Demo Abstract: Distributed Cut Detection in Sensor Networks

Harshavardhan Chenji[†], Prabir Barooah[§], Radu Stoleru[†], Tamás Kalmár-Nagy[‡]
[†] Department of Computer Science, Texas A&M University
[§]Department of Electrical and Computer Engineering, University of California, Santa Barbara
[‡]Department of Aerospace Engineering, Texas A&M University
cjh@cs.tamu.edu, pbarooah@ece.ucsb.edu, stoleru@cs.tamu.edu,
sensys08@kalmarnagy.com

ABSTRACT

Loss of connectivity in deployed wireless sensor networks can be quite disastrous for the network. A "cut" (which separates the network into two or more components incapable of communicating with each other) is usually hard to detect. An algorithm which enables each node in the network to detect whether a cut has occurred anywhere in the network is demonstrated.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; C.2.4 [**Computer-Communication Networks**]: Distributed Systems; B.8.1 [**Performance and Reliability**]: Reliability, Testing and Fault-Tolerance

General Terms

Algorithms, Design, Reliability, Experimentation

Keywords

Distributed Cut Detection, Wireless Sensor Networks

1. INTRODUCTION

Several challenges exist in wireless sensor networks (WSNs). One of them is the loss of network connectivity due to environmental factors, unreliable hardware and limited energy resources. For WSN deployments in harsh environments network connectivity becomes a serious issue. The structure and connectivity pattern of even a static WSN varies over time. The appearance of a "cut", i.e, the separation of the network into multiple components [1] renders the deployment unusable, hence the utmost importance of detecting a disconnected network.

We demonstrate a distributed method for cut detection - the Distributed Source Separation Detection (DSSD) [2]. We assume that there exists a specially designated node, called the "source node". Every node maintains a scalar state, which it updates regularly through communication with its immediate neighbors. When a cut occurs, the states of the nodes that are disconnected from the source node decays to 0, while the states of those nodes that are still connected to the source converge to a new steady state value. Hence, a node can detect a cut by monitoring its own state.

The DSSD algorithm can detect cuts that separate the network into multiple components of arbitrary shapes. Since the algorithm involves only nearest neighbor communication, there is no need of routing messages to any particular node (even the source node), which might create a bottleneck in communication. This algorithm is particularly well suited to wireless sensor networks since the nodes in such a network have limited computational capabilities. In spite of the fact that the algorithm uses only nearest neighbor communication, the algorithm's convergence is independent of the size of the network [2].

2. ALGORITHM

Here we briefly describe the DSSD algorithm proposed. One node of the network is denoted as the "source node". Let $\mathcal{G}(k) = (\mathcal{V}(k), \mathcal{E}(k))$ denote the sensor network that consists of all the nodes and edges of $\mathcal{G}$ that are still active at time k, where $k = 0, 1, 2, \ldots$ is an iteration counter. For ease of description, we index the source node as 1. Every node u maintains a scalar state $x_u(k)$ that is iteratively updated. At every iteration k, nodes broadcast their current states. Let $\mathcal{N}_u(k) = \{v|(u, v) \in \mathcal{E}(k)\}$ denote the set of neighbors of u in the graph $\mathcal{G}(k)$. Every node in $\mathcal{V}$ except the source updates its state as:

$$x_u(k+1) = \frac{1}{d_u(k)+1} \sum_{v \in \mathcal{N}_u(k)} x_v(k). \qquad (1)$$

where $d_u(k)$ is the number of active neighbors of u at time k. The source node updates its state as:

$$x_1(k+1) = \frac{1}{d_1(k)+1} \left(\sum_{v \in \mathcal{N}_1(k)} x_v(k) + s \right). \qquad (2)$$

where s is a design parameter that is called the *source strength*. The state update law described above can be thought of as an iterative method for computing the potentials of the nodes in a fictitious electrical network with unit resistors on every edge [2]. The evolution of the node states with and without the occurrence of cuts is stated next theorem. Note that we assume that the source node never fails.

THEOREM 1. *[2] Let the nodes of a sensor network with an initially connected undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ execute*

SenSys'08, November 5–7, 2008, Raleigh, North Carolina, USA.
ACM 978-1-59593-990-6/08/11.

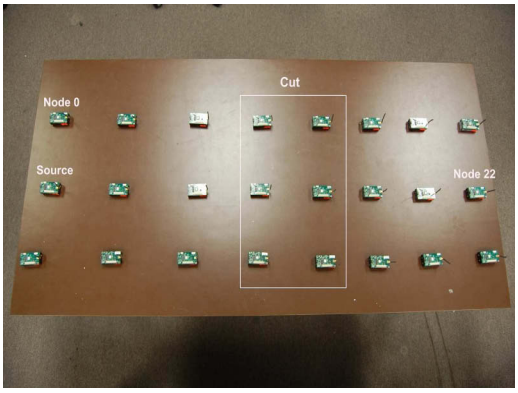

Figure 1: Demonstration setup

the DSSD algorithm starting at time $k = 0$ *with initial condition* $x_u(0) = 0,\ \forall u \in \mathcal{V}$.

1. *If no nodes fail, the state of every node converges to a positive number.*

2. *If a cut appears at a finite time* $\tau > 0$ *which separates the graph* $\mathcal{G}$ *into* N *disjoint connected components* $G_{\text{source}}, G_2, \ldots, \mathcal{G}_N$, *where the component* $\mathcal{G}_{\text{source}} = (\mathcal{V}_{\text{source}}, \mathcal{E}_{\text{source}})$ *contains the source node,then the state of every node disconnected from the source node converge to 0, i.e.,* $x_u(k) \to 0$ *as* $k \to \infty$ *for every* $u \notin \mathcal{V}_{\text{source}}$, *and the state of every node in* $\mathcal{V}_{\text{source}}$ *converges to a positive number.*

It is clear from the result above how the node states can be used to detect cuts. A node can determine if there has been a cut and if it has been separated from the source node by monitoring if its state has converged to zero. This is done in practice by choosing a small number ϵ and checking if the node state has become smaller than ϵ. The value of ϵ chosen depends on the source strength s as well as the size of the graph $\mathcal{G}$. Nodes that are still connected to the source are also able to detect that, one, a cut has occurred somewhere in the network, and two, they are still connected to the source node.

3. DEMONSTRATION

We demonstrate the execution of the DSDD algorithm in a network of 24 MicaZ motes, placed on a table in an equally spaced 8x3 grid, as depicted in Figure 1. A multihop network is created by ensuring that nodes communicate only with their immediate neighbors in space. Once the system is started, the DSDD algorithm executes and after about 60 iterations the states of the nodes converge. A laptop is used to display the convergence of the nodes states. Once the states converge, the cut is simulated by physically turning off the motes in the 4^{th} and 5^{th} columns of the 8x3 deployment, as depicted by the section "Cut" in Figure1). After the cut occurs, the new convergence values for the states are shown on the laptop. Figures 2 and 3 show the evolution of the states of two nodes in the network. As depicted in the figures, at iteration 60 a cut occurs in the network. Node 0 remains connected to the Source node and its state converges to a new value. Node 22 is disconnected from the Source node after the cut occurs and its state converges to

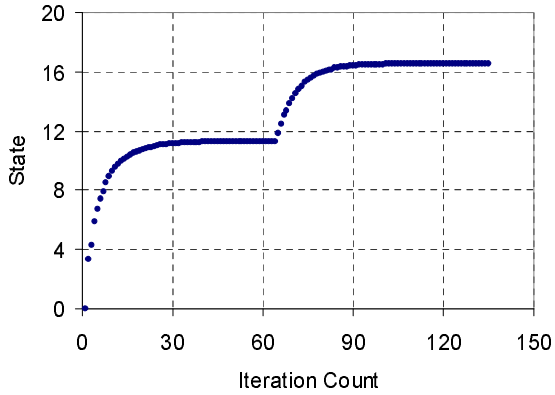

Figure 2: The state of Node 0, which remains connected to the Source after the cut.

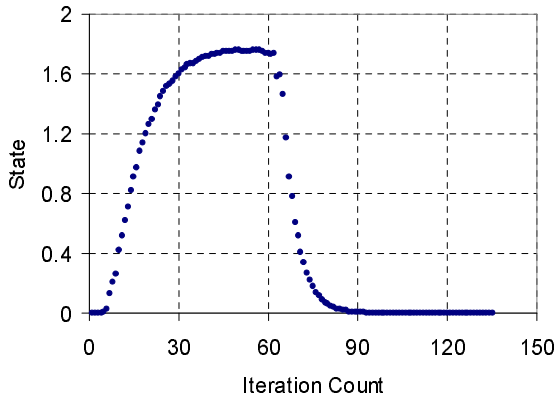

Figure 3: The state of Node 22, which is disconnected from the Source after the cut.

a value of 0. The demonstration also shows the effects of single node failures, when a cut does not occur.

4. REFERENCES

[1] N. Shrivastava, S. Suri and C. D. Tóth. Detecting cuts in sensor networks. In *Proceedings of the International Symposium on Information Processing in Sensor Networks (IPSN'05)*, 2005.

[2] P. Barooah. Distributed cut detection in wireless sensor networks. In 47th *IEEE Conferenece on Decision and Control*, 2008. Accepted for publication.

[3] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless and C. Gill. Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks In *Proceedings of the International Conference on Embedded Networked Sensor Systems (SenSys'03)*, 2003.