Advanced Computer Networks (CE6390)
Project on

ENERGY EFFICIENT COMMUNICATION IN WIRELESS SENSOR NETWORKS

Technical Design Document

v1.0

Submitted By,
Iyer Venkatesh Narayanan
Sahith Katukuri
Lokeshwar Kesavan

Preface

Purpose of the Document

To design and implement energy efficient communication application for wireless sensor networks using socket programming.

Overview

A wireless sensor network consists of a set of sensor devices that communicate with each other via wireless links. Building an infrastructure for wireless sensor network that guarantees energy efficient communication is an important problem. Assume that we want to deploy a sensor network in an area to take temperature measurements and collect some statistics such as minimum, average, or maximum temperature in the area. You will be given topology information for a sensor network deployment (see below for the topology file format). One of the sensor nodes in the network will be marked as a sink node. The sink node will issue queries into the network to collect a desired temperature statistic (min/average/max) from the network. One critical requirement in collecting this statistic is to ensure an energy efficient approach to disseminate the query to other sensor nodes and to ensure an energy efficient approach to collect the desired statistic from the nodes in the network. Your main job in this project will be to come up with a design for a spanning tree construction algorithm for efficient dissemination of query messages and to come up with a design for a protocol for energy efficient collection of the requested statistics in the network.

Scope

- Spanning Tree Algorithm design
- Data Collection protocol

Spanning Tree Algorithm Design

We are given the co-ordinates of each node (read from the topology file by the sink node). The Spanning tree is to be constructed rooted at the sink node. The sink node will calculate the distance between each node and compute the spanning tree as follows:

1. Calculate the distance (edge weights) between each node and check whether the calculated distance is greater than the coverage radius of the node.
2. If the distance is greater than the given coverage radius then assign the distance (edge weights) as 0 (zero). This indicates that there is no link between the 2 nodes.
3. Compute the distance (edge weights) between all the nodes given in the topology file and form an adjacency matrix given by 'adj[N][N]' where 'N' is the number of nodes.
4. This adjacency matrix is given to the algorithm to calculate a minimum weight spanning tree.

The algorithm starts with a tree consisting of a single vertex, and continuously increases its size one edge at a time, until it spans all vertices.

- Input: A non-empty connected weighted graph with vertices V and edges E (the weights can be negative).
- Initialize: Vnew = {x}, where x is an arbitrary node (starting point) from V, Enew = {}
- Repeat until Vnew = V:
  - Choose an edge {u, v} with minimal weight such that u is in Vnew and v is not (if there are multiple edges with the same weight, any of them may be picked)
  - Add v to Vnew, and {u, v} to Enew
- Output: Vnew and Enew describe a minimal spanning tree

```
MST (G, w, r)
{
    for each u ∈ G.V
        u.key = ∞
        u.parent = NIL
    r.key = 0
    Q = G.V
    while (Q ≠ ∅)
        u = Extract-Min(Q)
        for each v ∈ G.Adj[u]
            if (v ∈ Q) and w(u,v) < v.key
                v.parent = u
                v.key = w(u,v)
}
```

Data Dissemination & Collection Protocol

After computing the minimum spanning tree, the sink node needs to disseminate the information to each node specifying its parent & child. It also need to send query messages to the nodes and the nodes in the spanning tree needs to re-broadcast this message to its children and so on to pass the query to all the nodes in the network. This can be done using a specific data structure as follows:

```
Data-Structure {
        Integer     configuration-message
        Structure  Message
}
```

Structure Message can be of the following message type depending on the value of the configuration-message

0) Config-Message type (represented by integer 0)
1) Query-Message type (represented by integer 1)
2) Response-Message type (represented by integer 2)

```
Config-Message type {
        Parent    p
        Child      c[]
        Boolean  In-Spanning-Tree
}
```

```
Query-Message type {
        String    query
}
```

```
Response-Message type {
        Double    avg-temp
        Integer    number-of-nodes
}
```

The response message is sent to a node's parent. If the node is a parent then it calculates the average of its children and itself using the following function and then passes this average value along with the number of nodes participating in the average value to its parent and so on till this data reaches the sink node. The function to calculate the average is given as follows:

```
If ( Child[] = NULL )
        avg-temp  = my-temp
        number-of-nodes = 1
        Send data to parent
else
        Collect temperature values from each child
        Calculate-avg-temp (double my-temp) {
                Avg-temp = 0
                For each c in Child[]
                        Avg-temp = avg-temp + c.temp
                        number-of-nodes = number-of-nodes + 1
                Avg-temp = Avg-temp + my-temp
                number-of-nodes = number-of-nodes + 1
                Avg-temp = avg-temp / number-of-nodes
                If ( not sink-node )
                        Send data to parent
                Else
                        Return Avg-Temp
        }
```

Conclusion

Thus by using the proposed minimum weight spanning tree construction algorithm and data dissemination and collection protocol will ensure energy efficient communication between the wireless nodes in the network.