStars: Multihop scatternet formation for Bluetooth networks, IEEE Transactions on Computers 52(6), Special Issue on Wireless Internet (2003) 779–790.

[3] 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 (7–10 January 2002).

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

[5] Specification of the Bluetooth System, Vol. 1, Core, Version 1.1, `http://www.bluetooth.com` (22 February 2001).

[6] T. Salonidis, P. Bhagwat, L. Tassiulas and R. LaMaire, Distributed topology construction of Bluetooth personal area networks, in: *Proceedings of the IEEE Infocom 2001*, Anchorage, AK (22–26 April 2001) pp. 1577–1586.

[7] C. Law, A.K. Mehta and K.-Y. Siu, A new Bluetooth scatternet formation protocol, ACM/Kluwer Journal on Mobile Networks and Applications (MONET) 8(5), Special Issue on Mobile Ad Hoc Networks (2003).

[8] G. Tan, A. Miu, J. Guttag and H. Balakrishnan, An efficient scatternet formation algorithm for dynamic environment, in: *Proceedings of the IASTED Communications and Computer Networks (CCN)*, Cambridge, MA (4–6 November 2002).

[9] M. Ajmone Marsan, C.F. Chiasserini, A. Nucci, G. Carello and L. De Giovanni, Optimizing the topology of Bluetooth wireless personal area networks, in: *Proceedings of IEEE Infocom 2002*, New York (23–27 June 2002).

[10] S. Baatz, C. Bieschke, M. Frank, P. Martini, C. Scholz and C. Kühl, Building efficient Bluetooth scatternet topologies from 1-factors, in: *Proceedings of WOC 2002* (2002).

[11] C. Petrioli and S. Basagni, Degree-constrained multihop scatternet formation for Bluetooth networks, in: *Proceedings of the IEEE Globecom 2002*, Taipei, Taiwan, ROC, Vol. 1 (17–21 November 2002) pp. 222–226.

[12] I. Chlamtac and A. Faragó, A new approach to the design and analysis of peer-to-peer mobile networks, Wireless Networks 5(3) (1999) 149–156.

[13] C.F. Chiasserini, M. Ajmone Marsan, E. Baralis and P. Garza, Towards feasible topology formation algorithms for Bluetooth-based WPANs, in: *Proceedings of IEEE HICSS-36 2003*, Big Island, Hawaii (6–9 January 2003); personal communication.

[14] A.C.-C. Yao, On constructing minimum spanning trees in $k$-dimensional spaces and related problems, SIAM Journal on Computing 11(4) (1982) 721–736.

[15] The VINT Project, The ns Manual, `http://www.isi.edu/nsnam/ns/` (2002).

[16] S. Basagni, R. Bruno and C. Petrioli, Performance evaluation of a new scatternet formation protocol for multi-hop Bluetooth networks, in: *Proceedings of the 5th International Symposium on Personal Wireless Multimedia Communications, WPMC 2002*, Honolulu, Hawaii, Vol. 1 (27–30 October 2002) pp. 208–212.

[17] B.N. Clark, C.J. Colburn and D.S. Johnson, Unit disk graphs, Discrete Mathematics 86 (1990) 165–167.

[18] IBM, BlueHoc: Bluetooth Ad Hoc Network Simulator, Version 1.0, `http://www-124.ibm.com/~developerworks/projects/bluehoc` (June 2001).

**Stefano Basagni** holds a Ph.D. in electrical engineering from the University of Texas at Dallas (December 2001) and a Ph.D. in computer science from the University of Milano, Italy (May 1998). He received his B.Sc. degree in computer science from the University of Pisa, Italy, in 1991. Since Winter 2002 he is on faculty at the Department of Electrical and Computer Engineering at Northeastern University, in Boston, MA. From August 2000 to January 2002 he was an assistant professor of computer science at the Department of Computer Science of the Erik Jonsson School of Engineering and Computer Science, The University of Texas at Dallas. Dr. Basagni's current research interests concern research and implementation aspects of mobile networks and wireless communications systems, Bluetooth and sensor networking, definition and performance evaluation of network protocols and theoretical and practical aspects of distributed algorithms. Dr. Basagni has published over two dozens of referred technical papers. He is also co-author of book chapters. Dr. Basagni served as a Guest Editor of the special issue of the Journal on Special Topics in Mobile Networking and Applications (MONET) on Multipoint Communication in Wireless Mobile Networks as well as Guest Editor of the special issue on mobile ad hoc networks of the Wiley's Interscience's Wireless Communications & Mobile Networks journal, for which he serves in the editorial board. Dr. Basagni also served and serves on Technical Program Committees, including the ACM/SIGMOBILE MobiCom TPC, and ACM/SIGMOBILE MobiHoc TPC, IEEE Globecom and IEEE ICC. Dr. Basagni is a member of the ACM (including the ACM SIGMOBILE) and of the IEEE (Computer and Communication Societies).
E-mail: basagni@ece.neu.edu

**Raffaele Bruno** received the B.Sc. degree in telecommunications engineering in 1999 and the Ph.D. degree in information engineering in 2002 from the University of Pisa, Italy. Since January 2003 he joined the IIT Institute of the Italian National Research Council (CNR), where he is a Junior Researcher. His current research interests are in the area of wireless and mobile networks with an emphasis on efficient wireless MAC protocols, scheduling algorithms for Internet traffic integration and QoS.
E-mail: bruno@iit.cnr.it

**Gabriele Mambrini** received the laurea degree in computer science from the University of Rome,"La Sapienza", Italy, in March 2003. He is currently with the Italian Interuniversities Consortium for Supercomputing Applications (CASPUR), in Rome. His current research interests include wireless networking, automatic text categorization and indexing, computer security.
E-mail: mambrini@caspur.it

**Chiara Petrioli** received the Laurea degree with honors in computer science in 1993, and the Ph.D. degree in computer engineering in 1998, both from Rome University "La Sapienza", Italy. She is currently Assistant Professor at the Computer Science Department at Rome University "La Sapienza". Her current work focuses on ad-hoc and sensor networks, Bluetooth, energy-conserving protocols and QoS in IP networks. Prior to Rome University she was research associate at Politecnico di Milano, and was working with the Italian Space Agency (ASI) and Alenia Spazio. Dr. Petrioli is an area editor of the ACM/Kluwer Wireless Networks journal, the Wiley InterScience Wireless Communications & Mobile Computing journal and the Elsevier Ad Hoc Networks journal. She has served in the organizing committee and technical program committee of several leading conferences in the area of networking and mobile computing including ACM MobiCom, ACM MobiHoc, IEEE ICC. Dr. Petrioli was a Fulbright scholar, and is member of ACM, IEEE, ACM SIGMOBILE, IEEE Communications Society.
E-mail: petrioli@di.uniroma1.it