ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

An Efficient Detection of Disconnected Links in Wireless Sensor Network

[1]M.Rajalakshmi, [2]G.Balaraju
[1]M Tech Student, [2]Asst. professor
[1,2]Dept of Computer Science and Engineering,
[1,2]Sri Sunflower College of Engineering and Technology, Lankapally, Krishna dist, A.P.

ABSTRACT: Wireless sensor networks (WSNs) have emerged as a promising new technology to monitor large regions at high spatial and temporal resolution. The inherent nature of WSNs such as unattended operation, battery-powered nodes, and harsh environments pose major challenges. 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. In this Paper, We propose a distributed algorithm that allows every node to monitor the topology of the (initially connected) graph and detect if a cut occurs. For reasons that will be clear soon, one node of the network is denoted as the "source node". The algorithm consists of every node updating a local state periodically by communicating with its nearest neighbors. The state of a node converges to a positive value in the absence of a cut. If a node is rendered disconnected from the source as a result of a cut, its state converges to 0. By monitoring its state, therefore, a node can determine if it has been separated from the source node. In addition, the nodes that are still connected to the source are able to detect that, one, a cut has occurred somewhere in the network, and two, they are still connected to the source node. The algorithm is iterative, a faster convergence rate is desirable for it to be effective. The convergence rate of the proposed algorithm is not only quite fast, but is independent of the size of the network. As a result, the delay between the occurrence of a cut and its detection by all the nodes can be made independent of the size of the network. This last feature makes the algorithm highly scalable to large sensor networks.

Keywords: Wireless Sensor Network, Cut Detection,
.

I.INTRODUCTION

Wireless sensor networks (WSNs), consisting of large numbers of low-cost and low-power wireless nodes, have recently been employed in many applications: disaster response [1], military surveillance [2], and medical care [3] among others. The inherent nature of WSNs such as unattended operation, battery-powered nodes, and harsh environments pose major challenges. 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.

In this paper we consider the problem of detecting *cuts* in wireless sensor networks. A sensor network is modeled as a graph G = (V , E) whose node set V correspond to the wireless sensors and whose edges E consists of pairs of nodes (u, v) that can communicate directly with each other. A *cut* is defined as the failure of a set of nodes so that the removal of those nodes and the edges incident on them from the original graph results in the separation of the graph into two or more components.rida.

We propose a distributed algorithm that allows every node to monitor the topology of the (initially connected) graph and detect if a cut occurs. For reasons that will be clear soon, one node of the network is denoted as the "source node". The algorithm consists of every node updating a local

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

state periodically by communicating with its nearest neighbors. The state of a node converges to a positive value in the absence of a cut. If a node is rendered disconnected from the source as a result of a cut, its state converges to 0. By monitoring its state, therefore, a node can determine if it has been separated from the source node. In addition, the nodes that are still connected to the source are able to detect that, one, a cut has occurred somewhere in the network, and two, they are still connected to the source node. We call it the *Distributed Source Separation Detection* (DSSD) algorithm.

Since the algorithm is iterative, a faster convergence rate is desirable for it to be effective. The convergence rate of the proposed algorithm is not only quite fast, but is independent of the size of the network. As a result, the delay between the occurrence of a cut and its detection by all the nodes can be made independent of the size of the network. This last feature makes the algorithm highly scalable to large sensor networks.

II.RELATED WORKS

Shrivastava *et. al.* [1], the challenges posed by the possibility of network partitioning in WSNs has been recognized in several papers (see, e.g. [2], [3], [4]) but the problem of detecting when such partitioning occurs seems to have received little attention. Kleinberg *et. al.* have studied the problem of detecting network failures in *wired* networks, and proposed schemes for the case when k edges fail independently [5], [6].

To the best of our knowledge, the work by Shrivastava *et. al.* [1] is the only one that addresses the problem of detecting cuts in wireless sensor networks. They developed an algorithm for detecting ϱ linear cuts, which is a linear separation of ϱn nodes from the base station. The reason for the restriction to linear cuts is that their algorithm relies critically on a certain duality between straight line segments and points in 2D, which also restricts the algorithm in [1] to sensor networks deployed in the 2D plane. The algorithm developed in [1] needs a few nodes called *sentinels* that communicate with a base station either directly or through multi-hop paths. The base station detects ϱ-cuts by monitoring whether it can receive messages from the sentinels.

In contrast to the algorithm in [1], the DSSD algorithm proposed in this paper is not limited to ϱ-linear cuts; it can detect cuts that separate the network into multiple components of arbitrary shapes. Furthermore, the DSSD algorithm is not restricted to networks deployed in 2D, it does not require deploying sentinel nodes, and it allows every node to detect if a cut occurs.

The DSSD (distributed source separation detection) algorithm involves only nearest neighbor communication, which eliminates the need of routing messages to the source node. This feature makes the algorithm applicable to mobile nodes as well. Since the computation that a node has to carry out involves only averaging, it is particularly well suited to wireless sensor networks with nodes that have limited computational capability. Simulations are reported in [7] that illustrate the capability of the algorithm to detect cuts in mobile networks, and also its ability to detect if a "reconnection" occurs after a cut. The DSSD algorithm has been demonstrated in an wireless testbed with MicaZ motes [8].

III.PROBLEM STATEMENT

Consider a sensor network modeled as an undirected graph G = (V , E), whose node set V represents the sensor nodes and the edge set E consists of pairs of nodes (u, v) such that nodes u and v can exchange messages between each other. Note that we assume inter-node communication is symmetric. An edge (u, v) is said to be incident on both the u and v. The nodes that share an edge with a particular node u are called the *neighbors* of u. A *cut* is the failure of a set of nodes $V < V$ such that the removal of the nodes in Vcut and
the edges that are incident on Vcut from G results in G being divided into multiple connected components. Recall that an undirected graph is said to be connected if there is a way to go from every node to every other node by traversing the edges, and that a *component* Gc of a graph G is a maximal connected subgraph of G (i.e., no other connected subgraph G′c of G contains Gc as its subgraph). We are interested in devising a way to detect if a subset of the nodes has been disconnected from a distinguished node, which we call the *source node*, due to the occurrence of a cut.

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

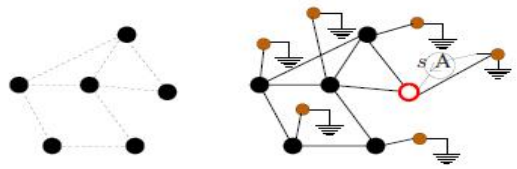

Fig. 1. A graph describing a sensor network (left), and the associated electrical network (right). In the electrical network, one node is chosen as the source that injects s Ampere current into the network, and additional nodes are introduced (fictitiously) that are grounded, through which the current flows out of the network. The thick line segments in the electrical network are resistors of 1 resistance.

IV. PROPOSED SYSTEM

The algorithm is based on an electrical analogy.Given an undirected graph G = (V , E)with, say,n nodes and m edges that describes the sensor network, we first designate one of the nodes as the *source node*. The algorithm is designed to detect when nodes get disconnected from the source node. We now construct a fictitious graph Gelec = (V elec, E elec) where V elec = V < V fict, where V fict consists of n − 1 nodes, one node for every node in V except the source node, and every node in V is connected to its corresponding fictitious node in V fict with a single edge. These edges constitute the extra edges in E elec that were not there in E . Now an *electrical network* (Gelec, 1) is imagined by assigning to every edge of Gelec a resistance of 1 . Figure 1 shows a sensor network and the corresponding electrical network.

The DSSD algorithm consists of two phases. One is a state update law, which a simple iterative procedure to compute the node potentials in the electrical network (Gelec, 1) when s Ampere current is injected at the source node and extracted through the nodes Vfict, with all the nodes in Vfict grounded. The *source strength* s is a design parameter. The other phase of the algorithm consists of monitoring the state of a node, which is used to detect if a cut has occurred.We now describe the two phases below. Note that the separation into two phases is merely for conceptual clarity, they are carried out simultaneously at every node.

State update law

Let G(k) = (V (k), E (k)) denote the sensor network that consists of all the nodes and edges of G that are still active at time k, where k = 0, 1, 2, . . . is an iteration counter. For ease of description, we index the source node as 1. Every node u maintains a scalar state xu(k) that is iteratively updated. At every iteration k, nodes broadcast their current states. Let Nu(k) = {v|(u, v) ▫ E (k)} denote the set of neighbors of u in the graph G(k). Every node in V except the source updates its state as:

$$x_u(k+1) = \frac{1}{d_u(k)+1} \sum_{v \in N_u(k)} x_v(k), \quad x_u(0) = 0,\ u \neq 1,$$

where $d_u(k) := |N_u(k)|$ is the number of active neighbors of u at time k. If we count the fictitious node corresponding to u as one of u's neighbors whose state is held fixed at 0, then the above can be thought of as an average of the neighbors' states. The source node updates its state as:

$$x_1(k+1) = \frac{1}{d_1(k)+1} \left(\sum_{v \in N_1(k)} x_v(k) + s \right) \quad x_1(0) = 0.$$

The description above assumes that all updates are done synchronously, or, in other words, every node shares the same iteration counter k. In practice, especially with wireless communication, an asynchronous update is preferable. To achieve this, every node keeps in its buffer a copy of the last received state of each of its neighbors. If in a particular iteration, a node does not receive messages from a neighbor during a time-out period, it updates its state using the last successfully received state from that neighbor. When a node fails, its neighbors will cease to receive messages from it permanently. When a node does not receive broadcasts from one of its neighbors for sufficiently long time, it removes that neighbor from its neighbor set. From then on, the node carries on the algorithm with the remaining neighbors.

We will need the following terminology. Given an electrical network (G, 1), two nodes u and o, and a set of nodes U, all in G, suppose all of the nodes in U are shorted together and grounded. The potential at u (with respect to the ground) when a

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

current source of s Ampere is connected between o and the ground is called *the potential difference between* u *and* U *with a current flow of* s *between* o *and* U in the network (G, 1).

The evolution of the node states with and without the occurrence of cuts is stated in the next theorem. Note that we assume that the source node never fails. The proof of the theorem is provided in the Appendix.

Theorem 1: Let the nodes of a sensor network modeled
as an undirected graph G(t) = (V , E (t)) that is initially
connected (i.e., G(0) is connected) iteratively update their stated by (2) and (2) with an arbitrary initial condition xu(0), u < V .

- If no nodes or edges fail, so that G(t) = G(0) for all t, the state of every node converges to the potential difference between itself and V fict with a current flow of s between the source and V fict in the electrical network (Gelec, 1). Furthermore, the steady state potential is positive for every node.
- If a node u gets disconnected from the source at time τ > 0 and remains disconnected from the source for

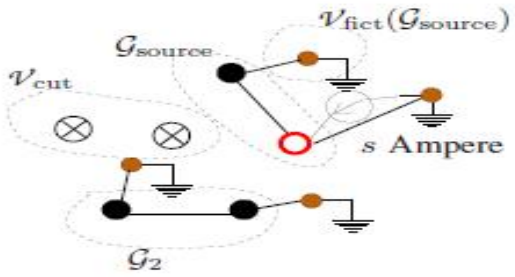

Fig.2. The connected components of the electrical network after a cut occurs in the graph shown in Figure 1. all k >τ, then its state converges to 0,i.e.,xu(k) → 0 as k − τ → ∞.

State monitoring for cut detection

Theorem 1 shows how the occurrence of a cut in the network is manifested in the states of the nodes. By analyzing their own states, nodes can detect if a cut has occurred.

Suppose a cut occurs at some time τ > 0 which separates the network into n components Gsource, G2, . . . , Gn, the component Gsource containing the source node. Since there is no source (and therefore no current injection) in each of the components G2, . . Gn disconnected from the source, it follows from Theorem 1 that the state of every node in each of these components will converge to zero. When the potential at a particular node drops below a particular threshold value, the node can declare itself cut from the source node. In fact, there may be additional node failures (and even increase in the number of components) after the cut appears. Since the state of a node converges to 0 if there is no path to the source, additional time variation in the network will not affect cut detection.

If additional failures do not occur after the cut occurs, it follows from Theorem 1 that the states of the nodes that are in the component Gsource (which contains the source) will converge to new steady state values. So, if a node detects that its state has converged to a steady state, then changed, and then again converged to a new steady state value that is different from the initially seen steady state, it concludes that there has been a cut somewhere in the network.

A node detects when steady state is reached by comparing the derivative of its state (with respect to time) with a small number ϙ that is provided a-priori. The parameters s and ϙ are design variables.

V. CONCLUSION

In this paper, the proposed algorithm can also be used for detection of "reconnection". If a component that is disconnected due to a cut gets reconnected later (say, due to the repairing of some of the failed nodes), the nodes can detect such reconnection from their states. The DSSD Algorithm efficiently identifies the cuts in wireless sensor network compare to existing systems.

References

1. N. Shrivastava, S. Suri, and C. D. T´oth, "Detecting cuts in sensor networks," in *IPSN '05: Proceedings of the 4th international*

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

symposium Information processing in sensor networks, 2005, pp. 210–217.

2. A. Cerpa and D. Estrin, "ASCENT: Adaptive Self-Configuring sEnsor Networks Topologies," in *IEEE Infocom*. New York, NY: IEEE, June 2002.
3. X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated coverage and connectivity configuration in wireless sensor networks," in *SenSys03*, Los Angeles, California, USA, November 57 2003.
4. X. J. Du, M. Zhang, K. E. Nygard, S. Guizani, and H.-H. Chen, "Selfhealing sensor networks with distributed decision making," *International Journal of Sensor Networks (IJSNET)*, vol. 2, no. 5/6, 2007.
5. J. Kleinberg, "Detecting a network failure," *Internet Mathematics*, vol. 1, pp. 37–56, 2003.
6. J. Kleinberg, M. Sandler, and A. Slivkins, "Network failure detection and graph connectivity," in *the 15th ACM-SIAM Symposium on Discrete Algorithms*, 2004.
7. P. Barooah, H. Chenji, R. Stoleru, and T. Kalm´ar-Nagy, "Detecting separation in robotic sensor networks," 2008, submitted to the IEEE wireless communications magazine.
8. H. Chenji, P. Barooah, R. Stoleru, and T. Kalm´ar-Nagy, "Demo abstract: Distributed cut detection in sensor networks," in *6th ACM Conference on Embedded Networked Sensor Systems (SenSys'08)*, November 2008.
9. G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed. The John Hopkins University Press, 1996.
10. F. Chung, "Spectral graph theory," Regional Conference Series in Mathematics, Providence, R.I., 1997.

Rajya lakshmi M, She completed her B.Tech from jntu Kakinada. At present she is persuing her M.Tech in Sri sunflower college of engineering and technology.