Distributed Algorithm for Fast Detecting the Failure Nodes Chronicles

VENKAT RAO KOMMURI *, NAGUL SHAIK **, ANIL KUMAR DASARI***

*Department of Computer Science Engineering, Nimra college (NIST)

** Department of Computer Science Engineering, Nimra college (NIST)

Summary
The Distributed Cut Detection algorithm we propose here enables every node of a wireless sensor network to detect Disconnected from Source events if they occur. Second, it enables a subset of nodes that experience CCOS events to detect them and estimate the approximate location of the cut in the form of a list of active nodes that lie at the boundary of the cut/hole. The DOS and CCOS events are defined with respect to a specially designated source node. The algorithm is based on ideas from electrical network theory and parallel iterative solution of linear equations. Numerical simulations, as well as experimental evaluation on a real Wireless Sensor Networks system consisting of micaZ motes, show that the algorithm works effectively with large classes of graphs of varying size and structure, without requiring changes in the parameters. For certain scenarios, the algorithm is assured to detect connection and disconnection to the source node without error. A key strength of the DCD algorithm is that the convergence rate of the underlying iterative scheme is quite fast and independent of the size and structure of the network, which makes detection using this algorithm quite fast.

Keywords
Application, Cut, Detection, Electrical, Fast, Network, Nodes, Parameter, Research, Wireless network

1. INTRODUCTION

A sensor network typically consists of hundreds, or even thousands, of small, low-cost nodes distributed over a wide area. The nodes are expected to function in an unsupervised fashion even if new nodes are added or old nodes disappear (e.g., due to power loss or accidental damage). While some networks include central location for data collection, many operate in an entirely distributed manner, allowing the operators to retrieve aggregated data from any of the nodes in the network. Furthermore, data collection may only occur at irregular intervals. For example, many military applications strive to avoid any centralized and fixed points of failure. In fact, node failure is expected to be quite common due to the typically limited energy budget of the nodes that are powered by small batteries. Failure of a set of nodes will reduce the number of multi hop paths in the network. Such failures can cause a subset of nodes—that have not failed to become disconnected from the rest, resulting in a "cut." Two nodes are said to be disconnected if there is no path between them.

We consider the problem of detecting cuts by the nodes of a wireless network. We assume that there is a specially designated node in the network, which we call the source node. The source node may be a base station that serves as an interface between the network and its users; the reason for this particular name is the electrical analogy introduced. Since a cut may or may not separate a node from the source node, we distinguish between two distinct outcomes of a cut for a particular node. When a node u is disconnected from the source, we say that a Disconnected from Source (DOS) event has occurred for u. When a cut occurs in the network that does not separate a node u from the source node, we say that Connected, but a Cut Occurred Somewhere (CCOS) event has occurred for u. By cut detection we mean 1) detection by each node of a DOS event when it occurs, and 2) detection of CCOS events by the nodes close to a cut, and the approximate location of the cut. By "approximate location" of a cut we mean the location of one or more active nodes that lie at the boundary of the cut and that are connected to the source. Nodes that detect the occurrence and approximate locations of the cuts can then alert the source node or the base station.

2. Distributed Cut Detection (DCD)

The algorithm allows each node to detect DOS events and a subset of nodes to detect CCOS events. The algorithm we propose is distributed and asynchronous: it involves only local communication between neighboring nodes, and is robust to temporary communication failure between node pairs.

A key component of the DCD algorithm is a distributed iterative computational step through which the nodes compute their (fictitious) electrical potentials. The convergence rate of the computation is independent of the size and structure of the network. The DOS detection part of the algorithm is applicable to arbitrary networks; a node

Manuscript received February 5, 2014
Manuscript revised February 20, 2014

only needs to communicate a scalar variable to its neighbors. The CCOS detection part of the algorithm is limited to networks that are deployed in 2D Euclidean spaces, and nodes need to know their own positions.

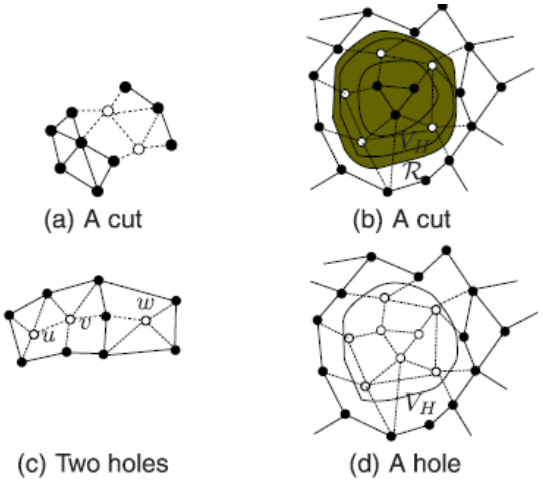


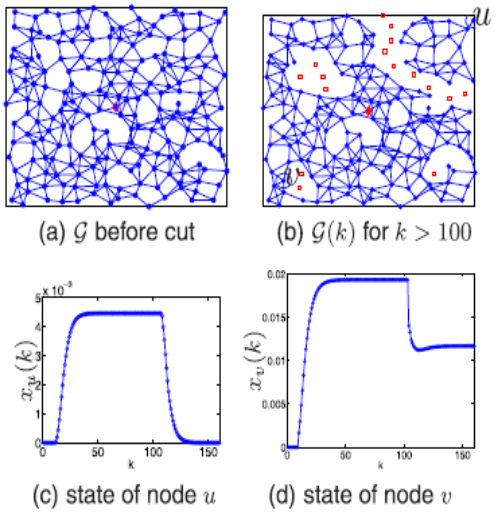
Fig. 1. Examples of cuts and holes.

2.1 DCD ALGORITHM BASED ON ELECTICAL ANALOGY

The DCD algorithm is based on the following electrical analogy. Imagine the wireless sensor network as an electrical circuit where current is injected at the source node and extracted out of a common fictitious node that is connected to every node of the sensor network. Each edge is replaced by a 1 Ω resistor. When a cut separates certain nodes from the source node, the potential of each of those nodes becomes 0, since there is no current injection into their component. The potentials are computed by an iterative scheme (described in the sequel) which only requires periodic communication among neighboring nodes. The nodes use the computed potentials to detect if DOS events have occurred (i.e., if they are disconnected from the source node). To detect CCOS events, the algorithm uses the fact that the potentials of the nodes that are connected to the source node also change after the cut. However, a change in a node's potential is not enough to detect CCOS events, since failure of nodes that do not cause a cut also leads to changes in the potentials of their neighbors. Therefore, CCOS detection proceeds by using probe messages that are initiated by certain nodes that encounter failed neighbors, and are forwarded from one node to another in a way that if a short path exists around a "hole" created by node failures, the message will reach the initiating node. The nodes that detect CCOS events then alert the source node about the cut.

When the sensor network G is connected, the state of a node converges to its potential in the electrical network (Gelec,1), which is a positive number. If a cut occurs, the potential of a node that is disconnected from the source is 0; and this is the value its state converges to. If reconnection occurs after a cut, the states of reconnected nodes again converge to positive values.

Therefore, a node can monitor whether it is connected or separated from the source by examining its state. The above description assumes that all updates are done synchronously. In practice, especially with wireless communication, an asynchronous update is preferable. The algorithm can be easily extended to asynchronous setting by letting every node keep a buffer of the last received states of its neighbors. If a node does not receive messages from a neighbor during the interval between two iterations, it updates its state using the last successfully received state from that neighbor. In the asynchronous setting every node keeps a local iteration counter that may differ from those of other nodes by arbitrary amount. Fig. 2 shows the evolution of the node states in a network of 200 nodes when the states are computed using the update law described above. The source node is at the center. The nodes shown as red squares in Fig. 2b fail at k =100, and thereafter they do not participate in communication or computation. Fig. 2c and 2d show the time evolution of the states of the two nodes u and v, which are marked by circles in Fig. 2b. The state of node u (that is disconnected from the source due to the cut) decays to 0 after reaching a positive value, whereas the state of the node v (which is still connected after the cut) stays positive.



2.2 DISTRIBUTED CUT DETECTION ALGORITHM

2.2.1 CCOS Detection

The algorithm for detecting CCOS (Connected, but a Cut Occurred Somewhere) events relies on finding a short path

around a hole, if it exists, and is partially inspired by the jamming detection algorithm proposed in The method utilizes node states to assign the task of hole-detection to the most appropriate nodes. When a node detects a large change in its local state as well as failure of one or more of its neighbors, and both of these events occur within a (predetermined) small time interval, the node initiates a PROBE message.

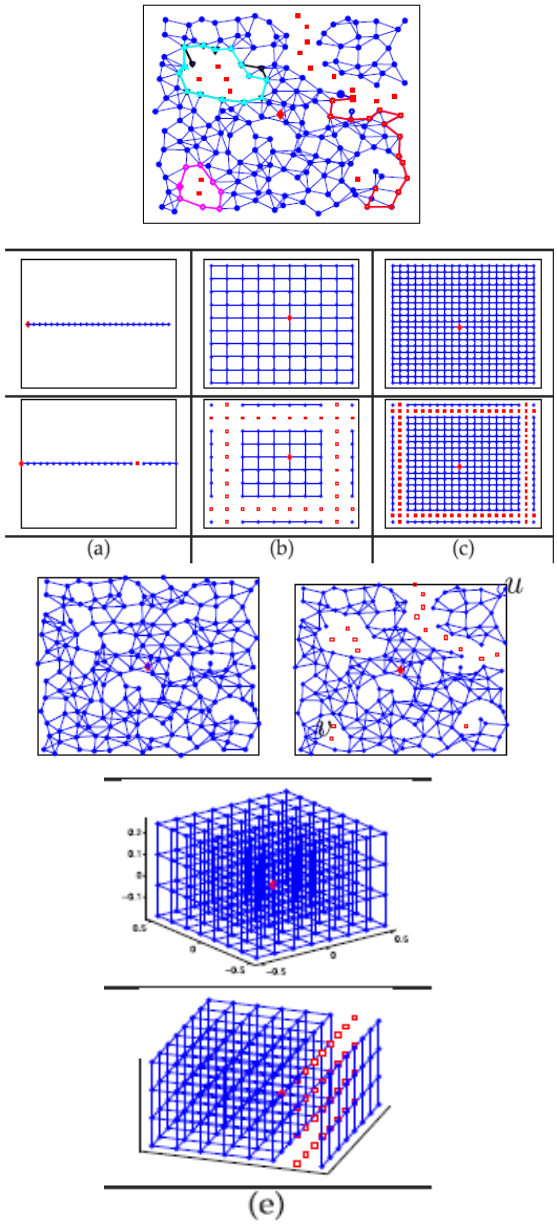
Each PROBE message p contains the following information:
a). a unique probe ID,
b). probe centroid Cp
c). destination node,
d). path traversed (in chronological order), and
e). the angle traversed by the probe around the centroid.

The probe is forwarded in a manner such that if the probe is triggered by the creation of a small hole or cut, the probe traverses a path around the hole in a counter-clockwise (CCW) direction and reaches the node that initiated the probe. In that case, the net angle traversed by the probe is 360 degree.Onthe other hand, if the probe was initiated by the occurrence of a boundary cut, even if the probe eventually reaches its node of initiation, the net angle traversed by the probe is 0. Nodes forward a probe only if the distance traveled by the probe (the number of hops) is smaller than a threshold value ‘max. Therefore, if a probe is initiated due to a large internal cut/hole, then it will be absorbed by a node (i.e., not forwarded because it exceeded the distance threshold constraint), and the absorbing node declares that a CCOS event has taken place. The location information needed by the nodes need not be precise, since it is only used to compute destinations of probe messages. The assumption of the network being 2D is needed to be able to define CW or CCW direction unambiguously, which is used in forwarding probes. At the beginning of iteration, every node starts with a list of probes to process. The list of probes is the union of the probes it received from its neighbors and the probe it decided to initiate, if any.

2.2.2 CCOS Detection Performance

Recall that the CCOS detection part of the algorithm is not applicable to 3D networks, so it was only tested on networks example, Fig. (a) Shows the path of the probes and their originating nodes in the network of Fig. 4d. Two probes were triggered by nodes close to the cut on the upper right corner; both of them were absorbed when the length of their path traversed exceeded ‘max hops, which led to correctly detecting CCOS events. Among three probes that were triggered by nodes near small holes in this network, one of them—near the hole in the upper left corner—failed to find a path back to its originating node, leading to an erroneous declaration of an CCOS event by the absorbing node. The probability of a CCOS1/0 error in this case is therefore 0.33. Table 3 summarizes the performance of the CCOS detection part of algorithm (executed with parameter values shown in Table 1). The CCOS detection error probabilities are 0 except in case of the network in Fig. 4d as described above.

3. Figures and Tables



4. Conclusion

Detecting cuts by the remaining nodes of a wireless sensor network. Algorithm that allows every node to detect when the connectivity to a specially designated node has been lost, and one or more nodes (that are connected to the special node after the cut) to detect the occurrence of the cut. The algorithm is distributed and asynchronous: every node needs to communicate with only those nodes that are within its communication range. The algorithm is based on the iterative computation of a fictitious "electrical potential of the nodes. Application of the DCD algorithm to detect node separation and reconnection to the source in mobile networks is a topic of on going research.

TABLE 1
List of Parameters that Have to Be Provided to the Nodes

Symbol	Name/description	Value
s	source strength	100
ϵ_{zero}	value below which the state is considered to be 0	10^{-10}
ϵ_{DOS}	value below which the normalized state is considered zero	10^{-3}
$\epsilon_{\Delta x}$	value below which the normalized state difference is considered zero	10^{-3}
τ_{guard}	time during which the normalized state difference has to be below $\epsilon_{\Delta x}$ for the state to be considered steady	3
τ_{drop}	number of failed consecutive transmissions before a neighbor is declared to have failed.	4
$\ell_{\max}$	maximum path length for a probe	15
$r^{\Delta ss}$	threshold ratio of change in the steady state for probe initiation	0.35

The numerical values shown here are used for all simulations and experimental evaluations reported in this document.

TABLE 2
DOS Detection Performance for the Networks Shown in Figs. 4

Network	(a)	(b)	(c)	(d)	(e)
Prob(DOS0/1 error)	0/0	0/0	0/0	0/0	0/0
Prob(DOS1/0 error)	0/0	0/0	0/0	0/0	0/0
DOS Delay (mean)	20	17	20	35	31
DOS Delay (std. dev.)	4.2	5.4	4.3	3.9	2

The two values of the probability shown in each cell correspond to $k = 60$ and $k = 160$, respectively.

TABLE 3
CCOS Detection Performance for Four Networks in Figs. 4a and 4d

Network	(a)	(b)	(c)	(d)
Prob(CCOS1/0 error)	0	0	0	0.33
Prob(CCOS0/1 error)	0	0	0	0
CCOS Delay	33	40	37	40

The Error Probabilities Are at $k = 160$.

REFERENCES

[1] G. Dini, M. Pelagatti, and I.M. Savino, "An Algorithm for Reconnecting Wireless Sensor Network Partitions," Proc. European Conf. Wireless Sensor Networks, pp. 253-267, 2008.

[2] N. Shrivastava, S. Suri, and C.D. To´ th, "Detecting Cuts in Sensor Networks," ACM Trans. Sensor Networks, vol. 4, no. 2, pp. 1-25, 2008.

[3] H. Ritter, R. Winter, and J. Schiller, "A Partition Detection System for Mobile Ad-hoc Networks," Proc. First Ann. IEEE Comm. Soc. Conf. Sensor and Ad Hoc Comm. and Networks (IEEE SECON '04), pp. 489-497, Oct. 2004.

[4] M. Hauspie, J. Carle, and D. Simplot, "Partition Detection in Mobile Ad-Hoc Networks," Proc. Second Mediterranean Workshop Ad-Hoc Networks, pp. 25-27, 2003.

[5] P. Barooah, "Distributed Cut Detection in Sensor Networks," Proc. 47th IEEE Conf. Decision and Control, pp. 1097-1102, Dec. 2008.

[6] A.D. Wood, J.A. Stankovic, and S.H. Son, "Jam: A Jammed-Area Mapping Service for Sensor Networks," Proc. IEEE Real Time Systems Symp., 2003.

[7] http://www.xbow.com/Products/Product_pdf_files/Wireless_ pdf/MICAZ_Datasheet.pdf, 2011.

[8] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System Architecture Directions for Networked Sensors," Proc. Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS),2000.

K VENKAT RAO is a student of Computer Science from NIMRA COLLEGE OF INFORMATION AND SCIENCE AND TECHNOLOGY, nimra nagar jupudi, presently pursuing M.tech from this college. He received M.C.A from Nagarjuna University in the year 2011. His research interested in Networks.

SHAIK NAGUL received his M.tech in computer science Engineering from Acharya Nagarjuna University in 2011.He is a good researcher. Who has presented nearly 5 various International journals and he is working as Asst.Professor & HOD in Dept. of Computer Science and Engineering. He is a good researcher in Networks and Neural networks.

ANILKUMAR DASARI received his M.tech in computer science Engineering from JNTUK and he published 4 International journals in various publications. He is a good researcher who has worked mostly on networks and networks security.