Group 05

Zhu Qi (A0224460N)
Emil Yong Kai Wen (A0169907M)
Tang Yuyi (A0232683A)
Chen Zuona (A0224670J)
Faisal Ahmad (A0034178E)
Ivander Jonathan Marella Waskito (A0200825R)

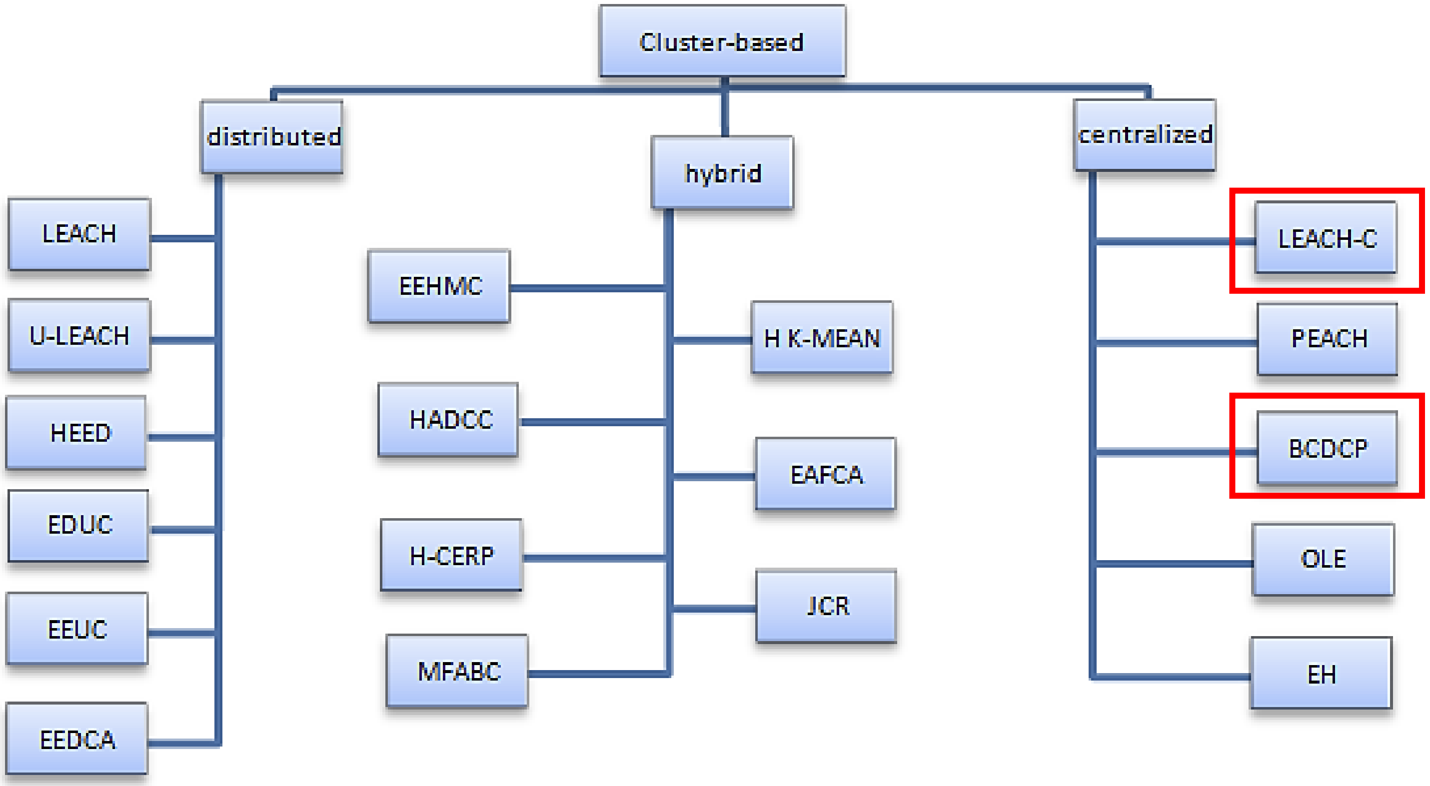
EE5132 Wireless and Sensor Networks
EE5024 IoT Sensor Networks

Simulation Study of Variations to LEACH Protocol

Variation A

Scope

- Part 1 – LEACH
  - Network Size Variation
- Part 2 – Centralised Variations
  - LEACH-C
  - BCDCP

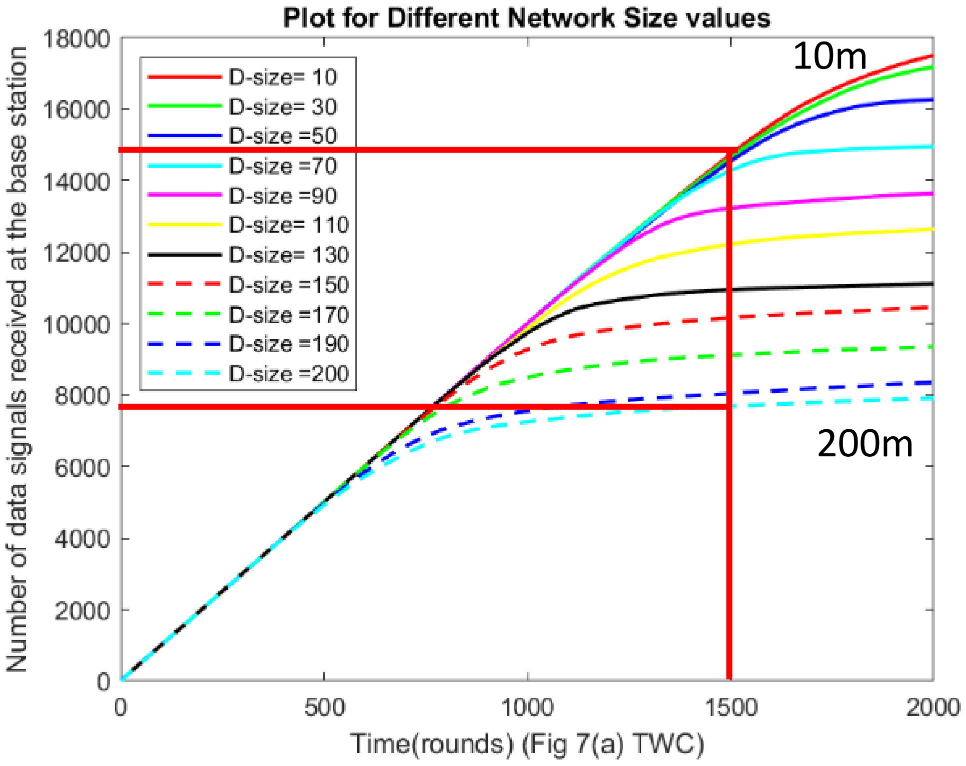


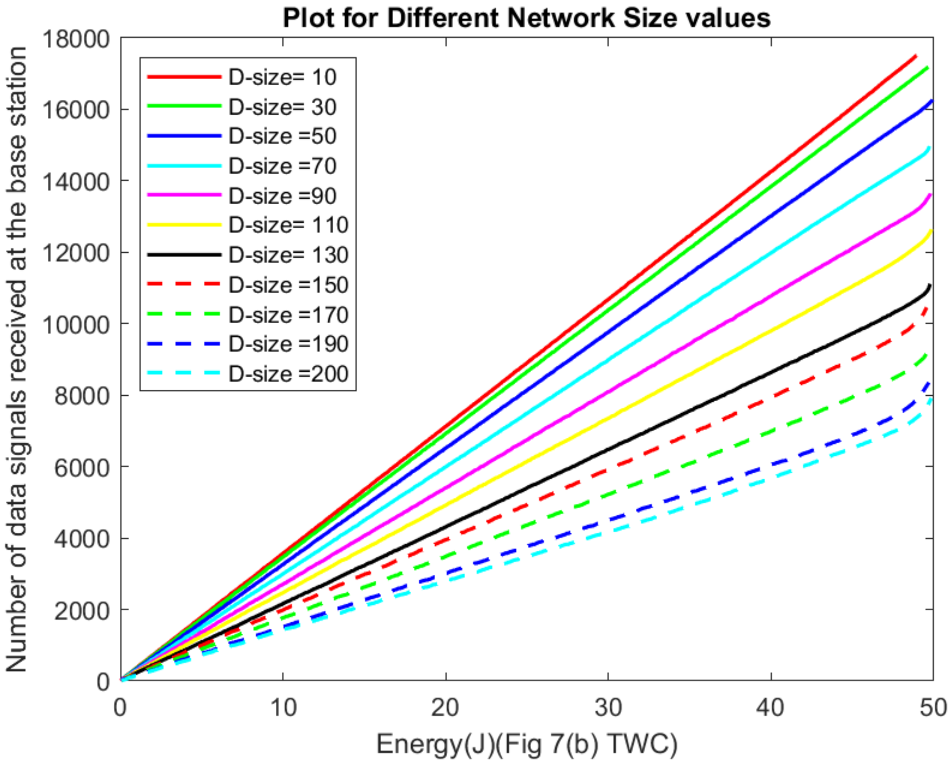
A. A. Hassan, W. Shah, M. F. Iskandar, and A. A.-J. Mohammed, "Clustering Methods for Cluster-based Routing Protocols in Wireless Sensor Networks: Comparative Study," vol. 12, no. 21, p. 11, 2017.

Part 1 – LEACH

Data delivery inversely proportional to network diameter

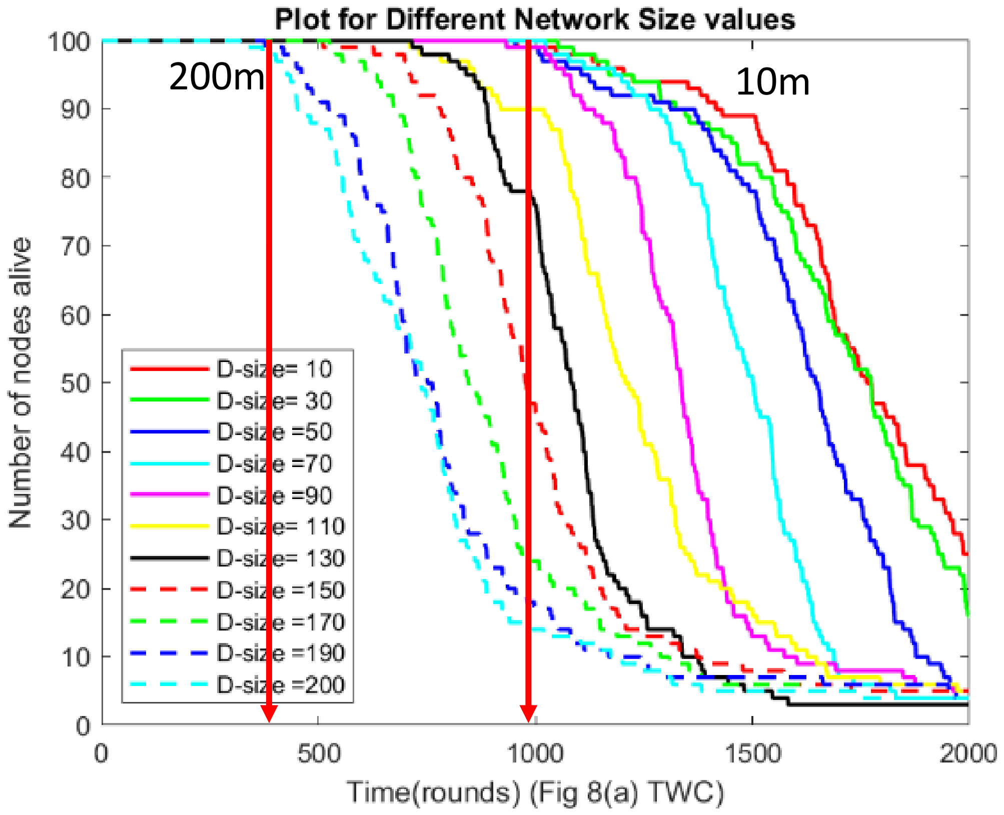
Less efficient data delivery in larger network

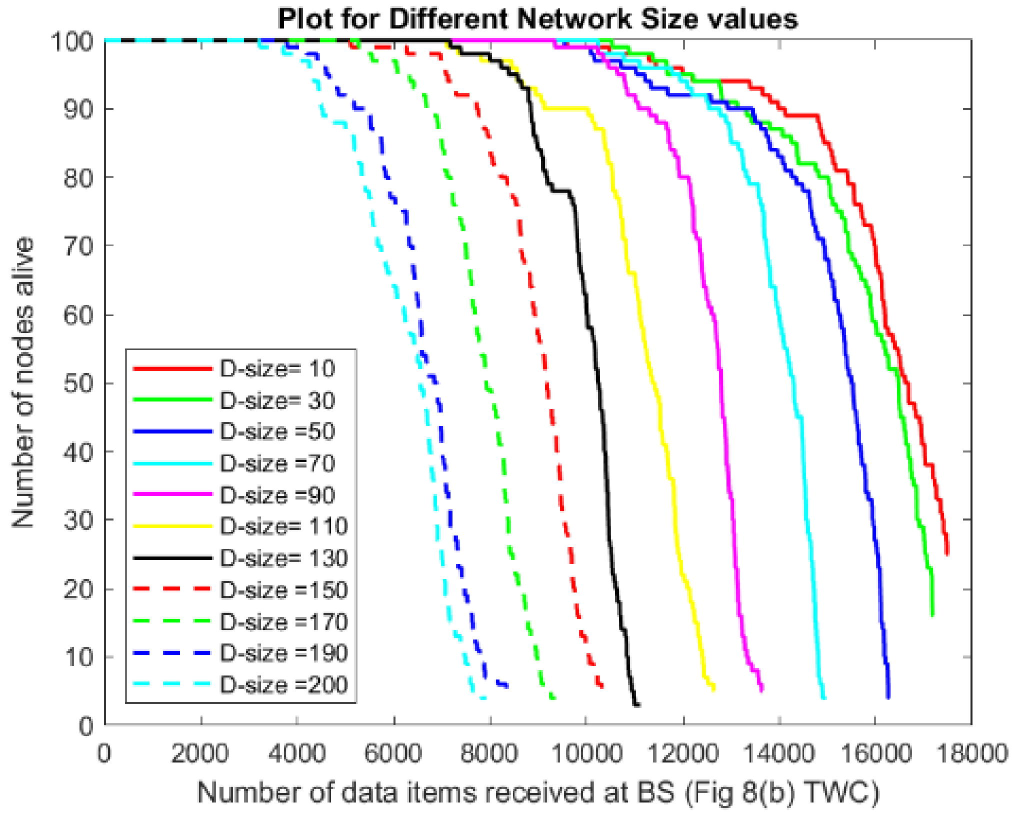




Part 1 – LEACH

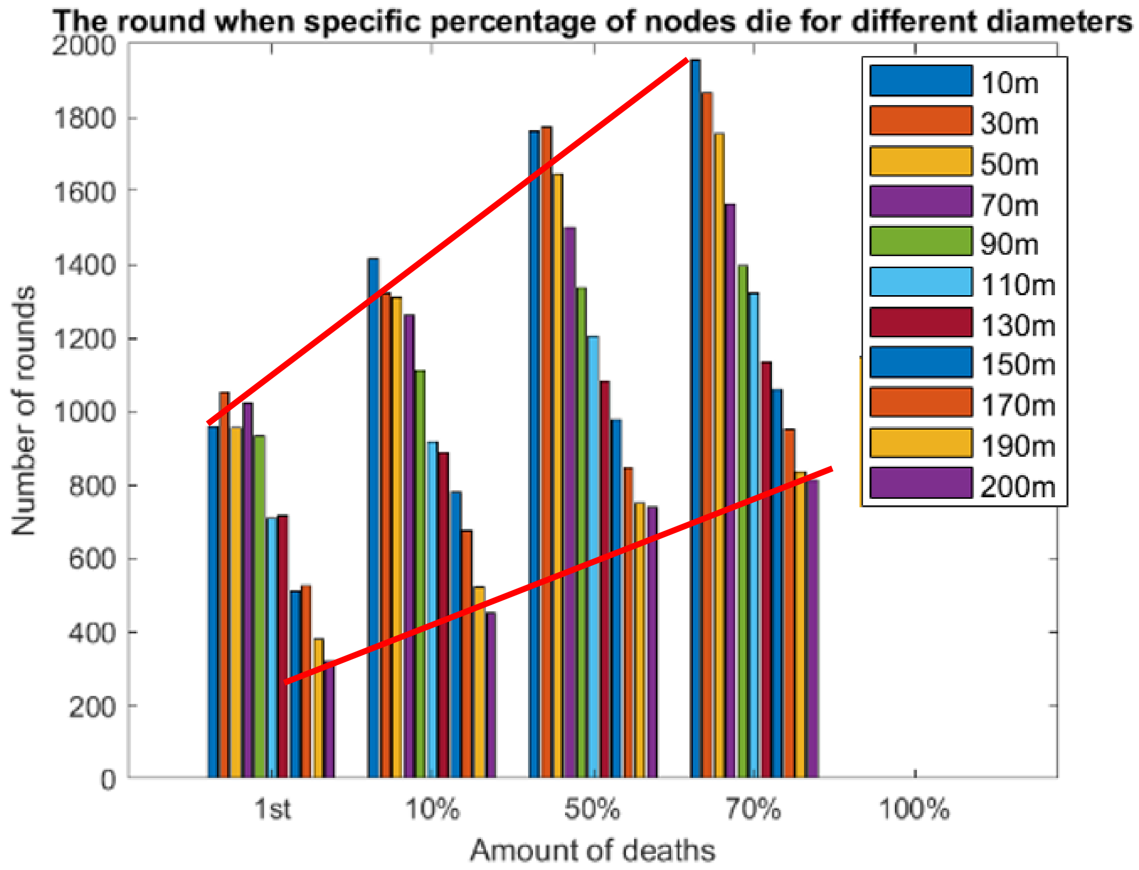
Network life & total data delivery is lesser for larger networks





Part 1 – LEACH

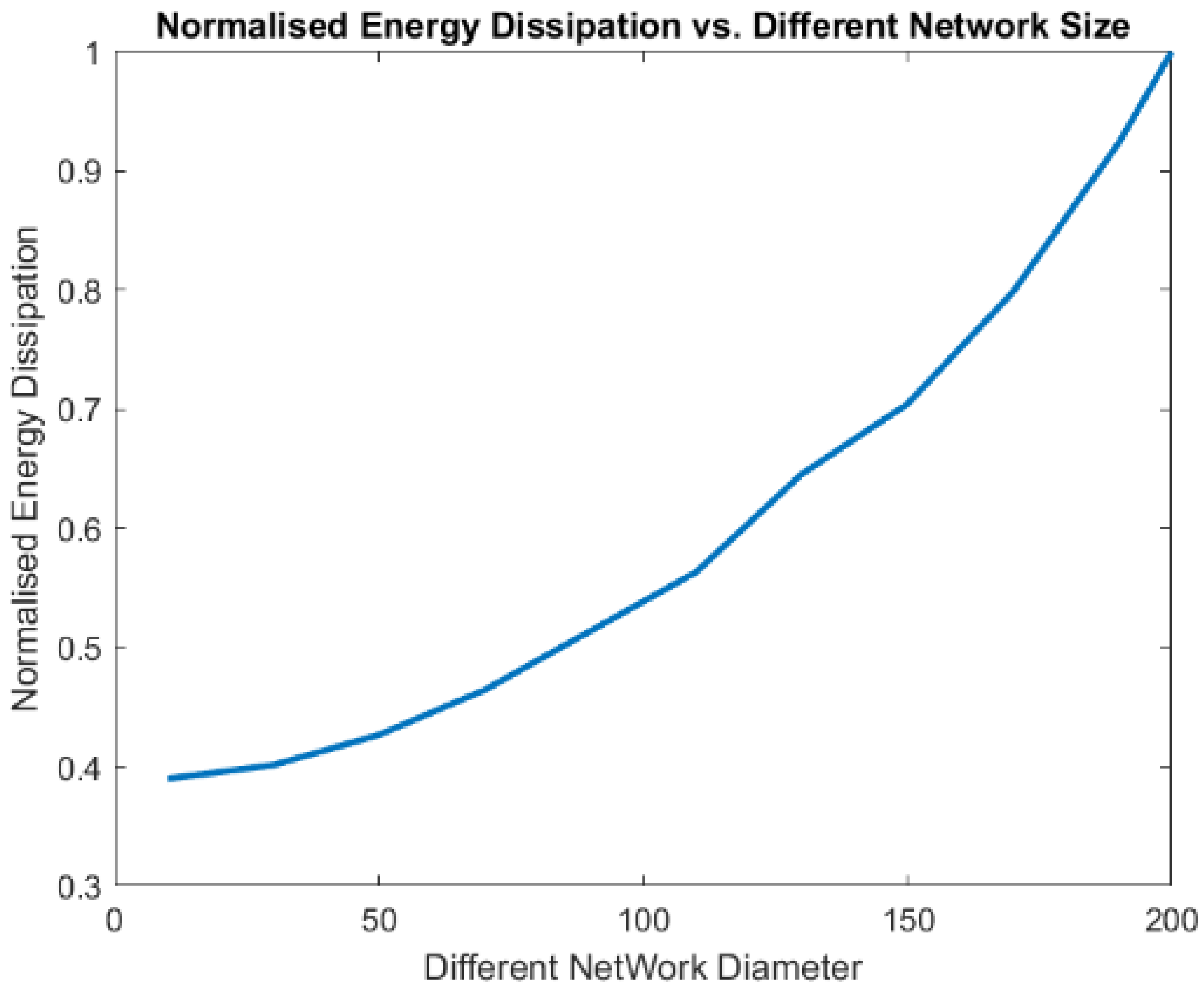
LEACH increases longevity for smaller networks more than larger networks



Part 1 – LEACH

Calculation of
Normalized Energy Dissipation:
1. Identify last rounds before any death occurs for different diameters.
2. Get the minimum round index.
3. Get the accumulated consumed energy at this minimum round index for different diameters.
4. Normalization: divide each energy value by the maximum value.

Exponential growth due to:
Energy $\propto$distance^2 or distance^4



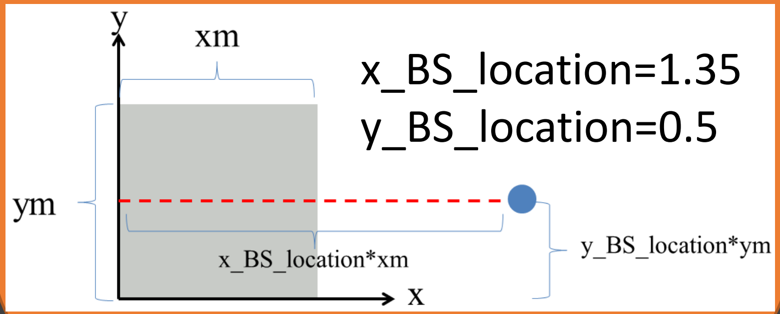
Part 2 – BS Location

Optimum k (number of clusters)

Num of nodes Network diameter

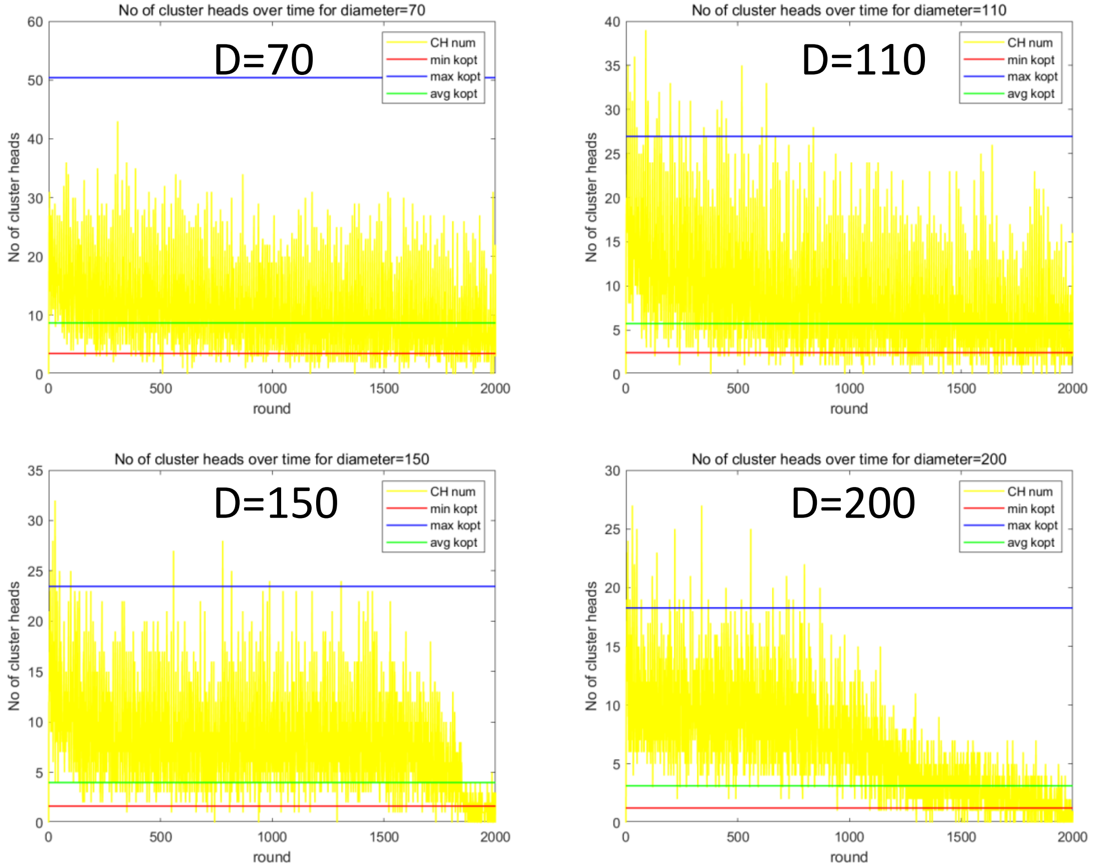
$$k_{opt} = \frac{\sqrt{N}}{\sqrt{2\pi}} \sqrt{\frac{\varepsilon_{fs}}{\varepsilon_{mp}}} \frac{M}{d_{toBS}^2}$$

Distance between node and BS

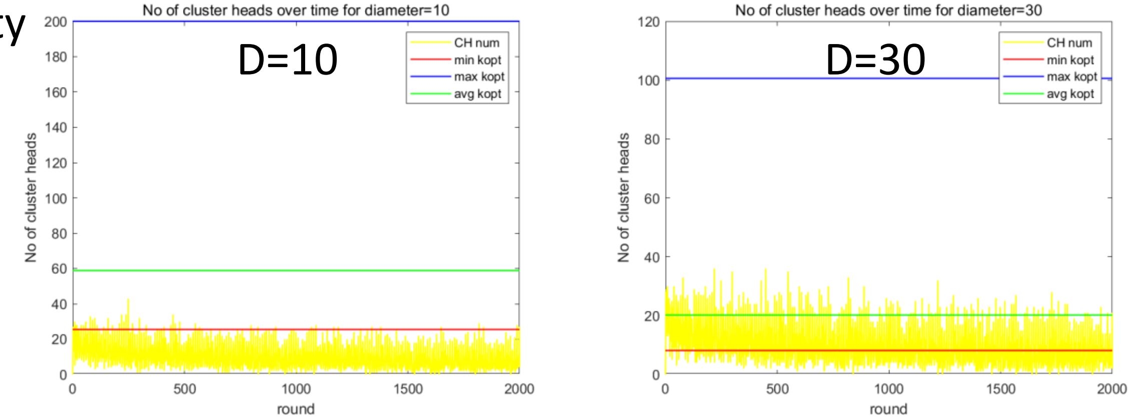


Experiment 1: BS outside network (not very far from network)

CH num
min kopt
max kopt
avg kopt

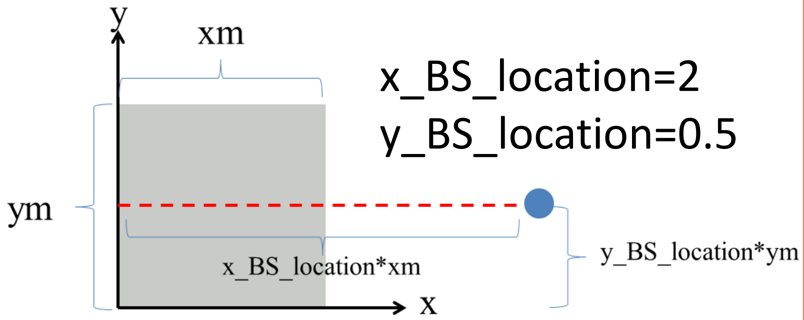


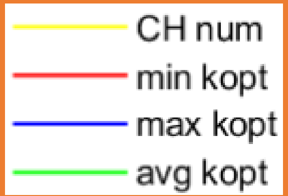
Abnormality



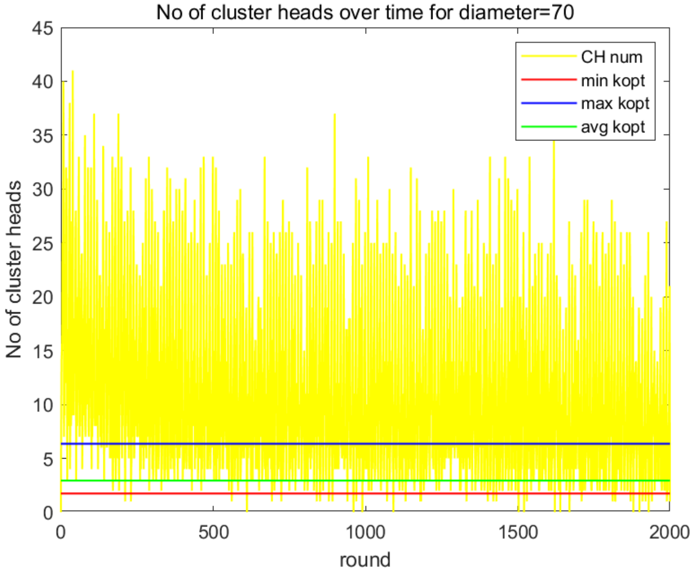
Part 2 – BS Location

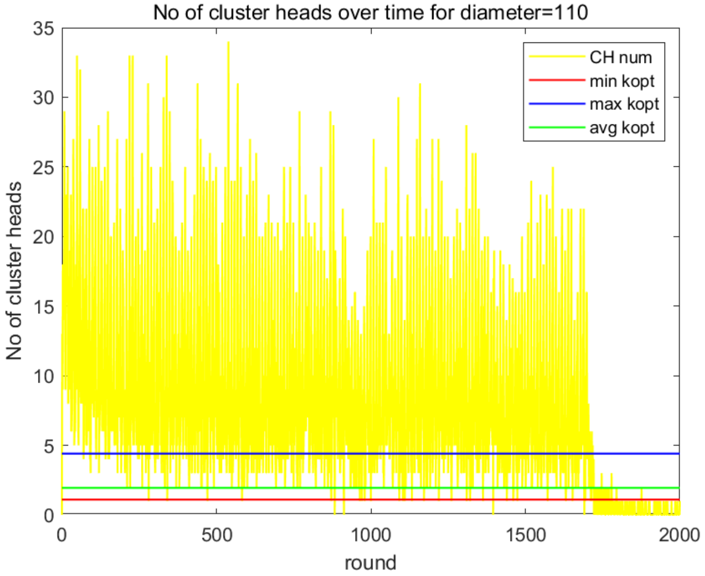
$$k_{opt} = \frac{\sqrt{N}}{\sqrt{2\pi}}\sqrt{\frac{\varepsilon_{fs}}{\varepsilon_{mp}}\frac{M}{d_{toBS}^{2}}}$$

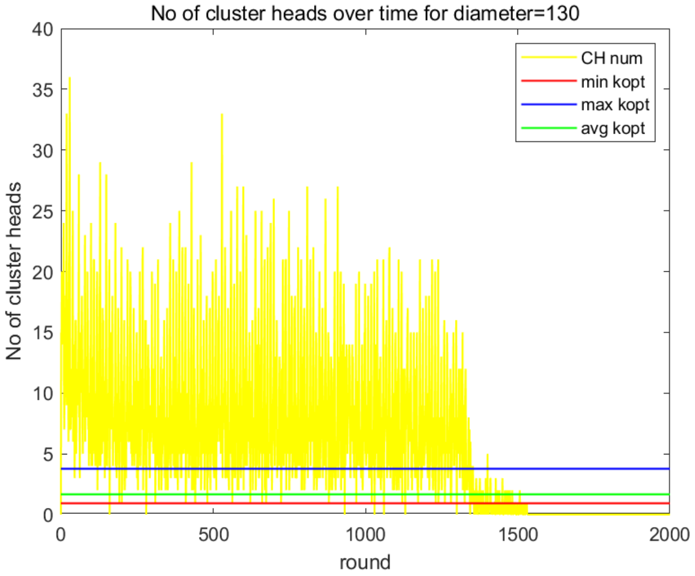


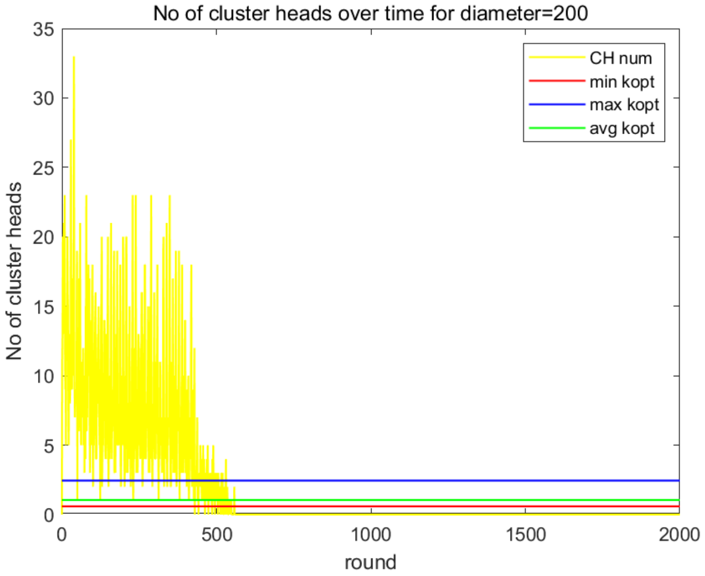


Experiment 2: BS outside of network, far from network



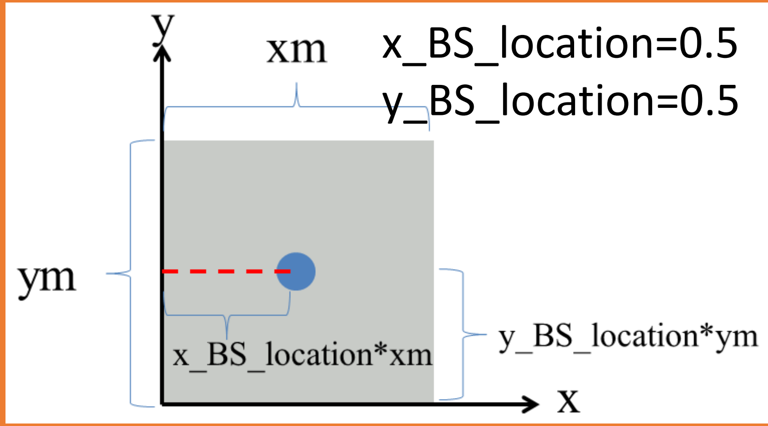


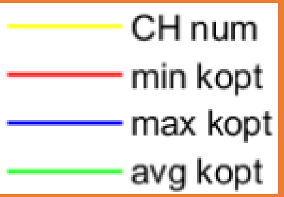




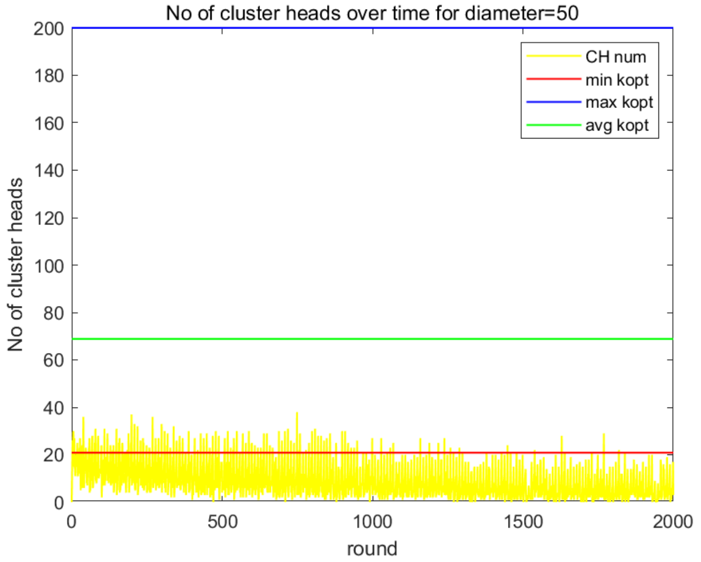
Part 2 – BS Location

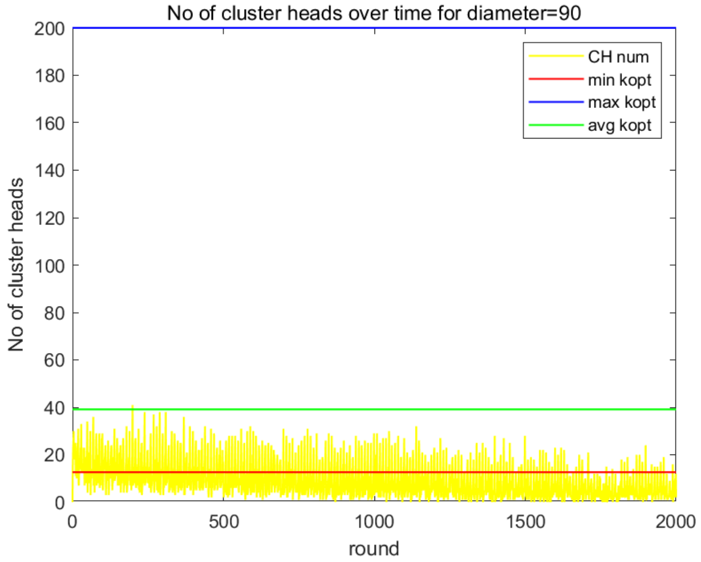
$$k_{opt} = \frac{\sqrt{N}}{\sqrt{2\pi}}\sqrt{\frac{\varepsilon_{fs}}{\varepsilon_{mp}}}\frac{M}{d_{toBS}^2}$$

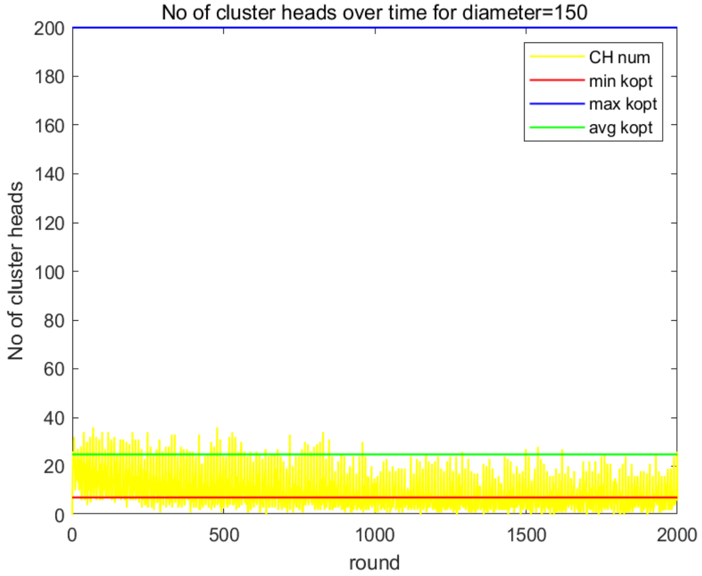


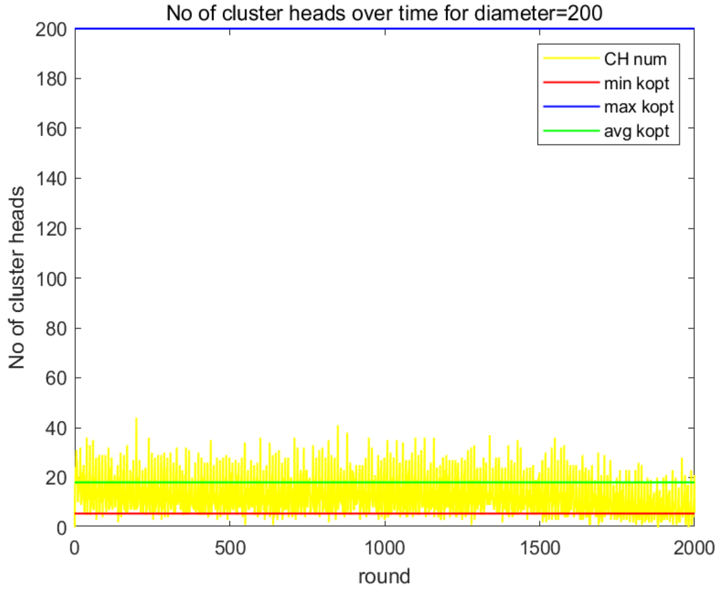


Experiment 3: BS inside network

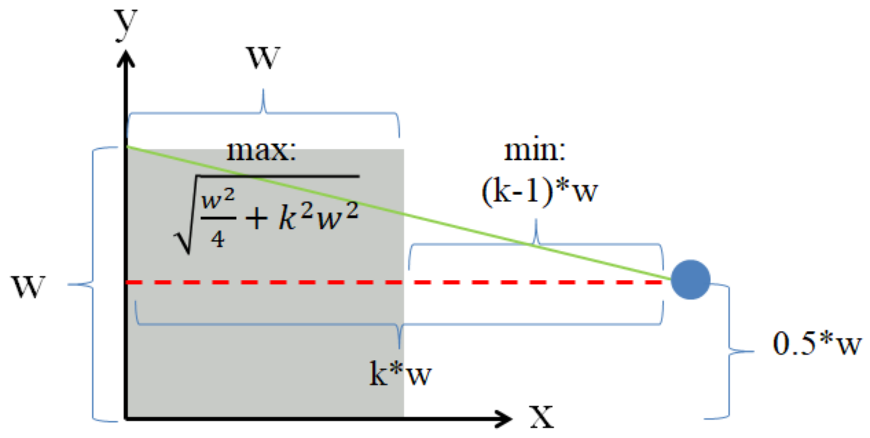








Part 2 – BS Location



Mathematic analysis

$$\begin{cases} min\ d_{toBS}^2 = (k-1)^2 w^2 \\ \max\ d_{toBS}^2 = \dfrac{w^2}{4} + k^2 w^2 \end{cases}$$

$$\begin{cases} \max k_{opt} = \sqrt{\dfrac{N\varepsilon_{fs}}{2\pi\varepsilon_{mp}}} \dfrac{1}{(k-1)^2 w} \\ \min k_{opt} = \sqrt{\dfrac{N\varepsilon_{fs}}{2\pi\varepsilon_{mp}}} \dfrac{1}{(\frac{1}{4}+k^2)w} \end{cases}$$

Considering validity of optimum k range, for further experiments, BS location is set as:

x_BS_location=1.5

y_BS_location=1.5

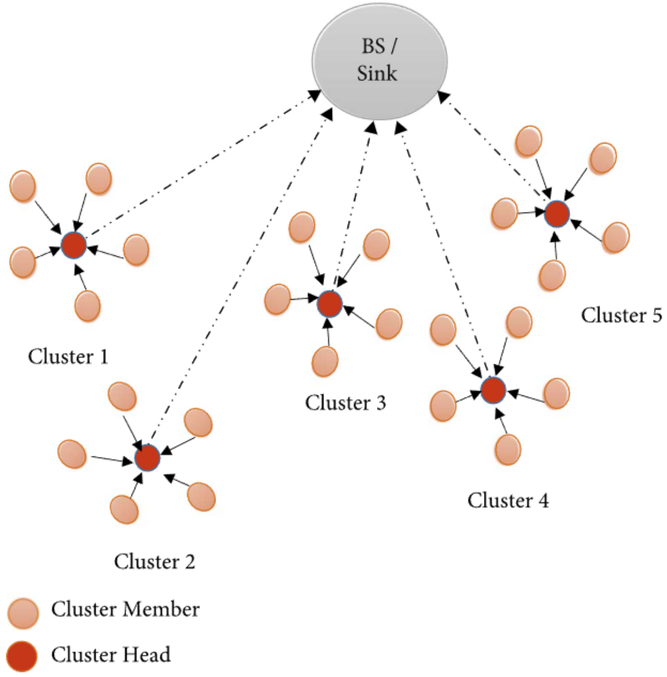
Part 2 – LEACH-C

- Improved method of LEACH
- Cluster head is still chosen stochastically based on the equation.
- Where each node generates random number
- Look at the overall network energy
- Find average and only those node's energy more than average can be CHs

$$T(r) = \frac{P}{1 - P * (r \bmod P^{-1})}$$

Where P is the cluser head probabilty

$$\forall nodes \in G$$

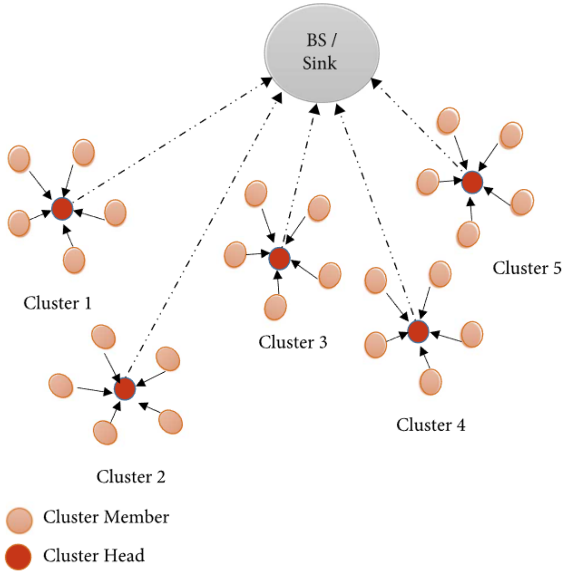


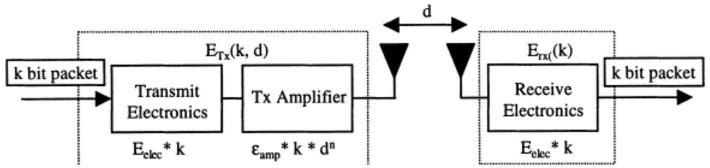
Part 2 – LEACH-C

- Uses Simple radio Model for radio hardware energy dissipation
- Transmitter : Electronics & Power Amp
- Receiver: Electronics
- Channel Model :
  - Free space (d^2)
  - Multipath (d^4)
- Controlled by d0
- Depends on distance between transmitter and receiver
- Network diameter matters

$$T(r) = \frac{P}{1-P*(r \bmod P^{-1})}$$ Where P is the cluser head probabilty

$$\forall nodes \in G$$





$$E_{Tx}(k,d) = E_{Txelec}(k) + E_{Txamp}(k,d)$$

$$E_{Tx}(k,d) = kE_{elec} + k\varepsilon_{fs}d^2 \qquad if\ d < d_0$$

$$E_{Tx}(k,d) = kE_{elec} + k\varepsilon_{mp}d^4 \qquad if\ d \geq d_0$$

$$d_0 = sqrt(\frac{\varepsilon_{fs}}{\varepsilon_{mp}})$$

Part 2 – LEACH-C

- Cluster head formation :
  - Check if it is alive
  - Check if it is in set G
  - Check random number
  - Check for remaining energy
  - Cluster head formation
- Cluster formation:
  - Sum of square distance as distance metrics
- Shortest distance = lower energy dissipation
- EDA: Consider for data fusion

```
if(S(i).E>0)
    temp_rand=rand;
    if ( (S(i).G)<=0)
        if (temp_rand<= (p/(1-p*mod(r,round(1/p)))))
            if (S(i).E>Eavg) % should be a CH
                countCHs=countCHs+1;
                packets_TO_BS=packets_TO_BS+1;
                PACKETS_TO_BS(r+1)=packets_TO_BS; %independent variable
                S(i).type='C';
                S(i).G=round(1/p)-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;
```

Cluster head formation

```
for i=1:1:n
    if ( S(i).type=='N' && S(i).E>0 ) % if it is a node OR alive
        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
```

Cluster formation

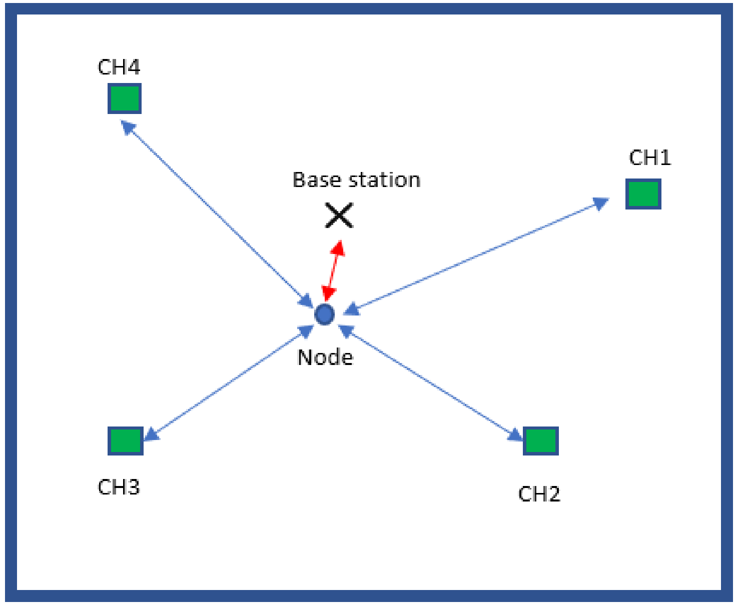
```
% Because CH is the intermediate node to the BS
    % Data received from other nodes also will have
    % energy dissapation.
    % Data Aggregation etc
S(C(min_dis_cluster).id).E = S(C(min_dis_cluster).id).E- ( (ERX + EDA)*pkt_size );
packets_TO_CH=packets_TO_CH+1;
```

$E_{Tx}(k,d)=kE_{elec}.$

Receiving Data packet

Part 2 – LEACH-C

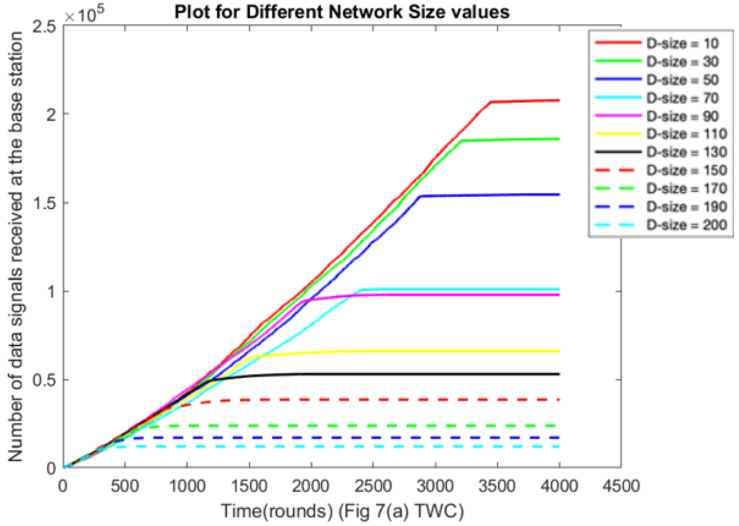
- Cluster head as intermediate node and data fusion
- Compacts and summaries data for the base station
- Reduce complexity and computational load for BS
- Larger network CHs is good
  - Provide fully connected network
  - Shortest path
- Nodes closer to BS, send it directly don't use CH, wasting energy dissipation

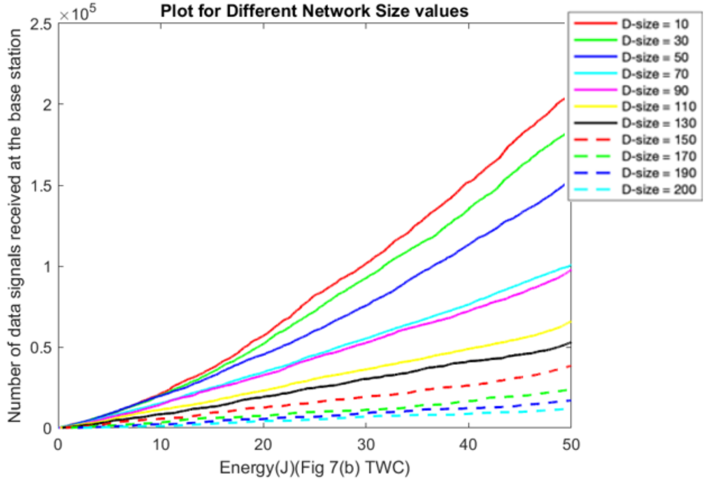


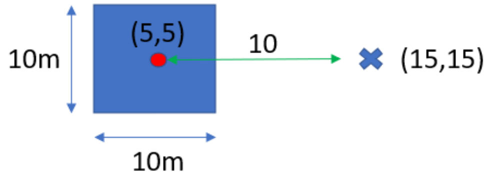

```
else % Closer to the BS than all other cluster head. Send to BS straight
    min_dis;
    if (min_dis>do)
        S(i).E=S(i).E- ( ETX*(pkt_size) + Emp*pkt_size*( min_dis * min_dis * min_dis * min_dis));
    end
    if (min_dis<=do)
        S(i).E=S(i).E- ( ETX*(pkt_size) + Efs*pkt_size*( min_dis * min_dis));
    end
    packets_TO_BS=packets_TO_BS+1;    % send pkt to BS
```

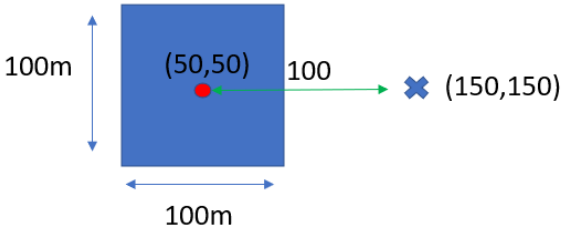
Part 2 – LEACH-C

- Data delivery to the BS is inversely proportional to the BS
- Base station in smaller network receives more data packet
- Main reason : Distance
- Lives longer and more energy efficient
- An increase in diameter equates to exponential increase for energy dissipation
- Larger network model likely to use multipath modelling









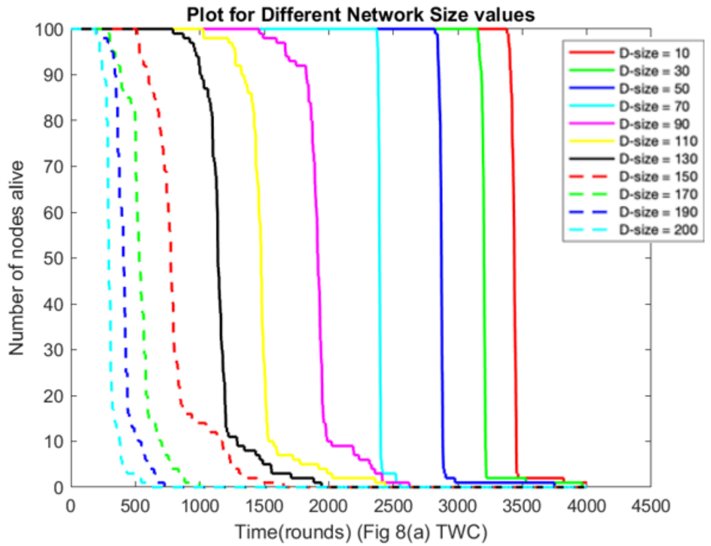
$$E_{Tx}(k,d) = E_{Txelec}(k) + E_{Txamp}(k,d)$$

$$E_{Tx}(k,d) = kE_{elec} + k\varepsilon_{fs}d^2 \qquad if\ d < d_0$$

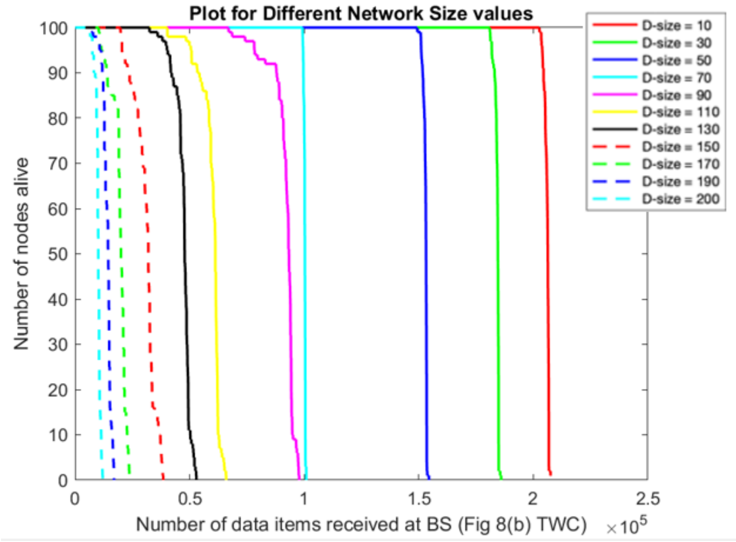
$$E_{Tx}(k,d) = kE_{elec} + k\varepsilon_{mp}d^4 \qquad if\ d \geq d_0$$

Part 2 – LEACH-C

- Larger network:
  - More dead nodes
  - Lesser data packets
- BS for larger network has lesser computational and complexity load

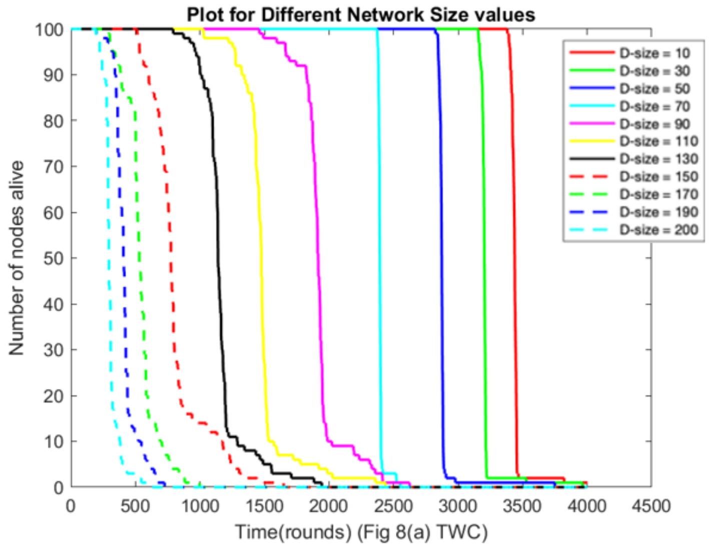


Leach C

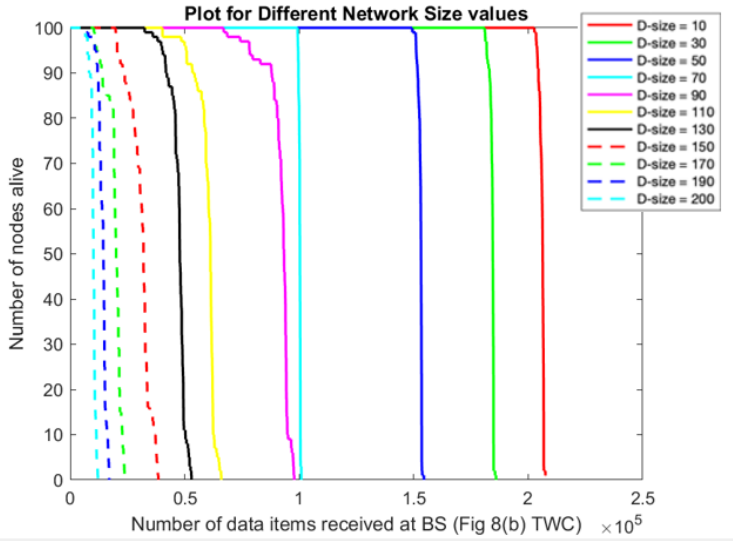


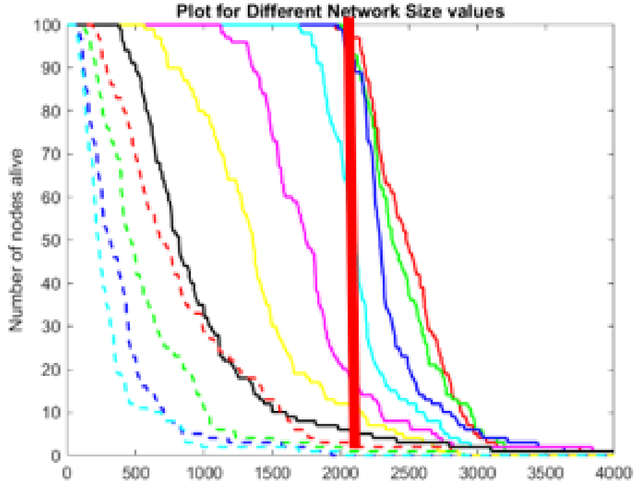
Part 2 – LEACH-C

- Larger network:
  - More dead nodes
  - Lesser data packets
- BS for larger network has lesser computational and complexity load
- Life time defination: looking at the first death
- Leach vs Leach C
  - First death occurs at much later round than Leach
  - Dies together
  - Evenly distributed



Leach C

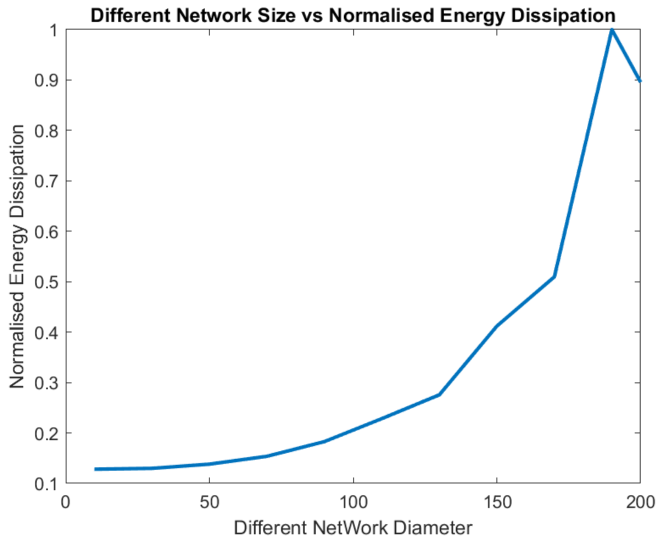




Leach

Part 2 – LEACH-C

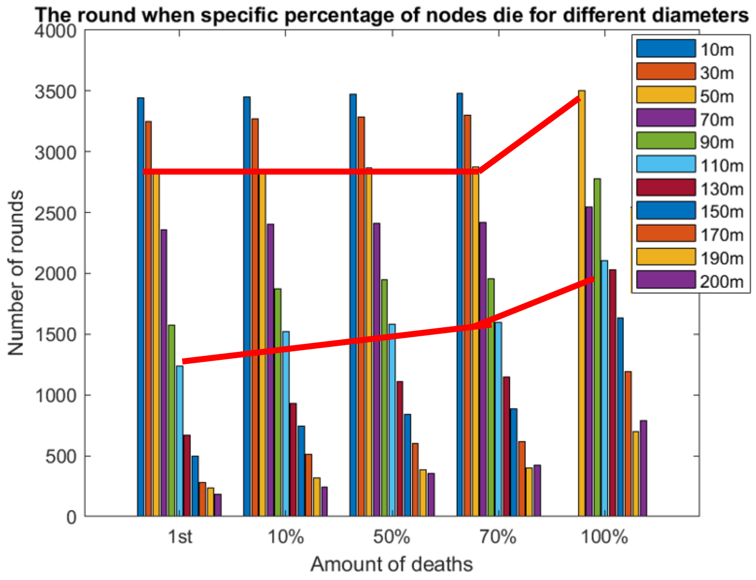
- Reinforce that larger network incurs larger energy dissipation
- Started off gentle and becoming steeper as network size increase
- Due fading and multipath model for energy dissipation



$$E_{Tx}(k,d) = E_{Txelec}(k) + E_{Txamp}(k,d)$$

$$E_{Tx}(k,d) = kE_{elec} + k\varepsilon_{fs}d^2 \qquad if\ d < d_0$$

$$E_{Tx}(k,d) = kE_{elec} + k\varepsilon_{mp}d^4 \qquad if\ d \geq d_0$$



Part 2 – BCDCP

- Improvement from LEACH and LEACH-C
- Uniform placement of cluster heads to minimize distance of packet transmission
- Performs balanced clustering where every cluster have roughly the same nodes
- Using CH-to-CH routing to save packet transmission distance further
- Fuses data packets to increase data compression

A Centralized Energy-Efficient Routing Protocol for Wireless Sensor Networks

SIVA D. MURUGANATHAN, DANIEL C. F. MA, ROLLY I. BHASIN, AND ABRAHAM O. FAPOJUWO, UNIVERSITY OF CALGARY

creation.

Performance of the proposed BCDCP protocol is assessed by simulation and compared to other clustering-based protocols (LEACH, LEACH-C, and PEGASIS). The simulation results show that BCDCP outperforms its comparatives by uniformly placing cluster heads throughout the whole sensor field, performing balanced clustering, and using a CH-to-CH routing scheme to transfer fused data to the base station. It is also observed that the performance gain of BCDCP over its counterparts increases with the area of the sensor field. Therefore, it is concluded that BCDCP provides an energy-efficient routing scheme suitable for a vast range of sensing applications.

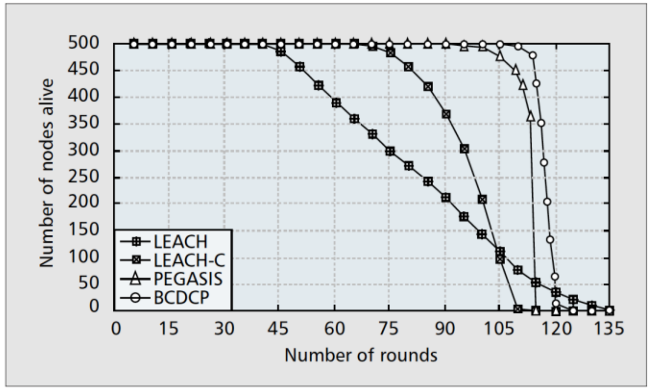


FIGURE 3. *A comparison of BCDCP's system lifetime with other clustering-based protocols.*

Part 2 – BCDCP

CH Candidate Selection:

- Creates a set of cluster head candidates with nodes that have E > Eavg
- Store these nodes in temp_S
- Note that addition of tolerance is to reduce rounding errors in matlab
- Similar to LEACH-C

```
% 1) Select nodes that are above average energy
% Get the average energy of all nodes
E_sum = 0;
for i=1:1:n
    if S(i).E > 0
        E_sum = E_sum + S(i).E;
    end
end
E_avg = E_sum / n;
STATISTICS.Total_energy(r)=E_sum;
STATISTICS.Avg_energy(r)=E_avg;

% Note down the nodes that are above average energy
temp_S = [];
for i=1:1:n
    if S(i).E > 0 && S(i).E - E_avg > - E_avg * 0.000000001 % multiply by tolerance
        temp_S = [temp_S S(i)];
    end
end
```

Filtering of nodes

Part 2 – BCDCP

Cluster creation

- Largest cluster split into two smaller balanced clusters
- Repeat until there are *Nch* clusters

```
% Balanced Iterative Clustering
clusters = {temp_S};
cluster_heads = {randsample(temp_S,1)};
nch = p * n;
m_size = n / nch;
while length(clusters) < nch
    % select largest cluster and split it to two
    j = 1;
    size = 0;
    for k=1:1:length(clusters)
        if length(clusters{k}) > size
            j = k;
            size = length(clusters{k});
        end
    end

    if size < 2
        m_size = size;
        break
    end

    [c1, ch1, c2, ch2] = twoClustering(clusters{j}, DM);
    clusters{j} = c1;
    cluster_heads{j} = ch1;
    clusters = [clusters {c2}];
    cluster_heads = [cluster_heads {ch2}];
end
```

Creation of clusters

- **Step 1**: From the set $\mathbb{S}$ which contains all the nodes that are eligible to become cluster heads, choose two nodes, s_1 and s_2, that have the maximum separation distance.
- **Step 2**: Group each of the remaining nodes in the current cluster with either s_1 or s_2, whichever is closest.
- **Step 3**: Balance the two groups so that they have approximately the same number of nodes; this forms the two sub-clusters.
- **Step 4**: Split $\mathbb{S}$ into smaller sets $\mathbb{S}_1$ and $\mathbb{S}_2$ according to the subcluster groupings performed in step 3.

Part 2 – BCDCP

Cluster creation

- Largest cluster split into two smaller balanced clusters
- Repeat until there are *Nch* clusters
- Two furthest nodes in the cluster to be cluster heads
- Split the nodes based on distances to the cluster heads put in *c1* and *c2*

```
function [c1,ch1,c2,ch2] = twoClustering(c, DM)

    dist = -1;
    ch1 = -1;
    ch2 = -1;
    % find largest distance
    for i=1:1:length(c)-1
        for k=i+1:1:length(c)

            dist_temp = DM(i,k);
            if dist_temp > dist
                dist = dist_temp;
                ch1 = c(i);
                ch2 = c(k);
            end
        end
    end

    c1 = [];
    c2 = [];
```

Spatial selection of two CHs

- **Step 1**: From the set $\mathbb{S}$ which contains all the nodes that are eligible to become cluster heads, choose two nodes, s_1 and s_2, that have the maximum separation distance.
- **Step 2**: Group each of the remaining nodes in the current cluster with either s_1 or s_2, whichever is closest.

```
% group nodes to the two clusters
for i=1:1:length(c)
    d_ch1 = DM(c(i).id, ch1.id);
    d_ch2 = DM(c(i).id, ch2.id);

    if d_ch1 < d_ch2
        c1 = [c(i) c1];

    else
        c2 = [c(i) c2];

    end
end
```

Cluster splitting

Part 2 – BCDCP

Cluster creation

- Largest cluster split into two smaller balanced clusters
- Repeat until there are *Nch* clusters
- Two furthest nodes in the cluster to be cluster heads
- Split the nodes based on distances to the cluster heads put in *c1* and *c2*
- Equalize number of nodes in the cluster
- Result is two clusters, *ch1* & *ch2*

```
% fix imbalances
diff = length(c1) - length(c2);
if diff ~= 0

    % if |c1| > |c2| find nodes in c1 closest to c2 then move them
    if diff > 0
        tomove = floor(diff / 2);
        dists = [];
        for i=1:1:length(c1)

            dists = [dists, DM(c1(i).id, ch2.id)];
        end
        dists;
        [X,idx] = sort(dists);

        c2 = [c1(idx(1:tomove)) c2];
        c1 = c1(idx(tomove+1:length(idx)));
```

- **Step 3**: Balance the two groups so that they have approximately the same number of nodes; this forms the two sub-clusters.

```
% if |c1| < |c2| find nodes in c2 closest to c1 then move them
elseif diff < 0
    tomove = - floor(diff / 2);
    dists = [];
    for i=1:1:length(c2)

        dists = [dists, DM(c2(i).id, ch1.id)];
    end
    dists;
    [X,idx] = sort(dists);

    c1 = [c2(idx(1:tomove)) c1];
    c2 = c2(idx(tomove+1:length(idx)));
end
```

Part 2 – BCDCP

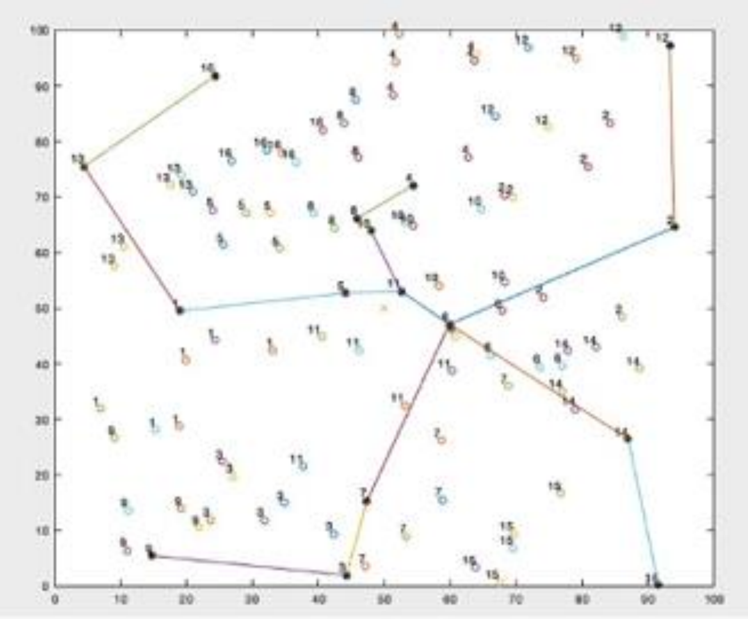
CH-to-CH Path Formation:

- Creating paths from each CH to the base station via CH-to-CH hops
- Uses MST formed by all the CHs.
- Edges have weight equal to distance, nodes are CHs.
- Implemented using MATLAB's graph objects and methods.
- Transfers aggregated data

```
% 3) Get shortest route to Base station for each cluster head

% Convert CHs from points cartesian plane into nodes in a graph
% Convert euclidean distances to weighted edges in a graph
% Convert point indexes to string, as keys to nodes
chs = length(cluster_heads);
s = [];
t = [];
w = zeros(chs*(chs + 1)/2);
edge_count = 1;
for i=1:1:length(cluster_heads)-1
    for k=i+1:1:length(cluster_heads)
        s = [s string(cluster_heads{i}.id)];
        t = [t string(cluster_heads{k}.id)];
        w(edge_count) = DM(cluster_heads{i}.id, cluster_heads{k}.id);
        edge_count = edge_count+1;
    end
end
edge_count = edge_count-1;

% Create a matlab graph object
G = graph(s(1:edge_count), t(1:edge_count), w(1:edge_count));

% Generate the minimum spanning tree
[T, pred] = minspantree(G);
```

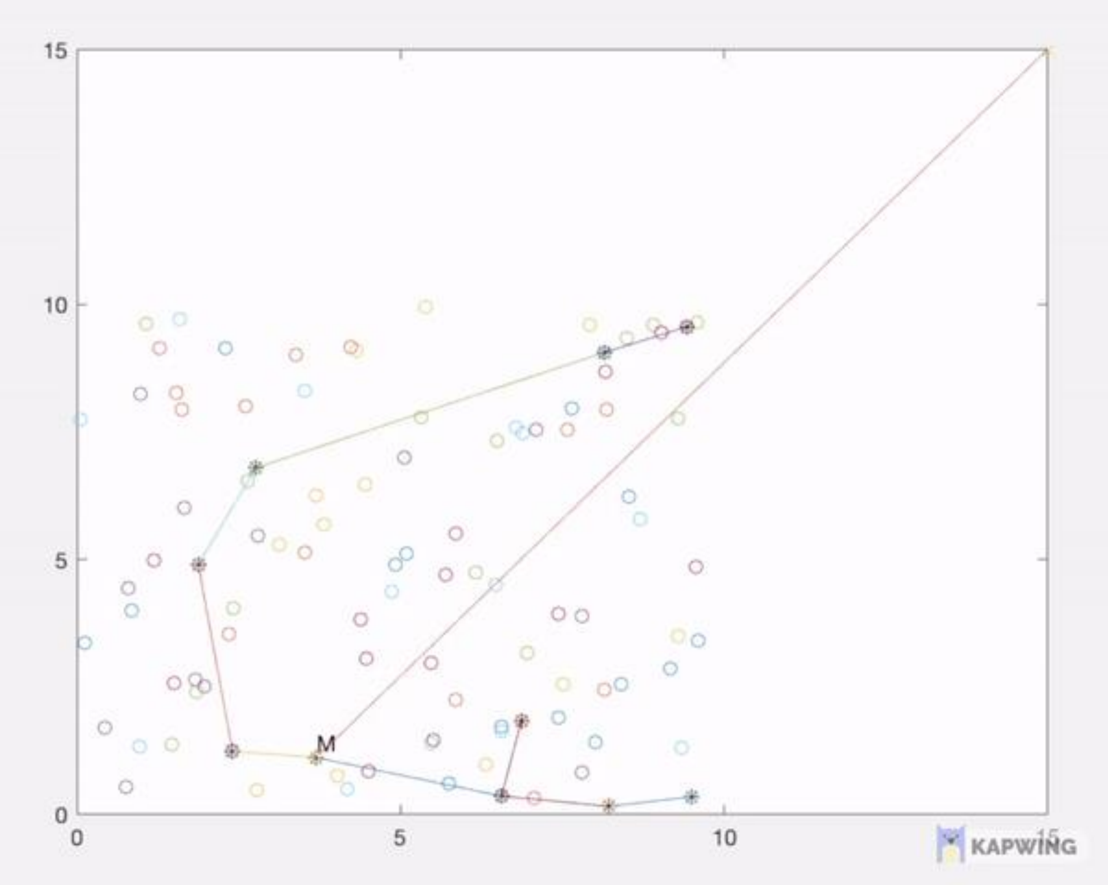
```
% random pick of main cluster head sending to Base Station
main_ch = randsample(cluster_heads,1);
main_ch = main_ch{1}.id;
```

```
% Find the shortest path in the minimum spanning tree
ch_path = shortestpath(T, string(ch), string(main_ch));
```



Part 2 – BCDCP

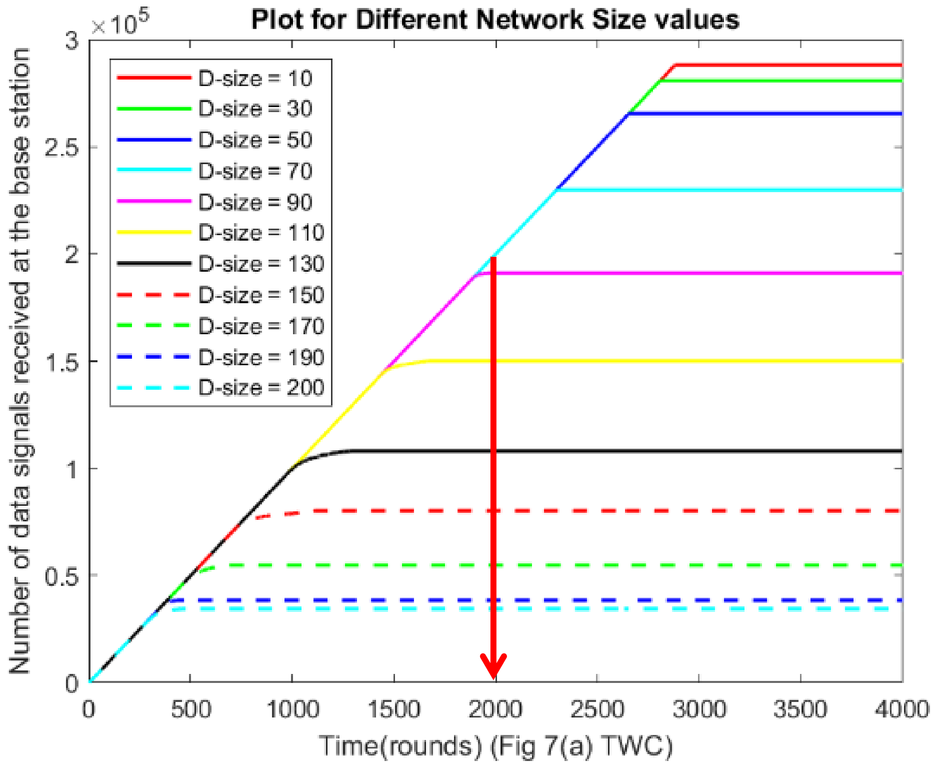
The Cluster Head selection & Routing of BCDCP in action

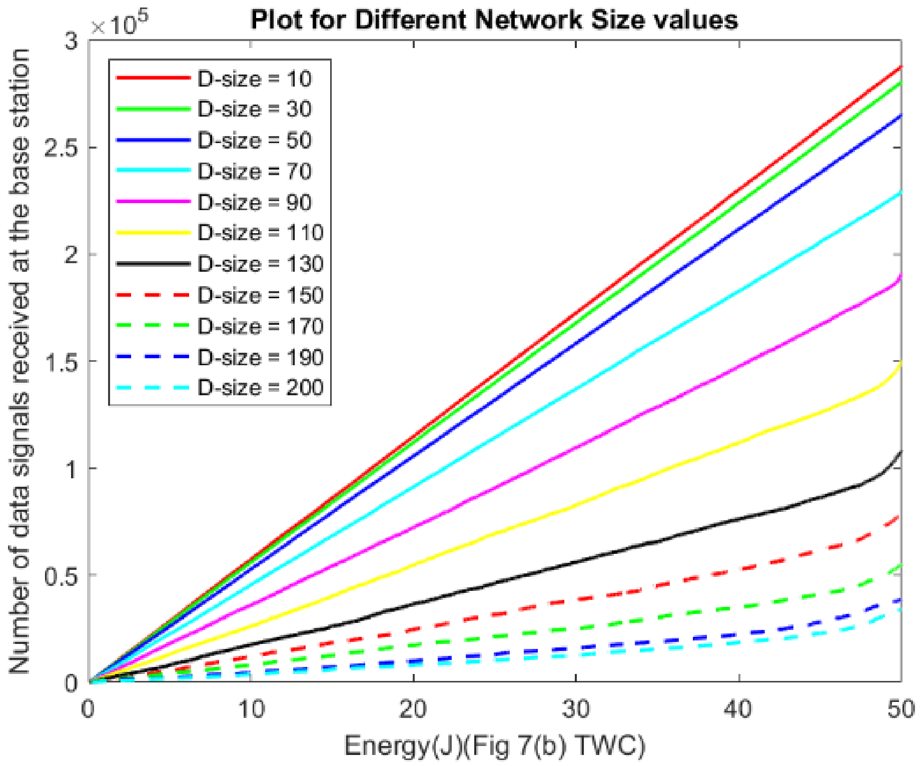


Part 2 – BCDCP Simulation Results

Data delivery inversely proportional to network diameter

Less efficient data delivery in larger network

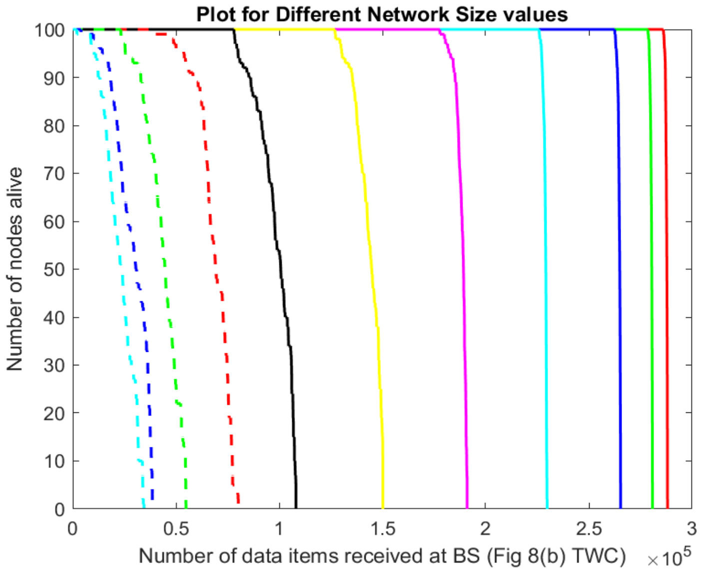


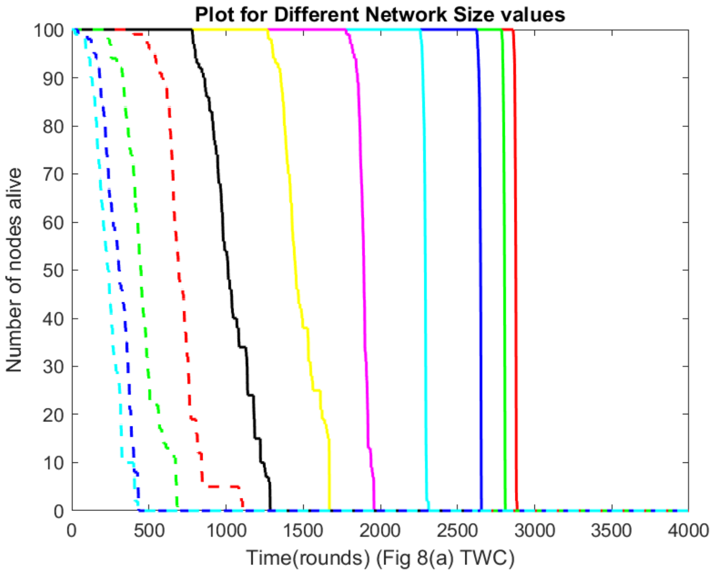


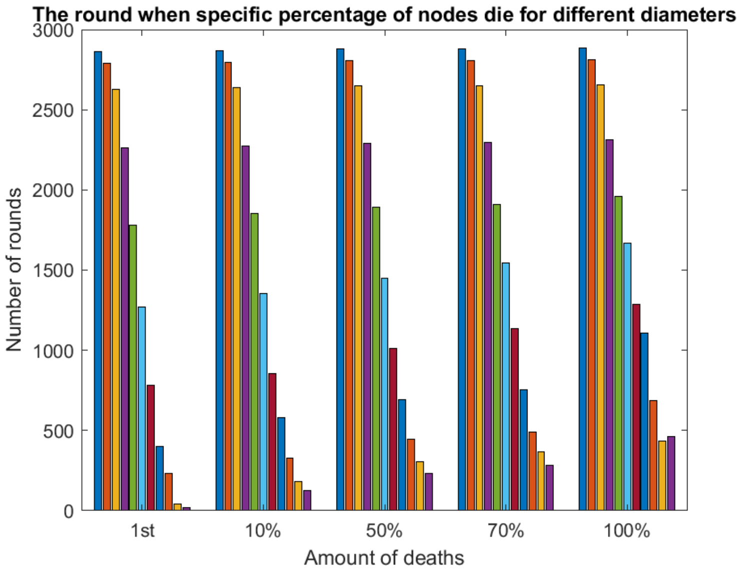
E0=0.5J, n=100, r=4000, BS (1.5xm, 1.5ym)

Part 2 – BCDCP Simulation Results

Network lifetime & total data delivery is lesser for larger networks







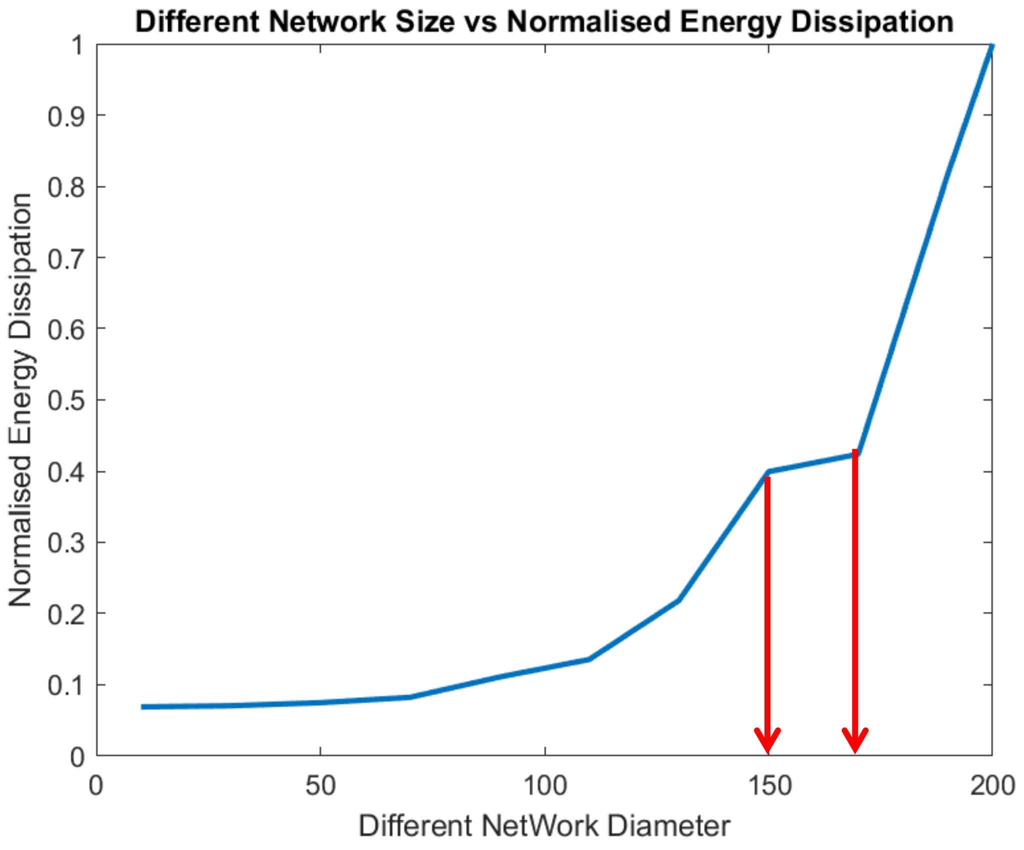
Part 2 – BCDCP Simulation Results

$$E_{Tx}(k,d) = E_{Txelec}(k) + E_{Txamp}(k,d) \qquad \text{- Equation 2}$$

$$E_{Tx}(k,d) = kE_{elec} + k\varepsilon_{fs}d^2 \qquad if\ d < d_0$$

$$E_{Tx}(k,d) = kE_{elec} + k\varepsilon_{mp}d^4 \qquad if\ d \geq d_0$$

$$d_0 = sqrt(\frac{\varepsilon_{fs}}{\varepsilon_{mp}})$$

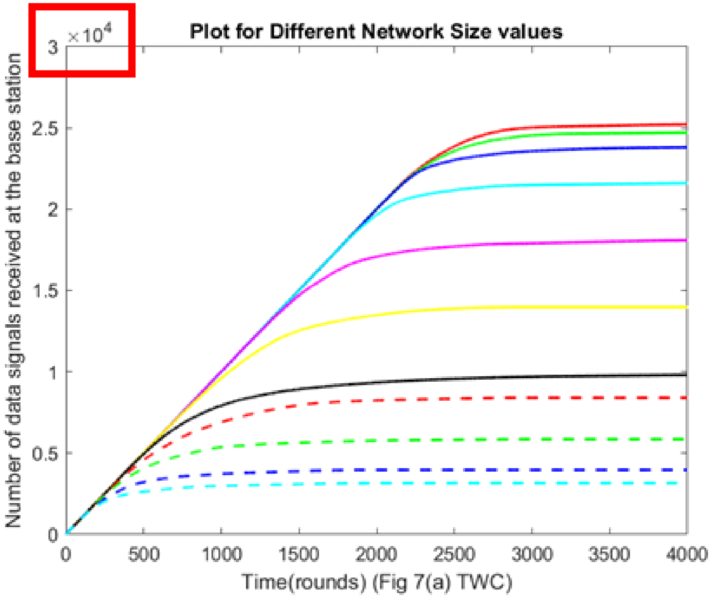


Comparison of All 3 Protocol Performance

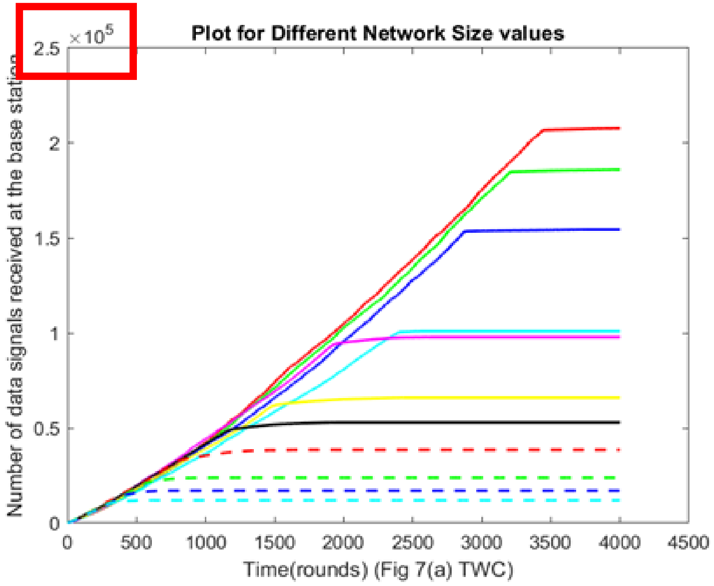
BCDCP & LEACH-C have higher data delivery

	LEACH	LEACH-C	BCDCP
Data magnitude	10^4	10^5	10^5

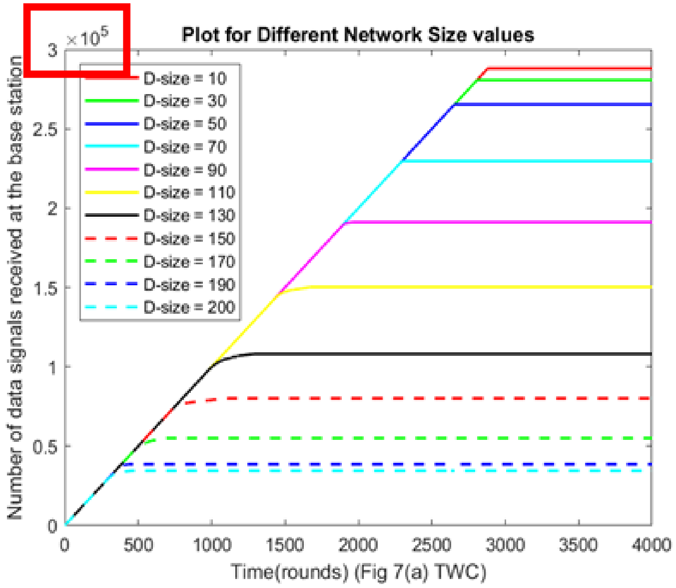
LEACH



LEACH-C



BCDCP

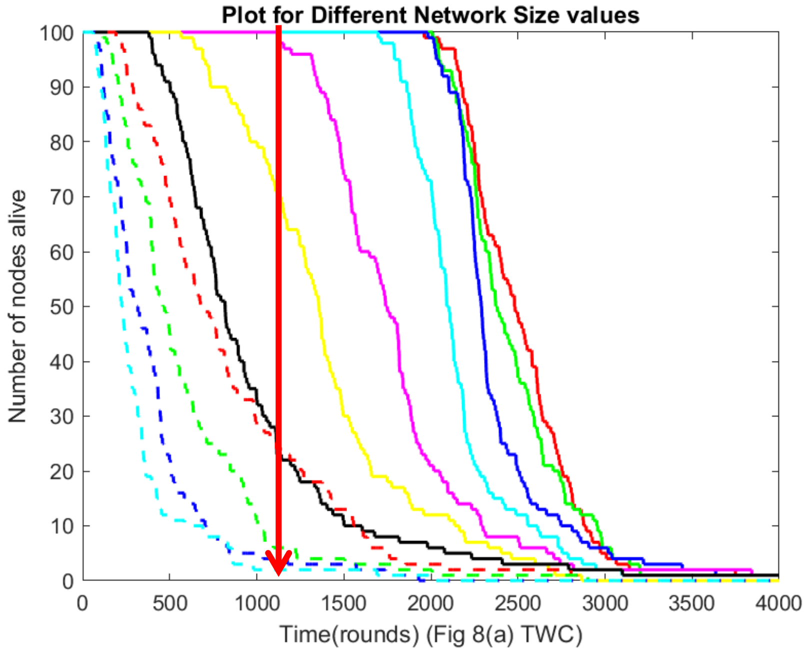


Comparison of All 3 Protocol Performance

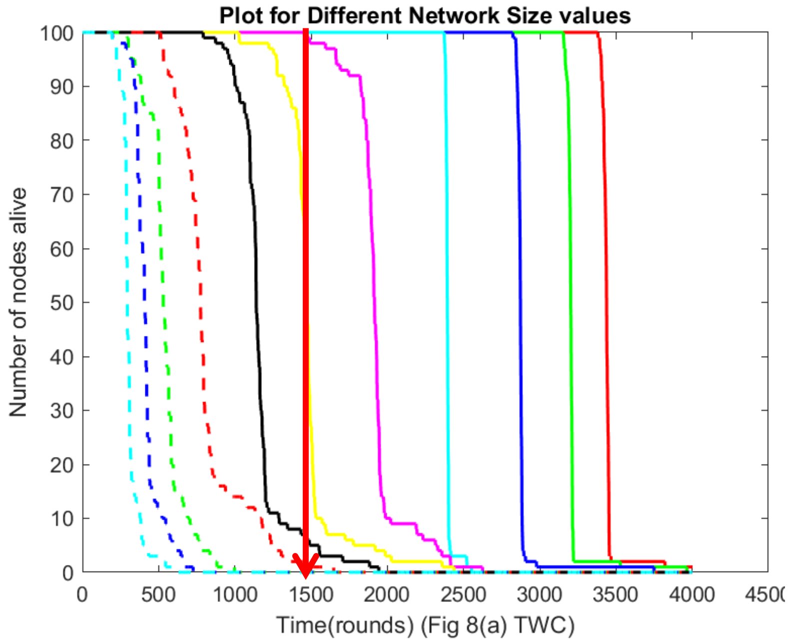
BCDCP & LEACH-C - longer network life but faster death rate

At d=90	LEACH	LEACH-C	BCDCP
First death	1100	1500 (35% ↑ than LEACH)	1750 (60% ↑ than LEACH)

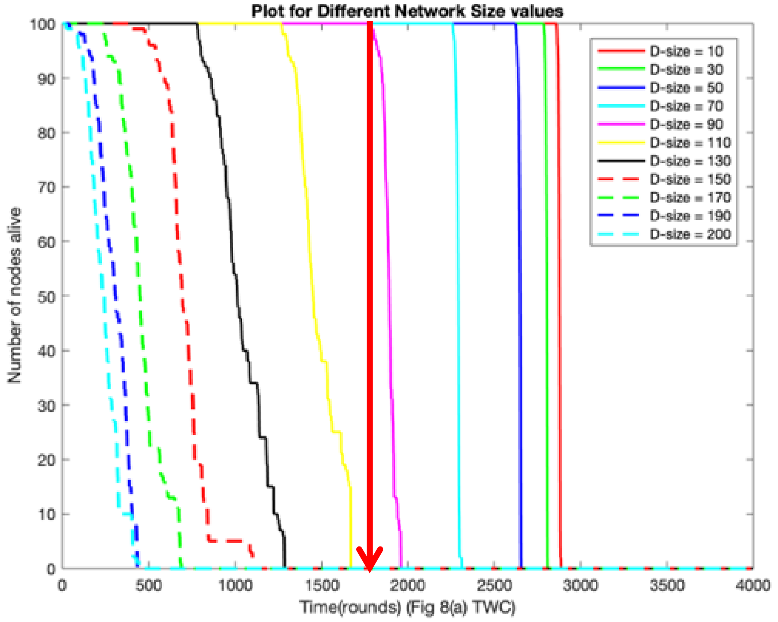
LEACH



LEACH-C



BCDCP



Conclusion

- LEACH-C and BCDCP are more energy-efficient than LEACH by having a better CH-selection based on remaining energy and location of nodes.
- Both BCDCP and LEACH-C do well in distributing the burden of sending messages evenly among all nodes, so that all nodes die at similar times.
- BCDCP is able to extend longevity of network due to its CH-to-CH spanning tree routing and cluster balancing that ensures equal cluster size and spatial distribution of clusters.

Thank You!