Volume 3, Issue 3, March 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

DOS Detection Using DCD Algorithm in Wireless Sensor Networks

B. Shyam Kumar
Assistant Professor, Department of Information Technology,
Teegala Krishna Reddy Engineering College
Hyderabad, Andhra Pradesh, India

Abstract -Wireless Sensor networks of sensor nodes are envisioned to be deployed in the physical environment to monitor a wide variety of real-world phenomena. Almost any sensor network application requires some form of self configuration, where sensor nodes take on specific functions in the network without manual intervention. These roles may be based on varying sensor node properties. Loss of connectivity in deployed wireless sensor networks can be quite disastrous for the network. A wireless sensor network can get separated into multiple connected components due to the failure of some of its nodes, which is called a "cut". Here, in this paper, problem of detecting cuts by the remaining nodes of a wireless sensor network has been mentioned. Also followed DCD algorithm that allows every node to detect when the connectivity to a specially designated node has been lost, and one or more nodes to detect the occurrence of the cut. Performance of DCD in outdoor also presented. This algorithm is distributed and asynchronous.

Keywords—wireless networks, sensor networks, distributed cut detection, disconnected from source, nodes.

I. INTRODUCTION

Wireless sensor networks (WSNs) are a promising technology for monitoring large regions at high spatial and temporal resolution. However, the small size and low cost of the nodes that makes them attractive for widespread deployment also causes the disadvantage of low operational reliability. A node may fail due to various factors such as mechanical/electrical problems, environmental degradation, battery depletion, or hostile tampering. 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.

Let us consider the problem of detecting cuts by the nodes of a wireless network. A specially designated node in the network, which we call the source node may be a base station that serves as an interface between the network and its users. 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 DOS (Disconnected frOm Source) event has occurred for u. To see the benefits of a cut detection capability, imagine that a sensor that wants to send data to the source node has been disconnected from the source node. Without the knowledge of the network's disconnected state, it may simply forward the data to the next node in the routing tree, which will do the same to its next node, and so on. However, this message passing merely wastes precious energy of the nodes. The cut prevents the data from reaching the destination.

Therefore, on one hand, if a node were able to detect the occurrence of a cut, it could simply wait for the network to be repaired and eventually reconnected, which saves onboard energy of multiple nodes and prolongs their lives. The ability to detect cuts by both the disconnected nodes and the source node will lead to the increase in the operational lifetime of the network as a whole. So a distributed algorithm is introduced to detect cuts, named the Distributed Cut Detection (DCD) algorithm. This algorithm 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 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 and a node only needs to communicate a scalar variable to its neighbors.

II. OUR APPROACH

2.1 Distributed Cut Detection

Consider a sensor network as a time-varying graph G(k) = (V(k), E(k)), whose node set V(k) represents the sensor nodes active at time k and the edge set E(k) consists of pairs of nodes (u, v) such that nodes u and v can directly exchange messages

between each other at time k. By an active node we mean a node that has not failed permanently. All graphs considered here are undirected, i.e., (i, j) = (j, i). The neighbors of a node i is the set Ni of nodes connected to i, i.e. $N_i = \{j|(i, j) \in E\}$. The number of neighbors of i, $|N_i(k)|$, is called its degree, which is denoted by $d_i(k)$. A path from i to j is a sequence of edges connecting i and j. A graph is called connected if there is a path between every pair of nodes. A component G_c of a graph G is a maximal connected subgraph of G (i.e., no other connected subgraph of G contains G_c as its subgraph).

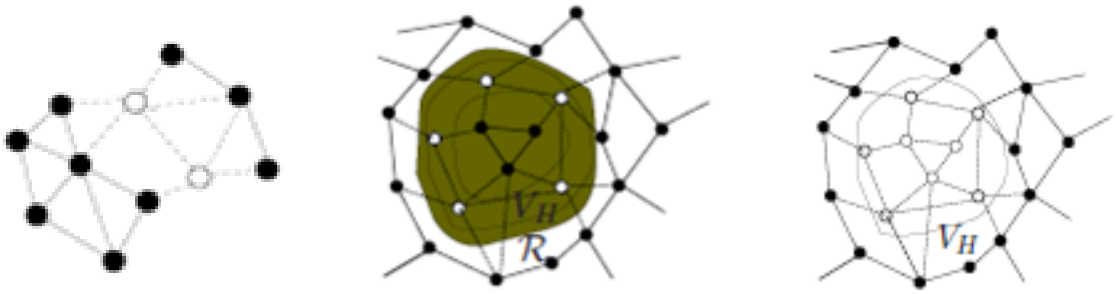

Fig. 1 Examples of cuts and holes.

In terms of these definitions, a cut event is formally defined as the increase of the number of components of a graph due to the failure of a subset of nodes. The number of cuts associated with a cut event is the increase in the number of components after the event. The problem we seek to address is twofold. First, we have to enable every node to detect if it is disconnected from the source .

III. EXPERIMENTAL DESIGN

3.1 Dos Detection using DCD Algorithm

The approach here is to exploit the fact that if the state is close to 0 then the node is disconnected from the source, otherwise not. In order to reduce sensitivity of the algorithm to variations in network size and structure, we use a normalized state. DOS detection part consists of steady-state detection, normalized state computation, and connection/separation detection. Every node i maintains a binary variable $\widehat{DOS}_i(k)$,which is set to 1 if the node believes it is disconnected from the source and 0 otherwise. This variable, which is called the DOS status, is initialized to 1 since there is no reason to believe a node is connected to the source initially. A node keeps track of the positive steady states seen in the past using the following method. Each node I computes the normalized state difference $\partial x_i(k)$ as follows:

$$\delta x_i(k) = \begin{cases} \frac{x_i(k)-x_i(k-1)}{x_i(k-1)} & \text{if } x_i(k-1) > \epsilon_{zero} \\ \infty & \text{otherwise} \end{cases}$$

where ϵ_{zero} is a small positive number. A node i keeps a Boolean variable PSSR (Positive Steady State Reached) and updates PSSR(k) ← 1 if $|\delta x_i(\kappa)| < \epsilon_{\Delta x}$ for $\kappa = k - \tau_{guard}, k - \tau_{guard} + 1, \ldots, k$ (i.e., for τ_{guard} consecutive iterations), where $\epsilon_{\Delta x}$ is a small positive number and τ_{guard} is a small integer. The initial 0 value of the state is not considered a steady state, so PSSR(k) = 0 for $k = 0, 1, \ldots . \tau_{guard}$. Each node keeps an estimate of the most recent "steady state" observed, which is denoted by $\hat{x}^{ss}_i(k)$. This estimate is updated at every time k according to the following rule: if PSSR(k) = 1, then $\hat{x}^{ss}_i(k) \leftarrow x_i(k)$, other-wise $\hat{x}^{ss}_i(k) \leftarrow \hat{x}^{ss}_i(k-1)$. It is initialized as $\hat{x}^{ss}_i(0) = \infty$.

Every node i also keeps a list of steady states seen in the past, one value for each unpunctuated interval of time during which the state was detected to be steady. Each node computes a normalized state $x^{norm}_i(k)$ as:

$$x_i^{norm}(k) := \begin{cases} \frac{x_i(k)}{\hat{x}_i^{ss}(k)} & \text{if } \hat{x}_i^{ss}(k) > 0 \\ \infty & \text{otherwise} \end{cases},$$

where $\hat{x}^{ss}_i(k)$is the last steady state seen by i at k, i.e., the last entry of the vector $\hat{x}^{ss}_i(k)$.

IV. PERFORMANCE EVALUATION

Two important metrics of performance for the DCD algorithm are (1) detection accuracy, and (2) detection delay. Detection accuracy refers to the ability to detect a cut when it occurs and not declaring a cut when none has occurred. DOS detection delay for a node i that has undergone a DOS event is the minimum number of iterations (after the node has been disconnected) it takes before the node switches its DOS_i flag from 0 to 1. In detecting DOS (disconnection from source) events, two kinds of inaccuracies are possible. A DOS0/1 error is said to occur if a node concludes it is connected to the source. A DOS1/0 error is said to occur if a node concludes that is disconnected from the source while in fact it is connected.

The algorithm's effectiveness is examined by evaluating the probabilities of the four types of possible errors enumerated above, as well as the detection delays. The probability of DOS0/1 error at time k is the ratio between the number of nodes that incur a DOS0/1 error (who believe they are connected but are not) at that time to the number of nodes that are disconnected from the source at that time. Probability of DOS1/0 error at k is the ratio between the number of nodes that incur a DOS1/0 error (who believe they are disconnected from the source but are in fact connected) to the number of nodes that are connected to the source at that time.

4.1 **Parameter Selection**

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

TABLE 2
DOS detection performance for the networks shown in Figure

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

4.2 **DOS Detection Performance**

In simulations with each of the five networks, the node failures occur at k=100. Performance of the DOS detection part of the algorithm in terms of error probabilities and detection delays are summarized in Table 2. The error probabilities shown are the ones that are empirically computed at k=60 and k=160, i.e., 60 iterations after deployment and after the node failures occurred, respectively. The mean and standard deviation of DOS detection delay for a network are computed by averaging over the nodes that detected DOS events. We see from Table 2 that the algorithm is able to successfully detect initial connectivity to the source and then DOS events for all the five networks without requiring the parameters to be tuned for each network individually.

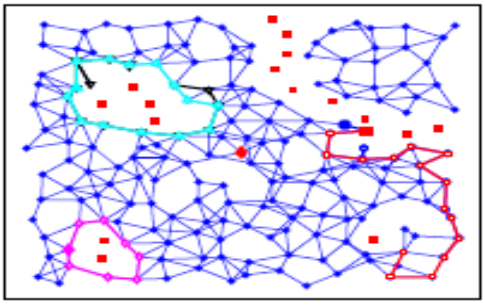

Fig. 2 The path of the probe messages in the network

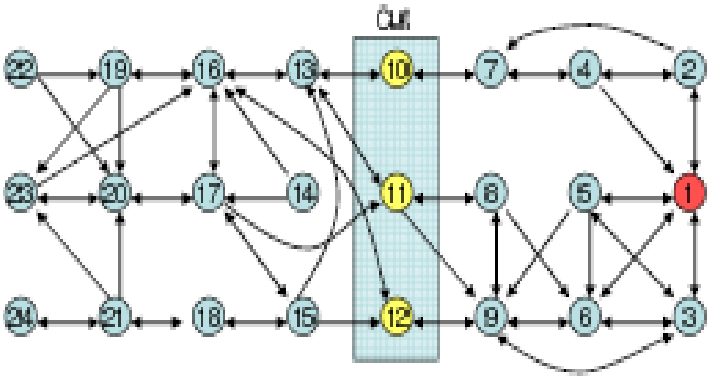

Fig. 3 The network for the outdoor deployment

All nodes disconnected from the source detected the DOS event correctly. The mean DOS detection delay is 19 iterations, with a standard deviation of 4. The DOS detection delays can be substantially reduced by choosing a larger value for €zero.

V. CONCLUSION

The DCD algorithm we propose here enables every node of a wireless sensor network to detect DOS (Disconnected frOm Source) events if they occur. 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. Application of the DCD algorithm to detect node separation and reconnection to the source in mobile networks could be the enhancement to this topic.

REFERENCES

[1] G. Dini, M. Pelagatti, and I. M. Savino, “An algorithm for reconnecting wireless sensor network partitions,” in *European Conference on Wireless Sensor Networks*, 2008, pp. 253–267.

[2] N. Shrivastava, S. Suri, and C. D. T´oth, “Detecting cuts in sensor networks,” *ACM Trans. Sen. Netw.*, 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,” in *First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (IEEE SECON 2004)*, Oct. 2004, pp. 489–497.

[4] M. N. Halgamuge, M. Zukerman, K. Ramamohanarao, and H. L. Vu, "An estimation of sensor energy consumption, "*Progress In Electromagnetics Research B*, Vol. 12, 259-295, 2009.

[5] Sankarasubramaniam, Y., I. F. Akyildiz, and S. W. McLaughlin, "Energy efficiency based packet size optimization in wireless sensor networks," *Proc. IEEE Int. Sensor Network Protocols and Applications Conf.*, Vol. 1, No. 8, 2003.

[6] Zou, Y. and K. Chakrabarty, "Target localization based on energy considerations in distributed sensor networks," *Proc. IEEE Int. Sensor Network Protocols and Applications Conf.*, Vol. 51, No. 58, 2003.

