Wireless Sensor Networks Graded Lab 2

Prasanna Natarajan
1410110298

Code:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                             %
% This is the LEACH [1] code we have used.                                    %
% The same code can be used for FAIR if m=1                                   %
%                                                                             %
% [1] W.R.Heinzelman, A.P.Chandrakasan and H.Balakrishnan,                    %
%     "An application-specific protocol architecture for wireless             %
%      microsensor networks"                                                  %
%     IEEE Transactions on Wireless Communications, 1(4):660-670,2002         %
%                         Mohammad Hossein Homaei                             %
%     Homaei@wsnlab.ir       %       Homaei@wsnlab.org                        %
%                www.wsnlab.ir  &   www.wsnlab.ir                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                             %
%                             LEACH Protocol                                  %
%                                                                             %
%                       By: Mohammad Hossein Homaei                           %
%                             homaei@wsnlab.ir                                %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear;
%%%%%%%%%%%%%%%%%%%%%%%%%%%% PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Field Dimensions - x and y maximum (in meters)
xm=100;
ym=100;
%x and y Coordinates of the Sink
sink.x=0.5*xm;
sink.y=0.5*ym;

%Number of Nodes in the field
n=100

%Optimal Election Probability of a node to become cluster head
p=0.4;
%Energy Model (all values in Joules)
%Initial Energy
Eo=0.5;
%Eelec=Etx=Erx
ETX=50*0.000000001;
ERX=50*0.000000001;
%Transmit Amplifier types
Efs=10*0.000000000001;
Emp=0.0013*0.000000000001;
%Data Aggregation Energy
```

```
EDA=5*0.000000001;
%Values for Hetereogeneity
%Percentage of nodes than are advanced
m=0.1;
%\alpha
a=1;
%maximum number of round
rmax=200
%%%%%%%%%%%%%%%%%%%%%%%% END OF PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%%
%Computation of do
do=sqrt(Efs/Emp);
%Creation of the random Sensor Network
figure(1);
for i=1:1:n
    S(i).xd=rand(1,1)*xm;
    XR(i)=S(i).xd;
    S(i).yd=rand(1,1)*ym;
    YR(i)=S(i).yd;
    S(i).G=0;
    %initially there are no cluster heads only nodes
    S(i).type='N';

    temp_rnd0=i;
    %Random Election of Normal Nodes
    if (temp_rnd0>=m*n+1)
        S(i).E=Eo;
        S(i).ENERGY=0;
        plot(S(i).xd,S(i).yd,'o');
        hold on;
    end
    %Random Election of Advanced Nodes
    if (temp_rnd0<m*n+1)
        S(i).E=Eo*(1+a)
        S(i).ENERGY=1;
        plot(S(i).xd,S(i).yd,'+');
        hold on;
    end
end
S(n+1).xd=sink.x;
S(n+1).yd=sink.y;
plot(S(n+1).xd,S(n+1).yd,'x');

TotalEnergyRemaining = zeros(rmax+1);
%First Iteration
figure(1);

%counter for CHs
countCHs=0;
%counter for CHs per round
rcountCHs=0;
```

```
cluster=1;


sum1=0;
iterator = 0;

countCHs;
rcountCHs=rcountCHs+countCHs;
flag_first_dead=0;
r=0;

for r=0:1:rmax
    r

    %Operation for epoch
    if(mod(r, round(1/p) )==0)
        for i=1:1:n
            S(i).G=0;
            S(i).cl=0;
        end
    end

    hold off;

    %Number of dead nodes
    dead=0;
    %Number of dead Advanced Nodes
    dead_a=0;
    %Number of dead Normal Nodes
    dead_n=0;

    %counter for bit transmitted to Bases Station and to Cluster Heads
    packets_TO_BS=0;
    packets_TO_CH=0;
    %counter for bit transmitted to Bases Station and to Cluster Heads
    %per round
    PACKETS_TO_CH(r+1)=0;
    PACKETS_TO_BS(r+1)=0;

    figure(1);

    for i=1:1:n
        %checking if there is a dead node
        if (S(i).E<=0)
            plot(S(i).xd,S(i).yd,'red .');
            dead=dead+1;
            if(S(i).ENERGY==1)
                dead_a=dead_a+1;
            end
            if(S(i).ENERGY==0)
```

```
                dead_n=dead_n+1;
            end
            hold on;
        end
        if S(i).E>0
            S(i).type='N';
            if (S(i).ENERGY==0)
                plot(S(i).xd,S(i).yd,'o');
            end
            if (S(i).ENERGY==1)
                 plot(S(i).xd,S(i).yd,'+');
            end
            hold on;
        end
    end
%    plot(S(n+1).xd,S(n+1).yd,'x');


    STATISTICS(r+1).DEAD=dead;
    DEAD(r+1)=dead;
    DEAD_N(r+1)=dead_n;
    DEAD_A(r+1)=dead_a;

    %When the first node dies
    if (dead==1)
        if(flag_first_dead==0)
            first_dead=r
            flag_first_dead=1;
        end
    end

    countCHs=0;
    cluster=1;
    for i=1:1:n
        if(S(i).E>0)
            temp_rand=rand;
            if ( (S(i).G)<=0)

                %Election of Cluster Heads
                if(temp_rand<= (p/(1-p*mod(r,round(1/p)))))
                    countCHs=countCHs+1;
                    packets_TO_BS=packets_TO_BS+1;
                    PACKETS_TO_BS(r+1)=packets_TO_BS;

                    S(i).type='C';
                    S(i).G=round(1/p)-1;
                    C(cluster).xd=S(i).xd;
                    C(cluster).yd=S(i).yd;
                    plot(S(i).xd,S(i).yd,'k*');
```

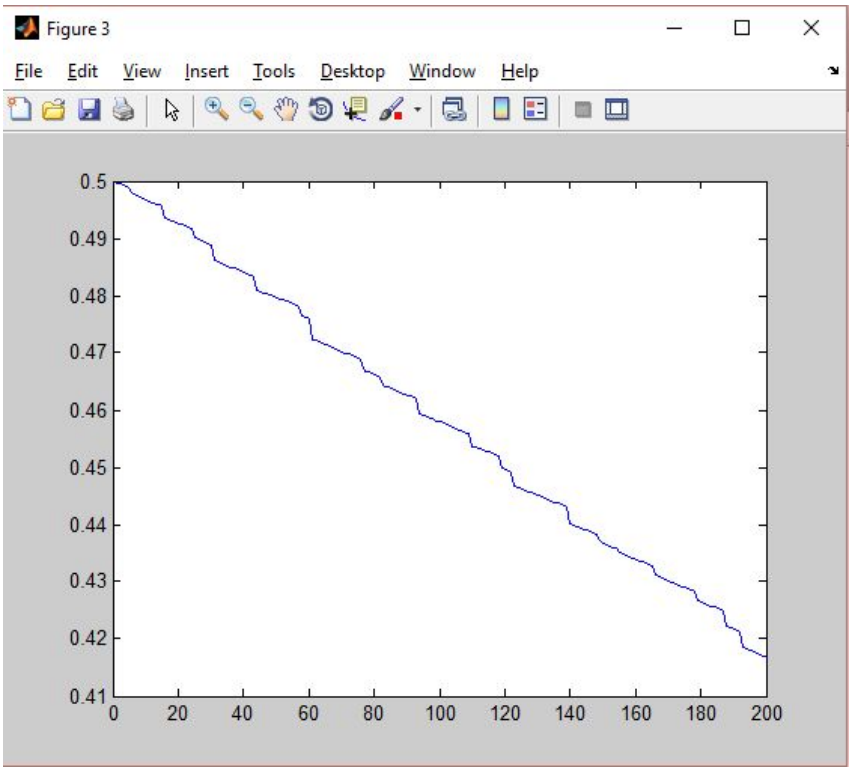
```
                    distance=sqrt( (S(i).xd-(S(n+1).xd) )^2 + (S(i).yd-(S(n+1).yd) )^2 );
                    C(cluster).distance=distance;
                    C(cluster).id=i;
                    X(cluster)=S(i).xd;
                    Y(cluster)=S(i).yd;
                    cluster=cluster+1;

                    %Calculation of Energy dissipated
                    distance;
                    if (distance>do)
                        S(i).E=S(i).E- ( (ETX+EDA)*(4000) + Emp*4000*(
distance*distance*distance*distance ));
                    end
                    if (distance<=do)
                        S(i).E=S(i).E- ( (ETX+EDA)*(4000)  + Efs*4000*( distance * distance
));
                    end
                end
            end
        end
    end

    STATISTICS(r+1).CLUSTERHEADS=cluster-1;
    CLUSTERHS(r+1)=cluster-1;

    %Election of Associated Cluster Head for Normal Nodes
    for i=1:1:n
        if ( S(i).type=='N' && S(i).E>0 )
            if(cluster-1>=1)
                min_dis=sqrt( (S(i).xd-S(n+1).xd)^2 + (S(i).yd-S(n+1).yd)^2 );
                min_dis_cluster=1;
                for c=1:1:cluster-1
                    temp=min(min_dis,sqrt( (S(i).xd-C(c).xd)^2 + (S(i).yd-C(c).yd)^2 ) );
                    if ( temp<min_dis )
                        min_dis=temp;
                        min_dis_cluster=c;
                    end
                end

                %Energy dissipated by associated Cluster Head
                min_dis;
                if (min_dis>do)
                    S(i).E=S(i).E- ( ETX*(4000) + Emp*4000*( min_dis * min_dis * min_dis *
min_dis));
                end
                if (min_dis<=do)
                    S(i).E=S(i).E- ( ETX*(4000) + Efs*4000*( min_dis * min_dis));
                end
                %Energy dissipated
                if(min_dis>0)
```

```
                    S(C(min_dis_cluster).id).E = S(C(min_dis_cluster).id).E- ( (ERX +
EDA)*4000 );
                    PACKETS_TO_CH(r+1)=n-dead-cluster+1;
                end

                S(i).min_dis=min_dis;
                S(i).min_dis_cluster=min_dis_cluster;
            end
        end
    end
    hold on;
    countCHs;
    rcountCHs=rcountCHs+countCHs;
    if(r == 1)
        TotalEnergyConsumed = TotalEnergyRemaining(1);
    end
    TotalEnergyRemaining(r+1) = findEnergy(TotalEnergyRemaining,S,n,r);

end


figure;
plot(1:rmax,TotalEnergyConsumed - TotalEnergyRemaining(1:rmax));
figure;
plot(1:rmax,TotalEnergyRemaining(1:rmax));
figure;
plot(1:rmax,DEAD_N(1:rmax));
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%    STATISTICS    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                                                                          %
%  DEAD  : a rmax x 1 array of number of dead nodes/round
         %
%  DEAD_A : a rmax x 1 array of number of dead Advanced nodes/round
         %
%  DEAD_N : a rmax x 1 array of number of dead Normal nodes/round                          %
%  CLUSTERHS : a rmax x 1 array of number of Cluster Heads/round                           %
%  PACKETS_TO_BS : a rmax x 1 array of number packets send to Base Station/round           %
%  PACKETS_TO_CH : a rmax x 1 array of number of packets send to ClusterHeads/round        %
%  first_dead: the round where the first node died                                         %
%                                                                                          %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Function:
function a = findEnergy(TotE,S,n,r)
%This sums up the total energy of the elements
%in the network
    a=0;
    for ct=1:1:n
        a = TotE(r+1) + S(ct).E;
    end
end
```

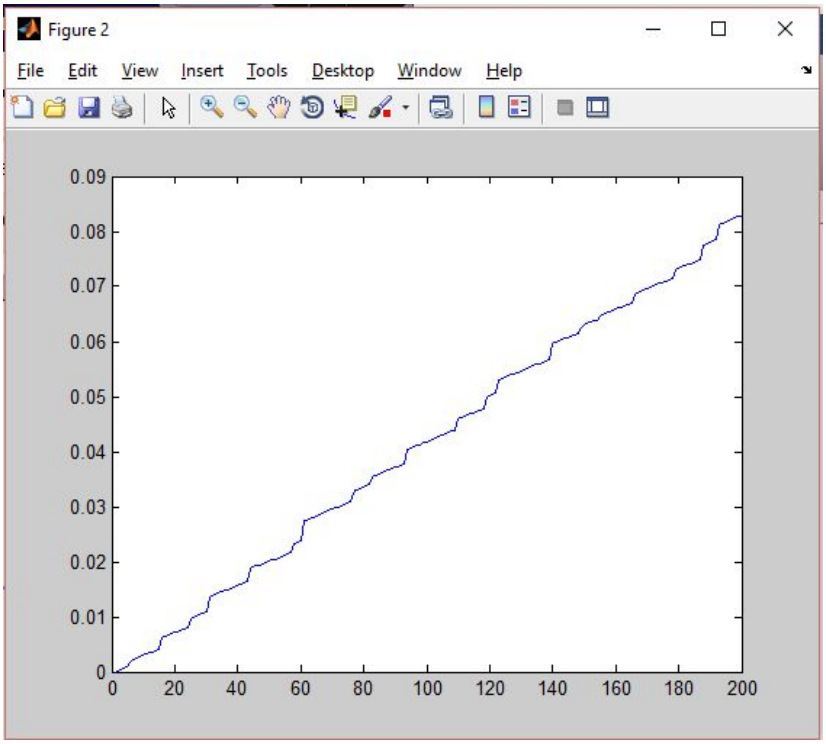
Screenshots:

1) p=0.1

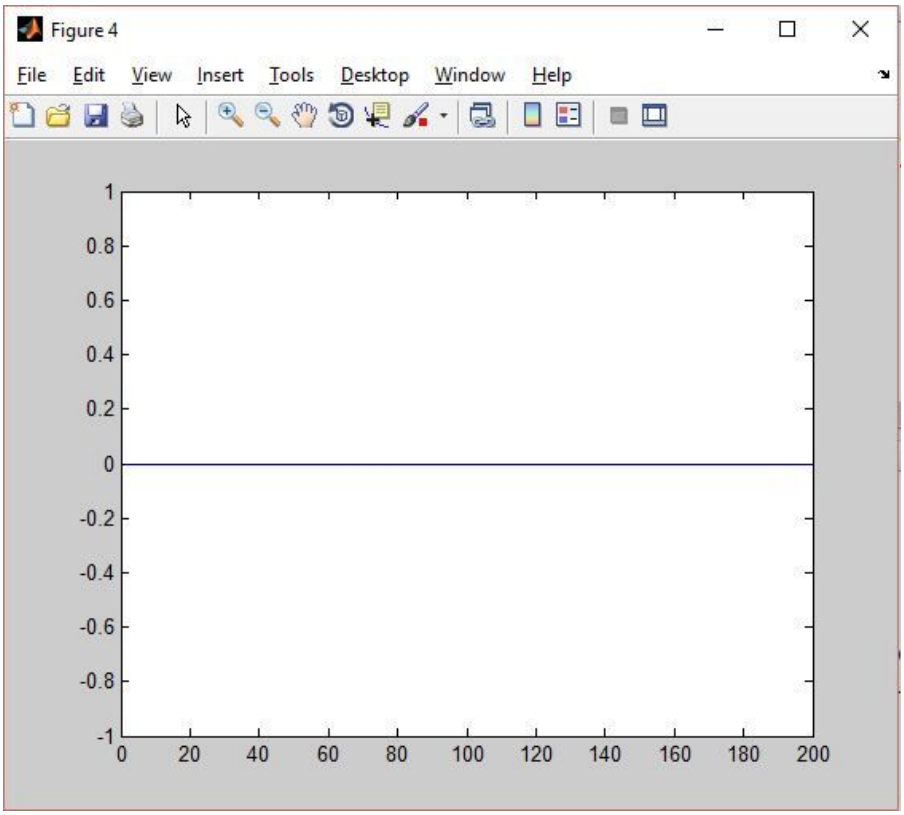
a) Energy Consumed vs Round number



b) Energy Remaining vs Round number

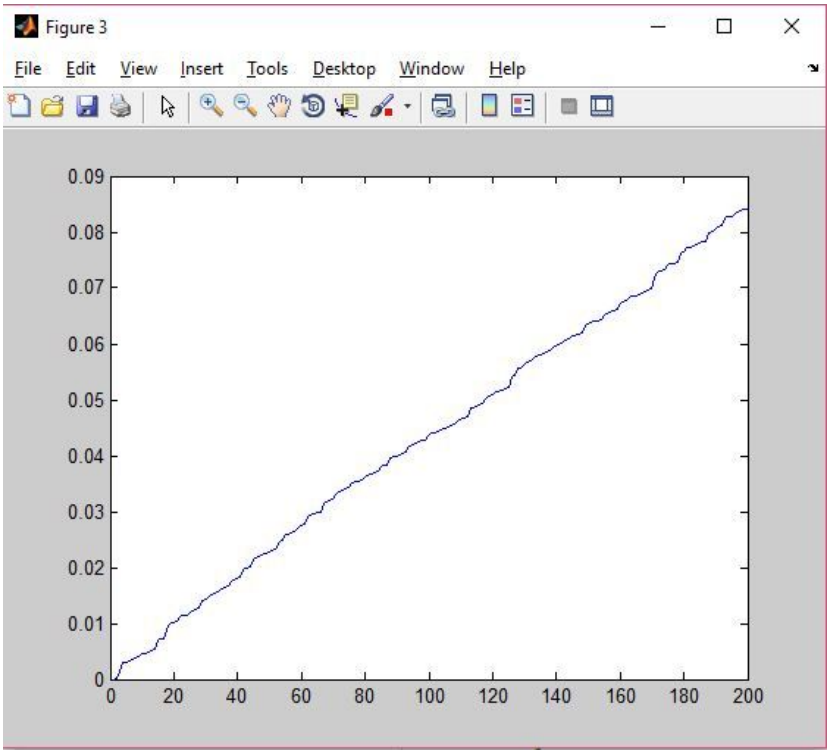


c) Number of Dead Nodes vs Round number

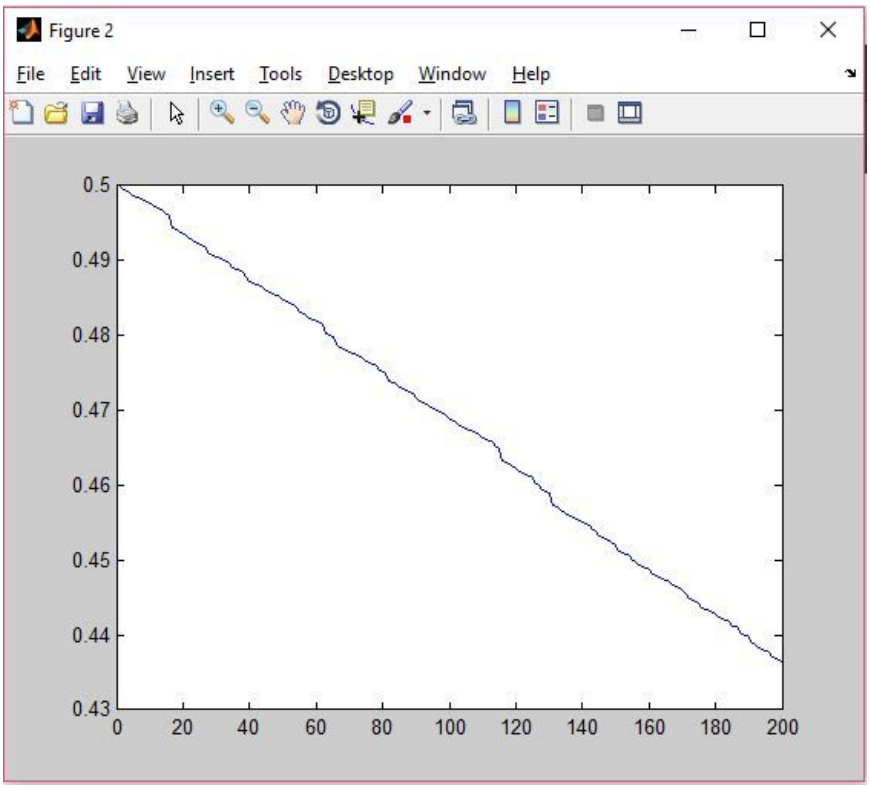


2) p=0.4

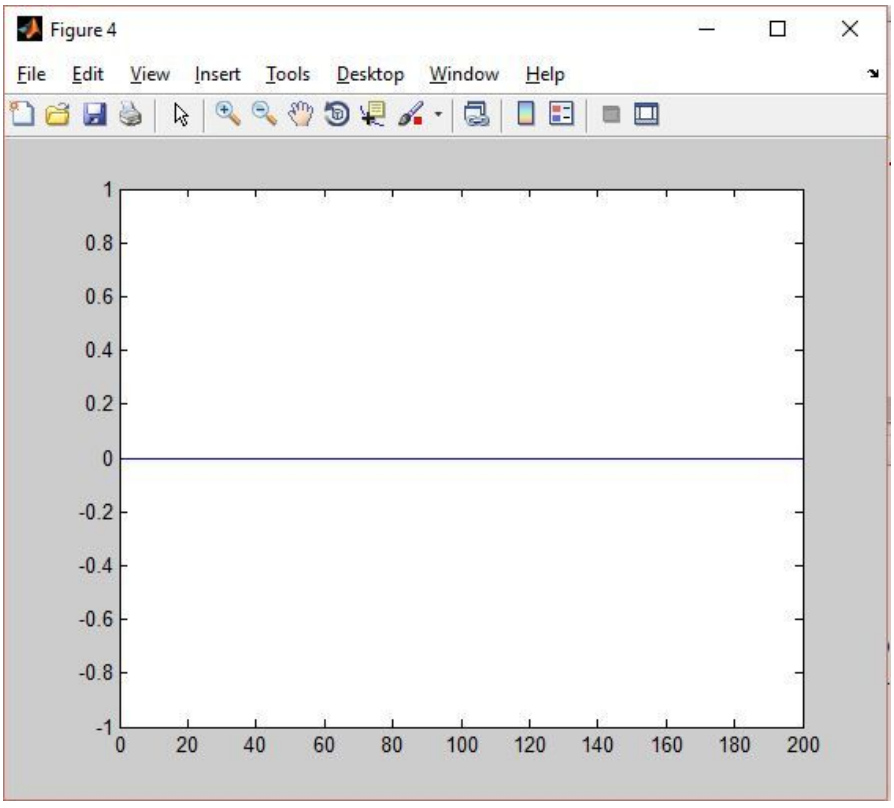
a) Energy Consumed vs Round number



b) Energy Remaining vs Round number



c) Number of Dead Nodes vs Round number



When p is changed from 0.1 to 0.4 the energy consumed increases.