Power Aware Chain Routing Protocol for Data Gathering in Sensor Networks

#Minh-Long Pham [1], Daeyoung Kim [1], Yoonmee Doh [2], Seong-eun Yoo [1]
[1]*Information and Communications University, Korea*
{longpm, kimd, seyoo}@icu.ac.kr
[2]*Electronics and Telecommunications Research Institute*
ydoh@etri.re.kr

Abstract

To prolong the network lifetime, we propose an energy efficient chain-based routing scheme and a distributed algorithm for constructing the routing chain based on the minimum cost tree. The chain construction algorithm calculates the transmission cost based on received signal strength. Therefore, it doesn't require global knowledge of nodes' location information and provides more accurate communication cost calculation among nodes under different practical deployment environments. The proposed power aware mechanism for leader node election in the chain ensures more uniform energy consumption among nodes. The simulation shows the new scheme provides more uniform energy consumption among nodes and better active network lifetime in different network settings as compared to previous chain-based protocols.

1. INTRODUCTION

In recent years, the combined advances in wireless communications, micro-electro-mechanical system (MEMS) technology, and digital electronics have driven the growth of wireless distributed sensor networks and this growth in turn has opened a wide range of applications. Sensor networks consist of a large number of tiny, inexpensive and constrained sensor nodes with sensing, computing and wireless communication capabilities. Since sensor nodes have limited power supply and cannot be easily recharged or replaced when batteries run out, the mechanism for efficient power consumption is very important. Many researches have been carried out for efficient energy consumption of sensor networks at different levels: hardware, operating systems, MAC, routing protocols, and so on.

Among the sources of energy consumption of sensor networks, the energy consumption on RF communication dominates**.** Therefore, routing protocols must be energy efficient to prolong the lifetime of sensor networks. The failure of a few nodes can cause sensor network to consume more energy, and significant topology changes. So a routing protocol also needs to balance the energy consumption of nodes to maximize network lifetime as a whole. Network lifetime can be measured by parameters such as time until certain percentage of nodes die or time until network cannot perform the assigned task. In this work, we consider the network lifetime until the first node dies and the active network operation time until halves of nodes die. When the first node dies, the network topology may need to be changed, and after halves of nodes die the network may not function properly.

Many researches have been carried out for sensor networks but the network protocols for sensor networks still need further extensive and intensive explorations. Most of existing routing protocols focus on one major technical issue whether it is data-centric approach [5] or power efficient approach [1], [2], [3]. In a typical monitoring application, after deploying sensor nodes in the field, users would like to query the data over a specific area for a certain time to monitor the status of environment. Then sensor nodes periodically send data back to the sink. In this kind of application the purpose of routing protocol is to minimize total energy consumption so that the lifetime of sensor networks as the whole can be prolonged as much as possible. Chain–based routing protocols have been proposed to reduce the total energy consumption for data gathering. In [1], PEGASIS uses a greedy algorithm for constructing the routing chain. In [3], authors provide a better algorithm for constructing the energy efficient chain called minimum total energy (MTE) chain. These chain construction algorithms use centralized approaches for constructing the chain and elect the leader node for transmitting data back to the sink by taking turn. However, if the remaining energy of each node is not taken into account in the leader election, the nodes with low remaining energy will easily run out of energy leaving just a small number of survival nodes performing the sensing task. This is also not good from viewpoint of the network lifetime as the whole. We need a more uniform consumption of energy so that sensor nodes can be able to work together most the time and die nearly at the same time.

To support this, we propose a power aware chain (PAC) routing protocol for energy efficient data gathering from the sensor field. The chain is constructed by using a distributed algorithm based on the minimum cost tree. The transmission cost among nodes is calculated using received signal strength thus does not require global knowledge of nodes' location information. This calculation also closely represents the exact transmission cost in different practical deployment environments. Moreover, the proposed power aware mechanism in leader election ensures more uniform energy

consumption among nodes than the previous approaches. Our new chain construction algorithm along with power aware mechanism for leader election provides longer network lifetime and at the same time guarantees the better active network operation time.
The rest of this paper is organized as follows: Section 2 reviews the related work in the literature. In section 3, we describe the details of the proposed scheme. Section 4 discusses the performance evaluation. Section 5 concludes the paper.

2. Related Work

Researches for energy efficient routing protocols for sensor networks have drawn increasing attention. One of the main sources of energy waste in communication is the idle listening, which consumes high energy almost the same as normal receiving mode as observed in [6] and [7]. So one of the most efficient mechanism for power saving is to turn off the radio transceiver whenever it is possible. TDMA based MAC layer protocol eliminates this kind of problems thus can be good candidate for energy efficiency. Therefore, some routing protocols have been developed based on TDMA MAC layer to take advantages of energy conservation.
LEACH [2] is a cluster-based sensor network routing protocol in which nodes are organized into clusters with one node assigned as a cluster head for each cluster. Each node in a cluster is assigned time slot to transmit data to the cluster head. The cluster head then compresses all the received data and transmits it to the sink thus reduces the number of direct transmission to the sink. Since cluster heads consume more energy than normal nodes, nodes take turn to become cluster heads by using probability.
In LEACH, cluster heads still consume high energy because they have to wake up all the time to receive data from all nodes in their clusters.
PEGASIS [1] reduces the total communication energy consumption compared to LEACH. PEGASIS organizes all nodes into the chain using greedy algorithm by adding the next closest node to the chain starting from the node farthest from the sink. It assigns one leader node to transmit data to the sink. Other nodes just transmit data to neighbour node along the chain and aggregate data before continuing sending data along the chain toward the leader node. It achieves better lifetime than LEACH by about 100 to 200 percent.
In [3], the authors present a new centralized algorithm for constructing the minimum total energy (MTE) chain. In each step of chain construction, it searches all remaining nodes and all possible insertion positions in the chain to select a node and a corresponding position in chain that increases the total transmission cost of the chain to the minimum amount. The node is then inserted into the chain at that position. MTE constructs the chain with smaller total transmission energy cost than PEGASIS but has more computation complexity. Also by dividing whole sensor field into four regions and construct the chain in each region, MTE gains better performance compared to PEGASIS in case of sparse-node distribution.
Both PEGASIS and MTE approaches use centralized chain construction. Firstly, their transmission cost calculation based on distance may not reflect the exact cost in different practical environments due to radio irregularity as indicated in [9]. Secondly, these centralized approaches may not scale well for large network or large number of nodes. Moreover, after sometimes nodes far away from the sink easily run out of battery since they consume more energy to transmit to sink as a leader. PAC addresses these issues by constructing the chain using a distributed algorithm. The transmission cost is calculated based on the received signal strength between nodes thus reflects more accurately the actual transmission cost between nodes in the field. By applying distributed algorithm for chain construction, PAC can be applied to larger networks and larger number of sensor nodes. Also, the chain construction bases on signal strength to calculate the transmission cost so that location information is not necessary. Furthermore, the power aware mechanism in leader election guarantees more uniform energy consumption among nodes. So that all nodes work together and die approximately at the same time, which provides better active network operation time than the case where there are only few nodes still function while almost other nodes have died.

3. Power Aware Chain (PAC) Scheme

A. PAC system model

In this work, we consider the wireless sensor network that consists of a sink and a large number of immobile sensor nodes. Sensor nodes are deployed at monitoring area to collect data back to a remote sink, which can be accessed by end users as shown in Figure 1. To save energy sensor node can adjust its transmission power using power control mechanism. All sensor nodes can reach the sink. All the nodes monitor the environment and periodically send sensed data back to the sink.

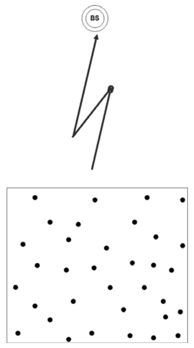

Fig. 1: PAC Sensor Network Model

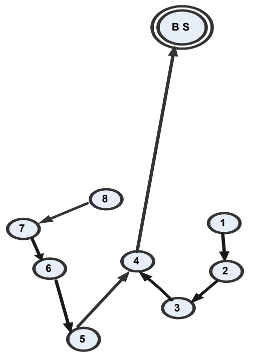

Fig. 2: A Routing Chain

For all the sensor nodes to energy efficiently send data to the sink, we develop PAC, a power aware chain routing protocol. In PAC, all nodes organize themselves into the energy efficient chain with one node elected as leader node to transmit data back to sink on behalf of all other nodes as shown in Figure 2. Each node aggregates received data from the previous node in the chain with its own collected data to produce an aggregated data packet. After nodes are deployed, they will start the chain construction period as discussed in section B. Then nodes collaborate to elect one leader node, which is responsible to collect data from all the nodes and transmit to the sink in the leader node election period. In normal operation, data gathering is divided into iterative rounds. In each round, every node in the chain is assigned a time slot according to its position in the chain to receive and send data. Data collection starts from one end of the chain towards the leader node, and then from the other end towards the leader node, finally the leader node combines the data and send to the sink.

For example, as shown in Figure 2, node 1 sends data to node 2; node 2 combines received data with its own data and forward it to node 3. After the data from node 3 reaches node 4, the same procedure occurs from node 8 through node 7, 6, and 5 towards node 4. Node 4, as a leader, then combines received data with its own data and sends fused data to the sink. After certain number of rounds, the leader node election is performed again to elect new leader node.

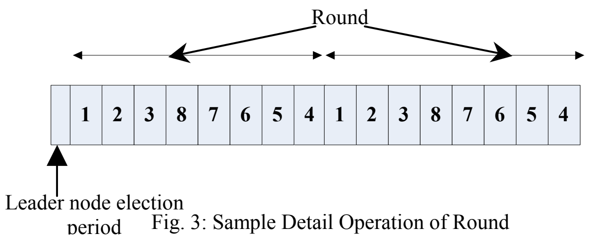

Fig. 3: Sample Detail Operation of Round

B. Chain construction algorithm

The chain construction consists of two steps: first building the Minimum Cost Tree among all nodes and then traverse the tree using depth first search to construct the chain.

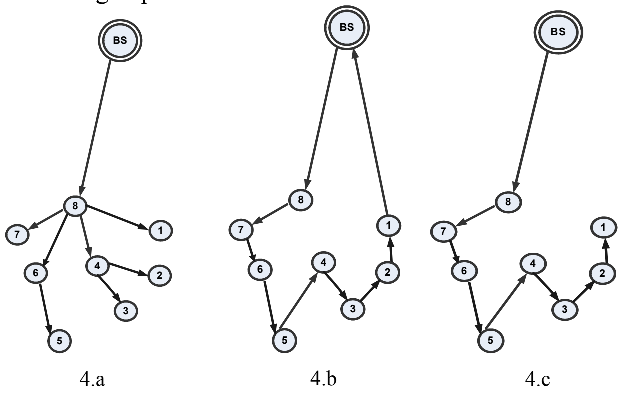

Fig. 4: Chain construction based on Minimum Cost Tree

a. Broadcast to build Minimum Cost Tree
b. Traverse the tree by depth first search
c. Construct the final chain from the tour

The Minimum Cost Tree is constructed as in Algorithm 1. This tree construction algorithm is based on the algorithm for finding minimum cost path from all nodes to the sink [4].

In previous work, each node only keeps the minimum cost to reach the sink. This simplifies the state information stored at each node. However, there may exist several paths towards the sink with the same cost that may cause duplicated data transmission to the sink. Also since the node when sending data does not specify the exact receiver by just broadcasting

Algorithm 1: Constructing the Minimum Cost Tree:

```
1.For each node i: $P_i=\infty$ ; $d_i=\infty$ ; parent=null; childList=null
2. BS broadcasts the initial ADV message (BS, 0)
3. Event: Node B receives an ADV message from node A ($A_{ID}$, $P_A$)
If ($P_B > P_A+P_{AB}$){
        $P_B=P_A+P_{AB}$
        $d_B=\alpha P_{AB}$
        parent =A
}
4. Event: Node B's backoff timer expires
        Sends the JOIN message ($B_{ID}$) to parent A
        Broadcasts ADV message ($B_{ID}$, $P_B$)
5. Event: Node B receives JOIN message ($C_{ID}$) from node C
  Add node C in the children list of B in the decreasing order of the cost $P_{BC}$
 Variables:
        $P_A$, $P_B$, $P_C$, $P_i$: costs of node A, B, C, i to reach the BS
        $P_{AB}$, $P_{BC}$: cost between node A and B, and between B and C
        $\alpha$: backoff coefficient
       $d_B$: delay timer of node B
```

with the minimum cost, this leads to redundant overhearing by nodes that are not involved in the real forwarding. We overcome these problems in our algorithm by using feedback messages to construct the tree explicitly. Thus after construction finish, each node knows the exact relay node (parent node) to forwards data to the sink and list of nodes that it will relay data for (children nodes). Therefore, transmission power is saved by adjusting to just the correct receiver as compared to broadcast message in previous work. This also removes the duplicate transmission to the sink and overhearing of unwanted message.

First each node initialises both the *minimum transmission cost to the sink* (either directly or relay through other nodes, hereafter called *function cost*) and *delay time* to infinite. Sink activates the tree construction by broadcasting the initial advertisement message with the sender as BS and function cost set to zero. When a node receives the advertisement message, it calculates and updates its function cost. If the sender's function cost plus transmission cost between the receiver and the sender is smaller than the receiver's current function cost, the receiver considers the sender as the relay node to send data back to the sink, and the new function cost is updated. The delay time is updated to proportional to the transmission cost between the receiver and sender in case the sender is chosen as the relay node. After the sleep delay, the node wakes up and rebroadcasts the advertisement with its own id as sender's id and the cost as its function cost. It also sends a JOIN message back to its relay node thus considers the relay node as the parent node in the tree. This specifies the exact parent node towards the sink thus eliminates redundant overhearing and duplicate paths.

After all, the tree with the sink as the root is constructed. This algorithm requires each node to send one broadcast message. By this algorithm, the Minimum Cost Tree is constructed with the path from each node to the sink is the minimum cost path. Each node stores the parent node and list of children nodes along with the associated transmission cost to them.

Once the Minimum Cost Tree construction is completed, the distributed depth first traversal is performed to build the chain starting from the sink as root node, as in Algorithm 2.

```
Algorithm 2: Construct the Depth First Search traversal of
Minimum Cost Tree
1. Each node V:
    V.setNodeStatus(NEW)
    For each child node U of V
        V.setEdgeStatus(U, NEW)
2. BS as the root sends out the token to the first child in children list
3. Event: Node U receives a token from node V
If (U.getNodeStatus()==NEW) then{
    U.setNodeStatus(VISITED)
    Extract the id and calculate transmission cost from the token
    and put into PREVIOUS (id, estimated_ cost)
    Send the CONNECT ($U_{ID}$) message to the PREVIOUS node
    Add the node U's id into token packet
    If there is child K of the node U{
        U.setEdgeStatus(K, SENT)
        Send the token packet to K
    } else{ // this is a left node
        U.setNodeStatus(DONE)
        Send the token back to parent node
        }
} else if (U.getNodeStatus()==VISISTED){
        Update the estimated cost in token by adding the
transmission cost between U and V
        If (K is child of U and U.getEdgeStatus(K)==NEW){
                //if there're unvisited child node, visit it
                U.setEdgeStatus(K, VISITED)
                Send token to K
        }else{    // all the child nodes have been visited
                U.setNodeStatus(DONE)
                Send the token back to parent node
        }
}
If (U is BS & U.getNodeStatus()==DONE) Terminate algorithm
4. Event: Node U receives CONNECT($V_{ID}$,estimated cost) packet
from node V
 Put the $V_{ID}$ and the cost for U, V into NEXT[$V_{ID}$, cost]
Send the ACK back to V.
Node V when receives the ACK from U can calculate transmission
cost and update the transmission cost between U and V

Variables
Each node V keeps the variables
NodeStatus={NEW, VISITED, DONE} (initial status is NEW)
EdgeStatus={NEW, VISITED } (initial status is NEW)
NEXT [nodeID, cost], PREVIOUS[nodeID, cost] is the variable for
keeping the next node and previous node in the tour
```

After the tour, the number of nodes in the tour is counted. The longest edge is removed from the tour to form the optimal chain. Sink sends a packet again along the chain to inform all the nodes of the number of nodes, and their orders in the chain. Each node then calculates its corresponding time slot using this information when the order of the leader node is known. For example, from the chain in Figure 2, timeslots are assigned to each node as in Figure 3. The number in each slot is id of the node that will send data within this time slot.

C. Leader node election

Periodically, the *leader node election* is performed to select new leader node in order to balance energy level among nodes. This is carried out by comparing the reserve values among nodes along the chain. The reserve value of node *i*, Ri, is calculated as the ratio between the power available (P_{ai}) of the node and the power needed for transmission from the node to the BS (P_{Txi}): $R_i= P_{ai} / P_{Txi}$. The higher the value means the more total energy available. The direct transmission cost from nodes to the sink is calculated in Minimum Cost Tree construction period, when nodes receive broadcast message from the sink.

Details of leader election are described below:

1. Each node *i* in a chain calculates its own reserve value $R_i=P_{ai}/P_{Tx\,i}$.
2. The node from the end of the chain starts sending its reserve value towards the leader node. Each node receives the packet, compares the current value in packet with its own value. If the value in the packet is higher than its own value, the node simply forwards the packet; otherwise it will modify the packet with its own value and forward the packet.
3. The leader node gets the packet with the highest reserve value, informs nodes in the chain of the new leader node.
4. The leader node election is performed again after a number of rounds.

The number of rounds for re-electing the leader node changes adaptively according to the current energy level of nodes. At beginning, the energy difference between nodes is small and nodes still have high energy. Once elected as a leader node, the node keeps this role for number of rounds. Then it initiates another leader node election period thus reduces the overhead associated with leader election. When energy level of nodes decrease, the number of rounds for re-electing new leader node also decreases thus avoiding one node consuming too much energy as the leader. When energy level of nodes become low, the leader election is taking place every round. This adaptive mechanism ensures that nodes with high energy level and near the sink have more chances to become the leader node. The election of a node near the sink as leader node also reduces the total transmission cost.

D. Fault tolerance

In data gathering phase, if a node detects the next node in the chain is dead, it will try to connect the next node in the chain in the next time slot. For example in the Fig. 4, when the node 7 dies, the node 8 will connect with node 6 in the next time slot thus bypasses the dead node 7. The chain is also be updated when the new leader election is performed to bypass the dead nodes.

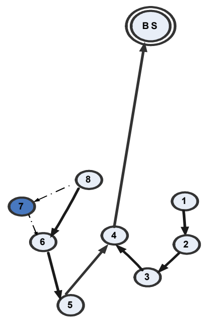

Fig. 5: Fault tolerance when node 7 is dead.

4. PERFORMANCE EVALUATION

A. Radio model and simulation setting

The calculation for communication and computation energy consumption is based on model discussed in [2] and [8]. The transmission energy consumption for k bits over distance d is calculated using both free space model and multipath fading model depending on the distance d between the transmitter and the receiver as follows:

$$E_{Tx}(k,d) = kE_{elec} + k*\varepsilon_{friss\text{-}amp}*d^2 \quad (d<d_{crossover})$$
$$= kE_{elec} + k*\varepsilon_{two\text{-}ray\text{-}amp}*d^4 \quad (d>=d_{crossover})$$

and the energy consumption for receiving k bits can be calculated as $E_{Rx}(k)=k*E_{elec}$, where E_{elec}=50 nJ/bit is the energy consumed for the radio electronics; $\varepsilon_{friss\text{-}amp}$ =10pJ/bit/m^2 and $\varepsilon_{two\text{-}ray\text{-}amp}$=0.0013pJ/bit/m^4 are radio power amplifier parameters; $d_{crossover}$ =87.7m. For the computation energy consumption we only consider the energy consumption for data aggregation E_{fuse}=5nJ/bit/signal, and consider that other processing energy consumption is negligible. A node receiving data packet from the previous neighbour in chain will combine this data with it own data to generate the aggregated data packet of the same size. The initial energy for each node is 0.5J.

Depending on the transmission cost between two nodes, the transmission power of the transmitter is adjusted to minimum level that still be heard by the receiver.

The operation as mentioned above is divided into rounds. In each round, each node sends a number of data packets. The total volume of all data packets is 2000 bits. After a certain number of rounds, the current leader node will initiate the new leader election. Each node will send a control packet of size 3 bytes to re-elect the leader node, which is considered as overhead. At the beginning, leader election is carried out every 20 rounds. After 60% of nodes die, leader node is re-elected every 5 rounds, and when there are 20% of remaining nodes, the leader node election is performed after every round.

The simulation is to further verify that the PAC has better performance than PEGASIS and minimum total energy (MTE) algorithm in terms of number of rounds against percentage of death nodes. The simulation also shows that nodes in PAC die at random position as compared to PEGASIS and MTE where nodes far away from the sink usually die first.

The comparison between PAC, PEGASIS and MTE is based on different network densities, network sizes, and sink positions.

B. Active network operation times

The first type of simulations compares the performance of the PAC protocol versus PEGASIS and MTE protocol in term of number of rounds versus the percentage of death nodes. The comparison is based on the number of operation rounds when the first node, 10%, 20%… 100% of nodes die. The higher the number of rounds when there are few death nodes shows the better performance of the protocol. It's better if large number of nodes alive and die nearly at the same time.

The simulation varies with different network sizes 50mx50m, 100mx100m with different node densities of 30, 50, 100,150, and 200 with base node at the position (25,200) and (50,200).

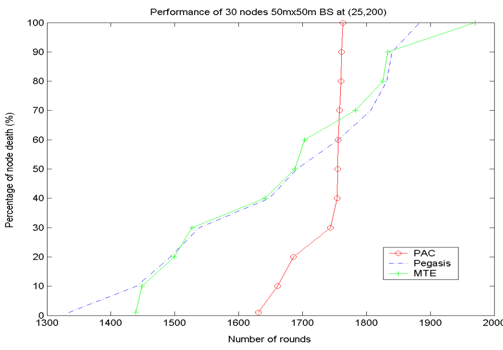

Fig. 6: Comparison of network lifetime for 30 nodes 50mx50m sensor network

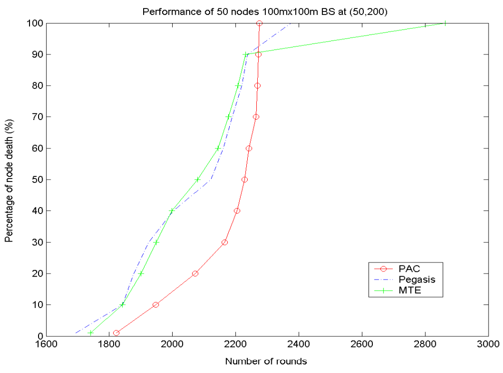

Fig. 7: Comparison of network lifetime for 50 nodes 100mx100m sensor network

As shown in Figure 6 and 7, the lifetime until the first node die of PAC is 10-20% higher than that of MTE and about 20-30% higher than that of PEGASIS. The number of rounds of PAC is 15-25% higher than that of MTE and PEGASIS when the node death percentage increases until more than 50% of nodes die. Moreover, in PAC, just after 30% first nodes die, almost remaining nodes die at the same time as compared to gradual death of remaining nodes in case of PEGASIS and MTE.

In PEGASIS and MTE, nodes take turn to become the leader. So nodes far away from the sink consume more energy as the leader nodes than nodes stay near the BS and easily run out of energy earlier. For example, let's consider the network of size 100mx100m, with the sink at (50,200) and the average distance between nodes is 25m. The transmission cost for the leader node far away from sink is about 10 times higher than that of the leader node close to the sink. It can be seen easily that the far way nodes will run out of energy first if nodes have equal chance to become leader nodes as in PEGASIS and MTE. In PAC, leader node election is based on current energy level and the power consumption to transmit to BS from each node; thus ensures more uniform energy consumption among nodes. Nodes consume all the energy and die almost at the same time together. Also the energy consumption is distributed evenly thus dead nodes are distributed randomly among the field.

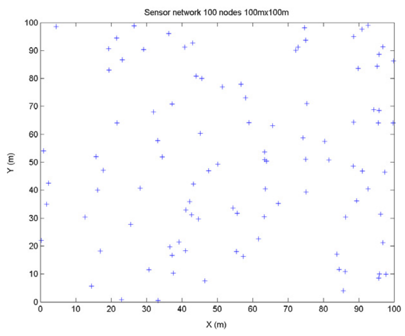

Fig. 8: Initial node distribution

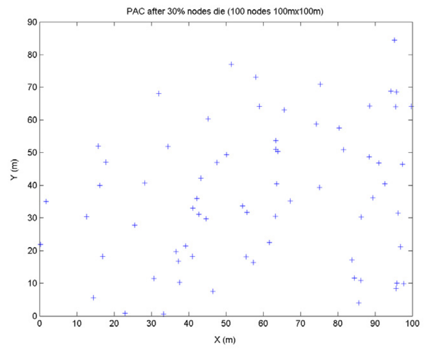

Fig. 9: Node distribution of PAC after 30% nodes die

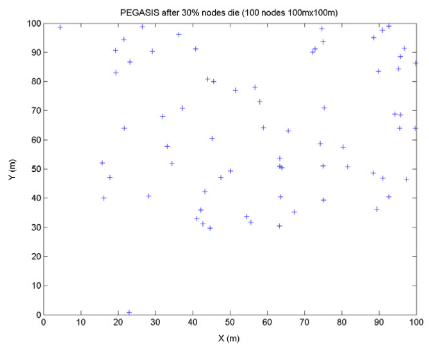

Fig. 10: Node distribution of PEGASIS after 30% nodes die

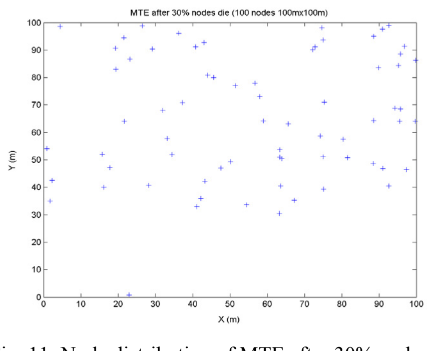

Fig. 11: Node distribution of MTE after 30% nodes die

Figure 8, 9, 10, 11 are the results of the second type of simulations that further illustrate the above analysis by showing the initial node distribution and node distribution of remaining live nodes after 30% of nodes die for PAC, PEGASIS and MTE. As can be seen from these figures, in the case of PAC, nodes die evenly at random position versus the case of PEGASIS and MTE, almost nodes far away form the sink die first.

5. Conclusion

In this paper, we present PAC, an energy efficient chain-based routing protocol for data gathering in sensor networks. The PAC protocol provides the distributed algorithm to construct the energy efficient chain based on minimum cost tree. It also takes into account the remaining energy level and power consumption of sensor nodes when electing the leader node so that nodes consume energy more uniformly. Simulation shows that PAC outperforms previous chain-based routing protocols (PEGASIS and MTE) in term of network lifetime. Nodes also die at random positions thus maintain the better network coverage as compared to previous work where nodes far away from the sink usually die first.

References

[1] S. Lindsey, C. Raghavendra, and K. Sivalingam, *Data Gathering Algorithms in Sensor Networks Using the Energy Metrics*, IEEE Transactions on Parallel and Distributed Systems, vol. 13, no. 9, Sep. 2002, pp. 924-935.

[2] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "*An Application-Specific Protocol Architecture for Wireless Microsensor Networks*" IEEE Trans. on Wireless Communications, Vol. 1, No. 4, Oct. 2002, pp. 660-670

[3] K. Du, J. Wu, and D. Zhou, *Chain-Based Protocols for Data Broadcasting and Gathering in Sensor Networks,* Proc. of Workshop on Parallel and Distributed Scientic and Engineering Computing with Applications (in conjunction with IPDPS), April 2003

[4] Fan Ye, Alvin Chen, Songwu Lu, Lixia Zhang, *A scalable Solution to Minimum Cost Forwarding in Large Scale Sensor Networks*. ICCCN 2001.

[5] Intanagonwiwat C., Govindan R. and Estrin D., *Directed diffusion: A scalable and robust communication paradigm for sensor networks* In Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking (MobiCOM '00), August 2000, Boston, Massachussetts.

[6] Matthew J. Miller and Nitin H. Vaidya, *On-Demand TDMA Scheduling for Energy Conservation in Sensor Networks*, University of Illinois at Urbana Champaign Technical Report, June 2004.

[7] Wei Ye, John Heidemann and Deborah Estrin *An energy-efficient MAC protocol for wireless sensor networks,* IEEE Infocom 2002.

[8] T. Rappaport, *Wireless Communications: Principles & Practice*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[9] Gang Zhou, Tian He, Sudha Krishnamurthy, John A. Stankovic. *Impact of Radio Asymmetry on Wireless Sensor Networks*, MobiSys'04, Boston, MA, June 2004.