An Ad hoc Scatternet Formation Algorithm on Bluetooth

Dong Kyoo Kim *, Yun Jong Choi ** and PooGyeon Park ***

* ** ***School of Electronic and Electrical Engineering Pohang University of Science and Technology(POSTECH)
Pohang, Kyungbuk, Korea (Tel: +82-54-279-5588; Fax:+82-2-279-2903; Email:kdk,scope75,ppg@postech.ac.kr)

Abstract: Bluetooth is a low cost, license-free, and short-range radio communication technology. This is gradually gaining de facto standard for applications of a personal area network over the world. To make it further popular, it should provide faster and more flexible connectivity than that supported by the existing bluetooth specification. This paper proposes a new Ad hoc scatternet formation algorithm that supports those connectivity under circumstances such that devices can not get in direct touch with one another; *say out-of-bounded case*. It also maintains the character that those connectivity minimizes the number of piconets when all devices are in the communication range with one another, *say in-bounded case*, as previous works.

Keywords: bluetooth, ad hoc, scatternet, formation

1. Introduction

Bluetooth is a low cost, low power consumption, license-free, and short-range radio communication technology. Its power consumption is usually less than a hundred mili-Watt, which is suitable for mobile and portable applications, with supporting up to 720 kilo-bits per seconds(kbps) data transmission rates. One device can establish a network with up to 7 devices under its omni-controlled medium, which is usually called a piconet. This small-scaled and simple medium access makes it a de facto standard for a wireless Personal Area Network(PAN). The current bluetooth specification [1] being announced in 2001 does not explicitly instruct us about ad-hoc connectivity. However, this is an obstacle for the bluetooth to be well-accommodated in variable applications. To solve the difficulty, the scatternet formation algorithm should be introduced in the bluetooth. The terminology scatternet indicates a network consisting of more than one piconet or a piconet itself. Some papers had been published to handle this difficulty [5][6]. Noticeable works among them are the BTCP (Bluetooth Topology Controuction Protocol) [2] and the BSFA (Bluetooth Scatternet Formation Algorithm) [3][4]. Their scatternet formation algorithms assumed that participants in the formation process could communicate directly with one another at anytime. [2] used an election-and-coordination based method, in which one device is elected among participants, then, the elected device coordinates and constructs the whole scatternet. Therefore, [2] can be worked well in the case of a meeting or a small conference which are usually well-planned and have a term for preparing the coordination. [3] also used the election-based method suitable for the same cases in [2]. It is faster than [2] in constructing a scatternet because its algorithm in [3] makes a temporary scatternet in the middle of the election. Both algorithms [2][3] have a restriction that devices participating in a scatternet construction must be kept in touch with one another, and we call this in-bounded case. However, this restriction is not suitable for the bluetooth-enabled devices in the practical situations such as hospitals, buildings or exhibition halls which are too spacious for the in-bounded condition to be satisfied; the circumstances that does not satisfy the in-bounded condition will be called out-of-bounded cases. Therefore, it is desirable that the restriction should be removed in order for the bluetooth to be applied for the more diverse applications. In this paper, we suggest a new scatternet formation algorithm, the ASF (Ad-hoc Scatternet Formation) algorithm that constructs scatternets in the out-of-bounded cases as well as the in-bounded cases. The ASF algorithm introduces a dynamic coordination method which enables devices participating in a coordination process to communicate with other devices even if the process is going on. It gives a useful instant communication capability to the devices which has not been supported in the previous coordination algorithms. Moreover, the ASF algorithm significantly enhances a device-discovery performance of the bluetoth through the slight modifications on the existing inquiry/inquiry scan-based mechanism. This modification does not have any physical-level change but has trival changes only to the firmware-level, so that the ASF algorithm can be easily applied to existing bluetooth devices without losing its backward compatibility.

Section 2.explains the modifications in the ASF algorithm. The coordination algorhtim is presented in Section 3.. In Section 4, the ASF algorithm is analyzed and the simulation results are presented. Finally, concluding remarks are provided in Section 5.

The authors would like to thank the Ministry of Education of Korea for its financial support toward the Electrical and Computer Engineering Division at POSTECH through its BK21 program.

2. Modification on the device discovery and the connection mechanism

2.1. Modification on search process for Device-discovery method

We utilize that there are the reserved addresses for DIAC (Dedicated Inquiry Access Code) in the bluetooth spec 1.1 [1], which are used for dedicated inquiry of specific classes of bluetooth devices. The ASF search process will exploit one of these DIACs to gather information about neighbor devices. We call it a SDIAC(Search DIAC); we will use a term, node, as the same means for a term, device. By using the SDIAC, we can make a more intelligent device-discovery method. In the search process, we will introduce a new search/search scan states. These states are similar to the inquiry/inquiry scan states except that the mandatory page scheme is not adopted; the mandatory page scheme is that a node having been responded to any device's GIAC(General Inquiry Access Code) is followed by a page scan state during $T_{mandatory\ pscan}$.

Because of not using mandatory page scheme, the node continues to stay in the search scan state after responding the inquiry message and can give its information to another search devices. In the ASF algorithm, the information about neighbor nodes collecting through the search process is very important for the dynamic coordination procedure; we will give a comprehensive explanation about this in the next section. The more information about neighbor nodes is collected, the more the coordination results might be desirable in the sense of any intended purpose.

In addition, we will propose a scheme that increases the probability of pairing among nodes. We divide the search scan procedure to three part; search scan, page and page scan. As is shown in Fig 1, a node in the search scan procedure changes its state every $T_{roundTO}$ during $T_{pairingTO}$. This increases the probability of a node being connected with other nodes by changing states frequantly and repeatedly. In this process, a state transition is performed in a random manner with the state transition probability p. In ASF algorithm, if a selected random value is smaller than p=1/3, a node becomes search state. Otherwise it becomes search scan state.

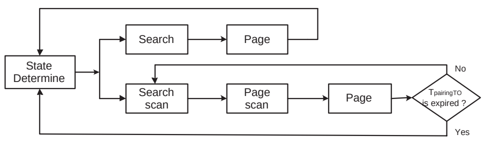

Fig. 1. The modified search / search scan procedure

2.2. Modify the connection mechanism for bluetooth

We begins with defining some notations used in this section. Subscript and capital T represent inquiry nodes, and subscript and capital R represent inquiry scan nodes. A table is defined for the dynamic coordination process and called a COTABLE. It consists of 4 lists. See Table 1 for detailed explanation. Each list member in the COTABLE consists of its bluetooth address and clock information, which are contents of the inquiry response packet. As stated earlier, we assume that each node has gathered its neighbor's information through the search operation. Node R receiving node T's inquiry request responds with its FHS packet and enters into the page scan state. Received the node R's response packet, node T initiates the page process. After paging process is finished, two nodes get into the half-connection state, which is stated in the latter Subsection 2.2.1 In the state, node T receives node R's COTABLE through its POLL packet's response, and node T performs the dynamic coordination process. Then it sends back the result COTABLE to node R. Through the process the groups evolve. The series of these descriptions are illustrated in Fig 2 as a message flow diagram.

Table 1. Table for Coordination : *COTABLE*

Leader(L)	Members(M)	Neighbors(N)	Bridges(B)

L : Leader itself M : Its member nodes
N : Its neighbor nodes B : Bridge nodes

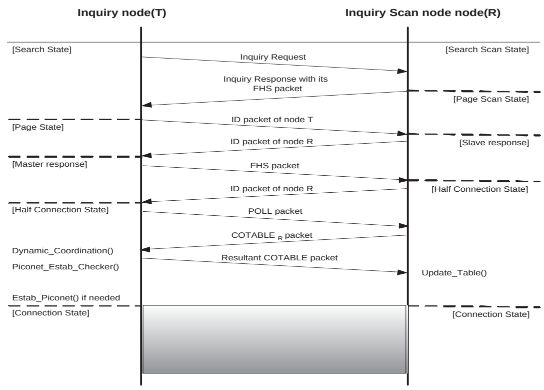

Fig. 2. The modified message flow for ASF algorithm

2.2.1 Half-Connection State

The half-connection is the state which the frequency hopping sequence of two Bluetooth devices is synchronized. In the Spec 1.1 [1], the master device sends a POLL packet and the slave device may respond to it with any packet in the frequency hopping sequence which is made by the master device's address and clock. We replace any response packet by the packet which contains the slave device's COTABLE and its leader's COTABLE. The master device having received the packet performs the coordination process and sends its result to the slave device. This is the half-connection state in which two devices are connected in the RF (Radio Frequency) layer. Further operations for the link-level connection is delayed until conditions for a piconet connection are satisfied.

3. ASF algorithm

Once we will discuss the environment on bluetooth devices before explain the ASF algorithm. A node means a bluetooth device separately. Generally, nodes are located in a space. Some nodes may be located in their communication range and others may not; we call it out-of-bounded case. If all the nodes are located in their communication range, they can be directly connected with one another; we call it in-bounded case. But, bluetooth devices under the ad hoc network are apt to lie on rather out-of-bounded case than in-bounded case because of their limited power. Therefore they cannot but be indirectly connected with one another. Under this environment, our goal is that we carry out a complete scatternet formation and further it has an optimal from piconet point of view – i.e the minimum number of piconet [7].

The ASF algorithm consists of two phases, the construction phase and the supplementary phase. The scatternet coordination is performed dynamically in the construction phase. Nodes in the search state gather information of their neighbor nodes as explained above. Nodes with the neighbor's information gathered in the search state enter into page state after $T_{searchTO}$ period, in which they participate in the dynamic coordination process. The coordination is performed every moments when two search and search scan nodes meet, and the coordinated nodes are partitioned into several groups; A *group* consists of one *leader* node and a number of its *member* nodes. A *leader* node is a node which participates in the coordination process, and a *member* node is a node which is half-connected to the leader node. The other details on the construction phase are provided in the following subsection. Secondly, the supplement phase is started after the construction phase has been finished; when either the number of members in a group are reached to the maximum number of slaves or a pre-defined construction time-out value is expired. In the supplement phase, a piconet is constructed and the established piconet schedules the order of nodes that will be in the search / search scan states and activates one node among its slaves in the order. The reason why only one node is activated at one given-duration is to save the overloads and to avoid unnecessary overlapped node detection.

3.1. The construction phase

It can be seen as a function which node T executes, when a node T receives node R's COTABLE as the POLL's response packet in Fig 2. There are four different cases according to their roles participating in the coordination process, leader-to-leader, leader-to-slave, slave-to-leader and slave-to-slave. Refer to Table 2, Table 3. From the table, we can see that Max_Include procedure is a main algorithm for the construction phase. It is used to merge two groups optimally when both of the leaders are in their communication range with each other.

As shown in Table 3, we can obtain a variety of results according as they are a node or a group. Once node vs node is a simple case. In case, a node T becomes a leader of a new group and a node R becomes affiliated with the node T as a member of the group. Next, in case of node vs group or group vs node, it is reasonable that a group tries to affiliate with a node. However, if a group has the maximum number of members(S_{max}), this group will not take a node any longer. Actually, this case means that this group is in the supplementary phase. Because a group which possesses S_{max} as a member already constructs a *piconet*. In this case, a node becomes a new leader instead of a group and makes a bridge to be connected with the group. At this time, an unshared member of the group is selected as a bridge. We can examine this case with examples through Fig 3. The last case, but it is more or less complicate, is group vs group. It is divided two cases according as connectivity exists or not regarding two group. Look at examples shown in Fig 4. Here, we notice some terminology *leaf* using in this case. When a group has only one bridge node,in other words, a group is located on the edge of a large scatternet, we call it *a leaf group*. Otherwise, we call it *a intermediate group*. In

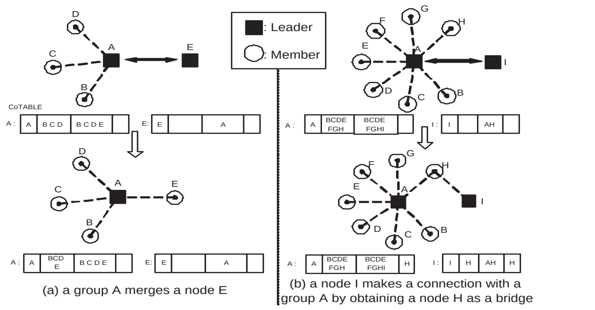

Fig. 3. Example : line 4-10 in Max_Include, Node vs Group

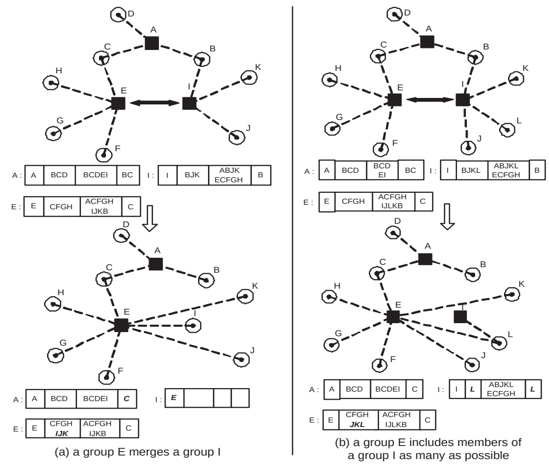

Fig. 4. Example : line 12-14 in Max_Include, Group vs Group, This illustrates the case in which connectivity among the groups exists

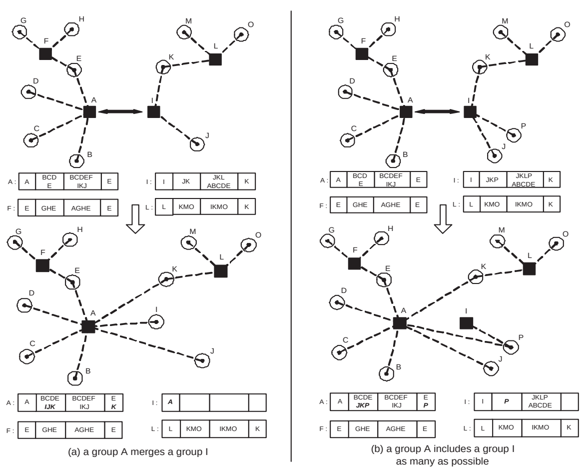

Fig. 5. Example : line 17-22 in Max_Include, Group vs Group, This illustrates the case in which connectivity among the groups does not exist

Table 2. Dynamic_Coordination Procedure

```
Dynamic_Coordination(COTABLE_R)
//L_T and S_T is a leader and a member of a node T respectively
//L_R and S_R is a leader and a member of a node R respectively
  Switch ( role(T), role(R) )
    case Leader : Leader
      Max_Include(COTABLE_{L_T}, COTABLE_{L_R})
    case Leader : Slave or Slave : Leader
      S ← S_R or S_T , L_1 ← L_T or L_R
      L_2 ← The leader of S_R or S_T
      if L_1 is reachable to L_2
        Max_Include(COTABLE_{L_1}, COTABLE_{L_2})
      else if S is reachable to all members L_2
        Role exchange between S and L_1
        Max_Include(COTABLE_{L_1}, COTABLE_{L_2})
      else Make a bridge as S
    case Slave : Slave
      if L_T is reachable to L_R
        Max_Include(COTABLE_{L_T}, COTABLE_{L_R})
      else if S_T is reachable to L_R and all members of L_T
        Role exchange between S_T and L_T
        Max_Include(COTABLE_{L_T}, COTABLE_{L_R})
      else if S_R is reachable to L_T and all members of L_R
        Role exchange between S_R and L_R
        Max_Include(COTABLE_{L_T}, COTABLE_{L_R})
      else if S_T is only reachable to L_R
        Make a bridge as S_T
      else if S_R is only reachable to L_T
        Make a bridge as S_R
      else
        A node T connects with a node R directly.
```

Fig 4, a group E and I will be a leaf group but a group A will be a intermediate group. Assume that a leaf E meets a leaf group I. In this case, a leaf group I disconnects the bridge node B and then a leaf group E merges a group I as Fig 4(a) or will try to include members of a leaf group I as many as possible as Fig 4(b). Here, though a leaf group I disconnects the bridge node B, total connectivity in the whole scatternet dose not have any problem because a group E will include a leaf group I and their connectivity still holds on. Rather it make a network simple.

Fig 5 illustrates the case in which two groups are not connected with each other. In this case, node T extracts a common nodes using a node R's COTABLE or vice versa. A common nodes means nodes that one group's neighbor lists in COTABLE is matched to the other group's member lists. In general, it is considered that all the bluetooth devices are not in their communication range – i.e out-of-bounded case. As a result, we can not force one group to include the other group unconditionally like in-bounded-case. So, we have to extract a reachable candidate from other COTABLE's member lists. To do so, we must have prior neighbor information and this can be done through the search process as stated above. Then, using common nodes, one group includes the

Table 3. Max_Include Procedure

```
Max_Include(COTABLE_T, COTABLE_R)
  // M_X means members of a group which a node X belongs to
  // N_X means neighbors of a group which a node X belongs to
Switch ( #(M_T), #(M_R) )
  case zero : zero            // Node vs Node
    L_T ← T ; M_T ← R ; L_R ← T
  case 0 : nz or nz : 0       // Node vs Group, nz : None Zero
    if #{M_nz} < S_max
      L_nz ← Node_nz
      M_nz ← M_nz + Node_0 ; L_0 ← Node_nz
    else
      B ← an unshared member of Group
      and make a bridge as B
  case default :              // Group vs Group
    if The connectivity between T and R exists
      if a group of T and R is a leaf
        A leaf is merged to other group as many as possible
      end if
    else
      {A} = N_T ∩ M_R //Extract common nodes(N_T,M_R);
      {B} = N_R ∩ M_T //Extract common nodes(N_R,M_T);
      if (#{A} ≥ #{B})  // Group T can include Group R
        a leader T includes {A}
      else
        a leader R includes {B}
```

other group as many as possible not to exceed S_{max} as member nodes.

3.2. The supplementary phase

This phase is occurred when the construction phase is over by following conditions; either the number of members in a group are reached to the maximum number(S_{max}) or a predefined construction timeout value is expired. In the supplement phase, a piconet is constructed and a link-level connection is also established at this time. The established piconet schedules the order of nodes that will be in the search/search scan states and activates one node among its members one after another – we call it a *revolving method*. The reason why only one node is activated at one given-duration is to save the overloads and to avoid unnecessary overlapped node detection. Through the revolving method, we can complete a scatternet formation with respect to nodes not connecting yet.

4. Simulation and Analysis

The simulation environment is as follows. We simulate ASF algorithm with Microsoft Visual C++. The bluetooth baseband and link layer protocols are implemented based on IBM BlueHoc simulator[8], which is an open source Bluetooth technology simulator on ns-2. And simulation results are measured every 0.315 μsec, which is equal to the clock tick on bluetooth. The frequency hopping kernel is implemented for the packet transmission. The simulation variables are as follows. The duration of $T_{searchTO}$ and $T_{pairingTO}$ is not fixed but has a uniform distribution about 1.28 sec and 4.8 sec on average respectively and $T_{roundTO}$ is fixed about 200 msec. The time of T_{pageTO} is 20 msec * the number of neighbor nodes. and $T_{searchscanTO}$ and $T_{pagescanTO}$ is fixed about 200 msec respectively. All results are averages of 100 trials.

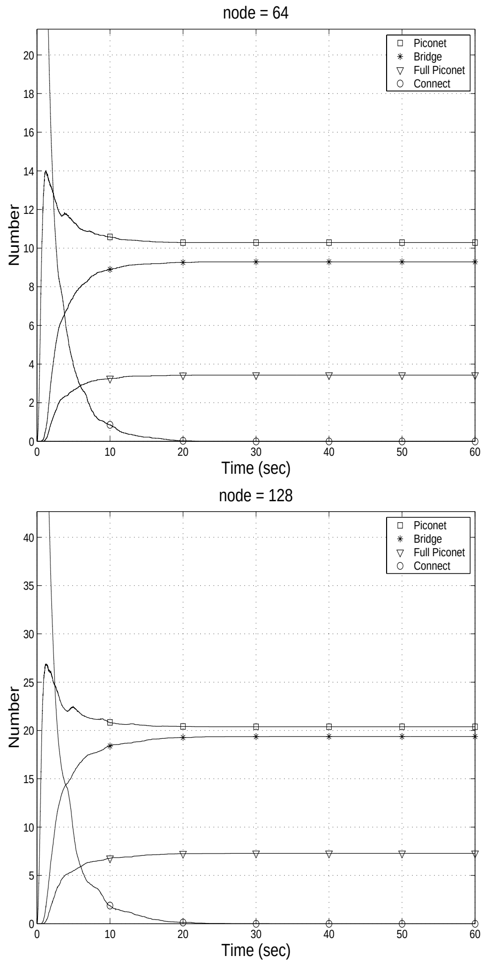

Fig. 6. BSFA simulation result according to the number of nodes about in-bounded case.

We have simulated the ASF algorithm according to two case; i.e. in-bounded and out-of-bounded case. Especially, in-bounded simulation is compared with existing BSFA algorithm. Look at the Fig 6. *Full piconet* means that a piconet master has 7 slaves and *connect* means a convergence measure. In other word, a scatternet formation is completed when *connect* becomes zero. As shown in Fig 6, ASF algorithm not only converges faster than BSFA according to the increment of nodes but also has the less number of piconet because full piconet numbers are larger than BSFA. Therefore, we become aware that ASF algorithm gets near to optimal.

Before explain the second simulation result, we will define some terminology.

$$\rho_i = \textit{the number of in-bounded neighbors at node } i \quad (1)$$

$$\rho_{avg} = \frac{\sum_{i=1}^{S} \rho_i}{S}, \text{ where S is the number of total nodes.} \quad (2)$$

ρ_{avg} means the number of neighbor to get in touch with

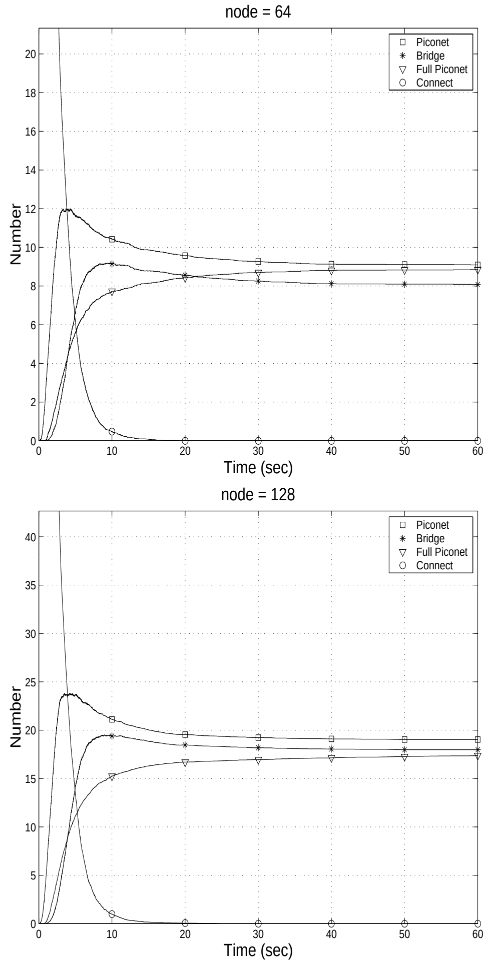

Fig. 7. ASF simulation result according to the number of nodes about in-bounded case.

on an average. So, it can be used out-of-bound measure. The more ρ_{avg} is high, the more nodes concentrate on one another. While the more ρ_{avg} is low, the more nodes are scattered. Out-of-bounded simulation is performed according to ρ_{avg} and the number of total nodes; 32, 64, and 128. As shown in Fig 10, the higher ρ_{avg} represents faster convergence. Moreover, we can see that a scatternet formation will be performed successfully if ρ_{avg} is over 5 as Fig 11. Further, if ρ_{avg} is over 14, we can obtain the same result as in-bounded case Fig 6,7.

5. Conclusion

This paper proposes an algorithm that can construct a scatternet on bluetooth even if in-bounded conditions are not satisfied. The ASF algorithm fully utilizes the Bluetooth's device discovery mechanism. Based on the knowledge of a device's neighbors, leader devices can determine how the coordination should be performed. The ASF algorithm also performs dynamic coordination and determine whether or not to make a piconet every time that a search node meets a scan node, so that the network can be gradually constructed. Then a large scatternet is formed effectively among the bluetooth devices.

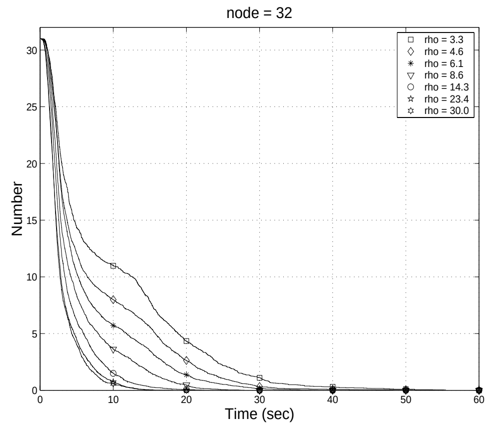

Fig. 8. Convergence according to ρ_{avg} when the number of nodes is 32.

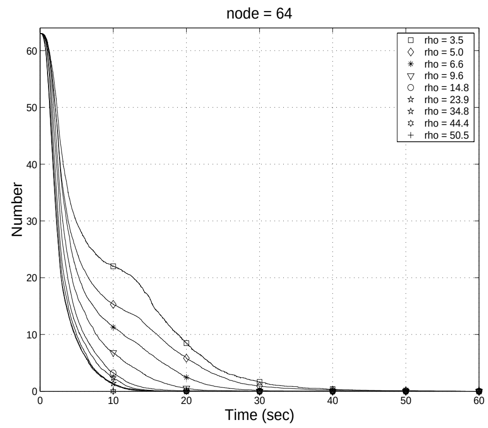

Fig. 9. Convergence according to ρ_{avg} when the number of nodes is 64.

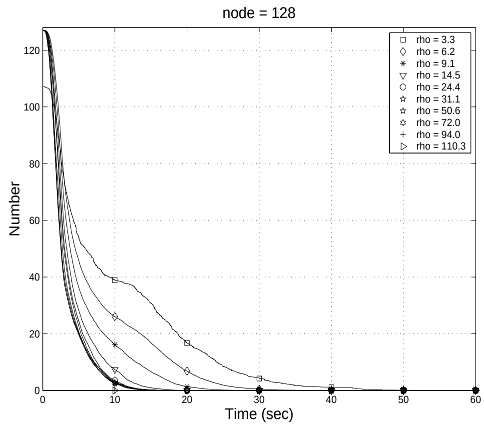

Fig. 10. Convergence according to ρ_{avg} when the number of nodes is 128.

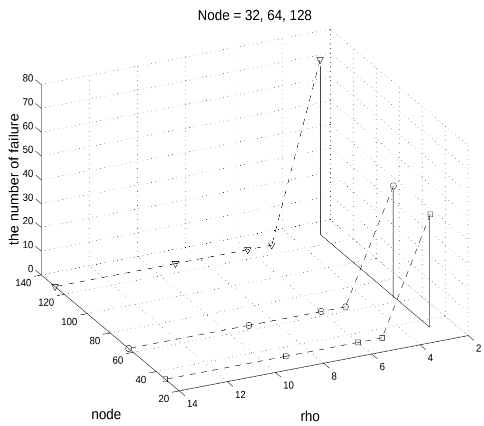

Fig. 11. the number of connection failure according to the number of nodes and ρ_{avg}.

References

[1] "Specification of the Bluetooth System 1.1," http://www.bluetooth.com.

[2] Theodoros Salonidis, Pravin Bhagwat, Leandros Tassiulas, and Richard LaMaire, "Distributed Topology Construction of Bluetooth Personal Area Networks," in *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE , Volume: 3* , 2001

[3] Ching Law and Kai-Yeung Siu, "A Bluetooth Scatternet Formation Algorithm," in *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE , Volume: 5* , 2001

[4] Ching Law, Amar K. Mehta, and Kai-Yeung Siu, "Performance of a new Bluetooth scatternet formation protocol," in *Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing 2001,* Long Beach, California, USA, Oct. 2001.

[5] G.V. Zaruba, S. Basagni, I. Chlamtac, "Bluetrees-scatternet formation to enable Bluetooth-based ad hoc networks," *IEEE International Conference on Communications,* vol.1, Page(s): 273 -277, 2001.

[6] S. Basagni, I. Chlamtac, and C. Petrioli, "Configuring BlueStars: Multihop Scatternet Formation for Bluetooth Networks", *IEEE Transactions on Computers,* 2002

[7] Gy. Miklos, A. Racz, Z. Turanyi, A. Valko, and P. Johansson, " Performance Aspects of Bluetooth Scaternet Formation," in *Proceedings of The First Annual Workshop on Mobile Networking and Computing,* 2000.

[8] BlueHoc : Bluetooth ad hoc network simulator by IBM, http://www-124.ibm.com/developerworks/projects/bluehoc