Implementation of Leach protocol on tinyOS

TinyOS free simulator

Fragkoulis Logothetis
Electrical and Computer Enginnering
Chania ,Crete

Abstract

LEACH uses localized coordination to enable scalability and robustness for dynamic networks, and incorporates data fusion into the routing protocol to reduce the amount of information that must be transmitted to the base station. Simulations show that LEACH can achieve as much as a factor of 8 reduction in energy dissipation compared with conventional routing protocols. In addition, LEACH is able to distribute energy dissipation evenly throughout the sensors, doubling the useful system lifetime for the networks we simulated.

I. Introduction

These project consists some pieces based on each other

- A C-programm that creates a grid topology ,to make a various of tests in the simulation
- The basic part of the project is to simulate the Leach protocol ,using timers and send/receive messages implementing a cumulative function according to the Leach .

II. Analyze Parts of the Project

A. *Building a grid -Topology*

It is useful to control the operation of our program creating topology files, including multiple sensors, by a "Automatic" way. In this programm we are prompted to create a program which :

ㅎ It takes as integer 1 parameters (we will refer to it as diameter D) and a floating point number (we will refer to it as signal range).

ㅎ Will create D * D nodes

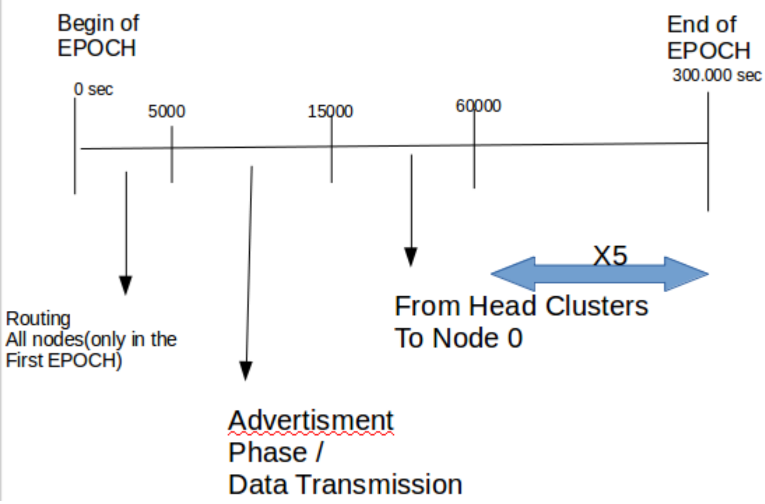
ㅎ Considering that the horizontal and vertical distances of the nodes in the grid are equal to 1, it is easy for any node to find all the nodes which are less than or equal to its range . For example, if the range is 1.5, then a central one node has 8 neighbors .

ㅎ If for each node you find its neighbors, then you can this information use it to create a topology file.

B. *Simulate Tag Protocol*

Each sensor would produce a random value every 60 seconds in order to give his father appropriate informationto find average and dispersion of measurements across the network.AVG and VARIANCE functions are simultaneously calculated . The simulation ends after 60 seconds. The base station (node 0) should prints the final result in each season.The value of each node with a K-identifier in each season will be a random integer number in interval [K..K + 20].

C. *What it is needed to do in each EPOCH ?*

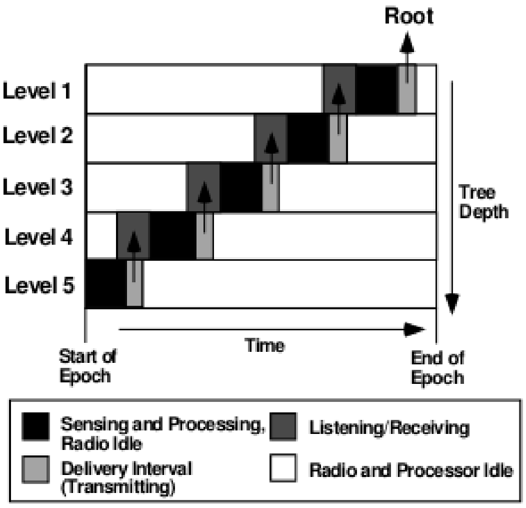


D. *(First-First Job) Routing Procedure and Wake Up Time*

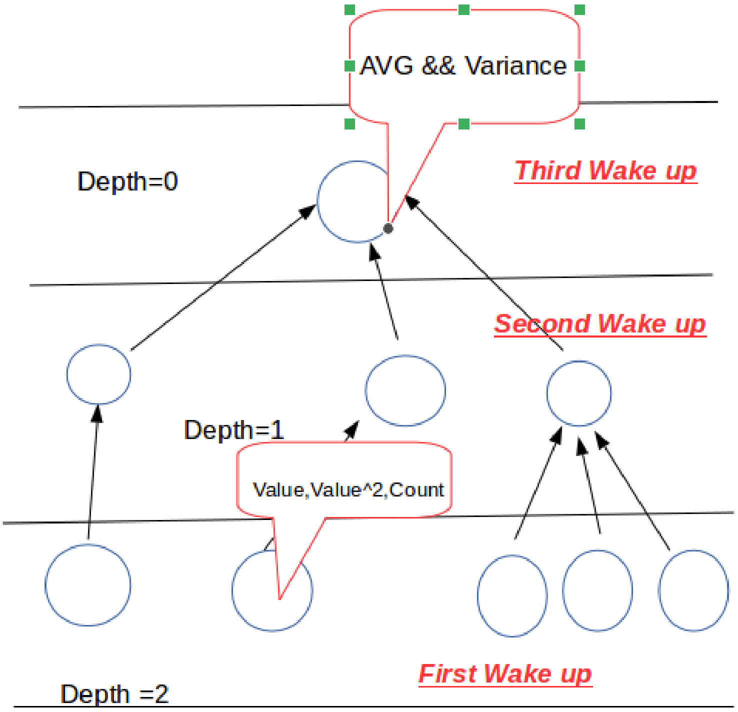
The EPOCH DURATION lasts 60.000 ms. During the half of the first 5000 sec a routing procedure is token place .Each node sends a broadcast message to incorporate some new nodes in their sub-tree. When a new node receives a Broadcast message ,then it defined the sender of the message as its parent . Then the node re-broadcast the routing message . This node will wake up from sleeping mode after :

ROUTING_TIME - (PROCCESSING_TIME*(curdepth+1)) + (MAX_SENSORS – curdepth) * PROCCESSING_TIME + rand()%80 + (AdvertismentPhase+DataTransimition)

The worst latency will be took place when the route of the nodes are like a connected list. So I have used this approach to built the Timer .Some nodes will be the route of the Head-Clusters to transit their value to Node 0 ,so their values will be zero.



Time Diagram of a Routing Tree

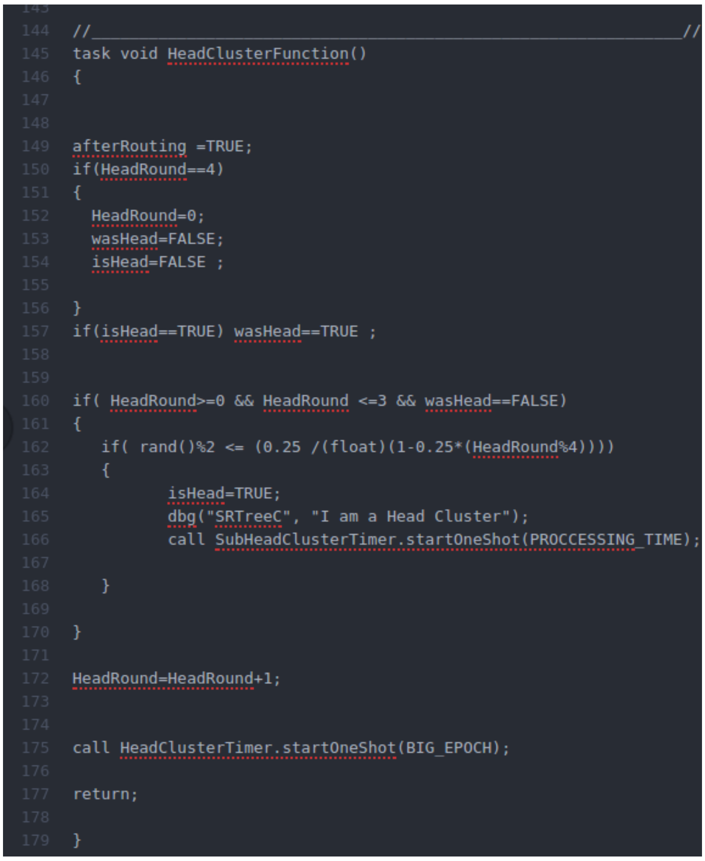


E. ***Election of Head-Clusters***

Initially, when clusters are being created, each node decides whether or not to become a cluster-head for the current round. This decision is based on the suggested percentage of cluster heads for the network (determined a priori) and the number of times the node has been a cluster-head so far. This decision is made by the node n choosing a random number between 0 and 1. If the number is less than a thresh-old T (n) , the node becomes a cluster-head for the current round. The threshold is set as:

$$T(n) = \begin{cases} \frac{P}{1 - P*(r \bmod \frac{1}{P})} & \text{if n} \in \text{G} \\ 0 & \text{otherwise} \end{cases}$$

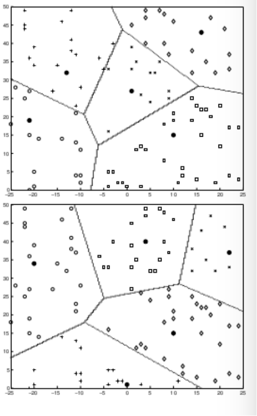
Using this threshold, each node will be a cluster-head at somepoint within 1/p rounds. During round 0 (r = 0), each node has a probability P of becoming a cluster-head. The nodes that are cluster-heads in round 0 cannot be cluster-heads for the next 1/p -1 rounds. Thus the probability that the remaining nodes are cluster-heads must be increased, since there are fewer nodes that are eligible to become cluster-heads. After 1/P-1 rounds, T = 1 for any nodes that have not yet been cluster-heads, and after 1/P rounds, all nodes are once again eligible to become cluster-heads.

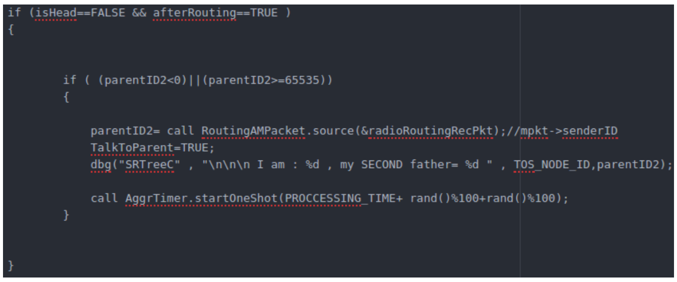


```
//________________________________________________________________________//
task void HeadClusterFunction()
{


afterRouting =TRUE;
if(HeadRound==4)
{
  HeadRound=0;
  wasHead=FALSE;
  isHead=FALSE ;

}
if(isHead==TRUE) wasHead==TRUE ;


if( HeadRound>=0 && HeadRound <=3 && wasHead==FALSE)
{
   if( rand()%2 <= (0.25 /(float)(1-0.25*(HeadRound%4))))
   {
          isHead=TRUE;
          dbg("SRTreeC", "I am a Head Cluster");
          call SubHeadClusterTimer.startOneShot(PROCCESSING_TIME);

   }

}

HeadRound=HeadRound+1;


call HeadClusterTimer.startOneShot(BIG_EPOCH);

return;

}
```

F. *Advertisement Phase*

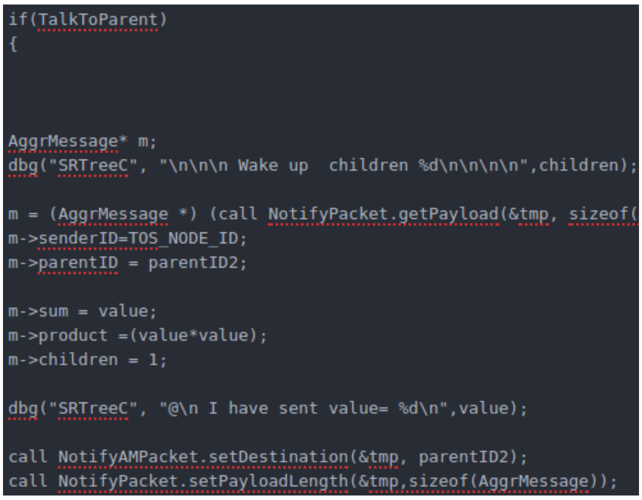
Each node that has elected itself a cluster-head for the current round broadcasts an advertisement message to the rest of the nodes. For this “cluster-head-advertisement” phase, all cluster-heads transmit their advertisement using the same transmit energy. The non-cluster-head nodes must keep their receivers on during this phase of set-up to hear the advertisements of all the cluster-head nodes. After this phase is complete, each non-cluster-head node decides the cluster to which it will belong for this round. Assuming symmetric propagation channels, the cluster-head advertisement heard with the largest signal strength is the cluster-head to whom the minimum amount of transmitted.





```
if (isHead==FALSE && afterRouting==TRUE )
{



        if ( (parentID2<0)||(parentID2>=65535))
        {

            parentID2= call RoutingAMPacket.source(&radioRoutingRecPkt);//mpkt->senderID
            TalkToParent=TRUE;
            dbg("SRTreeC" , "\n\n\n I am : %d , my SECOND father= %d " , TOS_NODE_ID,parentID2);

            call AggrTimer.startOneShot(PROCCESSING_TIME+ rand()%100+rand()%100);
        }


}
```

G. *From Sub-Nodes to Their Head-Clusters*



```
if(TalkToParent)
{




AggrMessage* m;
dbg("SRTreeC", "\n\n\n Wake up  children %d\n\n\n\n",children);

m = (AggrMessage *) (call NotifyPacket.getPayload(&tmp, sizeof(
m->senderID=TOS_NODE_ID;
m->parentID = parentID2;

m->sum = value;
m->product =(value*value);
m->children = 1;

dbg("SRTreeC", "@\n I have sent value= %d\n",value);

call NotifyAMPacket.setDestination(&tmp, parentID2);
call NotifyPacket.setPayloadLength(&tmp,sizeof(AggrMessage));
```

H. *From Head-Clusters to Node 0*

I haven’t used the TDMA procedure as following our teacher instructions I have incorporated the Tag Protocol to achieve the Transmission procedure .After a certain time, which is determined a priori, the next round begins with each node determining if it should be a cluster-head for this round and advertising this information.

G. *More work*

- *If a node isn’t a Head-Cluster in a round and it didn’t hear any “cluster-head-advertisement” ,then it elect itself as a Head-Cluster for this round without broadcast an advertisement message .*
- *I haven’t used the CSMA MAC and TDMA protocol,but I have included the Tag Protocol To transmit the information from Head-Cluster to Node 0*
- *Both to avoid collisions and to minimize the transmitted packets ,if a node has no information to send (Ex. Intermediate node) then it won’t send .*