ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

Slit Spotting in Wireless Sensor Networks

[I]M.Venkateswarlu [II]K.Anu Priya
[I]Student, [II]Assistant Professor
[I,II]Dept. of IT, LBRCE, JNTUK University, Myalavaram, Andhra Pradesh, India

Abstract
In recent trend, wireless sensor networks are beginning to be deployed at an accelerated pace. A wireless sensor network is a wireless network consisting of spatially distributed autonomous devices utilizing sensors to monitor physical or environmental conditions. But a few nodes may crash because of mechanical, electrical problems, environmental degradation, battery depletion, or hostile tampering, which is called a "CUT". The issue of detecting cuts by the neighboring nodes of a wireless sensor network The proposed algorithm will attain (i) every node to detect when the connectivity to a specially designated node has been lost (ii) one or more nodes to detect the occurrence of the cut. The algorithm is distributed and asynchronous: every node essential to communicate with only those that are within its communication range. The algorithm is rest on iterative computation of a fictitious "electrical potential" of the nodes. The convergence rate of the underlying iterative scheme is maverick of the size and structure of the network.

Keywords
Wireless sensor networks, Cut detection, electrical potential.

I. Introduction

A wireless sensor and actuator network (figure 1) is a collection of small randomly dispersed devices that provide three essential functions; the propensity to monitor physical and environmental conditions, often in real time, such as temperature, pressure, light and humidity; the ability to operate devices such as switches, motors or actuators that control those conditions; and the ability to yield efficient, reliable communications via a wireless network. The implementation of this last capability is the most eccentric to WSANs. Since they are designed for low traffic monitor and control applications, it is not necessary for them to console the high data throughput requirements that data networks like Wi-Fi require. Typical WSAN over-the-air data rates range from 20 kbps to 1 Mbps. Consequently they can operate with much lower power consumption, which in turn allows the nodes to be battery powered and physically small. Wireless sensor networks (WSNs) are a promising technology for monitoring extensive regions at high spatial and temporal resolution. However, the compact size and low cost of the nodes that makes them attractive for widespread deployment also causes the disadvantage of low-operational reliability. Failure of a set of nodes will diminish 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 finding 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.

WIRELESS sensor network (WSN) typically consists of a large number of small, low-cost sensor nodes distributed over a large area with one or possibly more powerful sink nodes convocation readings of sensor nodes. The sensor nodes are integrated with sensing, processing and wireless communication capabilities. Wireless sensor networks (WSNs) have emerged as a promising new technology to monitor large regions at high spatial and temporal resolution. Virtually any physical variable of interest can be monitored by equipping a wireless device with a sensor and networking these sensors together with the help of their sson-board wireless communication capacity.

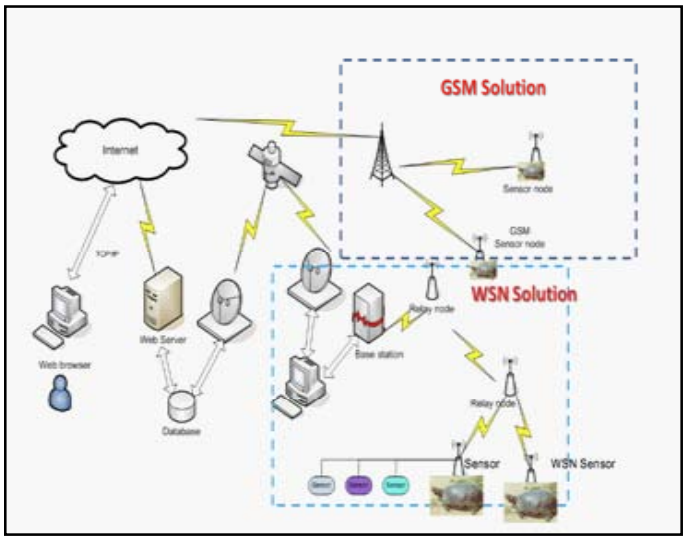

Fig.1: WSN architecture

Although these advances are expected to extend the lifetime of the wireless sensor nodes, due to their extremely limited energy budget and environmental degradation, node failure is expected to be quite common. This is especially true for sensor networks deployed in harsh and hazardous situations for critical applications, such as forest fire monitoring. In addition, the nodes of a sensor network deployed for defense applications may be subject to malicious tempering. When a number of sensors crashed, whether due to running out of energy, environmental degradation, or malicious intervention, the resulting network topology may become detached.

II. Literature Review

The recent advancements in MEMS technologies and low power, short range radio have enabled a rapid development of wireless sensor network in the last few years. These advancements have practically feasible and easy to deploy large scale, low power, and inexpensive networks. In near future the ever-present deployment of sensor networks is predicted .Sensor nodes typically route measurements to the base station in a tree-like fashion; the base station is at the root of the tree. The Base Station can also serve as a gateway for WSNs to exchange data and control information with other networks .There are many sensor network issues discussed

by reaches to resolve:

A. Concept of Boundary

Boundary of a sensor network means, the nodes residing on the edges of a sensor network or of the holes inside the network. Suppose we have a sensor network p with all nodes deployed randomly. There can be two kinds of nodes; the nodes which reside on the boundary and the interior nodes. There may be some holes in the network as well. Boundary and hole detection is an important problem in sensor networking applications. It is important for keeping track of the coverage range and events entering or leaving the region and for tracking communication with outside environment. It can be used for extracting further information about the structure and health of the network which is useful for routing, guiding and management purposes .

B. Holes Inside the Network

The group of destroyed sensors in a network creates a hole .The hole boundary separates all the faulty sensor nodes from the working nodes. A node fails mainly due to power depletion, however it could also happen due to physical destruction, and fluctuation in density or due to the existence of an obstacle or event .There can be two kinds of boundary nodes The nodes which reside on the boundary of the whole network and the nodes which resides on the boundary of a hole. The nodes which reside on the boundary of the network are termed as Network -Boundary Nodes and enclose all the nodes which constitute the whole network. On the other hand the nodes residing on the boundary of the hole(s) are termed as Hole- Boundary Nodes and enclose the destroyed sensors only. A network with no holes has only Network-Boundary Nodes. It could be destroyed nodes or it could be working nodes. The destroyed nodes are enclosed by Hole-Boundary Nodes while rest of the working nodes are the real interior node. Network-Boundary Nodes encloses all the nodes within the network. The hole identification is helpful in efficient routing and can help in geographic multi-casting. Information about the holes can be used to help path migration and to know when and how to shift the path from one side of the hole to the other side [4]. It can be also used in the information storage mechanism in sensor networks, such as the Geographic HashTable(GHT) and DIMENSIONS. Once holes are detected it could be easy to deploy nodes in the expected regions.

III. Related Work

Not only identifying the cuts in WSN detecting intruders is also essential for detecting intruder. We are developing two sensing models. Single-sensing detection: It is said that the intruder is detected under the single-sensing detection model if the intruder can be recognized by using the sensing knowledge from one single sensor. Multiple-sensing detection: the intruder can only be identified by using cooperative knowledge from at least k sensors.

IV. Problem Statement

A. Existing system

1) When a node u is disconnected from the source, we say that a Detach From Origin (DFO) event has occurred for u.
2) When a cut appear in the network that does not separate a node u from the source node, we say that Join, but a Slit Happen Abroad (JSHA) event has occurred for you.

B. Proposed system

We propose a distributed algorithm to detect cuts, named the Detach From Origin (DFO) algorithm.The algorithm allows each node to detect DFO events and a subset of nodes to detect JSHA events. We propose is distributed and asynchronous: it includes only local communication between neighboring nodes, and is robust to temporary communication failure between node pairs. The convergence rate of the computation is unique of the size and structure of the network.

1. Definitions and Problem Formulation

Time is measured with a discrete counter k= - ∞ ,…,-1,0,1,2,…. We model a sensor network whose node set represents the sensor nodes active at time k and the edge set ε(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. The problem we seek to address is twofold. First, we want to enable every node to detect if it is disconnected from the source (i.e., if a DFO event has occurred). Second, we want to enable nodes that lie close to the cuts but are still connected to the source (i.e., those that experience JSHA events) to detect JSHA events and alert the source node. There is an algorithm-independent limit to how accurately cuts can be detected by nodes still connected to the source, which are related to holes. provides a motivating illustration. This is discussed in detail in the Supplementary Material including formal definitions of "hole" etc. We therefore focus on evolving methods to distinguish small holes from large holes/cuts. We allow the possibility that the algorithm may not be able to tell a large hole (one whose circumference is larger than lmax) from a cut, Max discussion on hole detection part is limited to networks with nodes deployed in 2D.

2 State Update Law and Electrical Analogy

The DSS 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 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 DFO events have occurred (i.e., if they are disconnected from the source node).

To detect JSHA 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 JSHA events, since failure of nodes that do not cause a cut also induce changes in the potentials of their neighbors. Therefore, JSHA 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 JSHA events then alert the source node about the cut.

Every node keeps a scalar variable, which is called its state. The state of node i at time k is denoted by $x_i(k)$. Every node i initializes its state to 0, i.e., $x_i(0) = 0$, i. During the time interval between the kth and k+1th iterations, every node i broadcasts its current state $x_i(k)$ and listens for broadcasts from its current neighbors. Let N_i

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

(k) be the set of neighbors of node i at time k. Assuming successful reception, i has access to the states of its neighbors, i.e., $x_j(k)$ for j $N_i(k)$, at the end of this time period. The node then updates its state according to the following state update law (the index i=1 corresponds to the source node), where the source strength s (a positive number) is a design paramete

$$x_i(k+1) = \frac{1}{d_i(k)+1}\left(\sum_{j\in N_i(k)} x_j(k) + s\mathbf{1}_{\{1\}}(i)\right),$$

Where $d_i(k) := |N_i(k)|$ is the degree of node i at time k, and $1_A(i)$ is the indicator function if the set A. That is, $1\{1\}(i) = 0$ if i 1.

To understand the state update law's relation to the electrical analogy described earlier, given an undirected graph g= (v, ε), imagine a fictitious graph $G^{elec}= (v^{elec}, \varepsilon^{elec})$ as follows: The node set of the fictitious graph is $v^{elec}=v \{g\}$, where g is a fictitious grounded node; and every node in V is connected to the grounded node g with a single edge, which constitute the extra edges in ε^{elec} that are not there in E. Now an electrical network (G^{elec}, 1) is imagined by assigning to every edge of G^{elec} a resistance of 1.

When the sensor network G is connected, the state of a node converges to its potential in the electrical network (G^{elec}, 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.

It 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 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.

3. DFO Spotting

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. DFO detection part consists of steady-state detection, normalized state computation, and connection/separation detection. 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 $\delta x_i(k)$ as follows:

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

Where $\in_{zero}$ is a small positive number. A node i keeps a Boolean variable Positive Steady State Reached (PSSR) and updates PSSR (k)←1 if $|\delta x_i(k)| < \in \Delta_x$ for $k=\tau_{guard}, k-\tau_{guard}+1, \ldots, k$ (i.e., for τ_{guard} consecutive iterations), where $\in \Delta_x$ 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,…, τ_{guard}.

Each node computes a normalized state as

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

Where x is the last steady state seen by i at k , i.e., the last entry of the vector. If the normalized state of i is less than $\in_{DOS}$, where $\in_{DOS}$ is a small positive number, then the node declares a cut has taken place: DFO_i ←1If the normalized state is 1, meaning no steady state was seen until k, then DFO_i (k) is set to 0 if the state is positive (i.e. ,$x_i(k) > x \in_{zero}$) and 1 otherwise.

3. JSHA Spotting

The algorithm for detecting JSHA events relies on finding a short path around a hole, if it exists, and is partially inspired by the jamming detection algorithm proposed in [6]. 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. T he pseudo code for t he algorithm that decides when to initiate a probe is included in Section2 of the Supplementary Material, Each PROBE message p contains the following information:

1. a unique probe ID,
2. probe centroid Cp
3. destination node,
4. path traversed (in chronological order),
5. the angle traversed by the probe around the centroid. The

information required to compute and update these probe variables necessitates the following assumption for CCOS detection. 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 determined to initiate, if any. The manner in which the information in each of the probes in its list is updated by a node is described in Section 2 o f the Supplementary Material.

IV. Performance Evaluation

Performance of the DSS algorithm was tested using MATLAB simulations (conducted in a synchronous manner) and then on a

real WSN system consisting of micaZ motes [7]. Two important metrics of performance for the DSS 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. DFO detection delay for a node i that has undergone a DFO event is the minimum number of iterations (after the node has been disconnected) it takes before the node switches its DFO i flag from 0 t o 1. JSHA detection delay is the minimum number of iterations it takes after the occurrence of a cut before a node detects it. In detecting disconnection from source (DFO) events, two kinds of inaccuracies are feasible. A DFO 0/1 error is said to occur if a node concludes it is connected to the source while it is in fact disconnected, i.e., node i declares d DFO to be 0 while it should be 1. A DFO 1/0 error is said to occur if a node concludes that is disconnected from the source while in fact it is connected. In JSHA detection, again two kinds of inaccuracies are possible. A JSHA 0/1 error is said to occur when cut (or a large hole) has occurred but not a single node is able to recognize it. A JSHA 1/0 error is said to occur when a node concludes that there has been a cut (or large hole) at a particular location while no cut has occupied place near that location.

The algorithm's effectiveness is analyzed by evaluating the probabilities of the four types of possible errors enumerated above, as well as the detection delays. The probability of DFO 0/1 error at time k is the ratio between.

Table 1: List of Parameters that Have to Be Provided to the Nodes

Symbol	Name/description	Value
δ	Source strength	80
$\in$ zero	Valuebelow which the state is considered to be 0	10^{-8}
$\in$ zero	is considered zero Value below which the normalized state	10^{-2}
$\in \Delta_x$	Value below which the normalized state is considered zero	10^{-3}
τ guard	Time during which the normalized state difference has to be below x for the state to be considered steady	4
τ drop	Number of failed consecutive transmissions before a neighbor is declared to have failed.	5
ι max	Maximum path length for a prode	20

A. DFO Dispense Performance

In simulations with each of the five networks, the node failures occur at k ¼ 100. Performance of the DFO 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 ¼ 60 and k ¼ 160, i.e., 60 iterations after deployment and after the node failures occurred, respectively. The mean and standard deviation of DFO detection delay for a network are computed by averaging over the nodes that detected DFO events. We see from Table 2 that the algorithm is able to successfully detect initial connectivity to the source and then DFO events for all the five networks without requiring the parameters to be tuned for each network individually. Fig. 2 shows the path of the probes and their originating nodes in the network. 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 JSHA 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 JSHA event by the absorbing node. The probability of a JSHA 1/0 error in this case is therefore 0.33.

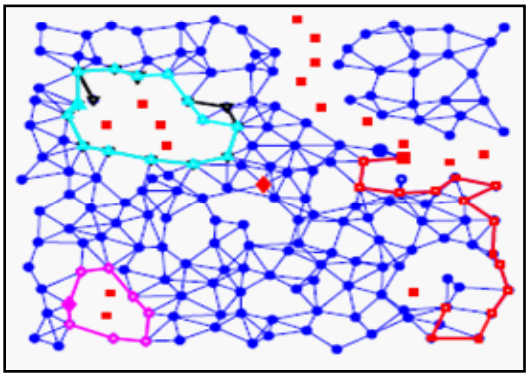

Fig.2: the path of the probe messages in the network.

B. System Execution and Evaluation

The algorithm was implemented using the nesC language on micaZ motes [7] running the TinyOS operating system [8]. The code uses 16 KB of program memory and 719 B of RAM. The system executes in two phases: the Reliable Neighbor Discovery (RND) phase and the DSS Algorithm phase. In the RND phase each mote broadcasts a beacon within a fixed time interval of 5 s for 15 such intervals. Upon receiving a beacon, the mote updates the number of beacons received from that particular sender. To determine whether a communication link is established, each mote first computes for each of its neighbors the Packet Reception Ratio (PRR), defined as the ratio of the number of successfully received beacons and the total number of beacons sent by a neighbor. A neighbor is deemed reliable if the PRR > 0:8. Next, the DSS algorithm executes.

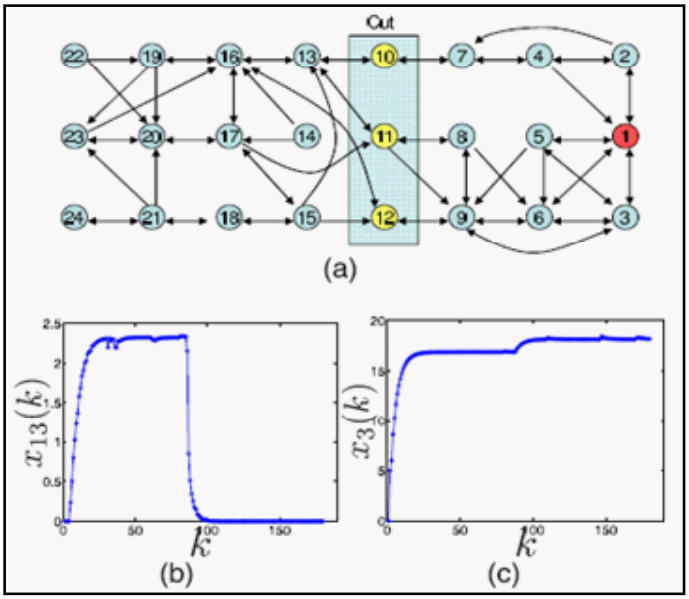

Fig. 3: (a) The network for the outdoor deployment. (b)-(c)The states of Nodes.

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

V. Conclusion

The presented algorithm performs expertly emerging with large class of Graphs of varying size, structure and without requiring changes in parameters. It assured to detect connection and disconnection to the source node with error-free. A key strength of the DCD algorithm is that the convergence rate of the underlying iterative approach is quite fast and discrete of the size and structure of the network, which makes detection using this algorithm.

References

[1] *G.Dini,M.Pelagatti, a d 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,"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 i n 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,RSzewczyk,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.*

Author Profile

Mallela.Venkateswarlu pursuing M. Tech Degree in Software Engineering from Lakireddy Balireddy College of Engineering, Vijayawada,AP, INDIA. Area of Interest: Networking, Image Processing.