Fault Tolerant Clustering Protocol for Data Delivery in Wireless Sensor Networks

Deepali Virmani
HOD (IT), BPIT, GGSIPU
deepalivirmani@gmail.com

Abstract

A wireless sensor network (WSN) typically consists of a large number of small sensor nodes with limited energy. Prolonged network lifetime, scalability, node mobility and load balancing are important requirements for many WSN applications. Wireless Sensor networks have limited energy resources so our main aim is to enhance network lifetime by energy balancing. Dynamic clustering plays an important role in enhancing network lifetime. In this paper we propose an idea of fault tolerant optimal path determination for forwarding data to the base station to enhance the network lifetime and implement fault tolerance at the same time. Simulation results prove the betterment of our proposed method over the existing Leach protocol and DCP.

Keywords: Cluster, Leach Protocol, Network lifetime, Fault tolerance

I. Introduction

Sensor networks are composed of small, battery operated sensors, whose main function is to collect and forward the required data to the base stations. A sensor network is comprised of sensing, processing, communication ability which enables observation and reaction to events and phenomena in a specified environment [13] [12]. Such a network provides a bridge between the real physical and virtual worlds. Networking of tiny sensor nodes makes it easy to obtain the data about physical phenomena which used to be very much difficult with the conventional ways. Wireless sensor network typically consist of hundreds to thousands of nodes. These nodes collect process and cooperatively pass this collected information to an administrator. The administrator typically is a civil, governmental, commercial, or an industrial entity. The WSN’s are a network of nodes that range from hundreds to thousands. Each node is connected to one or several sensors. Sensor nodes are usually resource constrained in terms of energy, processor, memory and bandwidth. The major constraint being limited energy as the sensor nodes are directly dependent on the battery life [1, 11]. Sensor nodes are deployed to gather information and desired that all the nodes works continuously and transmit information as long as possible. This addresses the lifetime problem in wireless sensor networks, so our focus lays on making them live for more amount of time. The base station (administrator) is a master node which is generally fixed and assumed to have unperturbed power supply. The main task of base station is to gather information from various sensor nodes and process it for further dissemination [2].

The sensor nodes are highly resource constrained. Energy is one of the major issues of the sensor nodes. Sensor nodes spend their energy during transmitting the data, receiving and relaying packets. Energy consumption is further increased with the distance, as energy consumption is directly proportional to the square of the distance among the nodes [3]. Hence, designing routing algorithms that maximize the life time until the first battery expires.
is an important consideration. In order to minimize energy consumption and increase lifetime of the network, distance must also be kept under consideration [4, 5]. Scalability being another issue of concern as a sensor consists of hundreds or thousands of nodes [6]. In sensor networks these issues are addressed in cluster based architecture particularly in LEACH [7] and DCP [14]. But Leach supports only static cluster formation where cluster head runs out of energy more quickly due to extra responsibility of data aggregation. Dcp provides dynamic clustering but it does not support optimal path formation and fault tolerance in forwarding data. In this paper we focus on key concern areas with sensor networks like minimizing energy conservation, maximizing network lifetime, real time communication, fault tolerance.

II. Literature Survey

The literature [7] LEACH (Low-Energy Adaptive Clustering Hierarchy) proposed a clustering-based routing protocol that minimizes global energy usage by distributing the load to all the nodes at different points in time. A fixed clustering based approach to prolong the lifetime of sensor networks employing data aggregation is described in literature [4]. The literature [8] RDAG is an approach of Data aggregation by reducing the number of transmissions which is an effective approach to save energy by using the concept of LEACH. The literature [5] and [9] says that energy consumption in a WSN varies with the transmission range. Reducing the transmission range will reduce the power consumed in transmitting a packet toward the neighbours. The literature [14] DCP (Dynamic clustering Protocol) is an approach for dynamic cluster formation method where clusters are refreshed periodically based on residual energy, distance and cost in this sleep and wait technology is also applied to enhance the network lifetime.

The major drawback of LEACH [7] is that it does not support dynamic clustering so to overcome this we use DCP [14] to form clusters. We determine optimal paths for each cluster head to forward its data to the base station. Our proposed method supports fault tolerance in case a conflict occurs or a node goes down and path becomes dead. To best of our knowledge none of the previous work has included all these factors in a single work.

III. Methodology

A. Proposed Fault Tolerant Optimal Path Clustering Protocol (FTOCP)

The proposed FTOCP uses DCP [14] for formation of dynamic clusters. The cluster head behaves as the data aggregating node for each cluster. As soon as nodes forward the data to the cluster head they move to the wait state and remain in the sleep mode until they have something more to transfer [12]. The data when aggregated [8] at cluster head of each cluster is forwarded to the base station The proposed protocol helps in conserving energy by only allowing each cluster head to communicate with other cluster heads to enable formation of an optimal path which will consume the minimum amount of energy to the base station. The optimal path is determined using an assumption that transmission energy consumption will increase greatly as the transmission distance increases. A node which needs to transmit its data sends the data to its neighbour (one of the cluster heads) having the maximum value for a constant determined by using the relation [Energy α (Distance)²]. The corresponding cluster head forwards the data to its next neighbour having maximum constant value till it reaches the base station. Only the cluster heads with highest energy levels and nearest to base station are used for transmission and the energy of all the other nodes is conserved for future use. The proposed FTOCP also implements fault tolerance as well that is in case one of the nodes in the optimal path fails the sending node immediately looks for the best available path (next
node) at that instant which does not include the dead node there by tolerating fault in data transfer. This reduces the frequency of transmission failure.

The salient features of FTOCP include:

- Dynamic clustering
- Reducing number of nodes taking part in transmission
- Fault tolerance
- Minimum energy path for forwarding data
- Reduces communication overhead
- Prolonged Lifetime
- Sleep and wait
- Scalability for large number of nodes
- Energy balancing
- Time to time refresh of clusters

IV. Algorithm

 initialise():- Parameters of the node are assigned values here.
 cal_distance() :- Calculates distance of each node from all other nodes and stores them in the distance matrix.
 energy_max() :- Determines the node with max energy from the nodes which are not a part any cluster yet and returns the id of that node.
 status_update() :- Marks the status of the node as active or inactive.

<table>
<thead>
<tr>
<th>Make_cluster():</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Loop1: For i=1 to No_of_Nodes</td>
</tr>
<tr>
<td>2 Do</td>
</tr>
<tr>
<td>3 Emax_id=energy_max()</td>
</tr>
<tr>
<td>4 if(eminax_id!=-1) then</td>
</tr>
<tr>
<td>5 subsink[i]=node[emax_id].id</td>
</tr>
<tr>
<td>6 node[emax_id].ss_flag=1</td>
</tr>
<tr>
<td>7 increment cluster_id</td>
</tr>
<tr>
<td>8 node[emax_id].cluster_no= cluster_id</td>
</tr>
<tr>
<td>9 loop2: For j=1 to No_of_Nodes</td>
</tr>
<tr>
<td>10 Do</td>
</tr>
<tr>
<td>11 if(node[j].cluster_no==0) then</td>
</tr>
<tr>
<td>12 if(distance[emax_id][j]&lt;=range) then</td>
</tr>
<tr>
<td>13 node[j].cluster_no=cluster_id</td>
</tr>
<tr>
<td>14 end loop2</td>
</tr>
<tr>
<td>15 end loop1</td>
</tr>
</tbody>
</table>
In this algorithm the node with the highest energy level looks for nodes within its transmission range to form a cluster and appoint itself as the cluster head of the cluster formed. The process goes on till all the nodes are not covered under some or the other cluster.

```c
int minimum(int a[], int m[], int n)
{
    int min = 9999;
    int t = 0;
    Loop: For i = 1 to n
        Do
            if (m[i] != 1) then
                if (m[i] >= a[i]) then
                    min = a[i];
                    t = i;
                end Loop
        End Do
    End Loop
    return t;
}
```

In this algorithm, the minimum cost is determined and node with minimum cost is returned.

```c
void printpath(int s, int i, int p[])
{
    if (i == s)
    {
        print BS_node;
        trace[head][++pathnode] = BS_node;
    } else if (p[i] == 0)
    {
        print "no path from BS_node to cluster head[i-1]";
    } else
    {
        printpath(s, p[i], p);
        print cluster head[i-1];
        trace[head][++pathnode] = cluster head[i-1];
    }
}
```

In this algorithm, the path from base station to the cluster head is printed and also stored inside the trace matrix.
path_maker()

1    head=0,pathnode=0,counter=0, num_of_vertices= no_of_clusters +1;
2    Loop1: For i=0 to (num_of_vertices -1)
3        Loop2: For i=0 to (num_of_vertices -1)
4            Do
5            distance_clusterheads[i][j]=distance between cluster head[i] and
6                cluster head[j];
7            end Loop2
8    end Loop1
9    Loop: For i=0 to (num_of_vertices -1)
10       Do
11       distance_BS[i]=distance of cluster head[i] to base station
12    end Loop
13    Loop1: For i=0 to (num_of_vertices -1)
14       Loop2: For i=0 to (num_of_vertices -1)
15           Do
16           if(distance_cluster heads[i][j]<=transmission_range) then
17             adjacency[i][j]=1;
18             graph[i+1][j+1]= distance_BS[j];
19           else
20             adjacency [i][j]=0;
21           end Loop2
22    end Loop1
23    source= no_of_clusters +1;
24    Initialize matrices mark[ ]=0, pathestimate[ ]=9999, predecessor[ ]=0;
25    pathestimate[source]=0;
26    while(counter<num_of_vertices)
27       Do
28       u=minimum(pathestimate,mark,num_of_vertices);
29       mark[u]=1;
30       Loop: For i=1 to num_of_vertices
31          Do
32             if(graph[u][i]>0) then
33                if(mark[i]!=1) then
34                   if(pathestimate[i]>pathestimate[u]+graph[u][i]) then
35                      pathestimate[i]=pathestimate[u]+graph[u][i];
36                      predecessor[i]=u;
37                   end Loop
38             end while
39       Loop: For i=1 to num_of_vertices
40          Do
41             ++head, pathnode=0;
42          printpath(source,i,predecessor);
In this algorithm, the path from cluster heads to the base station is formed. The cluster head send its data to another cluster head which is within its transmission range and as well as nearest to the base station. This process is repeated till the path to base station is completed.

```plaintext
FTOCP():
1    while true,
2       do
3          while(cluster head[n]!=0)
4             do
5                if(energy_val[cluster head[n]-1]<=0) then
6                   print NETWORK DEAD
7                   return false
8                decrement energy_val[cluster head[n]-1] by 10 units
9                increment n
10           end while
11           loop1: for time=1 to REFRESH_TIME
12              do
13                 wait for 1 time unit
14                 update_status();
15                 loop2: for i=1 to No_of_Nodes
16                    do
17                      if(node[i].energy<=0) then
18                          print NETWORK DEAD
19                          return false
20                      if(node[i].ss_flag!=1) then
21                          if(node[i].active==1) then
22                              decrement energy_val[i] by 2 units
23                              active_status[i]=0;
24                              node[i].active=0;
25                          else
26                              decrement energy_val[i] by 1 unit
27                          end loop2
28                      end loop1
29                 cluster_id=0;
30                 initialise();
31                 cal_distance();
32                 make_cluster();
33                 path_maker();
34                 redisplay output
35             end while
36        end while
```

In this algorithm transmission of data takes place from all the other nodes to the cluster head. Cluster head behaves as the data aggregating node for that particular time interval. As soon as nodes forward the data to the cluster head they move to the wait state and remain in the sleep mode until they have something more to transfer. The cluster heads then transfer the data to the base station through an optimum path. After some interval of time called refresh time clusters are reformed and the process repeats.

V. Working Scenario

A. Optimal Path Formation
Cluster heads: filled circles ●
Nodes: empty circle ○
Optimal Path: dark line ———
The different colours represent the different clusters formed.

Figure 1. Cluster Representation

Figure 2.1. Initial Configuration

Figure 2.1 Shows initial configuration that is 10 cluster heads namely A, B . . . J. There is a base station BS. We need an optimum path between every cluster head and base station.
Figure 2.2. All Possible Links between Nodes

Figure 2.2 shows all possible links between the nodes. Each node has a fixed transmission range up to which it can send the data. So, there can a link between two nodes if are in each other’s transmission range.

Figure 2.3. Optimal Neighbour Selection

Figure 2.3 shows that in finding path from A to BS, A has links to two nodes B and D. It will select that node which consumes minimum energy. Since B consumes less energy than D, it selects B.

Figure 2.4. Optimal Path Formation

Figure 2.4 shows that in finding path from A to BS, A selects B. Similarly from B, node consuming minimum energy is selected i.e., node C. The process is repeated until we reach the final node i.e. BS and optimal path is formed.
Figure 2.5. Final Configuration

Figure 2.5 Shows final configuration that is 10 cluster heads namely A, B . . . J. Each connected to the BS through an optimal path that consumes minimum energy.

B. Fault tolerance

Dead Path:

Figure 3.1. Nodes Going Down

Figure 3.1. shows node B, E and C going down and hence the paths A-B, B-C, B-E become dead.

Figure 3.2. Secondary Neighbor Selection

Figure 3.2 shows that in finding path from A to BS, since the optimal path (A-B) is dead. It looks for the secondary neighbour from A i.e., node D and selects it.
Figure 3.3. Fault Tolerance

Figure 3.3 shows implementation of fault tolerance in finding path from A to BS, since the optimal path (A-B) is dead. It looks for the secondary neighbour from A i.e., node D and selects it. Now from D, it will look for the optimum path to reach the base station.

Figure 4.1. Optimal Path Formation along with Clustering

Figure 4.1 represents the formation of clusters along with best path available for each cluster head to forward its data to the base station via other cluster heads. This path always consumes minimum amount of energy for forwarding data.

Figure 4.2. New Clusters and Path after Refresh Time

Figure 4.2 represents the nodes and path to the base station for each cluster head after refreshing time when new clusters are formed and new cluster heads are appointed.

VI. Simulation Results

We simulated and evaluated the performance of the proposed protocol with the existing Leach. All the simulations are done in MATLAB in order to validate the efficiency of proposed protocol.
Table 1. Simulation Parameters

<table>
<thead>
<tr>
<th>No.</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Simulation Area</td>
<td>1000x1000</td>
</tr>
<tr>
<td>2.</td>
<td>No. of nodes</td>
<td>450</td>
</tr>
<tr>
<td>3.</td>
<td>Radio Propagation Model</td>
<td>Two way ground</td>
</tr>
<tr>
<td>4.</td>
<td>Channel Type</td>
<td>Wireless Channel</td>
</tr>
<tr>
<td>5.</td>
<td>Antenna Model</td>
<td>Antenna/Omni antenna</td>
</tr>
<tr>
<td>6.</td>
<td>Energy Model</td>
<td>Battery</td>
</tr>
<tr>
<td>7.</td>
<td>Round Duration Time</td>
<td>10s</td>
</tr>
<tr>
<td>8.</td>
<td>Initial Energy of each node</td>
<td>0.5J</td>
</tr>
</tbody>
</table>

Figure 5.1. Total Energy Dissipated in EDBC, DCP and FTOCP

Figure 5.1 shows the total energy consumption of the network over simulation time. The simulations prove FTOCP to be better than EDBC and DCP in terms of energy consumption. At the start of simulation, performance of all are comparable but as the simulation time proceeds FTOCP conserves a lot of energy due to its sleep wait and fault tolerant technology. As only the active nodes consume energy and nodes after forwarding data to cluster head go to sleep mode.

Figure 5.2. Node Lifetime
Figure 5.2 shows the total number of nodes running in the network over simulation time. The simulations prove FTOCP has lesser number of dead nodes than EDBC and DCP. At the start of simulation number of nodes running is same for all but as the simulation time proceeds node failure is lesser in FTOCP sleep wait and fault tolerant technology.

![Image of a graph showing network node count over time]

**Figure 5.3. Evaluation of Fault Tolerance**

Figure 5.3 shows the available paths of nodes running in the network over simulation time. The simulations prove FTOCP has lesser number of failures than EDBC and DCP. At the start of simulation available paths for all are comparable but as the simulation time proceeds possible available paths decrease slowly as compared to EDBC and DCP due to fault tolerance in FTOCP.

**Table 2. Comparison among Protocols**

<table>
<thead>
<tr>
<th>Evaluation Parameter</th>
<th>LEACH</th>
<th>RDAG</th>
<th>EDBC</th>
<th>DCP</th>
<th>FTOCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clustering</td>
<td>Static</td>
<td>Static</td>
<td>Static</td>
<td>Dynamic</td>
<td>Dynamic</td>
</tr>
<tr>
<td>Energy Efficiency</td>
<td>Very Low</td>
<td>Low</td>
<td>Moderate</td>
<td>High</td>
<td>Highest</td>
</tr>
<tr>
<td>Delay</td>
<td>Least</td>
<td>High</td>
<td>Moderate</td>
<td>High</td>
<td>Highest</td>
</tr>
<tr>
<td>Network Life</td>
<td>Low</td>
<td>Low</td>
<td>High</td>
<td>High</td>
<td>Highest</td>
</tr>
<tr>
<td>Fault Tolerance</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Packet Loss</td>
<td>Maximum</td>
<td>Moderate</td>
<td>Less</td>
<td>High</td>
<td>Minimum</td>
</tr>
</tbody>
</table>

Table 2 shows that performance of our proposed protocols DCP and FTOCP is better than existing protocols in mostly all aspects like energy efficiency, network lifetime, etc. However,
the delay parameter of proposed protocols is on the higher side but its value is kept within the real time delay parameter so that it does not affect the overall performance.

VII. Conclusions

Clustering is a useful topology-management approach to reduce the energy consumption and exploit data aggregation in wireless sensor networks. In this paper we have focused on dynamic clustering where clusters are refreshed periodically and cluster head is selected accordingly. The existing protocols LEACH and RDAG support static clustering and cluster head is fixed in the entire scenario. Cluster head being used as the data aggregator node runs out of energy. So in this paper we have proposed a dynamic clustering protocol (DCP) that supports dynamic clustering with support of sleep and wait technology, where the node that needs to transmit the message is only in wake state after forwarding the message the node changes the state to sleep. By this a lot of energy is conserved enhancing network lifetime indirectly. We have simulated and compared the results in MAT lab of our proposed protocol DCP with the existing LEACH. Simulation results our proposed protocol is better in terms of energy conservation and enhancement of network life time as sleep and wait technology enhances the lifetime of nodes. Delay is increased in our DCP but still it is in real time communication limits so increased delay is not a harmful factor.

References