Optimized Scatternet Topologies for Personal Area Networking in Dynamic Environments

Francesca Cuomo, Guido Di Bacco, Tommaso Melodia,
University of Rome "La Sapienza", Via Eudossiana 18, 00184 Rome, ITALY

Abstract— **In a scenario where different radio technologies cooperate to provide access to the Internet and advanced wireless services to mobile and nomadic users, Bluetooth is considered an enabling technology for the Personal Area Networking segment. To this aim, Bluetooth devices should be able to set-up a wireless multi-hop network with given topological characteristics and with limited formation delay. In this work SHAPER, a distributed algorithm for tree scatternet formation, is enhanced to work in a dynamic environment where devices enter and leave the Personal Area Network and require a fast interconnection with an optimized topology. We define a procedure (called SHAPER-OPT) that produces a meshed topology applying a Distributed Scatternet Optimization Algorithm (DSOA) on the network built by SHAPER. Nodes are shown to be able to easily join or leave the scatternet at any time, without compromising the long term connectivity. Benefits brought by DSOA are shown by performance analysis, while the delay for network set-up and reconfiguration in dynamic environments is shown to be within acceptable bounds.**

I. Introduction

Untethered networks of small hand-held electronic devices are very likely to be part of our daily lives in the near future. These networks are usually referred to as *Personal Area Networks* (PANs). Different PANs can be interconnected to enable sharing of information or seamlessly integrated with networks of sensor/actuator devices, to allow interaction with the physical environment. The IEEE 802.15.1 working group has recently released a standard for PANs based on the *Bluetooth* Industrial Specifications. Thanks to the *scatternet* concept [1], which allows more than 8 devices to be interconnected in a multi-hop fashion, Bluetooth is considered an enabling technology for these scenarios. A scatternet is composed of different *piconets*; each piconet has an overall 1 Mbit/s gross data rate, to be shared by at most 8 devices (also *nodes* in the following).

The Scatternet Formation issue has been extensively discussed in the last few years. Algorithms for scetternet set-up should have the following properties:

1) work in a multi-hop scenario (not all nodes are in radio visibility of each other);
2) guarantee self-healing behavior of the network. In particular, they must handle: i) entrance of new nodes in the network; ii) nodes' mobility or failure/deactivation;
3) limited formation time;
4) achieve optimized topology.

To the best of our knowledge, no existing algorithm guarantees all of the above. Existing solutions can be classified as single-hop ([3][4][5][6]) and multi-hop ([7][8][9][10]). Paper [3] addresses Bluetooth scatternet formation with a distributed selection of a leader device which assigns roles to the others. In [4] and [6] a distributed scatternet formation protocol is defined. In both cases the topologies are meshed. The works in [5] and [8] form tree-shaped scatternets. SHAPER [7] also forms tree-shaped scatternets, but is fully distributed, works in a multi-hop scenario, has very limited formation time and assures self-healing properties of the network.

A second class of multi-hop proposals consists of algorithms that produce connected scatternets, starting from previously formed piconets. In [9] and [10] the *BlueStars* and *BlueMesh* protocols are described, respectively. These protocols define rules for device discovery, piconet formation and piconet interconnection so as to achieve suitable properties for the formed scatternet.

Some other works discuss scatternet topology optimization. This issue is faced in [11] and [2] by adopting centralized approaches. In [11] the aim is minimizing the load of the most congested node in the network, while [2] discusses the impact of different metrics on the scatternet topology. A distributed approach based on simple heuristics is presented in [12].

To finish with, the work in [13] proposes a new approach to scatternet formation. Route discovery and construction is performed on-demand on the basis of real traffic conditions and traffic requests.

In this paper, we discuss the Scatternet Formation issue by considering also scatternet optimization and topology reconfiguration after mobility or deactivation of nodes. We rely on our previous work SHAPER [7] and enrich the self-healing behavior of our multi-hop algorithm with a topology optimization named *Distributed Scatternet Optimization Algorithm* (DSOA) [2]. The overall procedure (called SHAPER-OPT) derives from the application of DSOA on the scatternet built by SHAPER and produces a meshed topology. To the best of our knowledge, SHAPER-OPT is the first scatternet formation algorithm that contemporarily works in a multi-hop environment, presents self-healing properties (e.g., dynamically reconfigures the network topology after nodes entrance/exit) and optimizes the network topology. Moreover, it is the first time that a dynamic behavior is extensively studied in the framework of scatternet formation algorithms.

The paper is organized as follows. In Section II our approach is described by means of the main procedures which compose it. In Section III simulative performance results concerning the interworking of SHAPER and DSOA are discussed, while Section IV gives the main conclusions.

This work was partially supported by the Italian FIRB Project VICOM.

II. SHAPER-OPT

In this Section SHAPER-OPT, i.e., the set of procedures for scatternet formation, topology optimization and network self-healingness is introduced and discussed.

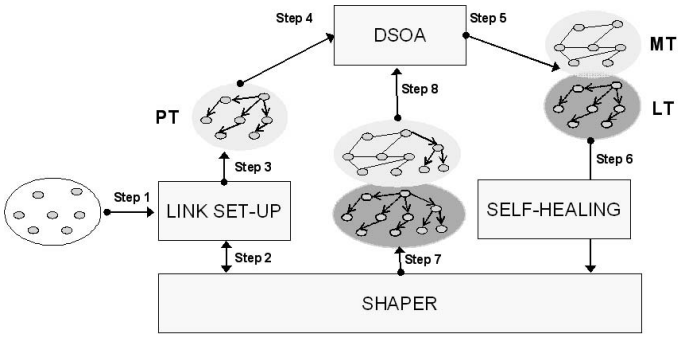

Fig. 1. Evolution of a scatternet during its lifetime

A. Description of SHAPER-OPT

Let us refer to Figure 1. Starting from a fully disconnected scenario, all nodes apply SHAPER [7] to form a tree-shaped connected scatternet. In a distributed fashion, adjacent nodes start forming links. A set of merging procedures are employed by SHAPER to interconnect randomly formed links (and piconets) in a loop free topology (Step 2). These procedures define rules for forming trees compliant with the BT Specifications in a multi-hop scenario. In general, during SHAPER, different trees are merged and converge to a unique tree (Step 3). We will refer to this physical tree-shaped scatternet as *Primitive Topology* (PT). This structure is then used to propagate messages that are used in the subsequent optimization phase. The root node broadcasts an INIT message to all nodes of the tree. This indicates that an optimizition procedure, named DSOA, is starting (Step 4). We will briefly explain the DSOA in the next subsection. At the end of DSOA the network presents an optimized *Meshed Topology* (MT) (Step 5). However, as will become clearer in the following, it can be advantageous to maintain the initial tree logical topology throughout the life of the network. This means that nodes maintain information about their parent and children as established at the end of Step 3, even if the relevant physical links have been torn down by DSOA in the MT. In the following, the set of these logical relationships will be referred to as *Logical Tree* (LT).

Thus, after the first execution of DSOA the network presents:

- a meshed physical topology;
- a tree logical topology.

Purposes of this tree-shaped logical structure are:

1) allowing to periodically re-apply DSOA to optimize the scatternet topology after dynamic events (mobility, failure of nodes, etc.);
2) simplifying the procedures that guarantee the connectivity of the network and the entrance of new nodes;
3) handling failure/mobility of nodes (SELF-HEALING procedures, Step 6 of Figure 1).

As for item 1), DSOA relies on a tree structure. The optimization starts from the physical (logical) root node and proceeds through all nodes of the network in a sequential way.

The combination of SHAPER and DSOA, together with the SELF-HEALING procedures, is called SHAPER-OPT. Thanks to the LT, SHAPER-OPT is able to merge networks that have autonomously formed. These networks could either be in a PT or in a MT configuration. As an example, in Figure 1 is represented the event of an autonomously formed tree (Step 7) that is included in the MT, by simply connecting a node of the tree to a node of the MT. This means that, at the LT level, the logical tree is extended with the inclusion of this new tree. At the MT level, the physical scatternet presents a hybrid topology (MT combined with PT). Periodically (Step 8), the DSOA is applied to re-optimize the overall physical topology.

B. The DSOA procedure

The aim of this procedure is to reconfigure the tree in an optimized physical topology that allows more efficient communication. DSOA is a distributed heuristic which allows a decentralized process of optimization of the scatternet topology: each node selects the best local topology on the basis of a matrix description of the network ($\mathbf{B}$) and on a given metric (see [2] for details). All nodes in the tree are sequentially visited starting from the root. At each step i, a node v_i receives information on the topology selected up to that step by the previous $(i-1)$-th nodes (described by the $\mathbf{B^{i-1}}$ matrix). Then, the node selects the role (master or slave) it will assume and the links to be established, with the aim of maximizing a global scatternet performance metric, calculated on the matrix. After node v_i has selected the "best" links to be established in accordance to the metric, it modifies the matrix giving rise to a new network topology description represented by $\mathbf{B^i}$, which is forwarded to the following node in the tree. At the end of DSOA, the root node receives the $\mathbf{B^N}$ matrix, where N is the number of nodes in the tree, containing the final topology selected by all nodes. The matrix is then broadcasted to all nodes. After receiving the matrix, the nodes distributedly set-up the new optimized topology. The periodical re-optimization of the network is regulated by a timer, named T_{DSOA}.

C. State Variables used to build and maintain the tree

To better explain the SHAPER procedures we introduce a number of state variables that define the behaviour of each node when SHAPER is executed.

Let us refer to a generic node i. It maintains and stores the following state variables:

- $comp_i$: is the component i is affiliated with. A component can be: i) a *Primitive topology* (PT, tree-shaped, not yet optimized); ii) a *Meshed topology* (MT, shaped by DSOA).

T_i represents the tree node i is affiliated with. When a node i is affiliated with a tree T_i in a PT, it stores and up-dates the following variables:

- $stat_i$: is the *status* of the node i, that can be *free* (F), *root* (R) or *non-root* (NR);

- $tree\_ID_i$: the Bluetooth Device ADDRess (BD_ADDR) of the root of the tree T_i (or of itself if free). This information univocally identifies all nodes in the same tree.

We always assume that a free node has $comp$=PT. When a node i is affiliated with a MT, it belongs also to a *logical tree* LT_i; it stores and up-dates the following parameters:

- $Lstat_i$: is the *status* of node i, F, R or NR in the logical tree;
- $tree\_ID_i$: the BD_ADDR of the root of LT_i;
- $Lpar_i$: is the parent node of i in LT_i;
- $Lch_i = \{Lch_i^1, Lch_i^2, .., Lch_i^7\}$: is the set of children nodes in LT_i;
- $role_i$: is the role i assumes in the MT, either master (M), slave (S) or gateway (G).

When DSOA is performed for the first time, every node assumes its logical state variables equal to the respective PT variables.

D. Merging Procedures

The merging procedures allow two nodes that meet to merge the components they belong to. We always refer to the two meeting nodes, through which the two components merge, as A and B. For the sake of simplicity, we assume that A is master and B is slave in the newly formed link. A sends a MERGE message to the slave B. This message contains $comp_A$ and $tree\_ID_A$. B verifies if the $tree\_ID_A$ equals its $tree\_ID$. In this case, the two nodes already belong to the same scatternet and the two components need not to be merged. If the $tree_ID$s are different, B continues the merging procedure depending on the components the two nodes are affiliated with. The following situations can occur:

1) SHAPER is applied in two cases: if a F node A meets a PT-node B or a F node B meets a PT-node A and the F node is included in the tree, or if a PT-node A meets a PT-node B;
2) The **`MainMerging`** procedure is applied when a PT-node meets a node which belongs to a MT or when an MT-node A meets an MT-node B.

In case 1) SHAPER produces a physical tree topology matching the logical tree one. In case 2), by applying the **`MainMerging`** procedure, we obtain a physical connected topology (ibrid PT-MT) and a logical tree. In this latter case DSOA is then applied to reoptimize the scatternet.

When the slave node B of the new link applies the **`MainMerging`** procedure, two scatternets with different $tree\_ID$ meet. Node B, if $role_B = M$, verifies from the MERGE message if $role_A \neq M$. In this case, it is more efficient to switch (role exchanging) the newly formed link between A and B. Then the MERGE message is sent by the new master and the **`MainMerging`** is re-entered. In all the other cases, the scatternet of node B has to update its $tree\_ID$ and merge its logical tree with the LT_A. To this aim, B broadcasts a message to the nodes of its LT_B (those with the same $tree\_ID$ as B). This message contains the $tree\_ID_A$. When a node receives this message, it updates its $tree\_ID$. At the end of this broadcast all nodes of the scatternet have the $tree\_ID_A$. Contemporarily to this $tree\_ID$ up-dating, B merges the LT_B and the LT_A. To this aim, B reconfigures its logical tree and changes its status to R. The status of the root of B changes to NR and the parent-child relation of the logical links between the old root and B is inverted. At the end of the reconfiguration of LT_B, node B is a new root and its tree can be merged with A's tree. This gives rise to a unique tree whose root is the root of LT_A.

`MainMerging`

1: **if** ($role_A \neq M$) and ($role_B$ = M) **then**
2: B: MasterSlaveSwitch
3: **else**
4: B: broadcast B's tree_ID
5: X = B ; Y = $Lpar_B$
6: **while** Y is different from B's root **do**
7: X: switch logical link with Y
8: **if** $role_Y$ = R **then**
9: $role_Y$ = NR
10: **else**
11: X = Y ; Y = $Lpar_Y$
12: **end if**
13: **end while**
14: **end if**

In the case of a master having already 7 links and trying to connect to a slave, the `Reconfig_links` procedure is called. This procedure relies on a property demonstrated in [8]. If a node has more than 5 neighbors, then at least 2 of them are neighbors themselves. Thus, the master parks the entering slave and forces the set-up of a link between 2 of its neighbors. The parked slave is then woken up and it becomes an active node in the piconet.

E. Self-Healing Procedures

The self-healing procedures of SHAPER-OPT rely on the capability of the nodes to check if the network preserves its connectivity. When a node, for a given time, can not send a packet or does not receive reply from a neighbor, it assumes that the neighbor has moved or switched off. It broadcasts a *Local Connectivity Check* (LCC) message which contains the BD_ADDR of the node that switched off. Then, it executes the **`ConnectivityCheck&Reconfigure`** procedure. This procedure is also performed by every node that receives an LCC message.

`ConnectivityCheck&Reconfigure`

1: i: contact $Lpar_i$ {parent node of i}
2: **if** (i does not receive reply from $Lpar_i$) **then**
3: i: delete $Lpar_i$; $stat_i$ = R
4: **for each** sub-tree of node i **do**
5: update the $tree\_ID$ with the BD_ADDR of i
6: **end for**
7: **end if**
8: **for each** Lch_i^x of i **do**
9: **if** (i does not receive reply from Lch_i^x) **then**
10: i: delete Lch_i^x
11: **end if**
12: **end for**

13: **for each** physical link of i with the neighbor node j **do**
14: **if** ($tree\_ID$ of $i \neq tree\_ID$ of j) **then**
15: i: build a logical connection with j as child
16: **for each** sub-tree of node i **do**
17: update the $tree\_ID$ with the $tree\_ID$ of j
18: **end for**
19: **end if**
20: **end for**

Node i contacts its Lch_i^x and its $Lpar_i$. If one of its children is unreachable, i deletes the logical link and expunges the child from the set Lch_i^x. If instead the node i can not reach its parent, it becomes R, updates its $Lpar_i$, changes its $tree\_ID$ with its BD_ADDR and sends a message to all its descendants containing the new $tree\_ID$. Then, i attempts to set-up logical links with its neighbors. For each physical link that i has with a neighbor j, i verifies if its $tree\_ID$ is different from j's $tree\_ID$. In this case i reconfigure its LT_i using the $tree\_ID_j$: for each logical descendant of i it updates the $tree\_ID_i$ with the $tree\_ID_j$, the status of the root of i changes to NR and, if $Lstatus_i \neq R$, the parent-child relation of the logical links between the old root and i is inverted. Finally i builds a logical connection as a children with j.

As an example, in Figure 2, when node 6 switches off (Box

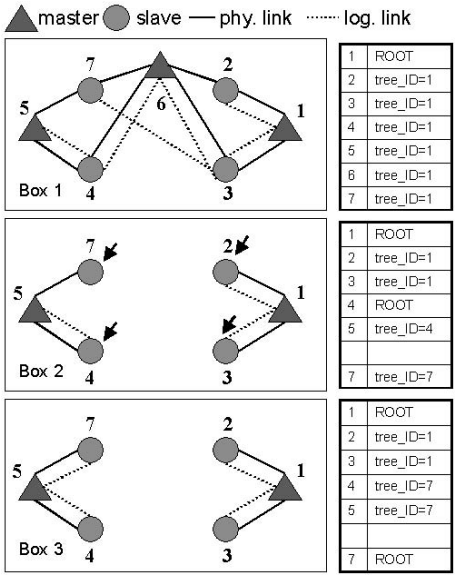

Fig. 2. Logical tree reconfiguration after a node derparture

1), nodes 2, 3, 4, and 7 broadcast an LCC message, thus nodes 4, 5 and 7 change their $tree\_ID$ (Box 2) and node 5 sets up logical links with its neighbors 4 and 7 (Box 3). At this point all connected nodes have the same $tree\_ID$. After the **`ConnectivityCheck&Reconfigure`** procedure the two scatternets can merge again in a unique one thanks to the **`MainMerging`** procedure.

III. Performance Evaluation

In this section we present simulation results referring to the optimization and the self-healing procedures of SHAPER-OPT. We extended the MIT's ns-2 Blueware platform [14] with SHAPER and DSOA. The inter-piconet scheduling mechanism is the one proposed by [14]. As for the parameters adopted in the SHAPER procedures, e.g. T_{comm}, we set the same values of [7], while T_{merg} is randomly extracted between $1.024s$ and $1.28s$. All figures report 95% confidence intervals.

A. Timing of the optimization procedures

Figure 3 plots the time needed to perform all the SHAPER-OPT components, as a function of the number of nodes, starting from a fully disconnected network. The nodes are randomly distributed in an area of $7\times7\ m^2$. The curve relevant to SHAPER shows the limited delay required to set-up a fully connected tree-shaped scatternet. The other three curves report the delays for the actuation of the INIT broadcasting, DSOA and $\mathbf{B^N}$ matrix broadcasting. DSOA consumes a time that is directly proportional to the number of nodes and represents, in the overall SHAPER-OPT mechanism, the most time consuming procedure. It is however to be noticed that, once SHAPER ends (after a few seconds), the network is fully connected and, as a consequence, DSOA optimization can proceed in parallel with the data communications. It is to be considered that DSOA delays can be reduced by assigning different packets' priorities in the scheduling mechanism.

B. Topology characteristics

A second set of simulations refers to the topology characteristics of PT and MT, respectively. The optimization metric used in DSOA was *average path capacity* which, as shown in [2], aims at jointly maximizing the overall scatternet capacity and minimizing the number of hops between a generic source-destination pair. In Figure 4 the mean number of piconets per scatternet is reported, as a function of the number of nodes. The MT reduces the number of piconets per scatternet with respect to the PT. This behavior could be advantageous for reducing the inter-piconet interference when the number of nodes is high.

Other topology characteristics are reported in Figure 5, where the number of links and the number of roles per node are compared for PT and MT. The mean number of links, which measures how meshed the scatternet is, changes from a value of 1.6 for the PT to values ranging from 2.4 to 4.6 for MT. A low average number of roles per node indicates that gateway nodes divide their time among a few piconets (thus reducing the interpiconet scheduling delay). However a more meshed topology (which is a desirable property) is usually associated to higher values of the average number of roles per node.

C. Self-Healing behavior

To evaluate the self-healing procedures of SHAPER-OPT we measured the time requested to reconfigure the scatternet topology after an "en mass" arrival in case of both PT and MT. All nodes are scattered in an area of $7\times7\ m^2$. We suppose that a node performs a communication task of a period of T_{comm}, uniformly distributed between 2.25 s and 6.3 s. With reference to Figure 6, we assume that a certain percentage of the nodes arrives at time t_0 while the rest

arrives after the initial scatternet (PT or MT) has been formed. Two different percentages for the second group of nodes have been considered: 10% and 20%. The figure reports the reconfiguration time due to the second burst of arrivals. The time needed to reconfigure both the topologies is shown to be comparable to the initial scatternet set-up time.

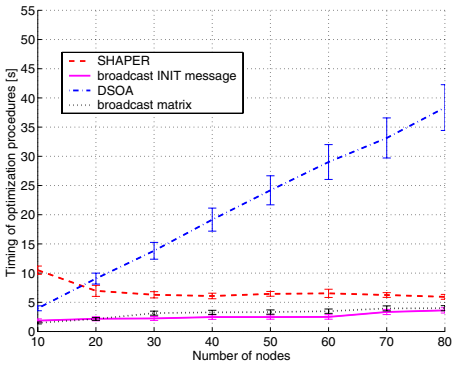

Fig. 3. Timing of SHAPER-OPT procedures

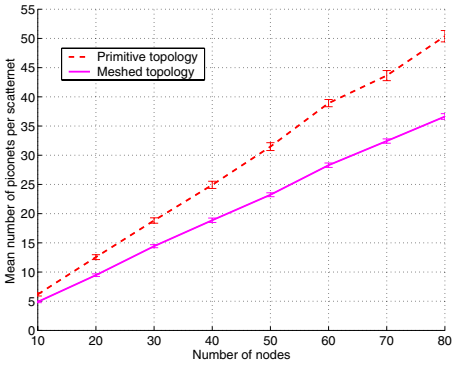

Fig. 4. Mean number of piconets for Primitive and Meshed Topologies

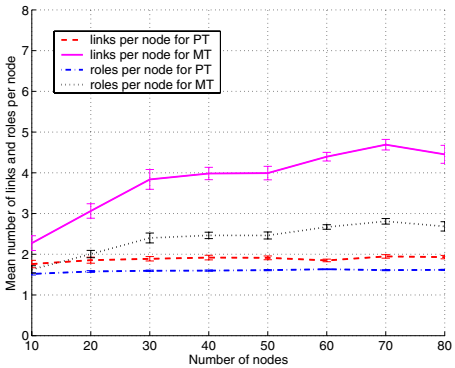

Fig. 5. Number of links and number of roles per node for Primitive and Meshed Topologies

IV. Conclusions

The article proposes a Scatternet Formation paradigm that combines a quick tree scatternet set-up with a distributed

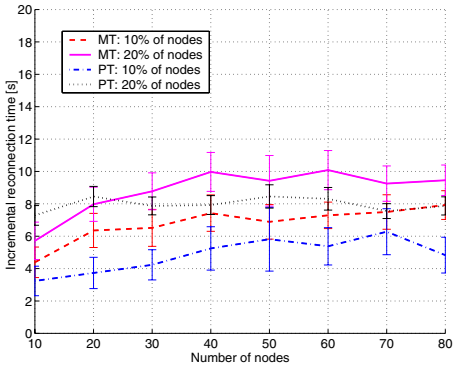

Fig. 6. Reconfiguration time in Primitive and Meshed Topologies

scatternet optimization algorithm, which gives rise to meshed topologies. A logical tree-structure is maintained to perform dynamical reconfiguration of the network. The procedures are described and performance results are reported, that show how the network can be dynamically reconfigured with limited delay. The optimization delay is compared to the time required to form a tree-shaped connected scatternet. To finish with, a comparison of the topological properties of tree-shaped and meshed topologies is provided.

References

[1] J. Haartsen, "The Bluetooth Radio System", *IEEE Personal Communications*, Vol. 7, n. 1, pp. 28-36, February 2000.

[2] T. Melodia, F. Cuomo, "Ad Hoc Networking with Bluetooth: Key Metrics and Distributed Protocols for Scatternet Formation", Ad Hoc Networks (Elsevier), Volume 2, Issue 2, Pages 109-202, April 2004.

[3] T. Salonidis, P. Bhagwat, L. Tassiulas, R. La Maire, "Distributed topology construction of Bluetooth personal area networks", Proc. of the *IEEE Infocom 2001*, pp. 1577-1586, April 2001.

[4] C. Law, A. Mehta, K-Y Siu, "Performance of a new Bluetooth scatternet formation protocol", ACM Symposium on Mobile Ad Hoc Networking and Computing 2001, Long Beach, California, USA, October 2001.

[5] G. Tan, A. Miu, J. Guttag, H. Balakrishnan, "An Efficient Scatternet Formation Algorithm for Dynamic Environments", in *IASTED Communications and Computer Networks (CCN)*, Cambridge, November 2002.

[6] H. Zhang, J. C. Hou, L. Sha, "A Bluetooth Loop Scatternet Formation Algorithm" Proc. of IEEE the *ICC 2003*, Anchorage, pp.1174-1180, 2003.

[7] F. Cuomo, G. Di Bacco, T. Melodia, "SHAPER: a Self Healing Algorithm Producing multihop bluetooth scattERnets", Proc. of the *IEEE Globecom 2003*, San Francisco, pp. 236-240 December 2003.

[8] G. Zaruba, S. Basagni, I. Chlamtac, "Bluetrees- Scatternet formation to enable Bluetooth-based personal area networks", Proc. of the *ICC 2001*, pp. 273-277, 2001.

[9] S. Basagni, C. Petrioli, "Multihop Scatternet Formation for Bluetooth Networks", Proc. of the *VTC 2002*, pp. 424-428, May 2002.

[10] C. Petrioli, S. Basagni "Degree-constrained Multihop Scatternet Formation for Bluetooth Networks", Proc. of the *IEEE Globecom 2002*, Taipei, November 2002.

[11] M. Ajmone Marsan, C. F. Chiasserini, A. Nucci, G. Carello, L. De Giovanni, "Optimizing the Topology of Bluetooth Wireless Personal Area Networks", *IEEE INFOCOM 2002*, New York, June 2002.

[12] C. F. Chiasserini, M. Ajmone Marsan, E. Baralis, P. Garza, "Towards Feasible Distributed Topology Formation Algorithms for Bluetooth-based WPANs", *36th Hawaii International Conference on System Science* (HICSS-36), Big Island, Hawaii, January 6, 2003.

[13] Y. Liu, M. J. Lee, T. N. Saadawi, "A Bluetooth Scatternet-route Structure for Multihop Ad Hoc Networks," IEEE Journal on Selected Areas in Communications, Vol. 21, n. 2, February 2003.

[14] G. Tan, "Blueware: Bluetooth Simulator for ns", MIT Laboratory for Computer Science, Cambridge, October 2002.