ISSN:0975-9646



# Energy Efficient Approach on Cut Detection in Wireless Sensor Networks

Poojarini Mitra, Sinthia Roy, Sayani Chandra, Ipsita Saha, Bidyutmala Saha

*Assistant Professor, Department of Computer Science and Engineering,*
*Guru Nanak Institute of Technology*
*Sodepur, Kolkata – 700114, WB, India*

*Abstract* - **Network connectivity is considered one of the most important aspects in Wireless Sensor Networks (WSN). Due to their low power configuration, WSN often suffers from disrupted connectivity resulting in data lose and significant energy loss of the whole network. The disruption in connectivity, referred to as network cut, divides the whole network into multiple components. In this article, we propose a low overhead scheme of detecting cuts in a WSN named Energy Efficient Approach on Cut Detection in WSN. The algorithm allows each node to detect whether a node is communicating or not. If a certain node is not communicating, the information is passed to the Base Station through other nodes which then takes the decision whether there is a cut or not. The algorithm is distributed and asynchronous in the sense that every node communicates within its communication range. Although the algorithm is iterative and involves only local communication, its convergence rate is quite fast and is independent of the size of the network. We prove the effectiveness of the proposed algorithms through simulations.**

*Keywords* – **Wireless Sensor Network, Cut Detection, Energy Efficiency**

## I. INTRODUCTION

Wireless Sensor Networks consists of a large number of low cost, low-power, low maintenance sensor nodes [1]. Sensor nodes usually consist of sensing, communicating, computing, storing and power components. The sensor nodes are capable of storing data sensed from the environment according to their specific application, collaborating and computing the data to perform some specific task and communicating those data to the higher level or the base station (BS) for decision making. BSs have enhanced capabilities over simple sensor nodes and can do complex data processing. Sensor nodes are often randomly deployed in huge numbers over a large area to monitor physical conditions like temperature, humidity, intensity, vibration, pressure, motion, pollutants etc. They are also deployed in military battlefield to track enemy movements.

Based on the characteristics, Wireless Sensor Networks has following main features [1]: a) Because of the limited energy supply for the wireless nodes, the wireless links between nodes in the Sensor Network are quite inconsistent for the communication participants. b) Due to high memory and energy constraint, sensor nodes have restricted computational capacity and communication bandwidth. c) They may remain unattended in hostile environment. Due to this, the major feature concerning WSN is energy efficient routing.

To offer energy efficient routing, one of the challenges is to ensure that the network is connected. The connectivity of the network can easily be disrupted due to unpredictable wireless channels, early depletion of node's energy, and physical tampering by hostile users. Network disconnection, typically referred as a *network cut*, may cause a number of problems. For example, ill-informed decisions to route data to a node located in a disconnected segment of the network might lead to data loss, wasted power consumption, and congestion around the network cut.

### *A. Wireless Sensor Network Topologies*

WSN nodes are typically organized in any one of the three types of network topologies [**2**]. In a Star topology, each node connects directly to a BS. In a Mesh network, nodes can connect to multiple nodes in the system and pass data through one of the path available to the BS. Finally, to offer fastest data transmission, in a Cluster Tree Network, each node connects to a node higher in the tree and then to the BS. In this way data is routed from the lowest node on the tree to the BS. In our algorithm we chose a Cluster Tree Network as the routing technique.

## II. RELATED WORK

The challenges of the cut detection problem have been highlighted in many papers [3]. Chong et al. [4] points out the problem from a security point of view. Chong et al. suggests that nodes deployed in a hostile environment are open to tampering, so the nodes must be able to detect the rig. In [5], Cerpa and Estrin stress the importance of the network cut detection problem in their self-configuring topology scheme but left it as a future work.

Kleinberg et al. [6] first considered the cut detection problem in a wired network. The authors introduced the concept of $(\epsilon, k)$-cut to be a network separation into $(1-\epsilon)n$ nodes and $\epsilon n$ set of nodes when $k$ independent edges are disabled. To detect the $(\epsilon, k)$-cut, a set of *agents*, denoted by a set *D*, is strategically deployed in the network to monitor the connectivity of the network. ----------Each agent periodically communicates with all other agents. Failed connections beyond some threshold are presumed to indicate the presence of a cut. The main result is that the size of the set $D$ must be $O(k^3\frac{1}{\epsilon}\log\frac{1}{\epsilon}+\frac{1}{\epsilon}\log\frac{1}{\delta})$ to successfully detect any kind of $(\epsilon, k)$-cut with probability$1-\delta$. However, in wireless sensor networks, due to their geometric structure, linear or other geometric shaped cuts are more likely than independent $k$disabled edges. Additionally, the number of agents required for this type of cut detection is very large.

Shrivastava et al. [7] proposed deterministic and randomized algorithms to detect network separation using a set of sentinel nodes to monitor for linear cuts in a network. The work is, in large part, based on [6]. Specifically, the authors defined the $\epsilon$ -cut where at least $\epsilon$ fractions of nodes are disconnected by the cut. However, Shrivastava minimized the number of required sentinels by reducing the problem to the linear cut, which is a more natural phenomenon for wireless sensor networks than independent $k$edge failures, and proved that there exist $O(\frac{1}{\epsilon})$ sentinels for any $\epsilon$ -cut with $\epsilon < 1$. This is a relatively small number of sentinels when compared with the result of [6]. The authors proposed a deterministic algorithm to find the minimum number of sentinels and introduced a fast randomized algorithm to compute the sentinels of $O(\frac{1}{\epsilon})$ size. However, Shrivastava's algorithm is limited to detecting linear cuts and fails to detect arbitrarily shaped cuts. Also, it is a centralized algorithm where information about a cut is only known to the base station.

In Ritter et al. [8], the authors select a source node and make it broadcast an alive message throughout the network. Border nodes detect a cut if they miss the alive message from the source node more than a given number of times.

The most recent cut detection algorithm is proposed by Barooah et al. [9] and overcomes several problems associated with previous solutions. Barooh's algorithm, DSSD, can not only detect an arbitrarily-shaped cut, but also enables every node in the network to autonomously detect a cut in a distributed manner. Each node maintains a positive scalar called state and updates this scalar iteratively based on received states from one-hop neighbors. After a number of iterations, if a node is connected to the source that updates state for the network, its state converges to some positive value. Otherwise, if there exist a cut between the node and the source, its state rapidly decays to 0. Thus, a node can independently determine its connectivity status by monitoring its state.

The decision on cut detection is made based on available data sets, e.g., state is updated based on received states, and convergence is determined based on the history of state changes. Thus, ensuring the validity of data sets is an important process for accurate cut detection. Outlier detection is a statistical analysis tool often used to identify problems in data sets like measurement error or abnormal data distribution.

Outlier detection can be categorized into two main streams: a parametric approach, which assumes a priori known distribution of the data, and a non-parametric approach that does not rely on a specific distribution. With known data distribution, the parametric approach detects outliers with very high precision. However, in many cases, finding a matching distribution is very hard. Probabilistic models that infer distribution based on sample data compensate for this difficulty but often show high false positive rates [10]. Non-parametric approaches using distance-based and density-based methods attempt to overcome this limitation. Knorr and Ng [11] proposed the first distance-based algorithm, where a point is regarded as an outlier if its distance to a $k^{th}$ nearest neighbor point is greater than a certain threshold. One disadvantage is that the threshold must be defined. Ramaswamy et al. [12] studied distance-based detection, where a point is said to be an outlier if the distance to kth nearest neighbor is greater than that of $n-1$ other points. Recently, Zhang et al. [13] introduced an algorithm for finding an outlier based on the sum of distances to the point's k nearest neighbors. However, all distance-based solutions fail to detect outliers in clustered data. Density-based outlier detection schemes [14], [15] gracefully solve this problem. Each data point is given a score called Local Outlier Factor (LOF) based on its local density, which is bounded by a specific value MinPts. In [14], an outlier is determined by score. In [15], the bounding value MinPts is determined autonomously using statistical values such as inter-cluster distances and cluster diameters.

The problem of network partition in sensor networks has been raised in several articles, but it appears not to have been investigated on the basis of energy efficiency. The main problem of WSN is energy constraint. All the algorithms discussed till now do not focus on the power drainage that takes place due to the calculations and transmission.

## III. PROBLEM FORMULATION

After surveying various algorithms and articles on cut detection mechanism in Wireless Sensor Networks, we have proposed a new and optimal technique to determine a cut in the configured network. The Energy Efficient Approach on Cut Detection in Wireless Sensor Networks is simplistic in its own regard. We have focused and projected our technique on energy efficiency. Energy of a sensor node is one of the major causes of disability and dying out of sensor nodes, thereby forming a cut in the network. We have used a tree like structure in a network to illustrate the technique. A tree has successor, predecessor, parent, uncle, siblings, root, and leaf nodes. Some of these terminologies will be used in our problem to identify a particular node in which cut has taken place.

In our technique, we will use a very small snapshot of the problem. The network will be a congested one in relation to the actual scenario. The algorithm comprises mainly of Parent Discovery Phase and the Data Forwarding Phase. After the network has been set up, the Parent Discovery Phase is started, which takes help of Breadth First Search (BFS) and shortest path algorithm and saves the parent and uncle nodes in each sensor's dedicated memory. In the Data Forwarding Phase after detecting the cut, the BS again recursively implements the parent discovery phase. The Data Forwarding Phase detects cut in simple way by sending data as well as parent node's unique address to the uncle sensor node if it is unable to reach its next hop parent node, thereby notifying the BS which takes care in recovering the cut and running the Parent Discovery Phase again.

## IV. PROPOSED MECHANISM

The algorithm comprises of various phases. The initial requirements and specifications of each node are mentioned and executed accordingly. The memory is a storage system associated to each node. The packet formation generally depends upon size and configuration of the desired network. The processing elements are placed accordingly with a RAM chip to help out the examination of packets. A unique Identification number is assigned to each sensor node and have similarities with other nodes within the same network. The nodes are then connected in a tree topology to form a network. The initial phase is the Network Configure Phase which makes the connections/paths between various sensor nodes. In the Parent Discovery Phase at first The Base Station sends a node discovery message to all the nodes at 1 hop distance. On receiving the node discovery message, the nodes which are at one hop from Base Station consider it as its parent. All the nodes then send a node discovery message to all the nodes which are at 1 hop distance along with the information of its predecessor. If a node receives more than one node discovery message it selects its parent depending on the least number of predecessor of its parent. As the node discovery messages are getting broadcast, a node may receive more than one node discovery message from nodes which are siblings i.e., nodes having equal number of predecessor each. Then it selects any one of them as its parent. On receiving a node discovery message and after selecting a parent it stores the information of its parent node as well as the information of the siblings of its parent also called the uncle nodes. This process continues till the leaf nodes are reached. After the Parent Discovery Phase, the Data Forwarding Phase is started. The child node sends data to its parent. The parent in return sends back an acknowledgement. If an acknowledgement is not received, the child node sends the data again. At the same time it sends a beacon to the uncle nodes with the information of its parent. The uncle nodes then send the beacon to its parent i.e., the grandparent of the child node. If the grandparent receives more than one beacon having the information of the same node, it forwards the beacon to the higher level, otherwise discards it. On receiving a beacon, the Base Station sends a beacon to the node whose information is contained in it. If it doesn't gets back any reply, the Base Station again starts the Parent Discovery Phase by discarding the node.

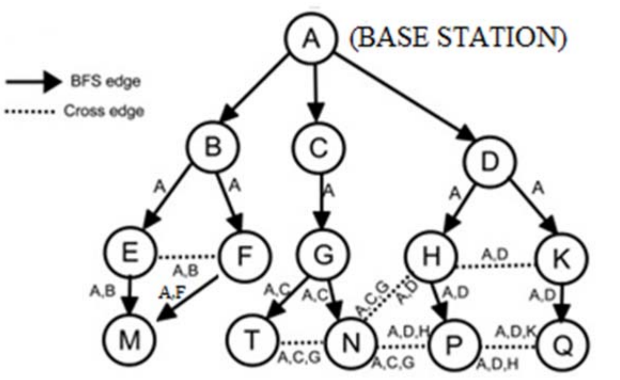


Fig. 1 Parent Discovery Phase

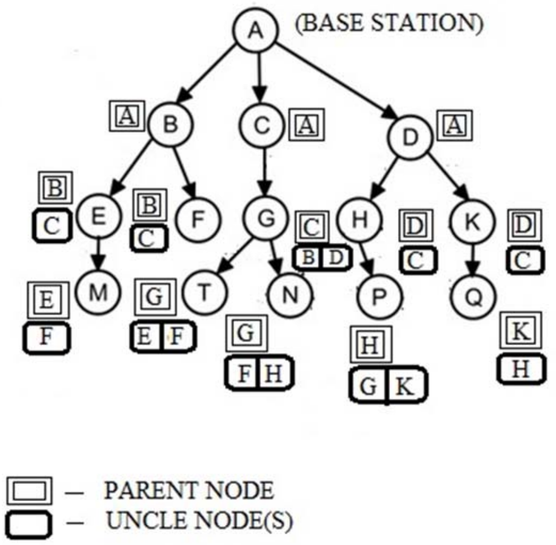


Fig. 2 After the selection of Parent and Uncle Node(s)

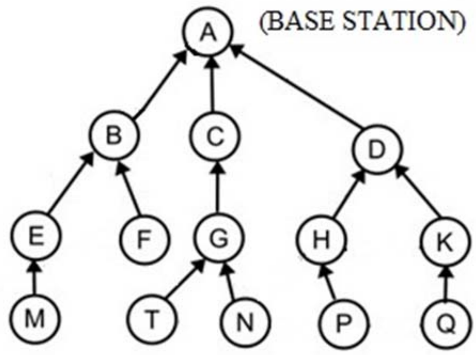


Fig. 3 Data Forwarding through Parent Node

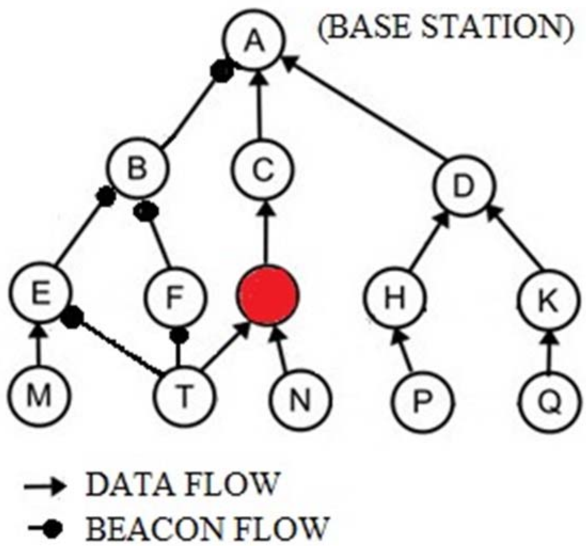


Fig. 4 Forwarding of Beacon by Uncle Nodes of 'T' when its parent node 'G' node has died out.

## V. SIMULATION

In this section, we use simulations to study the performance of our scheme. We have implemented Energy Efficient Approach on Cut Detection in Wireless Sensor Networks in OMNeT++ [16], a discrete event simulator. We chose OMNeT++ as a base platform and its Castalia [17] framework as a starting point for our implementation.

OMNeT++ is a public-source, component-based, modular and open-architecture simulation environment. In general, OMNeT++ is a discrete event simulator which can be used for wide variety of purposes like modeling of traffic of telecommunication networks, protocols, queuing networks, multiprocessors and distributed hardware systems, validating hardware architectures, evaluating performance of complex software systems and so on.

An OMNeT++ model consists of hierarchically nested modules. Modules communicate through message passing. They can send messages either directly to their destination or along a predefined path, through gates and connections. Modules can have their own parameters, which can be used to customize module behavior and to parameterize the model's topology.

OMNeT++ simulation can make use of several user interfaces for different purposes. In our simulation we use Tcl/Tk graphical user interface for debugging and presentation and faster command-line interface for simulations of large networks.

The simulator and user interfaces and tools are portable – they work on Windows, Mac OS and several Unix-like systems, using various C++ compilers.

Castalia framework is a simulator for Wireless Sensor Networks (WSN), Body Area Networks (BAN) and generally networks of low-power embedded devices. It is based on the OMNeT++ platform and can be used by researchers and developers who want to test their distributed algorithms and/or protocols in realistic wireless channel and radio models, with a realistic node behavior especially relating to access of the radio. Castalia can also be used to evaluate different platform characteristics for specific applications, since it is highly parametric, and can simulate a wide range of platforms.

The environment settings are shown in Table 1.

TABLE 1 ENVIRONMENTAL SETTINGS FOR SIMULATION

| Parameters | Values |
|---|---|
| WSN area | 70 X 70 |
| Number of nodes | 200 |
| Deployment | Uniform |
| Transmission power consumption | 0.002 * dist J |
| Reception power consumption | 0.02 J |
| Initial energy | 5 J |
| * dist is the transmission distance | |

The performance of Energy Efficient Approach on Cut Detection in WSN protocol is compared with a network having no cut detection scheme implemented. The Network Lifetime is used as an evaluation measure.

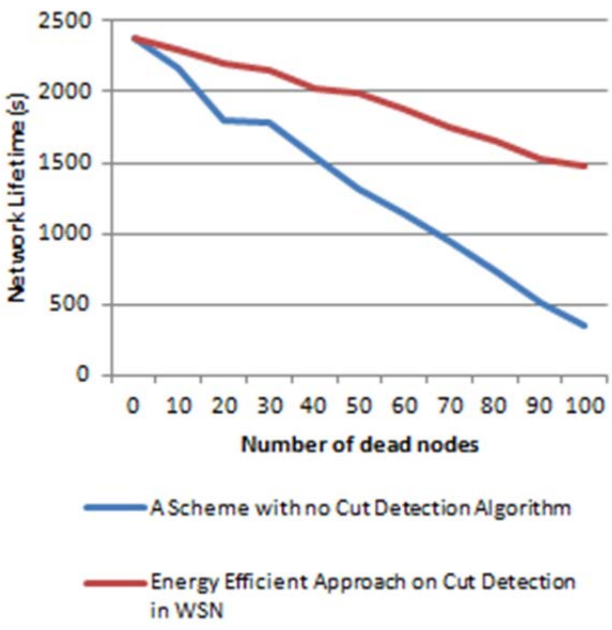


Fig. 5 Comparing Network Performance between A Scheme with no Cut Detection Algorithm and Energy Efficient Approach on Cut Detection in WSN to show the Network Lifetime

From Fig. 5, it can be observed that on implementing our algorithm the network lifetime increases.

## VI. CONCLUSION AND FUTURE SCOPE

Due to the large variety of applications, the notion of cut has been interpreted in many different ways. However, in general, cut can be considered as a measure of quality of service of a sensor network, because lesser cuts of a region lead to better measurements of the underlying physical phenomenon. Usually, the possibility of cuts also depends on the deployment scheme for a static sensor network. Because if in an area the activity of the nodes is greater than the nodes placed in other areas, the energy of those nodes are going to get exhausted much faster resulting in cuts. In this case, the problem is to identify a minimum number of locations in the deployment region where nodes could be placed to achieve maximum coverage. This is called deterministic deployment. One could also assign weights to different parts of the region depending on how critical the information in those parts are, and accordingly choose sparser or denser deployment of nodes. In many situations, however, deterministic deployment is not possible; this could be either because the region is inaccessible and hostile to human beings or because of the lack of a priori knowledge of the region itself. In such situations, nodes could be deployed randomly or according to some statistical distribution, such as uniform, Gaussian, Poisson, etc. This leads to the concept of stochastic coverage, and one could use a variety of computational geometry tools to provide a better coverage of the region. As our future endeavor we are thinking of developing an algorithm which will result in better coverage-connectivity.

## REFERENCES


[1] Application and Multidisciplinary Aspects of Wireless Sensor Networks: Concepts, Integration and Case Studies by Liljana Gavrilovska, Srdjan Krco, Veljko Milutinovik, Ivan Stojmenovic, Roman Trobec, Springer Science & Business Media, 05-Oct-2010 - Computer Communication Networks

[2] Shrestha, A.; A Performance Comparison of Different Topologies for Wireless Sensor Networks. Conference on Technologies for Homeland Security, 2007 IEEE; Page(s): 280 – 285

[3] V. Park, M. Corson, A highly adaptive distributed routing algorithm for mobile wireless networks, in: INFOCOM ’97, vol. 3, 1997, pp. 1405–1413.

[4] C.-Y. Chong, S. Kumar, Sensor networks: evolution, opportunities and challenges, Proceedings of the IEEE 91 (8) (2003) 1247–1256.

[5] A. Cerpa, D. Estrin, ASCENT: adaptive self-configuring sensor networks topologies, IEEE Transactions on Mobile Computing 3 (3) (2004) 272–285.

[6] J. Kleinberg, Detecting a network failure, in: Proceedings of the 41st Annual Symposium on Foundations of Computer Science, 2000, 231

[7] N. Shrivastava, S. Suri, C. Tóth, Detecting cuts in sensor networks, ACM Transactions on Sensor Networks 4 (2) (2008) 1–25.

[8] H. Ritter, R. Winter, J. Schiller, A partition detection system for mobile ad-hoc networks, in: Proceedings of IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON), 2004, pp. 489–497

[9] P. Barooah, Distributed cut detection in sensor networks, in: Proceedings of Conference on Decision and Control (CDC), 2008, pp. 1097–1102.

[10] M.V. Mahoney, P.K. Chan, Learning nonstationary models of normal network traffic for detecting novel attacks, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp. 376–385.

[11] E. Knorr, R. Ng, V. Tucakov, Distance-based outliers: algorithms and applications, The VLDB Journal 8 (3–4) (2000) 237–253.

[12] S. Ramaswamy, R. Rastogi, K. Shim, Efficient algorithms for mining outliers from large data sets, SIGMOD Record 29 (2) (2000) 427–438.

[13] J. Zhang, H. Wang, Detecting outlying subspaces for high-dimensional data: the new task, algorithms, and performance, Knowledge and Information Systems 10 (3) (2006) 333–355.

[14] M. Breunig, H. Kriegel, R. Ng, J. Sander, LOF: identifying density-based local outliers, SIGMOD Record 29 (2) (2000) 93–104.

[15] S. Papadimitriou, H. Kitagawa, P. Gibbons, C. Faloutsos, LOCI: fast outlier detection using the local correlation integral, in: International Conference on Data Engineering, 2003, p. 315.

[16] OMNeT++ Discrete Event Simulation System. [Online] http://www.omnetpp.org/.

[17] Castalia Framework Castalia – A simulator for WSNs. [Online] http://castalia.npc.nicta.com.au/.