

An Innovative Graded Snip Discovery protocol for Secure Data Transmission in multi hop path Networks

Dr. M. Sreedevi¹ and Dr. P. Chandra Babu Naidu²

¹Department of Computer Science and Engineering

² Department of Electrical and Electronics Engineering

Madanapalle Institute of Technology & Science, Madanapalle,
Andhra Pradesh, India

dr.msreedevi14@gmail.com and paduchuri.chandrababu@gmail.com

Abstract

Wireless Sensor Networks (WSNs) are challenging large regions at high spatial and temporal resolution. 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 “snip”. We considered the problem of detecting snips 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. Since a snip may or may not separate a node from the source node, we distinguish between two distinct outcomes of a snip for a particular node. When a node is disconnected from the source, identified that an SOS (Separated from Source) event has occurred for u.

When a snip occurs in the network that does not separate a node u from the source node, LSOS (Link, but a Snip Occurred Somewhere) event has occurred for u . By snip detection we can (i) discover by each node of a SOS event when it occurs, and (ii) discover of LSOS events by the nodes close to a snip, and the approximate location of the snip.

Our innovative algorithm to discover snips, named the Graded Snip Discovery (GSD) algorithm, allows each node to detect SOS events and a subset of nodes to detect LSOS events.

Key Words and Phrases: Wireless Sensor Networks (WSN), snip, network separation, multi hop

1 Introduction

Wireless sensor networks (WSNs) are a promising technology for monitoring huge areas next to great spatial and of time to resolve. 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 [11] in the network. Resulting a snip due to such failures and set of nodes disconnected from the rest.

To see the benefits of a snip discovery [15] [17] 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. A method of repairing a disconnected network by using mobile nodes has been proposed in [1]. Algorithms for discovering snips, as the one proposed here, can serve as useful tools for such network repairing methods. A review of prior work on snip discovery in sensor networks, e.g. [2], [3], [4] and others is included in the Supplementary Material.

2 Related Work

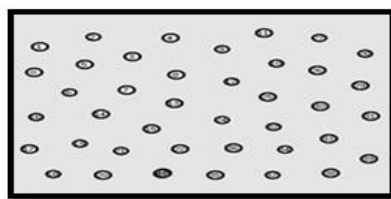
In a network with excessive loss and/or prolong, comparable to wireless networks, conventional TCP suffers from performance degradation because of conventional retransmissions and reordering [11], [20]. On cell phones, the concern could also be exacerbated due to restricted vigor [1]. Some options [2], [3] optimize congestion manipulate to atone for the performance degradation triggered by way of packet loss. Different efforts [4] [5] were made to introduce network coding into TCP over wireless networks. Huang et al. [4] show that community coding helped decrease packet loss likelihood and scale back the retransmissions, thus making improvements to the throughput. Sundararajan et al. [5] advise a scheme to combine random linear community coding with TCP. In comparison with the literature work, FMTCP employs coding mechanisms that essentially hinder retransmissions and reduces reordering without influencing the fairness of transmission.

Wireless Multimedia Sensor Networks (WMSNs) [14],[16],[19] has many challenges such as nature of wireless media and multimedia information transmission. Consequently traditional mechanisms for network layers are no longer acceptable or applicable for these networks. Wireless sensor network can get separated into multiple connected components due to the failure of some of its nodes, which is called a “snip”. Existing snip discovery system deployed only for wired networks having single path routing approach and it is Unsuitable for dynamic network reconfiguration.

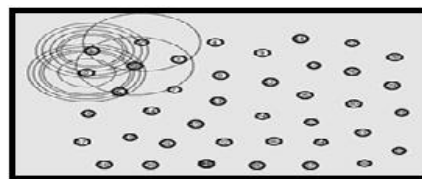
3 Graded Snip Discovery

The Graded snip Discovery algorithm allows each node to be detect to the Disconnected from Source events and the subset of nodes to be detect the LSOS (Linked, but a Snip Occurred Somewhere) events. These algorithm to be used for graded to the number of nodes and asynchronous, then it is will be involves to only local communication between and neighboring nodes, and it is robust to temporary communication failure between node pairs. A key component of the Graded Snip Discovery algorithm is a graded iterative nodes [18].

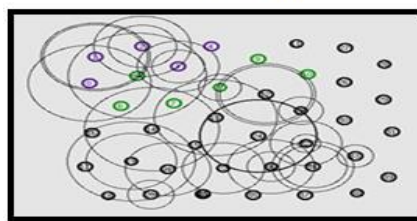
These nodes are computational step by step through the nodes are their (fictitious) electrical potentials. The computation is an independent of the convergence rate and size of the structure of the network. Phase is restrained to networks with nodes deployed in 2nd.



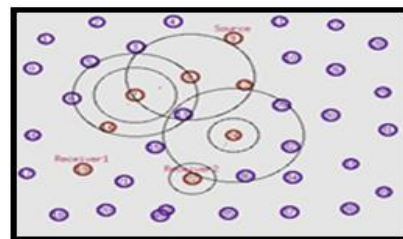
(a) Search for nodes.



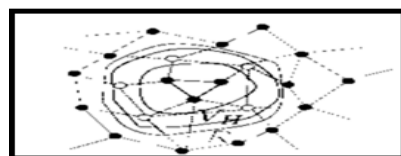
(b) Swap one node to another node.



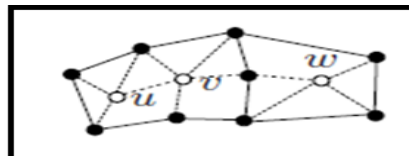
(c) Searching for neighbor nodes



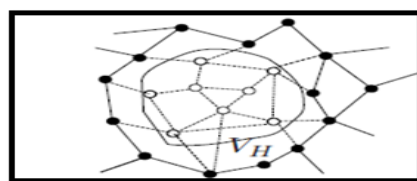
(d) Source and Destination Nodes to set for two paths



A Snip



(e) Two holes



(f) A hole

Fig.1. Examples of snips and holes. Filled circles represent active nodes and unfilled filled circles represent failed nodes. Solid lines represent edges, and dashed lines represent edges that existed before the failure of the nodes. The hole in (d) is indistinguishable from the snip in (b) to nodes that lie outside the region R .

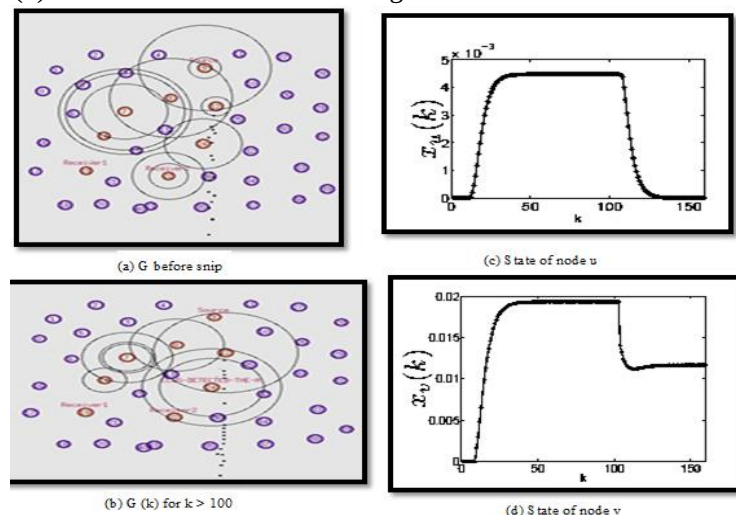


Fig.2. (a)-(b): A sensor network with 200 nodes, shown before and after a snip. The snip occurs, at $k=100$, due to the failure of the nodes shown as red squares. The source node is at the center. (c)- (d): The states of two nodes u and v as a function of iteration number.

When the sensor community G is connected, the state of a node converges to its advantage inside the electrical network, which is a constructive variety. If a scale down occurs, the potential of a node that's disconnected from the supply is 0; and that is the valued at its state converges to. If reconnection happens after a diminish, the states of reconnected nodes again converge to optimistic values. For this reason, a node can display whether or not it's connected or separated from the give with the support of inspecting its state. The above description assumes that everyone updates are applied synchronously. In observe, peculiarly with wireless conversation, an asynchronous change is primary.

The algorithm will also be comfortably accelerated to asynchronous environment by way of letting each node preserve a buffer of the final bought states of its neighbors. If a node does now not receive messages from a neighbor for the duration of the interval between two iterations, it updates its state utilizing the final efficiently obtained state from that neighbor. Within the asynchronous surroundings each node continues a neighborhood iteration counter a good way to differ from these of extraordinary nodes with the help of arbitrary amount.

Assess to endorse the evolution of the node states in a community of 200 nodes when the states are computed making use of the replace regulation described above. The supply node is on the core. The nodes shown as red squares in assess 2(b) fail at good enough=one hundred, and thereafter they do not participate in conversation or computation. Determine 2 (e-f) suggests the time evolution of the states of the 2 nodes u and v , which are marked by the use of circles in assess 2(b). The state of node u (that's disconnected from the deliver when you consider that of the scale back) decays to zero after attaining a optimistic worth, whereas the state of the node v (which remains to be linked after the snip) stays constructive.

The Graded Snip Discovery (GSD) Algorithm

Procedure GSD(C) consider S =Source node; Neighbors of node S are A, B . ack=active; dack=inactive

If the node A is active i.e. ack state then

Wait for 400ms.

Send file to node A .

else if the node A is deactive node failed i.e. dack state then
file sending to A failed.

If the node B is active i.e. ack state then

Wait for 500ms.

Send file to node B .

else if the node B is deactive node failed i.e. dack state then
file sending to B failed.

Here we briefly describe the proposed GSD algorithm [8]. One of the nodes of the network is a specially designed node which is always active called as "source node". Let $G = (V, E)$ denote the undirected sensor network that consists of all the nodes and edges of G that are active at time k .

where $k = 0, 1, 2 \dots$ is an iteration (repetitive) counter. Every node p of node set V maintains a scalar state $(x_p)(k)$ that is iteratively updated. Let the nodes of the graph G execute the GSD algorithm with initial condition as $(x_u)(0) = 0 \forall p \in V$.

1. If no snip occurs or else no node fails then state of every node converges to a positive number.
2. If a snip occurs at a time $T \geq 0$ which separates the graph G into N connected components $(G_s), \dots, G_N$, where the component (G_s) $((V_s), (E_s))$ contains the source node, then
 - (a) The state of every node disconnected from the source node converges to 0 (deactive) and
 - (b) The state of every node in (V_s) converges to a positive number.

Hence by monitoring the states of the nodes one can know about the status of the network connection. For effectiveness we proposed a prototype model by taking small number of nodes and their corresponding edges in the graph G . Hence the nodes can effectively detect first if there is any snip occurred and second they are still connected to source. We modified this algorithm by adding additional parameters to reduce redundant information at destination. We designed it in such a way that once the file is sent from a node, it is sent to its respective neighbors so that each and every node has the information. If there is any node failure from where information cannot be forwarded and a snip is detected, the information at the nodes is combined and then sent to the destination without the occurrence of redundancy. This approach is simulated successfully in Java environment and the expected results have found.

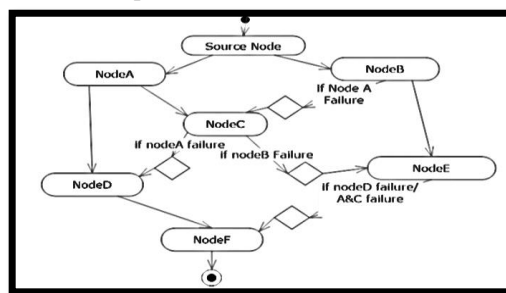


Fig (3). GSD algorithm

4 PERFORMANCE EVALUATION

A. Choice of parameters

The parameters ϵ_{zero} , ϵ_{DOS} , $\epsilon_{\Delta x}$, T^{guard} , T^{drop} , l^{max} and $r^{\Delta\text{ss}}$ have to be specified to all the nodes a-priori. The parameter s has to be specified only to the source node. A detailed discussion on the choice of parameters and their effect on the GSD algorithm's performance is provided with the Supplementary Material. The main conclusions are that (i) ϵ_{zero} should be chosen as small as possible and s should be chosen as large as possible to minimize discovery error, (ii) a smaller value of the parameter ϵ_{DOS} decreases probability of DOS1/0 error but increases DOS discovery delay, and (iii) the rest of the parameters do not seem to have a significant effect on the algorithm's performance. The values of the parameters used in all the experimental evaluations reported in this paper are shown in Table 1.

List of parameters that have to be provided to the nodes. The numerical values shown here are used for all simulations and experimental evaluations reported in this document.

TABLE 1

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 is normalized state is considered zero	10^{-3}
$\epsilon_{\Delta x}$	Value below which is 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
l^{max}	Maximum path length for a probe	15
$r^{\Delta\text{ss}}$	Threshold ratio of change in the steady state for probe initiation	0.35

B. DOS Discovery Performance

DOS discovery performance for the networks shown in Figure 4. The two values of the probability shown in each cell correspond to $k=60$ and $k=160$, respectively.

TABLE 2

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

In simulations with each of the five networks, the node failures occur at $k=100$. Performance of the DOS discovery part of the algorithm in terms of error probabilities and discovery 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 discovery 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.

C. LSOS Discovery Performance

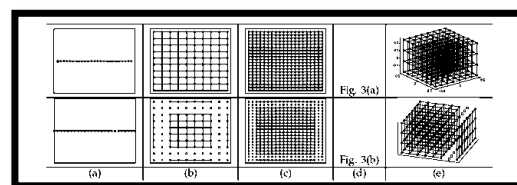


Fig. 4. Five networks before and after node failures: (a) 25-node 1D line network, (b) 100-node 2D grid, (c) 400-node 2D grid, (d) 200-node 2D random network, and (e) 256-node 3D grid ($8 \times 8 \times 4$).

Recall that the LSOS discovery part of the algorithm is not applicable to 3D networks, so it was only tested on networks 4(a)-(d). As a specific example, Figure 5 shows the path of the probes and their originating nodes in the network of Figure 4(d). Two probes were triggered by nodes close to the snip 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 LSOS 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 a LSOS event by the absorbing node. The probability of a LSOS1/0 error in this case is therefore 0.33. Table 3 summarizes the performance of the LSOS discovery part of algorithm (executed with parameter values shown in Tables 1). The LSOS discovery error probabilities are 0 except in case of the network.

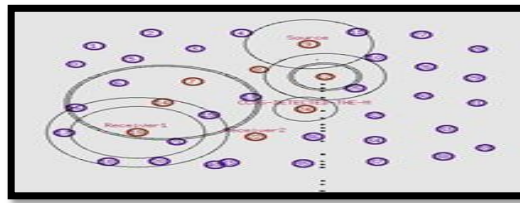


Fig. 5. The path of the probe messages in the network of Figure 4(d). Each probe path is marked with a distinct legend (circle, triangle, square, etc.), and the node that initiated the probe is shown as the one with the larger legend.

TABLE 3

Network	a	b	c	d
Prob(LSOS1/0 error)	0	0	0	0.33
Prob(LSOS0/1 error)	0	0	0	0
LSOS Delay	33	40	37	40

LSOS discovery performance for four networks in Figures 4(a)-(d). The error probabilities are at $k=160$.

Conclusion

The GSD algorithm we proposed here enables every node of a wireless sensor network to detect DOS (Disconnected frOm Source) events if they occur. Second, it enables a subset of nodes that experience LSOS (Link, but Snip Occurred Somewhere) events to detect them and estimate the approximate location of the snip in the form of a list of active nodes that lie at the boundary of the snip/hole. The DOS and LSOS 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 WSN system consisting of mica Z motes, show that the

algorithm works effectively with a 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 GSD 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.

References

- [1] Y. Cui, X. Ma, H. Wang, I. Stojmenovic, and J. Liu, "A survey of energy efficient wireless transmission and modeling in mobile cloud computing," *Mobile Network. Appl.*, vol. 18, no. 1, pp. 148–155, 2013.
- [2] C. Lai, K. Leung, and V. Li, "Enhancing wireless TCP: a serialized timer approach," in *Proc. IEEE INFOCOM*, 2010, pp. 1–5.
- [3] S. Mascolo, C. Casetti, M. Gerla, M. Sanadidi, and R. Wang, "TCP westwood: Bandwidth estimation for enhanced transport over wireless links," in *Proc. 7th Annu. ACM MobiCom*, 2001, pp. 287–297.
- [4] Y. Huang, M. Ghaderi, D. Towsley, and W. Gong, "TCP performance in coded wireless mesh networks," in *Proc. 5th Annu. IEEE SECON*, 2008, pp. 179–187.
- [5] J. Sundararajan, D. Shah, M. Médard, M. Mitzenmacher, and J. Barros, "Network coding meets TCP," in *Proc. IEEE INFOCOM*, 2009, pp. 280–288.
- [6] K. Kim, Y. Zhu, R. Sivakumar, and H. Hsieh, "A receiver-centric transport protocol for mobile hosts with heterogeneous wireless interfaces," *Wireless Netw.*, vol. 11, no. 4, pp. 363–382, 2005.
- [7] D. MacKay, "Fountain codes," *IEE Proc., Commun.*, vol. 152, no. 6, pp. 1062–1068, 2005.
- [8] *Distributed Cut Detection Algorithm* Jacinth Samuel December 12, 2013.
- [9] J. Iyengar, P. Amer, and R. Stewart, "Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths," *IEEE/ACM Trans. Netw.*, vol. 14, no. 5, pp. 951–964,

Oct. 2006.

[10] T. Dreibholz, M. Becke, H. Adhari, and E. Rathgeb, "On the impact of congestion control for concurrent multipath transfer on the transport layer," in *Proc. 11th IEEE ConTEL*, 2011, pp. 397–404.

[11] Sreedevi, M & Seshadri, R (2012). "A Framework to Improve Virtual Private Network Architecture for Secure Data Transmission Using CMT-PA and Multi-Homing." *International Review on Computers and Software*, 7, 1419-1427.

[12] P.Avanthi, M.Sreedevi (2015). "Smart suggest search engine with Location Sensing Recommendation." *International Journal of Applied Engineering Research*, 10, 40205-40208.

[13] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," in *Proceedings of international conference on Architectural support for programming languages and operating systems (ASPLOS)*, 2000

[14] A. D. Wood, J. A. Stankovic, and S. H. Son, "Jam: A jammed-area mapping service for sensor networks," in *IEEE Real Time System Symposium*, 2003.

[15] M. Hauspie, J. Carle, and D. Simplot, "Partition detection in mobile ad-hoc networks," in *2nd Mediterranean Workshop on Ad-Hoc Networks*, 2003, pp. 25–27.

[16] 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.

[17] 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.

[18] 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.

[19] P. Barooah, "Distributed cut detection in sensor networks," in *47th IEEE Conference on Decision and Control*, December

2008, pp. 1097 – 1102.

[20]http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICAZ_Datasheet.pdf.

