%%%%%%%%%%%%%%%%%%%%%%%%%%%% SEP %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc

close all

clear all;

xaxis=100; %dimensions of sensor network

yaxis=100;

base\_station.x=0.5\*xaxis; %distance of base station from the network

base\_station.y=0.5\*yaxis;

n = 800; %no of nodes

p= 0.1; %probability of a node to become cluster head

initial\_energy=0.5; %energy supplied to each node

Energy\_Transmitter=50\*0.000000001; %energy dissipated per bit at transmitter per node

Energy\_Receiver=50\*0.000000001; %energy dissipated per bit at receiver per node

Energy\_Frequency=10\*0.000000000001; %amplification coefficient of free-space signal when d is less than d0

Emplifier=0.0013\*0.000000000001; % multi-path fading signal amplification coefficient when d is greater than d0

Residual\_e1(1)=0;

Residual\_e2(1)=0;

Residual\_e3(1)=0;

Residual\_e4(1)=0;

Residual\_e5(1)=0;

Data\_Aggregation\_Energy=5\*0.000000001; % data aggregation energy per node

maximum\_lifetime=9999; %no of rounds

do=sqrt(Energy\_Frequency/Emplifier); %distance between cluster head and base station

%Values for Heterogeneity

m=0.2; %Percentage of nodes that are advance nodes

b=0.3; %Percentage of nodes that are intermediate nodes

a=3; %energy factor for advance nodes

u=a/2; % energy factor for intermediate nodes

h=100; %value for hard threshold

tEmpi=50;

tEmpf=200;

Residual\_e3(1)=0;

INFINITY = 999999999999999;

for i=1:1:n

S(i).xd=rand(1,1)\*xaxis; %it will distribute the nodes in 1 dimension in x axis randomly.

S(i).yd=rand(1,1)\*yaxis; %it will distribute the nodes in 1 dimension in y axis randoml.

S(i).G=0; % initially there are no cluster heads

S(i).E=initial\_energy;

S(i).type='N'; %initially there are no cluster heads only nodes

end

S(n+1).xd=base\_station.x; %total no of nodes is n and with base station it is n+1

S(n+1).yd=base\_station.y;

countCHs=0;

cluster=1; %first cluster is selected

flag\_first\_dead=0;

flag\_teenth\_dead=0;

flag\_all\_dead=0;

dead=0; % initially no node is dead

first\_dead=0;

teenth\_dead=0;

all\_dead=0;

allive=n; %initially all nodes are alive

%counter for bit transmitted to Bases Station and to Cluster Heads

packets\_TO\_BS=0;

packets\_TO\_CH=0;

% counter for sleep nodes

s=0;

%Creation of the random Sensor Network

for i=1:1:n

Se(i).xd=rand(1,1)\*xaxis;

XR(i)=S(i).xd;

Se(i).yd=rand(1,1)\*yaxis;

YR(i)=S(i).yd;

Se(i).G=0;

%initially there are no cluster heads only nodes

Se(i).type='N';

temp\_rnd0 = i;

%Random Election of Normal Nodes

if (temp\_rnd0>=m\*n+1)

Se(i).E=initial\_energy;

Se(i).ENERGY=0;

%%%plot(St(i).xd,St(i).yd,'o');

hold on;

end

%Random Election of advance Nodes

if (temp\_rnd0<m\*n+1)

Se(i).E=initial\_energy\*(1+a);

Se(i).ENERGY=1;

%%%%plot(St(i).xd,St(i).yd,'+');

hold on;

end

end

Se(n+1).xd=base\_station.x; %total no of nodes is n and with base station it is n+1

Se(n+1).yd=base\_station.y;

%plot(S2(n+1).xd,S2(n+1).yd,'x');

%counter for CHs

countCHs2=0;

%counter for CHs per round

rcountCHs2=0;

cluster2=1;

countCHs;

rcountCHs2=rcountCHs2+countCHs2;

flag\_first\_dead2=0;

alive=n;

for r=0:1:maximum\_lifetime

r

warning('off','all');

%Election Probability for Normal Nodes

pnrm=( p/ (1+a\*m) );

%Election Probability for Advanced Nodes

padv= ( p\*(1+a)/(1+a\*m) )

%Operation for epoch

if(mod(r, round(1/pnrm) )==0)

for i=1:1:n

if(Se(i).ENERGY==0)

Se(i).G=0;

Se(i).cl=0;

end

end

end

%Operation for epoch

if(mod(r, round(1/padv) )==0)

for i=1:1:n

if(Se(i).ENERGY==1)

Se(i).G=0;

Se(i).cl=0;

end

end

end

%Number of dead nodes

dead2=0;

%Number of dead Advanced Nodes

dead\_a2=0;

%Number of dead Normal Nodes

dead\_n2=0;

%counter for bit transmitted to Bases Station and to Cluster Heads

packets\_TO\_BS2=0;

packets\_TO\_CH2=0;

%counter for bit transmitted to Bases Station and to Cluster Heads

%per round

PACKETS\_TO\_CH2(r+1)=0;

PACKETS\_TO\_BS2(r+1)=0;

for i=1:1:n

%checking if there is a dead node

if (Se(i).E<=0)

%plot(Se(i).xd,Se(i).yd,'red .');

dead2=dead2+1;

if(Se(i).ENERGY==1)

dead\_a2=dead\_a2+1;

end

if(Se(i).ENERGY==0)

dead\_n2=dead\_n2+1;

% plot(Se(i).xd,Se(i).yd,'red .');

end

end

if Se(i).E>0

Se(i).type='N';

if (Se(i).ENERGY==0)

%plot(Se(i).xd,Se(i).yd,'o');

end

if (Se(i).ENERGY==1)

%plot(Se(i).xd,Se(i).yd,'+');

end

end

end

%plot(Se(n+1).xd,Se(n+1).yd,'x');

STATISTICS.DEAD2(r+1)=dead2;

STATISTICS.ALIVE2(r+1)=alive-dead2;

DEAD2(r+1)=dead2;

DEAD\_N2(r+1)=dead\_n2;

DEAD\_A2(r+1)=dead\_a2;

if dead2==n

break;

end

%When the first node dies

if (dead2==1)

if(flag\_first\_dead2==0)

first\_dead2=r

flag\_first\_dead2=1;

end

end

countCHs2=0;

cluster2=1;

for i=1:1:n

if(Se(i).E>0)

temp\_rand=rand;

if ( (Se(i).G)<=0)

%Election of Cluster Heads

if( ( Se(i).ENERGY==0 && ( temp\_rand <= ( pnrm / ( 1 - pnrm \* mod(r,round(1/pnrm)) )) ) ) ) % for normal nodes

countCHs2=countCHs2+1;

packets\_TO\_BS2=packets\_TO\_BS2+1;

PACKETS\_TO\_BS2(r+1)=packets\_TO\_BS2;

Se(i).type = 'C';

Se(i).G = round(1/p)-1;

C(cluster2).xd = Se(i).xd;

C(cluster2).yd = Se(i).yd;

%plot(Se(i).xd,Se(i).yd,'k\*');

distance=sqrt( (Se(i).xd-(Se(n+1).xd) )^2 + (Se(i).yd-(Se(n+1).yd) )^2 );

C(cluster2).distance = distance;

C(cluster2).id = i;

X(cluster2)=Se(i).xd;

Y(cluster2)=Se(i).yd;

cluster2=cluster2+1;

distanceBroad = sqrt(xaxis\*xaxis+yaxis\*yaxis);

if (distanceBroad > do)

Se(i).E = Se(i).E- ( (Energy\_Transmitter)\*(100) + Emplifier\* 100\*( distanceBroad\*distanceBroad\*distanceBroad\*distanceBroad ));

else

Se(i).E = Se(i).E- ( (Energy\_Transmitter)\*(100) + Energy\_Frequency \* 100\*( distanceBroad\*distanceBroad));

end

%Calculation of Energy dissipated

distance;

if (distance > do)

Se(i).E = Se(i).E- ( (Energy\_Transmitter+Data\_Aggregation\_Energy)\*(4000) + Emplifier \* 4000\*( distance\*distance\*distance\*distance ));

else

Se(i).E = Se(i).E- ( (Energy\_Transmitter+Data\_Aggregation\_Energy)\*(4000) + Energy\_Frequency \* 4000\*( distance \* distance ));

end

packets\_TO\_BS2 = packets\_TO\_BS2+1;

PACKETS\_TO\_BS2(r+1) = packets\_TO\_BS2;

end

if( ( Se(i).ENERGY==1 && ( temp\_rand <= ( padv / ( 1 - padv \* mod(r,round(1/padv)) )) ) ) ) % for advance nodes

countCHs2=countCHs2+1;

packets\_TO\_BS2=packets\_TO\_BS2+1;

PACKETS\_TO\_BS2(r+1)=packets\_TO\_BS2;

Se(i).type = 'C';

Se(i).G = round(1/p)-1;

C(cluster2).xd = Se(i).xd;

C(cluster2).yd = Se(i).yd;

%plot(Se(i).xd,Se(i).yd,'k\*');

distance=sqrt( (Se(i).xd-(Se(n+1).xd) )^2 + (Se(i).yd-(Se(n+1).yd) )^2 );

C(cluster2).distance = distance;

C(cluster2).id = i;

X(cluster2)=Se(i).xd;

Y(cluster2)=Se(i).yd;

cluster2=cluster2+1;

distanceBroad = sqrt(xaxis\*xaxis+yaxis\*yaxis);

if (distanceBroad > do)

Se(i).E = Se(i).E- ( (Energy\_Transmitter)\*(100) + Emplifier\* 100\*( distanceBroad\*distanceBroad\*distanceBroad\*distanceBroad ));

else

Se(i).E = Se(i).E- ( (Energy\_Transmitter)\*(100) + Energy\_Frequency \* 100\*( distanceBroad\*distanceBroad));

end

%Calculation of Energy dissipated

distance;

if (distance > do)

Se(i).E = Se(i).E- ( (Energy\_Transmitter+Data\_Aggregation\_Energy)\*(4000) + Emplifier \* 4000\*( distance\*distance\*distance\*distance ));

else

Se(i).E = Se(i).E- ( (Energy\_Transmitter+Data\_Aggregation\_Energy)\*(4000) + Energy\_Frequency \* 4000\*( distance \* distance ));

end

packets\_TO\_BS2 = packets\_TO\_BS2+1;

PACKETS\_TO\_BS2(r+1) = packets\_TO\_BS2;

end

end

end

end

STATISTICS.COUNTCHS2(r+1)=countCHs2;

STATISTICS.CLUSTERHEADS2(r+1) = cluster2-1;

CLUSTERHS2(r+1)= cluster2-1;

for i=1:1:n

if ( Se(i).type=='N' && Se(i).E>0 )

if(cluster2-1>=1)

min\_dis=sqrt( (Se(i).xd-Se(n+1).xd)^2 + (Se(i).yd-Se(n+1).yd)^2 );

min\_dis\_cluster2=1;

for c=1:1:cluster2-1

temp=min(min\_dis,sqrt( (Se(i).xd-C(c).xd)^2 + (Se(i).yd-C(c).yd)^2 ) );

if ( temp<min\_dis )

min\_dis=temp;

min\_dis\_cluster2=c;

end

end

%Energy dissipated by associated Cluster Head

min\_dis;

if (min\_dis>do)

Se(i).E=Se(i).E- ( Energy\_Transmitter\*(4000) + Emplifier\*4000\*( min\_dis \* min\_dis \* min\_dis \* min\_dis));

end

if (min\_dis<=do)

Se(i).E=Se(i).E- ( Energy\_Transmitter\*(4000) + Energy\_Frequency\*4000\*( min\_dis \* min\_dis));

end

%Energy dissipated

if(min\_dis>0)

Se(C(min\_dis\_cluster2).id).E = Se(C(min\_dis\_cluster2).id).E- ( (Energy\_Receiver + Data\_Aggregation\_Energy)\*4000 );

PACKETS\_TO\_CH2(r+1)=n-dead2-cluster2+1;

end

Se(i).min\_dis=min\_dis;

Se(i).min\_dis\_cluster2=min\_dis\_cluster2;

end

end

end

eee=0;

for i=1:n

eee=eee+Se(i).E;

end

STATISTICS.PACKETS\_TO\_CH2(r+1)=packets\_TO\_CH2;

STATISTICS.PACKETS\_TO\_BS2(r+1)=packets\_TO\_BS2;

Residual\_e3(r+1)=eee; % residual energy of nodes

end

x=1:1:r;

y=1:1:r;

w=1:1:r;

v=1:1:r;

for i=1:r;

x(i)=i;

y(i) = n- DEAD2(i); % remaining number of live nodes

w(i)= Residual\_e3(i);

v(i)= STATISTICS.PACKETS\_TO\_BS2(i);

end

figure(1)

plot(x,y,'r');

hold on;

figure(2)

plot(x,w,'r');

hold on;

figure(3)

plot(x,v,'r');

hold on;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% TSEP %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc

close all

clear all;

xaxis=300; %dimensionss of sensor network

yaxis=300;

base\_station.x=0.5\*xaxis; %distance of base station from the network

base\_station.y=0.5\*yaxis;

n = 800; %no of nodes

p= 0.1; %probibilty of a node to become cluster head

initial\_energy=0.5; %energy supplied to each node

Energy\_Transmitter=50\*0.000000001; %energy dissipated per bit at transmitter per node

Energy\_Receiver=50\*0.000000001; %energy dissipated per bit at reciever per node

Energy\_Frequency=10\*0.000000000001; %amplification coefficient of free-space signal when d is less than d0

Emplifier=0.0013\*0.000000000001; % multi-path fading signal amplification coefficient when d is greater than d0

Residual\_e1(1)=0;

Residual\_e2(1)=0;

Residual\_e3(1)=0;

Residual\_e4(1)=0;

Residual\_e5(1)=0;

Data\_Aggregation\_Energy=5\*0.000000001; % data aggregation energy per node

maximum\_lifetime=9999; %no of rounds

do=sqrt(Energy\_Frequency/Emplifier); %distance between cluster head and base station

%Values for Hetereogeneity

m=0.2; %Percentage of nodes that are advance nodes

b=0.3; %Percentage of nodes that are intermediate nodes

a=3; %energy factor for advance nodes

u=a/2; % energy factor for intermediate nodes

h=100; %value for hard threshold

tEmpi=50;

tEmpf=200;

Residual\_e3(1)=0;

INFINITY = 999999999999999;

for i=1:1:n

S(i).xd=rand(1,1)\*xaxis; %it will distribute the nodes in 1 dimension in x axis randomly.

S(i).yd=rand(1,1)\*yaxis; %it will distribute the nodes in 1 dimension in y axis randoml.

S(i).G=0; % initially there are no cluster heads

S(i).E=initial\_energy;

S(i).type='N'; %initially there are no cluster heads only nodes

end

S(n+1).xd=base\_station.x; %total no of nodes is n and with base station it is n+1

S(n+1).yd=base\_station.y;

countCHs=0;

cluster=1; %first cluster is selected

flag\_first\_dead=0;

flag\_teenth\_dead=0;

flag\_all\_dead=0;

dead=0; % initially no node is dead

first\_dead=0;

teenth\_dead=0;

all\_dead=0;

allive=n; %initially all nodes are alive

%counter for bit transmitted to Bases Station and to Cluster Heads

packets\_TO\_BS=0;

packets\_TO\_CH=0;

% counter for sleep nodes

s=0;

%Creation of the random Sensor Network

for i=1:1:n

St(i).xd=rand(1,1)\*xaxis;

XR(i)=S(i).xd;

St(i).yd=rand(1,1)\*yaxis;

YR(i)=S(i).yd;

St(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>=(b+m)\*n+1)

St(i).E=initial\_energy;

St(i).ENERGY=0;

%%%plot(St(i).xd,St(i).yd,'o');

hold on;

end

%Random Election of intermediate Nodes

if (temp\_rnd0<(b+m)\*n+1)&& (temp\_rnd0>=m\*n+1)

St(i).E=initial\_energy\*(1+u);

St(i).ENERGY=0.5;

%%%%plot(St(i).xd,St(i).yd,'+');

hold on;

end

%Random Election of Advanced Nodes

if (temp\_rnd0<m\*n+1)

St(i).E=initial\_energy\*(1+a);

St(i).ENERGY=1;

hold on;

end

end

St(n+1).xd=base\_station.x;

St(n+1).yd=base\_station.y;

%plot(St(n+1).xd,St(n+1).yd,'x');

%counter for CHs

countCHs2=0;

%counter for CHs per round

rcountCHs2=0;

cluster2=1;

countCHs;

rcountCHs2=rcountCHs2+countCHs2;

flag\_first\_dead2=0;

alive=n;

for r=0:1:maximum\_lifetime

r

warning('off','all');

cv = tEmpi + (tEmpf-tEmpi).\*rand(1,1);

%Election Probability for Normal Nodes

pnrm=( p/ (1+a\*m+u\*b) );

%Election Probability for intermediate Nodes

pint=( p\*(1+u)/ (1+a\*m+u\*b) );

%Election Probability for Advanced Nodes

padv= ( p\*(1+a)/(1+a\*m+u\*b) );

%Operation for epoch

if(mod(r, round(1/pnrm) )==0)

for i=1:1:n

if(St(i).ENERGY==0)

St(i).G=0;

St(i).cl=0;

end

end

end

if(mod(r, round(1/pint) )==0)

for i=1:1:n

if(St(i).ENERGY==0.5)

St(i).G=0;

St(i).cl=0;

end

end

end

if(mod(r, round(1/padv) )==0)

for i=1:1:n

if(St(i).ENERGY==1)

St(i).G=0;

St(i).cl=0;

end

end

end

%Number of dead nodes

dead2=0;

%Number of dead Advanced Nodes

dead\_a2=0;

%number of intermediate nodes

dead\_i2=0;

%Number of dead Normal Nodes

dead\_n2=0;

%counter for bit transmitted to Bases Station and to Cluster Heads

packets\_TO\_BS2=0;

packets\_TO\_CH2=0;

%counter for bit transmitted to Bases Station and to Cluster Heads

%per round

PACKETS\_TO\_CH2(r+1)=0;

PACKETS\_TO\_BS2(r+1)=0;

for i=1:1:n

%checking if there is a dead node

if (St(i).E<=0)

%plot(St(i).xd,St(i).yd,'red .');

dead2=dead2+1;

if(St(i).ENERGY==1)

dead\_a2=dead\_a2+1;

end

if(St(i).ENERGY==.5)

dead\_i2=dead\_i2+1;

end

if(St(i).ENERGY==0)

dead\_n2=dead\_n2+1;

% plot(S(i).xd,S(i).yd,'red .');

end

end

if St(i).E>0

St(i).type='N';

if (St(i).ENERGY==0)

%plot(S(i).xd,S(i).yd,'o');

end

if (St(i).ENERGY==0.5)

%plot(St(i).xd,St(i).yd,'^');

end

if (St(i).ENERGY==1)

%plot(S(i).xd,S(i).yd,'+');

end

end

end

%plot(S(n+1).xd,S(n+1).yd,'x');

STATISTICS.DEAD2(r+1)=dead2;

STATISTICS.ALIVE(r+1)=alive-dead2;

DEAD2(r+1)=dead2;

DEAD\_N2(r+1)=dead\_n2;

DEAD\_I2(r+1)=dead\_i2;

DEAD\_A2(r+1)=dead\_a2;

if dead2==n

break;

end

%When the first node dies

if (dead2==1)

if(flag\_first\_dead2==0)

first\_dead2=r

flag\_first\_dead2=1;

end

end

countCHs2=0;

cluster2=1;

for i=1:1:n

if(St(i).E>0)

temp\_rand=rand;

if ( (St(i).G)<=0)

%Election of Cluster Heads

if( ( St(i).ENERGY==0 && ( temp\_rand <= ( pnrm / ( 1 - pnrm \* mod(r,round(1/pnrm)) )) ) ) ) % for normal nodes

countCHs2=countCHs2+1;

packets\_TO\_BS2=packets\_TO\_BS2+1;

PACKETS\_TO\_BS2(r+1)=packets\_TO\_BS2;

St(i).type = 'C';

St(i).G = round(1/p)-1;

C(cluster2).xd = St(i).xd;

C(cluster2).yd = St(i).yd;

%plot(St(i).xd,St(i).yd,'k\*');

distance=sqrt( (St(i).xd-(St(n+1).xd) )^2 + (St(i).yd-(St(n+1).yd) )^2 );

C(cluster2).distance = distance;

C(cluster2).id = i;

X(cluster2)=St(i).xd;

Y(cluster2)=St(i).yd;

cluster2=cluster2+1;

distanceBroad = sqrt(xaxis\*xaxis+yaxis\*yaxis);

if (distanceBroad > do)

St(i).E = St(i).E- ( (Energy\_Transmitter)\*(100) + Emplifier\* 100\*( distanceBroad\*distanceBroad\*distanceBroad\*distanceBroad ));

else

St(i).E = St(i).E- ( (Energy\_Transmitter)\*(100) + Energy\_Frequency \* 100\*( distanceBroad\*distanceBroad));

end

%Calculation of Energy dissipated

distance;

if(cv>=h)

if (distance > do)

St(i).E = St(i).E- ( (Energy\_Transmitter+Data\_Aggregation\_Energy)\*(4000) + Emplifier \* 4000\*( distance\*distance\*distance\*distance ));

else

St(i).E = St(i).E- ( (Energy\_Transmitter+Data\_Aggregation\_Energy)\*(4000) + Energy\_Frequency \* 4000\*( distance \* distance ));

end

end

packets\_TO\_BS2 = packets\_TO\_BS2+1;

PACKETS\_TO\_BS2(r+1) = packets\_TO\_BS2;

end

if( ( St(i).ENERGY==0.5 && ( temp\_rand <= ( pint / ( 1 - pint \* mod(r,round(1/pint)) )) ) ) ) % for intermediate nodes

countCHs2=countCHs2+1;

packets\_TO\_BS2=packets\_TO\_BS2+1;

PACKETS\_TO\_BS2(r+1)=packets\_TO\_BS2;

St(i).type = 'C';

St(i).G = round(1/p)-1;

C(cluster2).xd = St(i).xd;

C(cluster2).yd = St(i).yd;

%plot(St(i).xd,St(i).yd,'k\*');

distance=sqrt( (St(i).xd-(St(n+1).xd) )^2 + (St(i).yd-(St(n+1).yd) )^2 );

C(cluster2).distance = distance;

C(cluster2).id = i;

X(cluster2)=St(i).xd;

Y(cluster2)=St(i).yd;

cluster2=cluster2+1;

distanceBroad = sqrt(xaxis\*xaxis+yaxis\*yaxis);

if (distanceBroad > do)

St(i).E = St(i).E- ( (Energy\_Transmitter)\*(100) + Emplifier\* 100\*( distanceBroad\*distanceBroad\*distanceBroad\*distanceBroad ));

else

St(i).E = St(i).E- ( (Energy\_Transmitter)\*(100) + Energy\_Frequency \* 100\*( distanceBroad\*distanceBroad));

end

%Calculation of Energy dissipated

distance;

if(cv>=h) % if current sensed value is greater than hard threshold then only transmission will be done

if (distance > do)

St(i).E = St(i).E- ( (Energy\_Transmitter+Data\_Aggregation\_Energy)\*(4000) + Emplifier \* 4000\*( distance\*distance\*distance\*distance ));

else

St(i).E = St(i).E- ( (Energy\_Transmitter+Data\_Aggregation\_Energy)\*(4000) + Energy\_Frequency \* 4000\*( distance \* distance ));

end

end

packets\_TO\_BS2 = packets\_TO\_BS2+1;

PACKETS\_TO\_BS2(r+1) = packets\_TO\_BS2;

end

if( ( St(i).ENERGY==1 && ( temp\_rand <= ( padv / ( 1 - padv \* mod(r,round(1/padv)) )) ) ) ) % for advance nodes

countCHs2=countCHs2+1;

packets\_TO\_BS2=packets\_TO\_BS2+1;

PACKETS\_TO\_BS2(r+1)=packets\_TO\_BS2;

St(i).type = 'C';

St(i).G = round(1/p)-1;

C(cluster2).xd = St(i).xd;

C(cluster2).yd = St(i).yd;

%plot(St(i).xd,St(i).yd,'k\*');

distance=sqrt( (St(i).xd-(St(n+1).xd) )^2 + (St(i).yd-(St(n+1).yd) )^2 );

C(cluster2).distance = distance;

C(cluster2).id = i;

X(cluster2)=St(i).xd;

Y(cluster2)=St(i).yd;

cluster2=cluster2+1;

distanceBroad = sqrt(xaxis\*xaxis+yaxis\*yaxis);

if (distanceBroad > do)

St(i).E = St(i).E- ( (Energy\_Transmitter)\*(100) + Emplifier\* 100\*( distanceBroad\*distanceBroad\*distanceBroad\*distanceBroad ));

else

St(i).E = St(i).E- ( (Energy\_Transmitter)\*(100) + Energy\_Frequency \* 100\*( distanceBroad\*distanceBroad));

end

%Calculation of Energy dissipated

distance;

if(cv>=h)

if (distance > do)

St(i).E = St(i).E- ( (Energy\_Transmitter+Data\_Aggregation\_Energy)\*(4000) + Emplifier \* 4000\*( distance\*distance\*distance\*distance ));

else

St(i).E = St(i).E- ( (Energy\_Transmitter+Data\_Aggregation\_Energy)\*(4000) + Energy\_Frequency \* 4000\*( distance \* distance ));

end

end

packets\_TO\_BS2 = packets\_TO\_BS2+1;

PACKETS\_TO\_BS2(r+1) = packets\_TO\_BS2;

end

end

end

end

STATISTICS.COUNTCHS2(r+1)=countCHs2;

STATISTICS.CLUSTERHEADS2(r+1) = cluster2-1;

CLUSTERHS2(r+1)= cluster2-1;

for i=1:1:n

if ( St(i).type=='N' && St(i).E>0 )

if(cluster2-1>=1)

min\_dis=sqrt( (St(i).xd-St(n+1).xd)^2 + (St(i).yd-St(n+1).yd)^2 );

min\_dis\_cluster2=1;

for c=1:1:cluster2-1

temp=min(min\_dis,sqrt( (St(i).xd-C(c).xd)^2 + (St(i).yd-C(c).yd)^2 ) );

if ( temp<min\_dis )

min\_dis=temp;

min\_dis\_cluster2=c;

end

end

%Energy dissipated by associated Cluster Head

if(cv>=h)

min\_dis;

if (min\_dis>do)

St(i).E=St(i).E- ( Energy\_Transmitter\*(4000) + Emplifier\*4000\*( min\_dis \* min\_dis \* min\_dis \* min\_dis));

end

if (min\_dis<=do)

St(i).E=St(i).E- ( Energy\_Transmitter\*(4000) + Energy\_Frequency\*4000\*( min\_dis \* min\_dis));

end

%Energy dissipated

if(min\_dis>0)

St(C(min\_dis\_cluster2).id).E = St(C(min\_dis\_cluster2).id).E- ( (Energy\_Receiver + Data\_Aggregation\_Energy)\*4000 );

PACKETS\_TO\_CH2(r+1)=n-dead2-cluster2+1;

end

end

St(i).min\_dis=min\_dis;

St(i).min\_dis\_cluster2=min\_dis\_cluster2;

end

end

end

eee=0;

for i=1:n

eee=eee+St(i).E;

end

STATISTICS.PACKETS\_TO\_CH2(r+1)=packets\_TO\_CH2;

STATISTICS.PACKETS\_TO\_BS2(r+1)=packets\_TO\_BS2;

Residual\_e6(r+1)=eee; % residual energy of nodes

end

x=1:1:r;

y=1:1:r;

w=1:1:r;

v=1:1:r

for i=1:r;

x(i)=i;

y(i) = n- DEAD2(i); % remaining number of live nodes

w(i)= Residual\_e6(i);

v(i)= STATISTICS.PACKETS\_TO\_BS2(i);

end

figure(1)

plot(x,y,'r');

hold on;

figure(2)

plot(x,w,'r');

hold on;

figure(3)

plot(x,v,'r');

hold on;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% DEEC %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all

xm=300;

ym=300;

sink.x=0.5\*xm; %location of sink on x-axis

sink.y=0.5\*ym; %location of sink on y-axis

n=800 %nodes

P=0.1; %probability of cluster heads

Eo=0.5; %initial energy

ETX=50\*0.000000001; %tx energy

ERX=50\*0.000000001; %rx energy

Efs=10\*0.000000000001; %free space loss

Emp=0.0013\*0.000000000001; %multi path loss

%Data Aggregation Energy

EDA=5\*0.000000001; %compression energy

a=1; %fraction of energy enhancment of advance nodes

rmax=7000 %maximum number of rounds

do=sqrt(Efs/Emp); %distance do is measured

Et=0; %variable just use below

A=0;

for i=1:1:n

S(i).xd=rand(1,1)\*xm; %generates a random no. use to randomly distibutes nodes on x axis

XR(i)=S(i).xd;

S(i).yd=rand(1,1)\*ym; %generates a random no. use to randomly distibutes nodes on y axis

YR(i)=S(i).yd;

S(i).G=0; %node is elegible to become cluster head

talha=rand\*a;

S(i).E=Eo\*(1+talha);

E(i)= S(i).E;

A=A+talha;

Et=Et+E(i); %estimating total energy of the network

%initially there are no cluster heads only nodes

S(i).type='N';

end

d1=0.765\*xm/2; %distance between cluster head and base station

K=sqrt(0.5\*n\*do/pi)\*xm/d1^2; %optimal no. of cluster heads

d2=xm/sqrt(2\*pi\*K); %distance between cluster members and cluster head

Er=4000\*(2\*n\*ETX+n\*EDA+K\*Emp\*d1^4+n\*Efs\*d2^2); %energy desipated in a round

S(n+1).xd=sink.x; %sink is a n+1 node, x-axis postion of a node

S(n+1).yd=sink.y; %sink is a n+1 node, y-axis postion of a node

countCHs=0; %variable, counts the cluster head

cluster=1; %cluster is initialized as 1

flag\_first\_dead=0; %flag tells the first node dead

flag\_teenth\_dead=0; %flag tells the 10th node dead

flag\_all\_dead=0; %flag tells all nodes dead

dead=0; %dead nodes count initialized to 0

first\_dead=0;

teenth\_dead=0;

all\_dead=0;

allive=n;

%counter for bit transmitted to Bases Station and to Cluster Heads

packets\_TO\_BS=0;

packets\_TO\_CH=0;

for r=0:1:rmax

r

if(mod(r, round(1/P) )==0)

for i=1:1:n

S(i).G=0;

S(i).cl=0;

end

end

Ea=Et\*(1-r/rmax)/n;

dead=0;

for i=1:1:n

if (S(i).E<=0)

dead=dead+1;

if (dead==1)

if(flag\_first\_dead==0)

first\_dead=r;

flag\_first\_dead=1;

end

end

if(dead==0.1\*n)

if(flag\_teenth\_dead==0)

teenth\_dead=r;

flag\_teenth\_dead=1;

end

end

if(dead==n)

if(flag\_all\_dead==0)

all\_dead=r;

flag\_all\_dead=1;

end

end

end

if S(i).E>0

S(i).type='N';

end

end

STATISTICS.DEAD(r+1)=dead;

STATISTICS.ALLIVE(r+1)=allive-dead;

countCHs=0;

cluster=1;

if dead==n

break;

end

for i=1:1:n

if Ea>0

p(i)=P\*n\*(1+a)\*E(i)/(n+A)\*(Ea);

%p(i)=P\*n\*S(i).E\*E(i)/(Et\*Ea);

if(S(i).E>0)

temp\_rand=rand;

if ( (S(i).G)<=0)

if(temp\_rand<= (p(i)/(1-p(i)\*mod(r,round(1/p(i))))))

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(i))-1;

C(cluster).xd=S(i).xd;

C(cluster).yd=S(i).yd;

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;

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

end

STATISTICS.COUNTCHS(r+1)=countCHs;

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=0;

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

if(min\_dis\_cluster~=0)

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

S(C(min\_dis\_cluster).id).E = S(C(min\_dis\_cluster).id).E- ( (ERX + EDA)\*4000 );

packets\_TO\_CH=packets\_TO\_CH+1;

else

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

packets\_TO\_BS=packets\_TO\_BS+1;

end

S(i).min\_dis=min\_dis;

S(i).min\_dis\_cluster=min\_dis\_cluster;

else

min\_dis=sqrt( (S(i).xd-S(n+1).xd)^2 + (S(i).yd-S(n+1).yd)^2 );

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

packets\_TO\_BS=packets\_TO\_BS+1;

end

end

end

STATISTICS.PACKETS\_TO\_CH(r+1)=packets\_TO\_CH;

STATISTICS.PACKETS\_TO\_BS(r+1)=packets\_TO\_BS;

RES(r+1)=0;

for i=1:1:n

RES(r+1)=RES(r+1)+S(i).E;

end

end

first\_dead

teenth\_dead

all\_dead

x=1:1:r;

y=1:1:r;

w=1:1:r;

v=1:1:r;

for i=1:r;

x(i)=i;

y(i) = n- STATISTICS.DEAD(i); % remaining number of live nodes

w(i)=RES(i);

v(i)=STATISTICS.PACKETS\_TO\_BS(i);

end

figure(1)

plot(x,y,'r');

hold on;

figure(2)

plot(x,w,'r');

hold on;

figure(3)

plot(x,v,'r');

hold on;

%%%%%%%%%%%%%%%%%%%%%% PROPOSED ALGORITHM %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clc;

clear all;

close all;

% Simulation Parameters

%Field dimensions in meters

xm = 200;

ym = 200;

%initial x and y coordinates of the Sink

sink.x = 0;

sink.y = ym\*0.5;

%Number of Nodes in the field

n = 600;

%Optimal Election Probability of a node

%to become cluster head

p\_opt = 0.1;

L = 4000; % packet length

%Energy Model

Eo = 0.5; %Initial Energy

E\_elec=50\*0.000000001; % energy consumed by radio electronics in transmit/receive mode(J/bit)

E\_fs=10\*0.000000000001; %energy consumed by the power amplifier on the free space model(J/bit/m2)

E\_mp=0.0013\*0.000000000001; %energy consumed by the power amplifier on the multi path model(J/bit/m4)

E\_DA=5\*0.000000001; % energy consumed for data aggregation(J/bit/signal)

INFINITY = 999999999999999;

%maximum number of rounds

rmax=15000;

%threshold distance

do= 70;

m = 0.75;

a = 2;

m0 = 0.525;

b = 2.5;

m1 = 0.225;

u = 3;

c=0.02;

z=0.71;

T\_absolute= z\*Eo;

cr=5; % compression ratio

S = struct;

S.xd = zeros(n,1);

S.yd = zeros(n,1);

S.E = zeros(n,1);

S.type = zeros(n,1);

S.G = zeros(n,1);

S.CH = zeros(n,1);

S.D = zeros(n,1);

S.T = zeros(n,1);

S.min\_dis\_cluster= zeros(n,1);

S.dis\_to\_cluster= zeros(n,1);

C = struct;

C.id= zeros(25,1);

C.xd= zeros(25,1);

C.yd= zeros(25,1);

C.E= zeros(25,1);

C.member= zeros(25,1);

R = struct;

R.id= zeros(100,1);

R.xd= zeros(100,1);

R.yd= zeros(100,1);

p = zeros(n,1);

E = zeros(n,1);

DEAD = zeros(rmax,1);

RES = zeros(rmax,1);

% Creation of four-level heterogeneous network

% Normal Nodes

for i=1:1:240

S(i).xd = xm\*rand(1,1);

S(i).yd = ym\*rand(1,1);

S(i).E = Eo;

S(i).type = 'N';

S(i).G = 1;

S(i).CH = 0;

S(i).D = 0;

S(i).R = 0;

S(i).min\_dis\_cluster=0;

S(i).dis\_to\_cluster=0;

figure(3);

plot(S(i).xd,S(i).yd,'o', 'MarkerSize', 3, 'MarkerFaceColor', 'g');

hold on;

end

% Advanced Nodes

for i=241:1:420

S(i).xd = xm\*rand(1,1);

S(i).yd = ym\*rand(1,1);

S(i).E = Eo\*(1+a);

S(i).type = 'A';

S(i).G = 1;

S(i).CH = 0;

S(i).D = 0;

S(i).R = 0;

S(i).min\_dis\_cluster=0;

S(i).dis\_to\_cluster=0;

figure(3);

plot(S(i).xd,S(i).yd,'o', 'MarkerSize', 3, 'MarkerFaceColor', 'b');

hold on;

end

% Super Nodes

for i=421:1:546

S(i).xd = xm\*rand(1,1);

S(i).yd = ym\*rand(1,1);

S(i).E = Eo\*(1+b);

S(i).type = 'S';

S(i).G = 1;

S(i).CH = 0;

S(i).D = 0;

S(i).R = 0;

S(i).min\_dis\_cluster=0;

S(i).dis\_to\_cluster=0;

figure(3);

plot(S(i).xd,S(i).yd,'o', 'MarkerSize', 3, 'MarkerFaceColor', 'r');

hold on;

end

% Ultra-Super Nodes

for i=547:1:600

S(i).xd = xm\*rand(1,1);

S(i).yd = ym\*rand(1,1);

S(i).E = Eo\*(1+u);

S(i).type = 'U';

S(i).G = 0;

S(i).CH = 0;

S(i).D = 0;

S(i).R = 0;

S(i).min\_dis\_cluster=0;

S(i).dis\_to\_cluster=0;

figure(3);

plot(S(i).xd,S(i).yd,'o', 'MarkerSize', 5, 'MarkerFaceColor', 'y');

hold on;

end

%E\_total= (n\*m1\*Eo\*(1+u))+(n\*m0\*Eo\*(1+b))+(n\*m\*Eo\*(1+a))+(n\*(1-m1-m0-m)\*Eo);

E\_total= 0;

for i=1:1:n

E\_total=E\_total+S(i).E;

end

% Selection of Rendezvous Nodes

rn=1;

for i=1:1:n

if (S(i).type=='S')

Rx= rand;

if ((ym\*(1-Rx)/2)<=S(i).yd && (ym\*(1+Rx)/2)>=S(i).yd)

if S(i).E > 0

S(i).R= 1;

R(rn).xd= S(i).xd;

R(rn).yd= S(i).yd;

R(rn).id = i;

rn=rn+1;

end

end

end

end

for r=1:1:rmax

r

RES(r)=0;

for i=1:1:n

if S(i).E>0

RES(r)= RES(r)+ S(i).E;

end

end

dead=0;

for i=1:1:n

%checking if there is a dead node

if (S(i).E<=0)

S(i).D=1;

dead=dead+1;

end

end

if (dead == n)

break;

end

DEAD(r+1)=dead;

for i=1:1:n

S(i).CH=0;

end

d\_toBS = 0.765\*(xm/2);

k\_opt = (sqrt(n/(2\*pi)))\*(sqrt(E\_fs/E\_mp))\*(xm/(d\_toBS^2));

d\_toCH = xm/(sqrt(2\*pi\*k\_opt));

E\_round = L\*((2\*n\*E\_elec) + (n\*E\_DA) + (k\_opt\*E\_mp\*(d\_toBS^4)) + (n\*E\_fs\*(d\_toCH^2)));

R = E\_total/E\_round;

N= round(n/(k\_opt))+1;

M= round(N/cr)+1;

q= mod(r,M+2);

% Average Energy of the network in this round

Ei(r)= 0;

for i=1:1:n

if S(i).E>0

Ei(r)=Ei(r)+S(i).E;

end

end

E(r)= Ei(r)/n;

% Calculation of election probability of nodes to become CH

for i=1:1:n

if S(i).E <= T\_absolute

p(i)= (c\* p\_opt \*(1+u)\*S(i).E)/((1+m\*(a+m0\*(-a+b+m1\*(-b+u))))\*E(r));

%S(i).G= 1;

else

if S(i).type == 'N'

p(i)= (p\_opt \*S(i).E)/((1+m\*(a+m0\*(-a+b+m1\*(-b+u))))\*E(r));

%S(i).G= 1;

end

if S(i).type == 'A'

p(i)= (p\_opt\*(1+a)\*S(i).E)/((1+m\*(a+m0\*(-a+b+m1\*(-b+u))))\*E(r));

%S(i).G= 1;

end

if S(i).type == 'S'

p(i)= (p\_opt\*(1+b)\*S(i).E)/((1+m\*(a+m0\*(-a+b+m1\*(-b+u))))\*E(r));

%S(i).G= 1;

end

if S(i).type == 'U'

p(i)= (p\_opt\*(1+u)\*S(i).E)/((1+m\*(a+m0\*(-a+b+m1\*(-b+u))))\*E(r));

%S(i).G= 1;

end

end

end

% Operation for epoch

for i=1:1:n

if( (mod(r, round(1/p(i))))==0) % checking if node has been CH in last 1/p rounds

S(i).G=0;

%S(i).cl=0;

end

end

for i=1:1:n

if S(i).G ==0 % i.e. if the node belongs to G

S(i).T = p(i)/(1-(p(i)\*(mod(r,round(1/p(i)))))); % Threshold Probability

else

S(i).T = 0;

end

end

% Election of CLUSTER HEADS (CHs)

if q==1

cluster=1;

for xcord=0:xm/6:((5\*xm)/6)

for ycord=0:ym/4:((3\*ym)/4)

for i=1:1:n

if (S(i).xd>= xcord && S(i).xd<(xcord+(xm/6)) && S(i).yd>= ycord && S(i).yd<(ycord+(ym/4)))

Rx = rand;

%disp('a')

if S(i).T >= 0.2

if S(i).R~=1

if S(i).E>0

%disp('b')

S(i).CH = 1; % node becomes CH

C(cluster).id= cluster;

C(cluster).xd= S(i).xd;

C(cluster).yd= S(i).yd;

C(cluster).E= S(i).E;

S(i).G = round(1/p(i))-1;

break;

end

end

end

end

end

cluster=cluster+1;

end

end

cluster1=cluster-1;

for xcord=0:xm/6:((5\*xm)/6)

for ycord=0:ym/4:((3\*ym)/4)

countCH=0;

for i=1:1:n

if (S(i).xd>= xcord && S(i).xd<(xcord+(xm/6)) && S(i).yd>= ycord && S(i).yd<(ycord+(ym/4)))

if S(i).CH==1

countCH=countCH+1;

break;

end

end

end

if countCH==0

maxenergy=0;

for i=1:1:n

if (S(i).xd>= xcord && S(i).xd<(xcord+(xm/6)) && S(i).yd>= ycord && S(i).yd<(ycord+(ym/4)))

if S(i).E > maxenergy

maxenergy=S(i).E;

end

end

end

for i=1:1:n

if (S(i).xd>= xcord && S(i).xd<(xcord+(xm/6)) && S(i).yd>= ycord && S(i).yd<(ycord+(ym/4)))

if S(i).E == maxenergy

for x=1:1:cluster1

if C(x).id==0

C(x).id=x;

C(x).xd=S(i).xd;

C(x).yd=S(i).yd;

break;

end

end

end

end

end

end

end

end

%determining number of nodes in each cluster

member=zeros(cluster1,1);

for xcord=0:xm/6:((5\*xm)/6)

for ycord=0:ym/4:((3\*ym)/4)

% INSIDE A CLUSTER

for j=1:1:cluster1

if (C(j).xd>= xcord & C(j).xd<=(xcord+(xm/6)) & C(j).yd>= ycord & C(j).yd<=(ycord+(ym/4)))

for i=1:1:n

if (S(i).xd>= xcord && S(i).xd<=(xcord+(xm/6)) && S(i).yd>= ycord && S(i).yd<=(ycord+(ym/4)))

if S(i).R~=1

if S(i).E>0

member(j)=member(j)+1;

end

end

end

end

end

end

end

end

end

gamma = 0.5; % weighted variable

w=zeros(n,1);

threshold = sqrt(((xm/18)^2)+((ym/12)^2));

% construction of sub-clusters in each cluster

for xcord=0:xm/6:((5\*xm)/6)

for ycord=0:ym/4:((3\*ym)/4)

% INSIDE A CLUSTER

% Determine the CH in this cluster

if q==1

E\_residual=0;

for j=1:1:cluster1

if (C(j).xd>= xcord & C(j).xd<=(xcord+(xm/6)) & C(j).yd>= ycord & C(j).yd<=(ycord+(ym/4)))

break;

end

end

E\_residual=C(j).E; % residual energy of cluster head

end

s=1;

t=1;

% GOING INTO SUB-CLUSTERS

for xcoor=xcord:xm/18:(xcord+(xm/6)-(xm/18))

for ycoor=ycord:ym/12:(ycord+(ym/4)-(ym/12))

% INSIDE A SUB-CLUSTER

% count the number of nodes in sub-cluster

nodes=zeros(n,1); % nodes transferring data to CH through relay nodes

for i=1:1:n

if (S(i).CH~=1 && S(i).R~=1)

if (S(i).xd>= xcoor && S(i).xd<=(xcoor+(xm/18)) && S(i).yd>= ycoor && S(i).yd<=(ycoor+(ym/12)))

nodes((10\*s)+t)=nodes((10\*s)+t)+1;

end

end

end

% Calculation of w for each node in this sub-cluster

if q==1

min\_w = INFINITY;

for i=1:1:n

if (S(i).R~=1)

if (S(i).xd>= xcoor && S(i).xd<=(xcoor+(xm/18)) && S(i).yd>= ycoor && S(i).yd<=(ycoor+(ym/12)))

% Calculation of summation term

term=0;

for k=1:1:n

if (S(k).xd>= xcoor && S(k).xd<=(xcoor+(xm/18)) && S(k).yd>= ycoor && S(k).yd<=(ycoor+(ym/12)))

term=term+(((sqrt(((S(i).xd - S(k).xd)^2)+((S(i).yd - S(k).yd)^2)))-(sqrt(((S(k).xd - C(j).xd)^2)+((S(k).yd - C(j).yd)^2))))^2);

end

end

w(i) = (gamma/C(j).E) + ((1-gamma)/nodes((10\*s)+t))\*term;

if w(i) < min\_w

min\_w = w(i);

end

end

end

end

end

% determining the relay node of the sub-cluster

if q==2

dis\_to\_relaynode=0;

for i=1:1:n

if (S(i).xd>= xcoor && S(i).xd<=(xcoor+(xm/18)) && S(i).yd>= ycoor && S(i).yd<=(ycoor+(ym/12)))

if w(i)==min\_w

S(i).relaynode=1;

% energy dissipated by relay nodes in inter-sub-cluster transmissions

distoCH= sqrt(((S(i).xd - C(j).xd)^2)+((S(i).yd - C(j).yd)^2));

if S(i).E>0

S(i).E = S(i).E - (nodes((10\*s)+t))\*L\*(E\_elec+E\_DA); % relay node receives and aggregates data

end

if S(i).E>0

S(i).E = S(i).E - ((nodes((10\*s)+t)+1))\*( (E\_elec\*(L)) + (E\_fs\*L\*((distoCH)^2))); % relay node transmits data to CH

end

% calculation of energy dissipated by nodes in intra-sub-cluster txns.

for k=1:1:n

if S(k).R~=1

if (S(k).xd>= xcoor && S(k).xd<=(xcoor+(xm/18)) && S(k).yd>= ycoor && S(k).yd<=(ycoor+(ym/12)))

dis\_to\_relaynode = sqrt( ((S(i).xd - S(k).xd)^2) + ((S(i).yd - S(k).yd)^2) );

if S(k).E>0

S(k).E = S(k).E - ( E\_elec\*(L) + E\_fs\*L\*((dis\_to\_relaynode)^2));

end

end

end

end

end

end

end

end

s=s+1;

t=t+1;

end

end

end

end

if q~=1

for o=1:1:cluster1 % energy dissipated by CHs during reception of data

if C(o).E >0

C(o).E = C(o).E - (E\_elec\*L\*member(o));

for k=1:1:n

if (S(k).xd==C(o).xd && S(k).yd==C(o).yd)

S(k).E=C(o).E;

end

end

end

end

end

%Energy dissipated in inter-cluster transmissions

if q~=1

dis\_to\_rn=0;

for l=1:1:cluster1

min\_dis\_to\_rn=INFINITY;

% determination of RN at least distance

for i=1:1:n

if S(i).E>0

if S(i).R==1

dis\_to\_rn= sqrt(((S(i).xd-C(l).xd)^2)+((S(i).yd-C(l).yd)^2));

if dis\_to\_rn < min\_dis\_to\_rn

min\_dis\_to\_rn= dis\_to\_rn;

end

end

end

end

%Energy dissipated by RNs for reception

for i=1:1:n

if S(i).E>0

if S(i).R==1

if (sqrt(((S(i).xd-C(l).xd)^2)+((S(i).yd-C(l).yd)^2)) == min\_dis\_to\_rn)

S(i).E = S(i).E - (E\_elec+E\_DA)\*L;

end

end

end

end

% energy dissipated by CHs for transmission of data to RNs

if min\_dis\_to\_rn > do

if C(l).E>0

C(l).E = C(l).E - ( E\_elec\*(L) + E\_mp\*L\*((min\_dis\_to\_rn)^4));

for k=1:1:n

if (S(k).xd==C(l).xd && S(k).yd==C(l).yd)

S(k).E=C(l).E;

end

end

end

else

if C(l).E>0

C(l).E = C(l).E - ( E\_elec\*(L) + E\_fs\*L\*((min\_dis\_to\_rn)^2));

for k=1:1:n

if (S(k).xd==C(l).xd && S(k).yd==C(l).yd)

S(k).E=C(l).E;

end

end

end

end

end

end

end

x=1:1:r;

y=1:1:r;

w=1:1:r;

for i=1:r;

x(i)=i;

y(i) = n- DEAD(i); % remaining number of live nodes

w(i)=RES(i);

end

figure(1)

plot(x,y,'r');

hold on;

figure(2)

plot(x,w,'r');

hold on;