An Energy-Efficient Scatternet Formation Algorithm for Bluetooth-Based Sensor Networks*

Sain Saginbekov and Ibrahim Korpeoglu

Department of Computer Engineering
Bilkent University
TR-06800 Ankara, Turkey

Abstract

In this paper, we propose an energy-efficient scatternet formation algorithm for Bluetooth based sensor networks. The algorithm is based on first computing a shortest path tree from the base station to all sensor nodes and then solving the degree constraint problem so that the degree of each node in the network is not greater than seven, which is a Bluetooth constaint. In this way, less amount of energy is spent in each round of communication in the sensor network. The algorithm also tries to balance the load evenly on the high-energy consuming nodes which are the nodes that are close to the base station. In this way, the lifetime of the first dying node is also prolonged. We obtained promising results in the simulations.

1 Introduction

Energy is a very important resource in wireless sensor networks. The choice of wireless communication technology for a sensor network has an effect on the lifetime of the network since different wireless technologies have different power parameters. Another factor that greatly impacts the energy spent in a sensor network is the routing scheme used in the sensor network. Because the routing scheme affects how many times and over what distances packets have to be tranmitted.

Bluetooth is one of the wireless technologies available today and that can be a condidate for being the wireless communication technology for some type of sensor networks. Its low-cost, low-power, small size are the main features that make it a good candidate technology. A Bluetooth scatternet spanning all the sensor nodes, however, has to be formed first, before transmitting data in the sensor network. And it is not well specified in Bluetooth standards how to form scatternets. Since the topology of a scatternet affects the routing scheme, it also affects the energy consumption in the sensor network. Therefore, it is important to form scatternets that are energy-efficient.

In this paper, we propose an energy efficient scatternet formation algorithm for Bluetooth based sensor networks. The main objective of the algorithm is to prolong the lifetime of a sensor network by reducing the total energy consumed during data transfer from all sensor nodes to the base station and by balancing traffic evenly load among the high energy consuming nodes which are the nodes that are close to the base station.

The proposed algorithm is a centralized algorithm that is executed at the base station. Therefore, it requires the base station to know the exact locations of nodes so that the base station can compute the reachabilty information among the nodes.

The paper also gives simulation results to evaluate the proposed algorithm. The simulation results show that the proposed algorithm causes a sensor network to spent total energy not signifantly more than the lower bound. They also show that balancing the load evenly prolongs the lifetime of the first dying node significantly.

*This work is supported by The Scientific and Technical Research Council of Turkey with Grant EEEAG-103E014.

The rest of the paper is organized as follows. In Chapter 2, we give some background information and describe some of the related work. In chapter 3, we give our model of a sensor network for which the algorithm is developed. In chapter 4, we descibe the proposed algorithm in detail. In Chapter 5 we provide and discuss some results to evalute the algorithm. Finally, in chapter 6, we give our conclusions.

2 Background and Related Work

Various wireless communication technologies can be considered as the underlying communication technology for a sensor network. However, not all of these candidate technologies satisfy the requirements and constaints of sensor networks and sensor nodes. A sensor node should be low-cost, consume low-power, and should be of small size. Bluetooth [1] can be a good candidate technology since it is low-power, low-cost, and has small form factor.

As stated in [2], today's available hardware platforms for sensor networks can be divided into four classes: special purpose sensor nodes, generic sensor nodes, high-bandwidth sensor nodes, and gateway nodes. The power requirements of those nodes increase respectively. Bluetooth technology can be used as part of sensor nodes that fall into the third class. Current sensor nodes that use Bluetooth as an underlying communication technology are *BT node* (developed in 2001) and *Imote 1.0* (developed by Intel Research in 2003). The less energy consuming 802.15.4 and Zigbee can be used in sensor nodes that are included in the second class. 802.15.4 provides a data rate in the order or 250 Kbps, whereas Bluetooth provides a raw data rate of 1 Mbps. Therefore, Bluetooth is better for sensor networks applications that are bandwidth demanding.

Bluetooth technology suppports nodes to form ad hoc networks that are self-configuring. This property of Bluetooth is also important for sensor networks which are also required to be usually self-configuring and dynamic.

Bluetooth devices can be categorized into three classes with respect to the energy consumption during transmission. A class 1 Bluetooth device has a communication range of 100 meters and a transmit power of 100 mW (20 dBm). A class 2 device has a communication range of 50 meters and a transmit power of 2.5 mW (4 dBm). A class 3 device has a communication range of 10 meters and a transmit power of 1 mW (0 dBm).

Bluetooth supports different power modes. This is another feature of Bluetooth that can be utilized by sensor network applications. The available power modes are: *active mode*, *sniff mode*, *hold mode*, and *park mode*. In sniff mode, master and slaves agree on certain time intervals to communicate. The master then sends packets to a slave only on the agreed periods. In this way, a slave can go into sniff mode and sleep. This enables a slave to spend less energy. Hold mode is used by slaves to switch to other piconets. For that a slave goes into hold mode in the current piconet, and then switches to another piconet and becomes active in that piconet. When a slave is in hold mode in one piconet, it does consume any energy, since it is not involed in any communication activity in that piconet. Park mode is a mode where the device is not active. In this mode a device also does not retain its temporary MAC address. In this way more than 7 slaves can be connected to a single master. Some of these slaves will be in active mode (at most 7 of them) and the others will be in park mode.

In a sensor network, nodes usually do not transmit and receive data continuously, but at regular times or when an event occurs. Therefore supporting sniff and hold modes makes Bluetooth a good technology that can utilize the characteristics of sensor networks for energy conservation.

There are two ways to form a Bluetooth network. The first way is forming a piconet. However, a piconet may have at most 8 devices in it, and therefore most sensor networks can not be established as a single piconet. The second way to form a Bluetooth network is forming a Bluetooth scatternet. A Bluetooth scatternet consists of overlapping Bluetooth piconets. These overlapping piconets are inter-connected together with some special Bluetooth nodes called bridges.

A bridge node takes part in two or more pi-

conets in a shared manner. Each piconet has a different frequency hopping sequence. Therefore, taking part in a piconet requires switching to the hopping sequence of that piconet.

Constructing a scatternet is not a well-defined process in Bluetooth standards. There are several studies in the literature [3, 4, 5, 6, 7] that propose algorithms to form scatternets. Whatever the algorithms is, the final scatternet topology must satisfy the Bluetooth constaints such as each node having at most a degree of seven. The construction of scatternet can consider also the requirements of applications that will be run over the constructed scatternet. For example, a sensor network application usually requires the data to flow from sensor nodes to a single base station. Hence, a scatternet can be constructed with this property in mind so that some metrics can be optimized or improved. One such metric is energy efficiency.

Once a Bluetooth scatternet is formed, how to route the traffic over the scatternet is another issue that has to be addressed. Depending on the topology characteristics of the scatternets, the topology may also determine the routing policy. For example, if the topology is tree-shaped, then the routing scheme to be used is trivial: a node should send packets to its parent or vice versa. Hence, the scatternet topology affects the routing scheme that is used. They may be thightly-coupled or completely de-coupled.

Depending on the sensor network applications, the routing scheme may be optimized in various ways for energy efficiency. There are routing schemes proposed for sensor networks that are based on various metrics. A routing scheme, for example, can try to reduce the total energy consumed in the network per unit time; another routing scheme may try to distribute the load on sensor nodes evenly so that all nodes die nearly at the same time.

There are many studies on Bluetooth scatternet formation problem and many studies on routing problem in sensor networks. Those studies are separate from each other. There are very few studies, however, that are investigating both of the problems.

One [8] of these few studies that consider both of the problems mentioned above, forms a scatternet for a sensor network application using a clustering approach. The protocol, abbreviated as DCP, is divided in two phases: a set-up phase and a steady-state phase. In set-up phase, each node learns its neighbors and at least one packet forward address (PFA). A node in a network can take one of the two roles: *cluster-member* or *cluster-head*. Cluster-heads are selected randomly with a given probability. In steady-state phase, PFA is used to forward the data to the base station. Cluster members in a cluster periodically forward sensed data to their cluster head, and the cluster head, after fusing or compressing data, forward the data to the base station. If the cluster head is not in the communication range of the base station, it forwards the data through another cluster head. In DCP, a node in the formed network is not necessarily a master or a slave. They allow more than seven nodes to connect to a single node. However, the authors do not describe how a node can get associated with more than seven nodes. Moreover, simulation results show that for a given probability the number of unconnected nodes is high for a communication range of 10 meters.

In [9], a Bluetooth based sensor network is formed using the Bluetree protocol mentioned in [3]. Since the main consideration in Bluetree algorithm is not energy consumption, it is not a very good choice for sensor networks.

The scatternet formation algorithm proposed in [10] is divided again into two phases: knowledge discovery phase and connection setup phase. In knowledge discovery phase, some characteristics about the sensor nodes are gathered by the base station. In the connection setup phase, base station starts selecting one-hop apart nodes as slaves, those slaves select their neighbors as slaves, and this process is repeated until the leaves are reached. Since there can be only up to seven slaves in a piconet, they propose a new technique to select nodes as slaves according to some criteria. They have used simulated annealing [11] for

this purpose.

3 Model and Problem Statement

A model for a sensor network depends on various factors: the type of sensor network application, the characterics of the information collected, the capabilities of sensor nodes, the properties of the communication technology, etc. Hence a model requires some reasonable assumptions to be made. We make the following assumptions in establishing our model for which we provide a solution.

- Nodes are using class 3 Bluetooth radio chips to communication with each other and with base station. This implies that the range of transmission is 10 meters and the transmit power is 1 mW (0 dBm).
- Each node is in the radio coverage of at least one another node.
- The devices are not capable of power control. Hence, the energy required to transmit a fixed size packet over a single hop is contant and is independent from the distance. However, the energy spent for transmitting a packet is proportional to the packet size.
- Sensor nodes and the base station are stationary.
- No data aggregation is used. This implies that if a node receives several packets, it does not combine them into a single packet to transmit them to the next node. It sends them separetly to the next node. This is because for some applications data aggregation may not be allowed due to application semantics.
- The base station has complete knowledge about the location of sensor nodes. Therefore the base station knows which nodes are directly reachable from a given node.
- All nodes do not have to be in the communication range of each other.
- All nodes are homogenous and using Bluetooth technology to talk with each other and to the base station.

Now the problem is constructing a Bluetooth scatternet spanning a set of given sensor nodes with the properties listed above so that the total energy spent during a single round of data transfer from all sensor nodes to the base station is kept as low as possible and the lifetime of the first-dying node is extended as much as possible.

Since not all nodes are in the range of base station and since the degree of a Bluetooth node can be at most seven, we need multi-hop communication for carrying data from sensor nodes to the base station. This, however, causes more load to be put on nodes closer to the base station, since no data agregation is applied. The sensor nodes that are one-hop away from the base station (i.e. directly reachable from the base station) consume more energy than the other nodes that are two or more hops away from the base station. Therefore, a node at the first level will be the first one to die assuming all nodes have initially the same amount of energy. The situation will be worse if there is an unbalance in the amount of traffic forwarded by these first level nodes.

4 Algorithm

In this section we describe our algorithm. But, first we define some terminology used in the paper.

A *round of communication* is the activity in which each node senses the environment (hence obtains data) and sends the data to the base station using mult-hop communication. After one round of communication data from all nodes reach to the base station. *Degree* of a node is the number of neighbors that node is connected to. *Parent* of a node X is the node that is directly connected to the node X and that has one less hop to the base station. *Possible parent* of node X is the node which is directly reachable by X and which has one or more less hops to the base station. *Possible brother* of a node X is the node which is a child of the parent of X. *Possible sibling* of a node X is a node which is at the same level

with X (a sibling does not have to be the child of the parent of X). *Level* of a node is the number of hops between the node and the base station. Hence the base station is at level 0. *Grandparent* of a node X is the first level node which is on the path between node X and the base station. Note that our definition for grandparent is different than the common definition which states that the grandparent of a node is the parent of the parent of that node.

4.1 Scatternet Construction Algorithm

To prolong the lifetime of a sensor network, the power consumption per round of communication has to be close to minimum and the energy consumption must be balanced among the sensor nodes (this is also the approach followed in [12]). Following this approach, our scatternet construction algorithm is divided into two parts. In the first part, our algorithm constructs a shortest path tree, rooted at the base station, over the reachability graph of all sensor nodes. We have an edge between two nodes in the reachability graph if these two nodes are in the Bluetooth communication range of each other. We mean with a shortest path tree as the tree where the path connecting a node in the tree to the base station has the shortest length among all possible paths that can connect that node to the base station in the reachability graph. Since each edge in the reachability graph has unit distance, a shortest path gives the minimum number of hops possible between the corresponding node and the base station.

After constructing the shortest path tree, the algorithm makes further arrangements over the tree so that the degree of each node in the tree is no greater than seven. This is a Bluetooth constraint.

The second part of the algorithm tries to balance the energy consumption of the first level nodes in the tree so that the lifetime of the first dying node is prolonged as much as possible. Both parts of the algorithm are run at the base station.

Our goal in the first part of the algorithm is to form a scatternet so that the power consumed in a round of communication is reduced as much as possible. If a data packet travels minimum number of hops, a shortest path, from a sensor node to the base station, the energy spend in the network for that packet will be minimum. If the data packets of all nodes are transported over the shortest paths, then the energy spent per round of communication will be minimum. However, a shorest path tree does not have any degree containt, and therefore some nodes may have a degree greater than seven. We solve this degree problem by re-arranging the connections. With this we may deviate from the shortest path tree, depending on the topology, but as the simulation results show, the heuristic still performs good.

For a given number of nodes, the minimum energy spent per round of communication can be approximated with the energy spent in a tree that is formed in such a way so that each node except the root has six children. The root can have seven children. We can call this a 6-ary tree. The total energy consumption (E) per round of communication in such a tree can be expressed as follows:

$$E = 7 \times \sum_{i=1}^{\lfloor \log_6 N \rfloor} 6^{i-1} \times i + (N - 1 - 7 \times \sum_{i=0}^{\lfloor \log_6 N \rfloor - 1} 6^i) \times \lceil \log_6 N \rceil) \times \alpha \quad (1)$$

where N is the number of nodes and α is a constant value denoting the energy spent to transmit and receive a single packet over a single hop. The equation is just an approximation since the tree is not an exact 6-ary tree because base station may have seven slaves and the leaf level may not be full. This equation, however, gives an approximate lower bound on the energy spent in a Bluetooth-based network per round of communication. We will use this lower bound in the simulations to compare our algorithm against it.

The energy consumption of each node of a 6-ary tree will be

$$E_{l,i=1..N} = (\sum_{j=0}^{\log_6 N - l} 6^j + 1) \times \alpha \quad (2)$$

where l is the level of that node.

The first part of the algorithm works as follows (related pseudo-codes are given in **Algorithm** 1 and **Algorithm** 2). First a shortest path tree spanning all nodes and rooted at the base station is formed using Breadth First Search (BFS) Algorithm or Dijkstra's Single-Source Shortest Paths Algorithm [13]. Lets call the tree formed in this way an SPT. An SPT formed in this way can have nodes whose degree is greater than seven. Therefore, after forming the SPT, the algorithm, starting from the leaves up to the root, checks all nodes if there exists a node that has more than six children, except the base station. Base station can have seven children. If it finds such a node X, then the children of node X is tried to be connected to some other possible parent whose number of children is less than six. If possible, this is repeated until the number of children of node X becomes at most six.

If we cannot reduce the number of children of X to six in this way (that means there is no alternative parent), then, starting from the child of X with minimum number of descendants, each child of X is tried to be connected to possible brothers or possible siblings. If possible, this is repeated until the number of children of X becomes at most six.

If, after this process, the number of children of X still exceeds six, then a child A of X with minimum number of descendants is connected to another child B of X where B has minimum number of descendants after A. After getting connected to B, A is disconnected from X. In this way the number of children of X is reduced by one. Then, if B's degree exceeds six, it is tried to be reduced using the same approach applied to X. Hence, a recursive algorithm is used here. Notice that, since the algorithm starts from the bottom, B had already solved its degree problem. So, B had to have at most six children before A is connected to it.

We call the tree obtained by the execution of the first part of our algorithm as an unbalanced degree constrained tree (a UDC tree).

Next we describe the second part of our al-

Algorithm 1 Scatternet Construction Algorithm

Input: Distance matrix or neighborhood matrix
Output: Balanced Degree Constrained Tree (BDC Tree)
Form Shortest Path Tree using Breadth First Search or Dijkstra's Single Source All Shortest Paths Algorithm
for each level k=numberOfLevels-1 to 1 **do**
  for each node n of level k **do**
    if $n.numberOfChildren > 6$ **then**
      for each child ch of n **do**
        for each possible parent pP of ch **do**
          if $pP.numberOfChildren < 6$ **then**
            disconnect ch from n
            connect ch to pP
            break
          end if
        end for
        if $n.numberOfChildren \leq 6$ **then**
          break
        end if
      end for
    end if
    if $n.numberOfChildren > 6$ **then**
      while $n.numberOfChildren \geq 7$ **do**
        Reconnect($n.child$ whose number of descendants is the minimum)
      end while
    end if
  end for
end for
if $root.numberOfChildren > 7$ **then**
  while $root.numberOfChildren \geq 8$ **do**
    Reconnect($root.child$ whose number of descendants is the minimum)
  end while
end if
Balance()

Algorithm 2 Reconnect(node)

```
boolean cont=true
tempParent=node.parent
for each node.possibleParents pP do
  if pP.numberOfChildren < 6 then
    disconnect node from tempParent
    connect node to pP
    cont=false
    break
  end if
end for
if cont then
  for each node.possibleSiblings pS do
    if pS.numberOfChildren < 6 then
      disconnect node from tempParent
      connect node to pS
      cont=false
      break
    end if
  end for
end if
if cont and number of possible brothers ≥ 1
then
  brother=child of tempParent whose number
  descendants is the minimum after node
  disconnect node from tempParent
  connect node to brother
  Reconnect(brother.child whose number of
  descendants is the minimum)
end if
```

gorithm (pseudocodes are given in **Algorithm** 3 and **Algorithm** 4).

4.2 Balancing Algorithm

In this part of the algorithm, first level nodes are balanced according to their number of descendants. Since the nodes that are one hop apart from the base station will drain more energy due to having more descendants than the other nodes, they will die first. These first level nodes have to forward their descendants data in addition to their own data. The situation will be worse if they are formed in an unbalanced manner, in other words if the number of descendants will differ a lot. The nodes with more descendants will die quicker than the nodes with less descendants. Furthermore, if the children of a dying node do not have any other possible parents, these children cannot forward their data to the base station after that node dies.

The Equation 3 expresses the amount of energy consumption at a node X ($E(X)$) as a function of its descendant nodes. The number of descendants of a node X ($D(X)$), on the other hand, can be expressed depending on the descendants of its set of children C (Equation 4).

$$E(X) = (D(X) + 1) \times \alpha \,. \tag{3}$$

$$D(X) = \mid C \mid + \sum_{i=1}^{|C|} D(C_i), \quad C_i \in C, \;\; 1 \leq i \leq \mid C \mid \;. \tag{4}$$

Figure 1 shows a sample network that is unbalanced at the first level. In this figure, node B has six descendants while node A has only one. The other first level nodes do not have any descendants. The dashed lines show the reachability information. If there is a dashed line between two nodes, the nodes are not connected with a Bluetooth link at the moment, but can be connected with a Bluetooth link if required. The balance of this tree can be improved at the first level, because the nodes D, F, and G can be connected to the nodes A, C, and F, respectively. When this re-arrangement is done, the tree will be more balanced at the first level. Note that we are only

concerned with balancing at the first level of the tree, since this is the level that will have nodes to die first. If we do not balance the tree, node B can die very fast. After the death of node B, nodes D, F, and G can be connected to other parents, namely to A, C, and F, respectively. But node E does not have any other possible parent to connect to. Node E and its descendants can only connect to node G. The new shape of the tree after balancing is shown in Figure 2.

The balancing should be done in a way so that degree constraints of nodes are not violated. Additionally, our balancing algorithm balances the descendants of first level nodes in such a way that the energy consumption in one round of communication is not increased in the resulting topology. In fact the energy consumption may even decrease. Although our algorithm is only concerned with balancing at the first level at the moment, if needed, it can be easily modified to balance other levels as well. We just have to call it recursively to balance other levels.

The idea of the algorithm can be illustrated using the Figure 3(a). In the figure, a number besides a node shows the number of descendants of that node. For the sake of simplicity, we will label nodes in the network with those numbers. The bold lines in the figure show the current connections between nodes, and the dashed lines show possible alternative connections.

In order to balance the energy consumption, we have to make the number of descendants of first level nodes as equal as possible. To achieve that, we look to the nodes at the second level (these nodes are the children of first level nodes) and find the one that has the maximum number of descendants. We then try to reconnect it to another parent in the first level. In the example shown in Figure 3, we start from node labeled with 15 because it is the maximum, indicating that this node has the maximum number of descendants (15 descendants). Since there is no other possible parent of 15 other than 16, we leave it as it is. Second maximum number is 11. We look all the possible parents of 11 and see which one has the least number of descendants

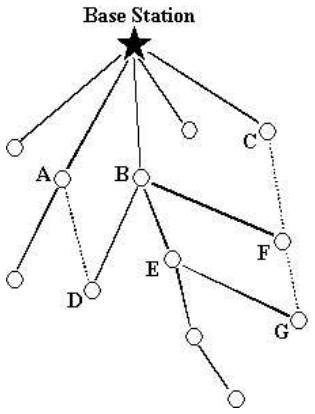

Figure 1: Unbalanced tree.

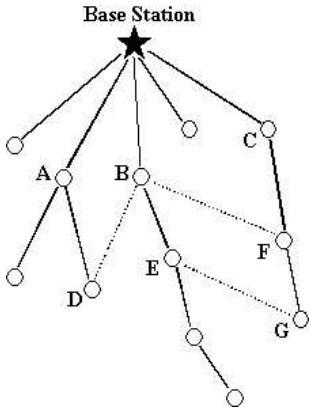

Figure 2: Balanced tree.

other than descendants due to node labeled with 11. Node labeled initially with 32 has 20 descendants $(32 - 11 - 1)$ and node labeled with 10 has 10 descendants. Therefore, we choose the node labeled with 10 in the first level as the new parent of the node labeled with 11 in the second level. We disconnect node 11 from 32 and connect it to node 10. New values of parents will be 20 and 22, whereas they were 32 and 10 earlier. So we achieve a better balance at the first level.

We continue applying the same procedure until all nodes at the second level are checked in the sorted order of their labels. Figure 3(b) shows the balanced configuration of nodes. As it can be seen in the figure, the node that was consuming

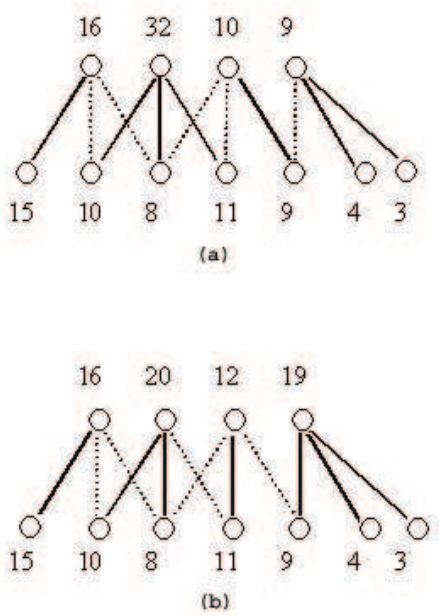

Figure 3: a) Unbalanced nodes; b) Balanced nodes

the maximum energy in the unbalanced configuration, is still the node that is consuming the maximum energy, but its energy consumption is reduced by about 30 %.

The pseudocode of our balancing algorithm is shown in **Algorithm** 3.

Algorithm 3 Balance()

```
for each level k=2 to numberOfLevels do
  Sort level k nodes in descending order according to their labels expressing the number of descendants
  for each node n of level k do
    newParent=Min(possible parents of n)
    if n.parent ≠ newParent then
      disconnect n from n.parent
      connect n to newParent
    end if
  end for
end for
```

We call the tree obtained after executing the second part of our algorithm as a balanced degree constrained tree (a BDC tree).

Algorithm 4 Min(array)

return the node whose grandparent's energy consumption is minimum and the number of children less than 6

4.3 Routing

Routing of sensor network data in a Bluetooth scatternet with a BDC tree topology is then very simple. Since only one path exists between each node and a base station, no routing tables or other nodes' addresses have to be maintained at nodes, except the address of the master node (parent) and the addresses of the children. Each node will forward its data to its master. Master then forwards the data to its master and so on until the data reaches the base station.

5 Simulations and Results

In this section we provide and discuss some results we obtained from our simulations run to evaluate our algorithm.

We have implemented our simulation model using Java programming language. Our simulations were static simulations without a time axis.

In our simulation experiments, we compare performance results for various scatternet topologies: unbalanced degree constrained tree (UDC Tree), balanced degree constrained tree (BDC Tree), shortest path tree (SPT), and 6-ary tree (a totally balanced and degree constrained tree). 6-ary tree gives the lower bound for energy consumption in a balanced tree satisfying Bluetooth node degree constraint. However, a 6-ary tree may not be always a feasible topology due to reachability constraints. An SPT tree gives a lower bound for energy consumption in a tree satisfying the reachability constraints but not satisfying degree constraints.

In our simulation model, different number of nodes, ranging from 75 to 500, are deployed randomly on an area of 50 m by 50 m. Since some of the nodes may not have any neighboring nodes after random deployment, we get rid those nodes and consider only nodes that have at least one neighbor that is part of the sensor network. For each simulation experiment we repeat running the

simulation 100 times and we take the average of these 100 measurements.

Figures 4, 5, 6, and 7 show how 150 nodes are scattered randomly over a region that is 50 m x 50 m, and how various topologies look like: shortest path tree, unbalanced degree constrained tree, and balanced degree constrained tree. As it can be seen in Figure 5, shortest path tree can have nodes that have node degree greater than seven. But, both UDC and BDC tree based scatternets satisfy the degree constraint: the number of children of each node is no more than six, except the root node which can have seven children. Notice that some nodes in the unbalanced tree are connected to different parents in the balanced version.

In Figure 8, energy consumption per round of communication versus number of nodes in the network is shown for various topologies. The lower bound of energy consumption for a given set of nodes is achieved if routing is done according to a shortest path tree. As it is seen in the figure, the energy consumption in sparsely deployed networks is almost equal to the lower bound. However, in densely deployed networks, the energy consumption is more than the lower bound. This is because some of the nodes have more than six children in a dense network, and enforcing the tree to be degree-constrained makes the tree to deviate from optimal routing.

Figure 9 shows the energy consumption of the *maximum energy consuming node* versus the number of nodes in the network for different topologies. BDC tree has better energy consumption values compared to UDC tree. Balancing algorithm reduces the energy consumption of the maximum energy consuming node by about 30 to 50%. Thus, the lifetime of the first dying node is increased by about 40 to 100% (assuming that all nodes have equal amount of initial energy).

Figure 10 shows the average number of slaves per node in a network. The number of nodes in the network is varied on the x-axis. The figure also shows the average number of hops between a node and the base station. The average number of slaves increases slightly as the number of

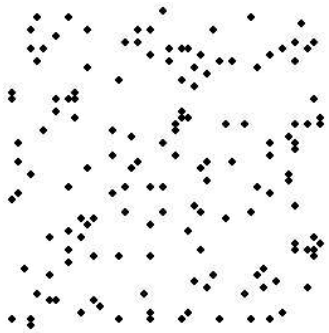

Figure 4: Randomply deployed sensor nodes.

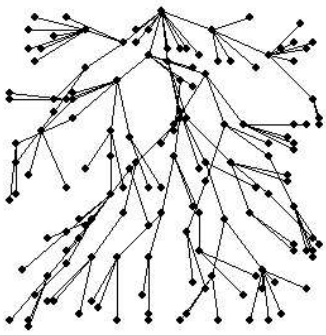

Figure 5: Shortest Path Tree formed from randomly deployed nodes.

nodes increases, as expected. The hop number is between 4 and 4.5. There is no significant change on the number of hops as a function of number of nodes. We think this is because as the network becomes denser, both the number of nodes which are nearer to the base station (small hop count) and the number of nodes which are further away (large hop count) increases with the same ratio, hence not changing the average value of hop count.

Figure 11 compares the total energy consumption per round in a BDC tree topology and in a 6-ary tree topology. 6-ary topology is an optimal configuration to consume minimum energy in a balanced tree satisfying Bluetooth constraints. We can see that energy consumed in a BDC tree is just slightly more than the lower bound.

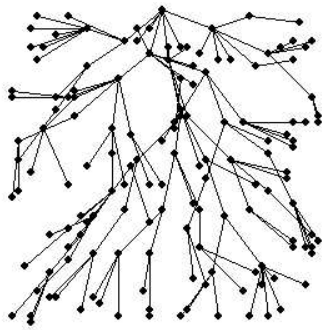

Figure 6: Unbalanced Degree Constrained Tree (UDC Tree).

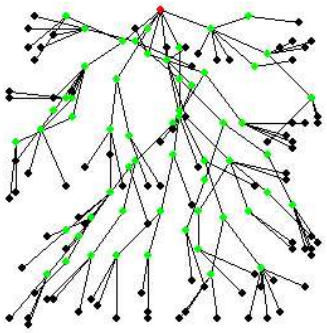

Figure 7: Balanced Degree Constrained Tree (BDC Tree). Light-color nodes are the M/S bridges, dark-color nodes are the slaves, and base station is a master.

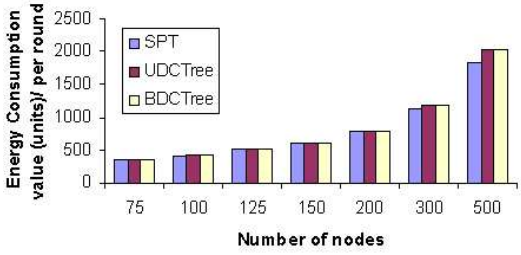

Figure 8: Average energy consumptions of SPT, UDC Tree, and BDC Tree per round.

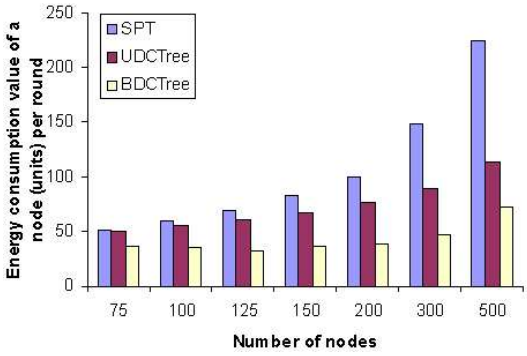

Figure 9: Average maximum energy consumptions of a node in SPT, UDC Tree, and BDC Tree per round.

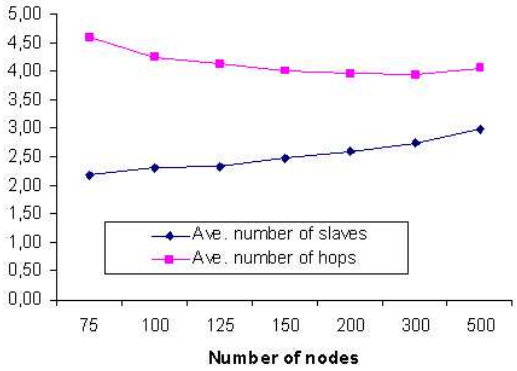

Figure 10: Average number of hops of BDC Tree as a function of node numbers.

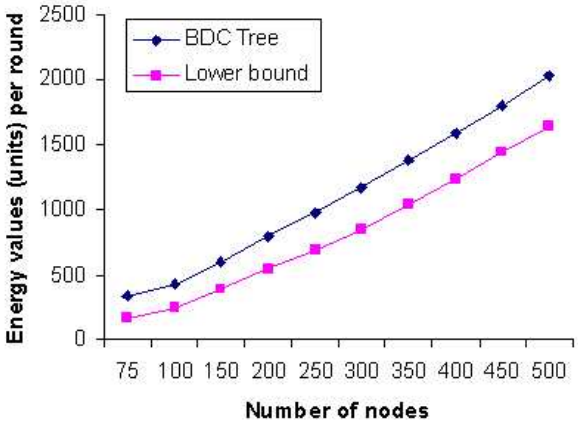

Figure 11: Comparison of energy consumptions of BDC Tree with lower bound.

6 Conclusions

Bluetooth is one of the wireless communication technologies that can be used as the communication technology for wireless sensor networks. Its low-cost, low power, and small size features make it a good candidate for wireless sensor networks.

In this paper we propose an algorithm about how to form an energy-efficient scatternet for Bluetooth based wireless sensor networks. The goal of the algorithm is to reduce the energy consumed in a round of communication and also to balance the traffic load on the high-energy consuming nodes. These nodes are the nodes that are close to the base station since all traffic has to be forwarded over these nodes.

The simulation results show that our algorithm consumes not significantly more energy than a possible lower bound. The simulation results also show that by balancing the load of the nodes closer to base station, the algorithm can prolong the lifetime of the first-dying node by up to 100%.

References

[1] B. A. Miller and C. Bisdikian. *Bluetooth Revealed: The Insider's Guide to an Open Specification for Global Wireless Communications*. Prentice Hall, 2001.

[2] J. Hill, M. Horton, R. Kling, and L. Krishnamurthy. The platforms enabling wireless sensor networks. *Communications of the ACM*, 47:41 – 46, June 2004.

[3] G. V. Zaruba, S. Basagni, and I. Chlamtac. Bluetrees-scatternet formation to enable bluetooth-based ad hoc networks. In *Proceedings of IEEE International Conference on Communications*, volume 1, pages 273–277, June 2001.

[4] B. J. Prabhu and A. Chockalingam. A routing protocol and energy efficiency techniques in bluetooth scatternets. *Design and Test of Computers*, Apr./May 2002.

[5] C. Law, A. K. Mehta, and K.-Y. Siu. A bluetooth scatternet formation algorithm. In *Proceedings 2001 ACM International Symposium on Mobile ad hoc networking and computing*, pages 183 – 192, Oct. 2001.

[6] C. Petroli, S. Basagni, and I. Chlamtac. Configuring bluestars: Multihop scatternet formation for bluetooth networks. *IEEE Transactions on Computers*, 52:779 – 790, June 2003.

[7] T. Salonidis, P. Bhagwat, L. Tassiulas, and R.LaMaire. Distributed topology construction of bluetooth personal area networks. *Proceedings Twentieth Annual Joint Conference of the IEEE Computer and Communication Societies IEEE INFOCOM 2001*, 3:1577 – 1586, Apr. 2001.

[8] H. Mathias, D. Jan, and T. Dirk. Energy-efficient data collection for bluetooth-based sensor networks. *IEEE Instrumentation and Measurement Technology Conference*, May 2004.

[9] M. Leopold, M. Dydensborg, and P. Bonnet. Bluetooth and sensor networks: A reality check. In *Proceedings of SenSys 2003*, Nov. 2003.

[10] V. Mehta and M. El Zarki. Fixed sensor networks from civil infrastructure monitoring - an initial study. In *First Annual Mediterranean Ad Hoc Networking Workshop(Med-hoc-Net 2002)*, Sept. 2002.

[11] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecci. Optimization by simulated annealing. *Science*, 220:671 – 680, 1983.

[12] H. O. Tan and Ibrahim Korpeoglu. Power-efficient data gathering and aggregation in wireless sensor networks. *ACM SIGMOD Record*, 32:66 – 71, Dec. 2003.

[13] T.H. Cormen, C. E. Leisorson, R. L. Rivest, and C. Stein. Introduction to algorithms, second edition, 2001.