Aim:Implement a Power Efficient Gathering in Sensor Information System using Simulation Tool

Pre-Requisite:

Sensor webs consisting of nodes with limited battery power and wireless communications are deployed to collect useful information from the field. Gathering sensed information in an energy efficient manner is critical to operate the sensor network for a long period of time. In W. Heinzelman et al. (Proc. Hawaii Conf. on System Sci., 2000), a data collection problem is defined where, in a round of communication, each sensor node has a packet to be sent to the distant base station. If each node transmits its sensed data directly to the base station then it will deplete its power quickly.

The LEACH protocol presented by W. Heinzelman et al. is an elegant solution where clusters are formed to fuse data before transmitting to the base station. By randomizing the cluster heads chosen to transmit to the base station, LEACH achieves a factor of 8 improvement compared to direct transmissions, as measured in terms of when nodes die. In this paper, we propose PEGASIS (power-efficient gathering in sensor information systems), a near optimal chain-based protocol that is an improvement over LEACH.

In PEGASIS, each node communicates only with a close neighbor and takes turns transmitting to the base station, thus reducing the amount of energy spent per round. Simulation results show that PEGASIS performs better than LEACH by about 100 to 300% when 1%, 20%, 50%, and 100% of nodes die for different network sizes and topologies.

Program:

```
%%%%%%%%%%%%%%%%%%%% Network Establishment Parameters %%%%%%%%%%%%%%%%%%%%
                        %%% Area of Operation %%%
                        % Field Dimensions in meters %
                        function [route_order,cl_head,turn_in] =
                        PEGASIS(Did,NDid,X,Y,Pwr,grid,n,sinkx,sinky,round,turn_in,max_energy
                        )
                        %Input:
                        % Did contains the Device number which starts from 1. Changes as
                        nodes die.
                        % NDid contains the unique ID of the devices as assigned in NetSim.
                        % X contains the X coordinates of the sensor nodes
                        % Y contains the Y coordinates of the sensro nodes
                        % Pwr contains the remaining energy of the sensors in milli Joules
                        % grid contains the length of the grid environment in Netsim in
                        meters
                        % n contains the total number of sensors alive in NetSim
                        % sinkx contains the X coordinate of the sink node
                        % sinky contains the Y coordinate of the sink node
                        % round contains the the number of calls made my NetSim to MATLAB
                        % turn_in keeps track of the turn variable used in the algorithm
                        % max_energy contains the max initial energy in the network
                        %Output to MATLAB workspace:
                        % route_order contains the order of devices as per the chain that is
                        formed
                        % cl_head contains the id of the node which is elected as cluster
                        head
                        % turn_in contains the value of the variable turn used in algorithm

                        xm=grid;
                        ym=grid;
                        x=0; % added for better display results of the plot
                        y=0; % added for better display results of the plot

                        % Round of Operation %
                        rnd=round;
                        % Current Number of operating Nodes %
                        operating_nodes=n;
                        d(n,n)=0;
                        dead_nodes=0;
                        selected=0;
                        turn=turn_in;
                        %%%%%%%%%%%%%%%%%%%%%%%%%%%% End of Parameters
                        %%%%%%%%%%%%%%%%%%%%%%%%%%%%
                        %%% Creation of the Wireless Sensor Network %%%
                        % Plotting the WSN %
                        fig1=gcf;
                        figure(fig1);
                        subplot(1,2,1);
```

```
clf;

for i=1:n
SN(i).id=Did(i);      % sensor's ID number
SN(i).Nid=NDid(i); % NetSim Device ID
SN(i).x=X(i);  % X-axis coordinates of sensor node
SN(i).y=Y(i);  % Y-axis coordinates of sensor node
SN(i).E=Pwr(i);     % nodes energy levels (initially set to be equal
to "Eo"
SN(i).cond=1;   % States the current condition of the node. when the
node is operational its value is =1 and when dead =0
SN(i).dts=0;    % nodes distance from the sink
SN(i).role=0;   % node acts as normal if the value is '0', if
elected as a cluster head it  gets the value '1' (initially all
nodes are normal)
SN(i).pos=0;
SN(i).closest=0;
SN(i).prev=0;
SN(i).dis=0;  % distance between two nodes headin towards to the
cluster head from position 1
SN(i).dis2=0;   % distance between two nodes headin towards to the
cluster head from position 2
SN(i).order=0;
SN(i).sel=0;    % states if the node has already operated for this
round or not (if 0 then no, if 1 then yes)
SN(i).rop=0;    % number of rounds node was operational
SN(i).tel=0;    % states how many times the node was elected as a
Cluster Head
order(i)=0;
hold on;
subplot(1,2,1);
plot(x,y,xm,ym,SN(i).x,SN(i).y,'ob',sinkx,sinky,'*r');
text(SN(i).x+(grid/100),SN(i).y,num2str(SN(i).Nid));
title_text=sprintf('Wireless Sensor Network (n=%d)',n);
title (title_text);
xlabel '(m)';
ylabel '(m)';
hold on;

end
text(sinkx-(grid/100),sinky-(grid/100),'sink');

if operating_nodes>1
% Calculates Distance Between Each Node and the Sink (Base Station)
%
for i=1:n
SN(i).dts=sqrt((sinkx-SN(i).x)^2 + (sinky-SN(i).y)^2);
%SN(i).Esink=Eelec*k + Eamp*k*(SN(i).dts)^2;
T(i)=SN(i).dts;
end


A=sort(T,'descend'); % Creates array A containing the distance
between each node and the sink,
% sorted in an asceding order
```

```
A_id(1:n)=0;
% Creates array A_id which is sorted in a way that it's elements are
% aligned with those of A. Contains the node ID
for i=1:n
for j=1:n
if A(i)==SN(j).dts
A_id(i)=SN(j).id;
end
end
end


% Creation of d Array with shortest distances %


for i=1:n
SN(i).closest=0;
for j=1:n
d(j,i)=sqrt((SN(i).x-SN(j).x)^2 + (SN(i).y-SN(j).y)^2);
if d(j,i)==0
d(j,i)=9999;
end
end
end


for i=1:n
[M,I]=min(d(:,i)); % finds the minimum distance of node to CH
[Row, Col] = ind2sub(size(d),I); % displays the Cluster Number in
which this node belongs too
SN(i).closest=Row; % assigns node to the cluster
SN(i).dis= d(Row,i); % assigns the distance of node to CH
end


% Choosing furthest node from sink %
for i=1:n
if SN(A_id(i)).E>0 && SN(A_id(i)).sel==0 && SN(A_id(i)).cond==1
set= A_id(i);
SN(set).sel=1;
SN(set).pos=1;
break;
end
end
order(1)=set;

temp=1;
while selected<n
min_dis=9999;
for i=1:n
if  SN(i).sel==0
d=sqrt((SN(i).x-SN(set).x)^2 + (SN(i).y-SN(set).y)^2);
if d<min_dis
min_dis=d;
```

```
next=i;
end
end
end
selected=selected+1;
SN(set).closest=next;
SN(set).dis=min_dis;
SN(next).sel=1;
SN(next).prev=set;
SN(next).dis2=sqrt((SN(set).x-SN(next).x)^2 +
(SN(set).y-SN(next).y)^2);
plot([SN(set).x SN(next).x], [SN(set).y SN(next).y])
hold on;
set=next;
temp=temp+1;
order(temp)=set;
end


order(n+1)=[];
SN(set).pos=2;
SN(set).dis=0;
SN(set).closest=0;
for i=1:n
if SN(i).closest==set && SN(i).pos==0;
SN(set).prev=i;
SN(set).dis2=sqrt((SN(i).x-SN(set).x)^2 + (SN(i).y-SN(set).y)^2);
end
end

% Data Transmission %


if operating_nodes>0

%         energy=0;

for i=1:n
SN(i).role=0;
end

% Cluster Head Election %
cluster_head=mod(turn,n) +1;
if SN(cluster_head).cond==0
while SN(cluster_head).cond==0
turn=turn+1;
cluster_head=mod(turn,n) +1;
end
end
if SN(cluster_head).cond==1
SN(cluster_head).role=1;
SN(cluster_head).tel=SN(cluster_head).tel+1;
figure(fig1);
subplot(1,2,1);
%figure(1)
```

```
plot(SN(cluster_head).x,SN(cluster_head).y,'.r','MarkerSize',20)
end


for i=1:n

if SN(order(i)).E<=0 && SN(order(i)).cond==1
SN(order(i)).cond=0;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if SN(order(i)).pos==1 && SN(order(i)).role==0
if operating_nodes==1
%done='OK'
else
t=i;
while SN(order(t)).cond==0 && t<n
t=t+1;
end
SN(order(t)).pos=1;
SN(order(t)).prev=0;
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if SN(order(i)).pos==2 && SN(order(i)).role==0
if operating_nodes==1
% done='OK';
else
t=i;
while SN(order(t)).cond==0 && t>1
t=t-1;
end
SN(order(t)).pos=2;
SN(order(t)).closest=0;
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if SN(order(i)).pos==0 && SN(order(i)).role==1
SN(order(i)).role=0;
after=i;
for l=after:n
if SN(order(l)).cond==1
break;
end
end
bef=i;
for j=bef:-1:1
if SN(order(j)).cond==1
break;
end
end
SN(order(j)).closest=order(l);
SN(order(l)).prev=order(j);
SN(order(j)).dis=sqrt((SN(order(l)).x-SN(order(j)).x)^2 +
(SN(order(l)).y-SN(order(j)).y)^2);
```

```
SN(order(l)).dis2=SN(order(j)).closest;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if SN(order(i)).pos==1 && SN(order(i)).role==1
SN(order(i)).role=0;
t=i;
while SN(order(t)).cond==0 && t<n
t=t+1;
end
SN(order(t)).pos=1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if SN(order(i)).pos==2 && SN(order(i)).role==1
SN(order(i)).role=0;
t=i;
while SN(order(t)).cond==0 && t>1
t=t-1;
end
SN(order(t)).pos=2;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if SN(order(i)).pos==0 && SN(order(i)).role==0
after=i;
for l=after:n
if SN(order(l)).cond==1
break;
end
end
bef=i;
for j=bef:-1:1
if SN(order(j)).cond==1
break;
end
end
SN(order(j)).closest=order(l);
SN(order(l)).prev=order(j);
SN(order(j)).dis=sqrt((SN(order(l)).x-SN(order(j)).x)^2 +
(SN(order(l)).y-SN(order(j)).y)^2);
SN(order(l)).dis2=SN(order(j)).dis;
end



operating_nodes=operating_nodes-1;
dead_nodes=dead_nodes+1;
SN(order(i)).closest=0;
SN(order(i)).prev=0;
SN(order(i)).dis=0;
SN(order(i)).dis2=0;
SN(order(i)).pos=101;
SN(order(i)).rop=rnd;
end


end
```

```
turn=turn+1;
factor =0;


if operating_nodes>2
%3D Plotting starts here
new_p=zeros(n,1);
Xc=zeros(n,1);
Yc=zeros(n,1);
for i=1:n
new_p(i)=SN(i).E;
Xc(i)=SN(i).x;
Yc(i)=SN(i).y;
end
new_p=new_p-2;
figure(fig1);
subplot(1,2,2);
tri = delaunay(Xc,Yc);
trisurf(tri,Xc,Yc,new_p);
axis([0 max(Xc) 0 max(Yc) 0 max_energy*2]);
title 'Energy Consumption';
xlabel('Sensor X position')
ylabel('Sensor Y position')
zlabel('Energy Consumed (J)')
lighting phong
shading interp
colorbar EastOutside
%3D Plotting ends here
end
end
else
cluster_head=n;
order(n)=n;
SN(cluster_head).role=1;
SN(cluster_head).tel=SN(cluster_head).tel+1;
figure(fig1);
subplot(1,2,1);
plot(SN(cluster_head).x,SN(cluster_head).y,'.r','MarkerSize',20)
turn=turn+1;
end

route_order=zeros(n,1);
for i=1:n
route_order(i)=SN(order(i)).id;
if SN(i).role==1
cl_head=SN(i).id;
end
end
turn_in=turn;

end
```

Execution:

1. C:\Use\Desktop\WirelsessNetworks
2. Windows/192.160.10.120

OUTPUT: