uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Milenko Jorgic

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.C.S.

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Localized Criteria for Detection of Critical Nodes and Link and K-Connectivity in Ad Hoc Networks

TITRE DE LA THÈSE / TITLE OF THESIS

Ivan Stojmenovic

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

M. Lanthier

A. Nayak

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Localized Criteria for Detection of Critical Nodes and Links and *K*-Connectivity in Ad Hoc Networks

by

Milenko Jorgic

Thesis submitted to the
School of Information Technology and Engineering
in partial fulfillment of
the requirements for the degree of

Master of Computer Science

Under the auspices of the
Ottawa-Carleton Institute for Computer Science

University of Ottawa
Ottawa, Ontario, Canada

January 2007

Your file Votre référence
ISBN: 978-0-494-32455-4
Our file Notre référence
ISBN: 978-0-494-32455-4

Acknowledgements

I would like to acknowledge a sincere gratitude to my research supervisor Prof. Dr. Ivan Stojmenovic for his continuing patronage throughout this effort. His guidance and numerous technical discussions we have had helped this research and learning experience to be successful and to get to this stage.

Quite often it was him who helped me to keep the focus on the larger problem and to work on solutions to associated challenging topics in this area. His hard work and continuing dedication to his research goals never stop to impress and inspire me. For all of his enduring support throughout all these years, I sincerely thank him.

Also I would like to thank my co-supervisor Professor Amiya Nayak. His chats and personal support was extremely useful in times when things did not look very well.

I would also like to thank Professor Marc Lathier from Carleton University for detailed and valuable suggestions on how to improve my thesis.

I have to express gratitude to my parents, my eternal inspiration. I thank them for all their support, for their faith in me and teaching that I should never give up.

Abstract

Abstract - Ad hoc network normally has critical connectivity properties before partitioning. The timely recognition is important in order to perform some data or service replication. Several existing centralized or globalized algorithms declare an edge or a node as critical if their removal will separate the network into several components. We introduce several localized definitions of critical nodes and critical links, using topological or positional information. A node is critical if the subgraph of p-neighbours of node (without the node itself) is disconnected. We propose three definitions of critical links, based on verifying common p-hop neighbours, loop length, and critical status of link endpoints, respectively. The experiments with random unit graph model of ad hoc networks show high correspondence of local and global decisions. Existing algorithms for testing k-connectivity are centralized. In this thesis, we introduce localized criteria for testing k-connectivity. In the first proposed local neighbor detection (LND) criterion, each node verifies whether or not itself and each of its p-hop neighbors have at least k neighbors. In the second local critical node detection (LCND) protocol, it also tests if the subgraph of its p-hop neighbours of a given node is k-connected. The third local subgraph connectivity detection (LSCD) protocol is based on communications between neighboring nodes to exchange the local decisions starting from k=1. All nodes declare themselves locally 1-connected. For k=2,3,..., iteratively, local decisions are propagated to p-hop neighbors. If node A is (k-1)-connected, all its p-hop neighbors are (k-1)-connected, and the graph consisting of p-hop neighbors of A (excluding A) is (k-1)-connected, then node A declares its neighborhood as k-connected. The experiments are carried with two ways of uniform generation of connected unit disk graphs. They show low percentage of false 'alarms', ability to locate critical areas in k-disconnected networks, and increased accuracy with increased local knowledge.

Table of Contents

Abstract iv

Preliminaries 10

Chapter 1...Introduction 11

1.1 Background and motivation 12

1.2 Problem Statement 13

1.3 Existing Solution 14

1.4 Assumptions and Limitations 16

1.5 Objectives and Contributions 17

1.5.1 CN criteria 17
1.5.2 CL criteria 18
1.5.3 LC criteria 19
1.5.4 LNC criteria 19
1.5.5 LND criteria 19
1.5.6 LCND criteria 20
1.5.7 LSCD criteria 20

1.6 Planned publications out of thesis 20

1.7 Simulation 21

1.8 Thesis Organization 22

Chapter 2 Literature review 23

Chapter 3 Localized algorithms for detection of critical nodes and links 30

3.1 CN (critical node) criteria 31

3.2 CL (critical link) criteria 34

3.3 LC (loop critical) criteria 35

3.4 LNC (link by node critical) criteria 37

Chapter 4 Localized criteria for detection of k-connectivity 39

4.1 Local neighbour detection (LND) criteria 40

4.2 Local critical node detection (LCND) criteria.....42
4.3 Local subgraph connectivity detection (LSCD) criteria.....44

Chapter 5 Performance Evaluation for detection of critical nodes and links.....45

5.1 Localized criteria for detection of critical nodes.....46
5.2 Localized criteria for detection of critical links.....49

Chapter 6 Performance evaluation for detection of k-connectivity.....54

6.1 Local_neighbour_detectio (LND).....58
6.2 Local subgraph connectivity detection (LSCD) 62
6.3 Local_critical_node_detection (LCND).....73

Chapter 7 Conclusions, Future work, and Open ideas.....85
Chapter 8 References 87
Chapter 9 Appendix.....89

List of Figures

Figure 1. Unit graph representation of multi-hop wireless network.....12
Figure 2. Example of *p*-hop neighbours in a unit graph.....33
Figure 3. Examples of critical links in a unit graph.....34
Figure 4. Example of loop length based critical links in unit graph.....37
Figure 5. Example of a 1-connected graph.....39
Figure 6. Example of 2-connected graph.....40
Figure 7. Example of 2-disconnected graph.....41
Figure 8. Example of 3-disconnected graph.....41
Figure 9. Example of 3-disconnected graph.....43
Figure 10. Example of 3-connected graph.....44
Figure 11. Example of a simulated network for d=15..... .53
Figure 12. Example of a simulated network for d=9.....61
Figure 13. Example of a simulated network for d=6.....65
Figure 14. Example of a simulated network for d=6.....66
Figure 15 Example of a simulated network for d=6.....75
Figure 16 Example of a simulated network for d=9.....76
Figure 17 Example of a simulated network for d=9.....77

List of Tables

Table 5-1. Detection ratios for p-top_CN, n= 500 nodes.....47
Table 5-2. Detection ratios for p-pos_CN, n= 500 nodes.....48
Table 5-2. Average numbers of critical nodes by p-top_CN and global algorithms.....48
Table 5-4. Average numbers of detected critical nodes by p-pos_CN and global algorithms.....49
Table 5-6. Detection ratios for p-top_CL criteria on connected graphs with 500 nodes.....49
Table 5-7. Detection ratios for p-pos_CL criteria on connected graphs with 500 nodes..... 50
Table 5-8. Average number of detected critical links by p-top_CL and global algorithms.....50
Table 6-1. Average number of detected critical links by p-pos_CL and global algorithms.....51
Table 6-2. Probability UDG is 1-connected for pairs (n, d).....51
Table 6-3. Probability that a connected UDG is 2-connected for pairs (n, d).....55
Table 6-4. Probability that a connected UDG is 3-connected for pairs (n, d).....56
Table 6-5. Detection ratio by LND.....56
Table 6-7. Detection ratio by LND.....58
Table 6-8. Detection ratio by LSCD.....59
Table 6-9. Detection ratio by LSCD.....63
Table 6-10. Detection ratio by LSCD.....63
Table 6-11. Detection ratio by LSCD.....64
Table 6-12. Detection ratio by LSCD.....67
Table 6-13. Detection ratio by LSCD.....67
Table 9-1. Detection ratio by LSCD.....81
Table 9-2 Detection ratio by LSCD..... 81
Table 9-3. Detection ratio by LSCD.....81
Table 9-4. Detection ratio by LSCD.....82
Table 9-5. Detection ratio by LSCD.....82
Table 9-6. Detection ratio by LSCD.....83
Table 9-7. Detection ratio by LSCD.....83
Table 9-8. Detection ratio by LSCD.....84

Table 9-9. Detection ratio by LSCD....84
Table 9-10. Detection ratio by LSCD....85
Table 9-11. Detection ratio by LCND....85
Table 9-12. Detection ratio by LCND....86
Table 9-13. Detection ratio by LCND....86
Table 9-14. Detection ratio by LCND....87
Table 9-15. Detection ratio by LCND....87
Table 9-16. Detection ratio by LCND....88
Table 9-17. Detection ratio by LCND....88
Table 9-18. Detection ratio by LCND....89
Table 9-19. Detection ratio by LCND....89
Table 9-20. Detection ratio by LCND....90
Table 9-21. Detection ratio by LCND....90
Table 9-22. Detection ratio by LCND....91
Table 9-23. Detection ratio by LCND....91
Table 9-24. Detection ratio by LCND....92
Table 9-25. Detection ratio by LCND....92
Table 9-26. Detection ratio by LCND....92

Preliminary

In this section the explanation of following expression is given:

1. Network Partitioning (Partition Detection) - is a form of network failure. A single connected network topology breaks apart into two or more network topologies separated from each other. Nodes within each partition are still able to communicate with each other but nodes in other partitions are unreachable.
2. Replication decisions – nodes decide to replicate (exchange) information between its neighbours in order to achieve fault tolerance
3. d – average number of neighbours
4. r – transmission radius
5. Unit Graph - A unit graph is a distance graph in which all edges are of length 1.

The simulation of the network and algorithms used for detection of network partitions and k-connectivity were implemented in C++ using OpenGL libraries. The simulation was done in a square area where nodes had random coordinates. The nodes were static (no movement on the nodes was allowed) and no nodes were making a transmission at the same time.

Chapter 1
Introduction

1.1 Background and Motivation

An ad hoc network is a collection of wireless mobile hosts forming a temporary network without the aid of any fixed infrastructure. They have potential application in civilian and military environments such as disaster relief, conference, wireless office, and battlefield. Ad hoc sensor networks for monitoring the environment are also being deployed.

In an ad hoc network a message sent by a node reaches all its neighboring nodes that are located at distances up to the transmission radius. A widely accepted basic graph-theoretical model for ad hoc networks is a *unit graph* model, defined in the following way. Two nodes A and B in the network are neighbors (thus joined by an edge) if and only if the Euclidean distance between their coordinates in the network is at most R, where R is the transmission radius which is equal for all nodes in the network. Due to the limited transmission radius, the routes between two nodes are usually created through several hops. Thus, it may be necessary to relay a packet over multiple nodes to reach the destination. In a connected (or 1-connected) graph, there is at least one path connecting every two nodes.

So far, the only available algorithms for partition detection of an ad-hoc network are globalized algorithms, which demand time and communication resources as well as a centralized infrastructure where the computation will take place. Also, by the time the centralized station finishes computing, in a highly mobile network, topology can easily change; hence the computed information can become obsolete. In order to obtain fast and accurate information on partition

detection, we need to resort to localized algorithms. Localized algorithms are fast, simple and there is no need for position information of all the nodes in a network. In localized protocols, each node makes its own decision based on the information available in its local neighborhood. More precisely we propose seven localized algorithms for detection of critical nodes, links and k-connectivity.

For instance, we experimented with 500 nodes in connected random unit graphs, in which over half of the locally estimated critical nodes and links were indeed globally critical even for p=1, where p is a p hop neighbour of a node, (the accuracy increases to over 70% for p=2 and over 80% for p=3), for an average number of neighbours ranging from 3 to 15. The errors mostly occur when alternative routes exist but are relatively long, and therefore may not provide satisfactory service in applications. Therefore our localized protocols provide faster and often more reliable partition warnings for possible timely replication decisions.

Suppose we have a user E and a server L (see Figure 1.). When user E requests a service or data from L, the request follows the path *ECBAIJL*. In this example, nodes *A, B* and *J,* and link *AB,* are critical, since removal (or movement) of any of them will partition the network. Node E may initiate some actions, such as replicating data from L in its own memory, or look for alternative services in a different part of the network, before it is too late.

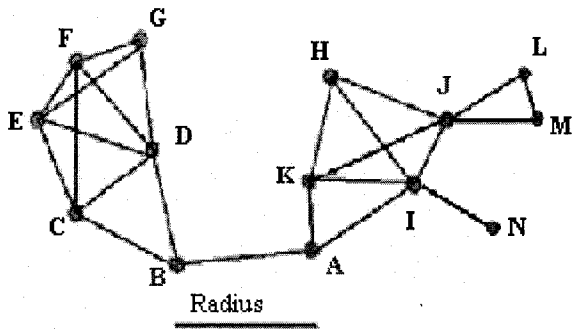

Figure. 1. Unit graph representation of multi-hop wireless network

1.2 Problem Statement

One of the most important problems in ad-hoc network is the problem of partition detection. Before an ad-hoc network separates, we have to decide if critical links or nodes should be reinforced or data between critical nodes or links replicated, hence keeping the integrity of the network. In order to avoid the loss of communication between the source and destination we need to discover all critical nodes and links so that the appropriate action can be taken. Current algorithms for detection of critical nodes and links are globalized in nature, which means that they require a central entity that possesses knowledge of all nodes/links in the network. This central entity is able to correctly detect all critical links and nodes. However in dynamic ad hoc and sensor networks, where mobility and frequent changes occur in sleep and active status of the nodes, such a central entity may not exist, or the collection of the network information to this entity may cause a huge communication overhead. Decisions may need to be taken in local neighbourhood, such as placing additional nodes for avoiding network partition. The purpose of this thesis is to show that,

with high probability, possible partition can be detected by localized algorithms, therefore greatly reducing communication overhead and the speed of detecting, allowing the network to replicate data or services in time if needed.

Connectivity is also one of the most important properties of a large-scale wireless ad hoc network. A network or graph is k-connected if it is still connected after the removal of any k-1 nodes. Fault tolerance in wireless ad hoc and sensor networks increases with increasing k for which the network is k-connected. Normally, a central entity (e.g., sink in a sensor network), knowing all nodes and edges of a graph, is able to make a unique decision on k-connectivity or k-disconnectivity of the whole graph. However, due to mobility and frequent changes such central entity may not exist, or the computation of network information to this entity may cause huge communication overhead. Such information on the whole network may also be unnecessary for the operation of ad hoc and sensor networks. Decisions may need to be taken based on connectivity properties in a local neighborhood, such as placing or activating additional sensors for better coverage and communication, or finding alternative services in ad hoc network due to local criticality. However, there is no unique definition of local k-connectivity, and, obviously, different nodes can make different decisions on local k-connectivity. Generally, each node makes a decision based on the information available from nodes located at most p hops away (that is, those with a shortest path between them of length at most p).

1.3 Existing Solutions

DFS (depth first search) algorithms were used to detect critical links in [ABS, T, GC]. These are centralized algorithms, and can be also implemented in a globally distributed manner. A centralized algorithm requires that a node should be aware of global topology. In practice, this method is inefficient and involves a quadratic (in number of nodes) communication overhead in order to update link information when nodes are moving, or when nodes change their status from active to sleeping and vice versa. In a globally distributed implementation, *DFS* can be performed in the network without global knowledge at any node, but with memorization at nodes.

We can modify this globalized algorithm in order to use it to discover k-connectivity. If the algorithm is ran and no links/node are declared critical, than we may conclude that the graph is 2-connected since removal of any single node in a graph will not affect the graph's connectivity.

This algorithm can be extended to test global 3-connectivity of a graph G. A detailed explanation of this algorithm is given later in this thesis.

Some of the existing solutions to delay or avoid network failure are proposed as follows: if critical nodes and links are detected, we might avoid the connectivity failure by brining another node to reinforce link. Other existing solutions suggest that a server should detect possible partitions and in each partition replicate vital information.

Another interesting solution proposes few replica allocation methods to improve data accessibility by replicating data items on mobile hosts. Also the problem of assigning transmission power to nodes so that the sum of powers is minimized and the network is k-connected was considered as well.

Few solutions were aimed at enhancing data access in an ad hoc network by detecting partitions in it [SCN]. They propose a data replication mechanism based on partition detection for allowing one node to access the data from another node even if connection is physically broken. Some solutions propose an approach in which nodes actively modify their trajectories to transmit messages [LR]. Also the problem of assigning transmission power to node so that the sum of powers is minimized and the network is k-connected was considered in [LR].

1.4 Assumptions and Limitations

The development of our proposals contains limitations and assumptions. We assume that each node uses omni-directional antennas so that it can receive a message from any of its neighbours. Each node is not aware of its neighbours coordinate position and the transmission pattern of every node is a circle with radius equal to the transmission radius R. We also make an assumption that there are no collisions between messages.

- The simulations for this work have the following assumptions:
- Ideal MAC and Physical layers.
- Transmission radius was the same for each node and was fixed.
- Only one broadcast at a time; no collision between messages.
- There were no obstacles between any two nodes; a message was received if and only if the distance between sending and receiving node was less than the transmission radius.
- All nodes were static while a broadcasting was in progress

- All nodes shared the wireless channel; when a node transmitted a message, all its neighbors heard the transmission.
- Assume omni-directional transmission was implemented. All neighbours get the same message.

1.5 Objectives and Contributions

In this thesis, we propose to apply localized algorithms instead of globalized ones for partition detection. Localized algorithms are distributed in nature and resemble greedy algorithms, where simple local behavior achieves a desired global objective. In this thesis we showed that each node/link could relatively accurately make a local decision and classify itself as critical or not. We introduced several localized definitions of detection of critical nodes and links, using topological or positional information. We propose the following four criteria: CN, CL, LC, LNC. The experiments (using random unit graph models of ad hoc networks) show high correspondence of local and global decisions.

We also describe three definitions and corresponding protocols for deciding local k-connectivity LND, LCND and LSCD. For some definitions (e.g., the first one given here), global k-connectivity may imply local k-connectivity at each node. However, for other definitions, globally k-connected networks may have nodes that are locally k-disconnected. This is due to additional connectivity to local neighbors being available by long routes in the whole network. The purpose of this thesis is to show that, with high probability, for uniform random node placements, the local decision at any node about k-connectivity corresponds to a global decision about k-

connectivity. This means that each node can make rather accurate estimation of global k-connectivity based on local information, therefore greatly reducing communication overhead and the speed of detecting global k-connectivity. We perform statistical analysis by randomly generating several topologies with widely distributed properties of transmission radius, node density and number of nodes.

1.5.1 CN criterion

The principal objective of this algorithm is to detect critical nodes with high accuracy. This algorithm can be used if a node does have position information of itself and its neighbours, or if it does not have any positional information just topological. As expected, if positional information is available the accuracy of the algorithm will be higher. For each node A, consider the subgraph of p-hop neighbours of A, where A and all its incident edges are excluded. In case of positional information, two nodes in that graph are connected if they are connected in the original graph. In case of topological information, two nodes in that graph are connected if they are connected in the original unit graph, and at least one of them is $(p\text{-}1)$-hop neighbor of A. A node A is p-critical if the corresponding subgraph of p-hop neighbours of A is disconnected. Based on information used, this is further specified as being a topologically or positionally p-hop critical node.

1.5.2 CL criterion

The idea of this algorithm is to detect critical links. If topological information is used, the algorithm, referred to as p-top_CL, applies the following criterion. AB is a critical link if the sets of p-hop neighbours of A and B (assuming that the link AB does not exist) are disjoint. For p=1, this reduced to the following 1-top_CL algorithm: AB is a critical link if A and B have no common neighbours (that is, there is no node C so that both AC and BC are in the unit graph). If position information is used, the corresponding p-pos_CL algorithm is defined as follows. AB is a critical link if the sets of p-hop neighbours of A and B (assuming again that the link AB is removed first from the graph) are disjoint, and there are no two nodes, one from each set, which are neighbours.

1.5.3 LC criterion

The idea behind this algorithm is very simple. If topological information is used, the algorithm, referred to as *p-top_CL*, applies the following criterion. AB is a critical link if the sets of *p*-hop neighbours of *A* and *B* (assuming that the link *AB* does not exist) are disjoint. For *p=1,* this reduced to the following *1-top_CL* algorithm: *AB* is a critical link if *A* and *B* have no common neighbours (that is, there is no node *C* so that both *AC* and *BC* are in the unit graph). If position information is used, the corresponding *p-pos_CL* algorithm is defined as follows. *AB* is a critical link if the sets of *p*-hop neighbours of *A* and *B* (assuming again that the link *AB* is removed first from the graph) are disjoint, and there are no two nodes, one from each set, which are neighbours.

1.5.4 LNC criterion

Perhaps the simplest definition of critical links is by using prior recognition of critical nodes, as follows. A link *AB* is declared *p*-pos_LNC (*p*-top_LNC) if both *A* and *B* are declared as *p*-pos_critical nodes (*p*-top_critical nodes, respectively).

1.5.5 LND criterion

The algorithm is based on verifying whether nodes in local neighbourhood of the given node all have degree at least *k*. In other words, every node verifies whether each node up to *p* hops away (each *p*-hop neighbour) has degree at least *k*. If indeed all neighbours *p*-hops away do have degree at least *k*, than the node declares the graph as being *k*-connected, otherwise *k*-disconnected.

1.5.6 LCND criterion

This algorithm is based on exchanging decisions with local neighbors to arrive at a consensus. All nodes declare themselves as being locally 1-connected. Each node then applies the algorithm (*p-top_CN* or *p-pos_CN*, depending on whether topological only or positional information as well is available) to decide whether or not it is locally 2-connected or critical (that is, 2-disconnected). The decisions are propagated to *p*-hop neighbors. If any of *p*-hop neighbors declared themselves as being critical then the graph is declared as 2-disconnected. Otherwise the node tests whether the graph consisting of its *p*-hop neighbors (excluding itself) is 2-connected. If so, it declares its neighborhood to be 3-connected, otherwise it is 3-disconnected.

1.5.7 LSCD criterion

This algorithm gives the best results for detection of k-connectivity however it does have the lowest performance. This algorithm is a combination of 'local' and 'global' algorithms. In this algorithm, we also test if the subgraph of p-hop neighbours of a given node is k-connected. A node A declares that the graph is k-connected, if and only if the following two conditions are both satisfied:

(i) each of p-hop neighbours of A has at least degree k, and
(ii) subgraph of p-hop neighbours of A, including A itself, is k-connected by LCND.

.

1.6 Planned publications out of thesis

Out of this thesis we made one publication and another one is in preparation:

1. Localized algorithms for detection of critical nodes and links for connectivity in ad hoc networks [JSHS]
2. detection of k-connectivity in wireless ad hoc, actuator and sensor networks [JGKNS]

1.7 Simulation

We simulated our proposed algorithms in C++ using OpenGL libraries. We have measured the accuracy of the proposed localized algorithms by comparing them with the corresponding globalized algorithm, that is, with the correct conclusion. Our experiments for detection of critical nodes and links were done on graphs containing 500 nodes for p=1, 2, 3 and d ranging from 3 to 15. For detection of k-connectivity we experimented with n=50, 100, 250 and 500, performed tests for p = 1, 2, 3 for connectivity k = 2, 3 and d ranging from 3 to 15.

Half of the locally estimated critical nodes and links were indeed globally critical even for p=1 (the accuracy increases to over 70% for p=2 and over 80% for p=3), for an average number of neighbours ranging from 3 to 15. The errors mostly occur when alternative routes exist but are relatively long, and therefore may not provide satisfactory service in applications.

All three proposed algorithms for detection of k-connectivity give excellent accuracy for local confirmation of global k-connectivity. However, they all have problems confirming k-disconnectivity. The LCND criterion appears to be the most accurate. The percentage of nodes in a k-connected graph, falsely declaring a graph to be k-disconnected is well under 10%, however the percentage of nodes in a k-disconnected graph that correctly classify the graph as k-disconnected is not that high. It ranges from approximately 3% to 30%.

The localized algorithms proposed in this work are indeed competitive to the globalized algorithms since the detection accuracy is very high and complexity of localized algorithms is significantly lower than the complexity of globalized algorithms.

1.8 Thesis Organization

The remainder of this work is organized as follows: We describe the existing work on critical links, node and k-connectivity detection in Chapter 2. In Chapters 3 and 4 we propose localized algorithms for detection of critical links/nodes and for detection of k-connectivity respectively. In Chapter 5 and 6 we compare our localized algorithms with the known global algorithm and record detection ratios. Finally, we conclude our work in Section 7 and discuss relevant open ideas.

Chapter 2

Literature Review

Critical nodes and links are only considered for a connected graph (or separately for each connected component of a graph). A node *A* is critical if its removal will disconnect the graph into two components. A straightforward algorithm for detecting critical nodes may consider, for each

node *A*, the subgraph obtained by its removal and removal of all its adjacent edges, and test whether this subgraph is connected. If it is not connected, the corresponding node *A* is a critical node. As a consequence, a node that has only one neighbor is not critical (e.g., node *N* in Fig. 1).

A faster global algorithm for detecting critical nodes was described by Duque-Anton, Bruyaux, and Semal [ABS], Tarjan [T], and Goyal and Caffery [GDS] who used *DFS*. These are centralized algorithms which can be also implemented in a globally distributed manner. While executing *DFS* on an undirected graph, the algorithms start at an arbitrarily chosen node which becomes the root. Then the algorithm keeps traversing fresh edges and marks nodes as "visited"; on the way we keep nodes are pushed into a stack data structure. This process is continued until a node is reached which is only connected to already visited nodes. At this point the algorithm backtracks up to a vertex which has edges connecting it to nodes which have hitherto not been visited. With a little thought it can be seen that such a node will always be a critical node of the graph. Alongside the identification of the critical node, it is easy to pop the downstream nodes from the stack into a set which corresponds to a bi-connected component. Since the above steps can be executed during *DFS* in the same pass, identification of critical nodes takes only linear time.

Critical links can de defined in several ways. One possible definition is that a link *AB* is critical if both endpoints *A* and *B* are critical nodes. However, two critical nodes may have alternate path between them. For example, in Fig. 4, nodes *O* and *Q* are critical, but alternate path between them exits via node *P*. It is therefore better to define critical link as the link connecting two critical nodes so that, when this link is eliminated from the graph, the graph becomes disconnected.

Ad hoc and sensor wireless networks have some critical connectivity properties before partitioning. Timely criticality recognition is important in order to perform some data or service replication, so that the network can continue functioning after a partition or fault does occur. Several existing centralized or globalized algorithms declare an edge or a node as *critical* if their removal will separate the network into several components. Because of the huge communication overhead involved, these solutions are impractical. Critical nodes and links are only considered for a connected graph (or separately for each connected component of a graph). As a consequence, node that has only one neighbour is not critical.

We can modify this globalized algorithm for detection of critical nodes and links (explained above) in order to use it to discover k-connectivity. If the algorithm runs and no links/node are declared critical, than we conclude that the graph is 2-connected since removal of any single node in a graph will not affect graph's connectivity.

This algorithm can be extended to test global 3-connectivity of graph G as follows. For each node X in G construct graph $F=G-\{X\}$ (eliminate X and all edges ending in X from G). Run the above *DFS* based 2-connected criterion on F. If F is 2-disconnected for any X then G is 3-disconnected.

In [GDS], once critical links in an ad hoc network are detected, two ways are proposed to delay or avoid their failure: changing the trajectory of one or both nodes forming the critical link and bringing another node to reinforce the link. Increased delivery rates were reported due to prolonged network connectivity. However, the communication overhead due to running *DFS* for detecting critical links was not measured. Karumanchi, Muralidharan, and Prakash [KMP], Li, and Rus [LR], Vahadat and Becker [VB], Park and Corson [PC] attempt to improve the communication without avoiding or delaying partitioning ('post-partitioning' approaches).

Hauspie, Simplot and Carle [HSC] proposed to evaluate stability of a path from a source to destination by a function that depends on disjoint path between them, and the hop distance of each of these paths. When the function reaches a threshold, data or service replication is performed. The protocol has a significant communication overhead for evaluating the function.

Koskinen [K] examined critical transmission ranges for bi-connectivity and tri-connectivity of ad hoc networks. He experimentally established an asymptotic behavior that these types of critical links can be determined by a function which is square root of the ratio of area and linear function of number of nodes (coefficients of that function are parameters to be determined).

. Shah, Chen, and Nahrstedt [SCN] aimed at enhancing data access in an ad hoc network by detecting partitions in it. They propose a data replication mechanism based on partition detection for allowing one node to access the data from another node even if the connection is physically broken. Each node of a connected group knows the behavior of other members in a group, because each node embeds a positioning system (GPS) by successive measures computes its velocity, and spreads that information to the other nodes. Due to this information, it is possible to predict when a node will leave its group. The 'node-to-leave' picks another node of the group to be a host of the data and performs a data replication on that node. The main advantage of this method is that each node knows exactly when the partition occurs (regular node movement is assumed, without sudden changes in direction). However, this method has two main disadvantages. First, a positioning system is needed. Second, the network is continuously and relatively highly loaded due to information exchange among nodes. While general ideas in [SCN] are good, the essential details are missing. The definition of group of nodes is not given, and one can even assume the whole connected network to be a group, since it also satisfies the vague definition given. The protocol for

predicting link breakage, based on predicting future locations and connectivity of two nodes, is described in detail (a similar protocol was described earlier by Stojmenovic, Russell and Vukojevic [SRV]) but there is no description of any protocol to detect group partitioning; it was left up to a node to decide without giving details on how it is actually decided.

Li and Rus [LR] propose an approach in which nodes actively modify their trajectories to transmit messages. They develop an algorithm that minimize the trajectory modifications under two assumptions: the movements of all the nodes in the system are known and not known, respectively.

In cooperative caching, discussed by Cao, Yin and Das [CYD], data from server are replicated on some nodes in ad hoc network so that access demands by other nodes can be satisfied by replicated files rather than original files, which should reduce traffic in the network or even provide service if the server becomes disconnected in the meantime. The described methods include caching data paths toward replicated copy by current node, or making another copy of data at the node, plus some hybrid method based on some criteria.

Hara [H] proposed three replica allocation methods to improve data accessibility by replicating data items on mobile hosts. The first method is to make a lot of replicas at each node, while the two others begin with a step in which each mobile host periodically broadcasts its host identifier and information on access frequencies to data items. After all mobile hosts complete their broadcasts, every host knows its connected mobile hosts. This partition detection method clearly requires a lot of communication overhead.

Hajiaghayi, Immorlica, and Mirrokni [HIM] considered the problem of assigning transmission power to nodes so that the sum of powers is minimized and the network is *k*-

connected. They use energy cost d^{α} for transmission between two nodes at distance d ($2 \leq \alpha \leq 6$) and the algorithms are globalized. To guarantee k-connectivity, [BHM] uses a solution where most nodes need to have $3k$ or more neighbors, by enforcing minimal angle between two selected neighbors.

In the partition detection system of [RWS] (Ritter, Winter, Schiller), some nodes become active (when the number of its neighbors is bellow a threshold) and send out beacon messages to other active nodes. Each active node also elects neighboring 'buddy' node to monitor its behavior. This approach has huge communication overhead due to regular broadcasting and routing tasks involved. Further, it only detects a partition after it happens, thus no action (e.g., data or service replication) is possible. Next, the partition will be detected anyway when a routing task is issued and destination is not found in the same partition. Otherwise, if there is no routing task, the fact that a partition (which may even be temporary) exists is irrelevant. Therefore the real purpose of running this protocol is unclear.

Several articles concentrate on the asymptotic behavior of connectivity when the number of nodes goes to infinity. Bettstetter [B] studied the property of k-connected graph. Clearly, in a k-connected graph, each node has at least p neighbors; otherwise, the removal of its neighbors will disconnect it from the rest of the graph. These two properties are studied analytically and experimentally in [B]. They both have very interesting behavior when measured as a function of transmission radius R (or corresponding average degree d). There is a sharp transition from probability near 0 to probability near 1 for a graph to be k-connected, and also to have minimum degree k. Moreover, these two transitions occur at close transmission ranges. Therefore, if a graph

has minimum degree k with high probability, it is k-connected with high probability [B]. Fast centralized algorithms for detecting 2 and 3-connectivity are also given in [B].

Xue and Kumar [XK] proved that, to guarantee a geometry graph over V connected, the number of nearest neighbours that every node has to connect is asymptotically $\Theta(\ln n)$. Li, Wang, Wan and Yi [LWWY] extended this work to study asymptotic network behavior with respect to k-connectivity. Zhang and Hou [ZH] investigate the minimum total power required to ensure asymptotic k-connectivity in heterogeneous wireless networks where nodes may transmit using different levels of power.

After doing an extensive literature review we were able to find only algorithms that can detect network partition by using global knowledge of the network. In [B] the property of k-connected graph was studied where the connectivity was in close relationship with average number of neighbours that gave us an idea to test this property in the local neighbourhood of a node. If the connectivity is not achieved in the local neighbourhood of a node than we can act and apply caching methods explained in [CYD] or apply the algorithm in [LR] where the nodes trajectory of the nodes is modified in order to achieve connectivity.

Chapter 3

Localized algorithms for detection of critical links and nodes

Localized algorithms that we have developed discover all critical nodes and links very quickly. However, these algorithms may detect some nodes and links as critical although they may not be globally critical. This is unavoidable since only local knowledge is used, therefore with such a restriction it is impossible for a node to learn about alternate connections in different parts of the network. On the other hand, in applications, often long alternate paths often do not provide

satisfactory service, thus the localized method may even provide a more useful decision. Moreover, the partition detection is done faster and with much less communication overhead.

We first define what local knowledge is available to nodes and how nodes gain it. We use the notion of p-hop knowledge. Two nodes are considered to be p-hop neighbours if and only if the shortest route between them has p or less hops. Awareness of itself only is represented as 0-hop knowledge. This may or may not include geographic position of the node. Localized algorithms that use position information can only be applied to nodes that are equipped with *GPS* or that find their relative coordinates by measuring signal strengths or time delays in mutual communication. Nodes collect p-hop knowledge by sending 'hello' messages to their neighbours in the graph of their $(p-1)$-hop neighbours. Thus 1-hop knowledge is a list of direct neighbours, with or without their geographic positions. We refer to p-hop knowledge as being *topological* and *positional* information, and corresponding knowledge as being *p-hop topological* or *p-hop positional* information, respectively. The 2-hop topological information is obtained by transmitting lists of 1-hop neighbours, and the subgraph of 2-hop neighbours therefore includes existing links between 1-hop neighbours, and between 1-hop and 2-hop neighbours, but not possible links between 2-hop neighbours. On the other hand, 2-hop positional information includes such links and information, since a node learns the position of 2-hop neighbours and may (by exchanging 'hello' messages), based on distances between them and the unit graph used, decide whether or not they are neighbours. Generalizing this, p-hop topological information, and corresponding subgraphs of p-hop neighbours, include all existing links between p-hop and $(p-1)$-hop neighbours, but not information on whether or not two *strictly* p-hop neighbours (that is, two nodes which are p-hop but are not $(p-1)$-hop neighbours) are connected. Since the information about any node in the

positional case includes its position, in case of p-hop positional information, such information is additionally available. Therefore, localized algorithms with position information have more information than localized algorithms with topological information, and are consequently more accurate; they discover less falsely detected critical links and nodes. Furthermore, if a node/link is declared critical by a local algorithm that uses position information, it is also declared as critical by the localized algorithm that uses the corresponding topological information. The reason is that the graph may only have fewer edges and thus the partition detected by positional information cannot be 'sealed' by the corresponding topological information.

We shall now describe our localized algorithms for detecting critical nodes and links. In each of topological and positional cases, we give one definition of critical nodes and three definitions of critical links. The four definitions are based on verifying common neighbours, loop length, and critical status of link endpoints, respectively.

3.1 Localized algorithms for detection of critical nodes (pos/top_CN criteria)

For each node A, consider the subgraph of *p-hop* neighbours of A, where A and all its incident edges are excluded. In case of positional information, two nodes in that graph are connected if they are connected in the original unit graph. In case of topological information, two nodes in that graph are connected if they are connected in the original unit graph, and at least one of them is $(k\text{-}1)$-hop neighbor of A.

A node *A* is a *p*-critical node if the corresponding subgraph of *p*-hop neighbours of *A* is disconnected. Based on information used, this is further specified as being a topologically or positionally *p-hop* critical node. The corresponding algorithms are referred to as being *p-top_CN* and *p-pos_CN.*

Clearly, if a node is globally critical, localized algorithm will detect it as such. Further, if a node is not declared as critical by *p-top_CN* it is also not declared as critical by *p-pos_CN* algorithm. That is, if a node is declared as critical by *p-pos_CN* it is also declared critical by *p-top_CN.*

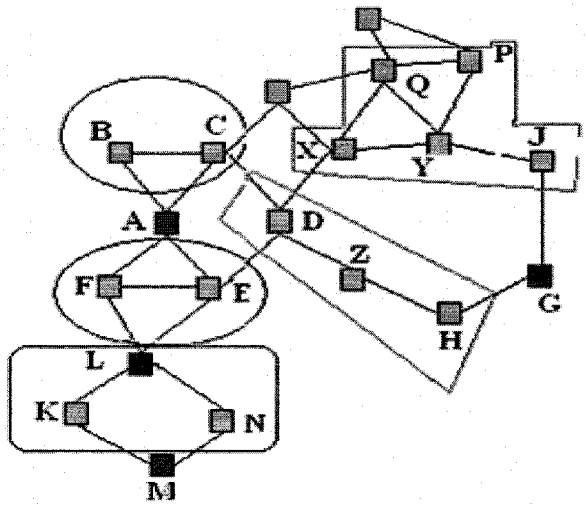

Figure. 2. Node *A* is 1-hop critical, node *G* is topologically 3-hop critical and positionally 2-hop critical, node *M* is 1-hop critical

We will now discuss particular cases and give some examples to illustrate. For *k=1,* if topological information is used, no links between neighbours exist in the decision graph, therefore all nodes are declared critical with the *1-top_CN* algorithm. This is obviously not very helpful.

Figure 2 illustrates the localized definitions of critical nodes. 1-hop neighbours of *A* can be divided into two sets *{B, C}* and *{E,F}* (circled in Fig. 2) which are disconnected. Therefore *A* is positionally (and therefore topologically) 1-hop critical. Node *A* is not 2-hop critical since its 2-hop neighbors *D* will connect the two sets. Node *M* is 1-hop critical since its 1-hop neighbours *K* and *N* are not connected. However, it is not 2-hop critical since its two hop neighbours *K, N,* and *L* create a connected subgraph (marked by a rectangle in Fig. 2), both topologically and positionally. Node *G* is topologically 3-hop critical since its 3-hop neighbours are divided in two subgraphs with vertices *{H, Z, D}* and *{J, Y, X, Q, P}* (the two sets are enclosed by polygons in Fig. 2) which are disjoint. However, when position information is added, node *G* can recognize that *X* and *D* are in fact neighbors, and that the two subgraphs are in fact connected, therefore *X* is not positionally 3-hop critical.

3.2 Localized algorithm for detection of critical links (pos/top_CL criteria)

If topological information is used, the algorithm, referred to as *p-top_CL*, applies the following criterion. *AB* is a critical link if the sets of *p-hop* neighbours of *A* and *B* (assuming that the link *AB* does not exist) are disjoint. For *p=1*, this reduced to the following *1-top_CL* algorithm: *AB* is a critical link if *A* and *B* have no common neighbours (that is, there is no node *C* so that both *AC* and *BC* are in the unit graph).

If position information is used, the corresponding *p-pos_CL* algorithm is defined as follows. *AB* is a critical link if the sets of *p*-hop neighbours of *A* and *B* (assuming again that the link *AB* is removed first from the graph) are disjoint, and there are no two nodes, one from each

set, which are neighbours. For $p=1$, the criterion is that A and B have no common neighbours that are within transmission range of each other (that is, neighbours).

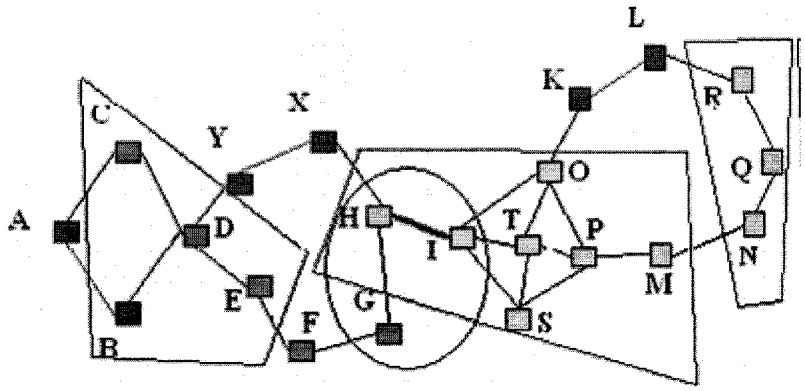

Figure. 3. Examples of critical links *HI, AB, XY* and *KL*

Consider the example graph in Figure 3. It has only one globally critical link, *HI.* Localized algorithms will detect some more critical links. For example, the *1-top_CL* algorithm declares the link *AB* as critical, because nodes *A* and *B* have disjoint 1-hop neighbours (*C* and *D*). However, the *1-pos_CL* algorithm declares correctly that the link *AB* is not critical, because *A* and *B* may together agree that *C* and *D* are in fact neighbours, therefore there exists an alternative path from *A* to *B* which does contain the link *AB*. The link *XY* is declared as critical with both the 2-*top_CL* and 2-*pos_CL* algorithms. The 2 hop neighbours of *X* (*H, G, I,* circled in Fig. 3) and *Y* (*D, C, B, E,* enclosed in a quadrilateral in Fig. 3), are disjoint and there are no two nodes, one from each set, which are neighbours, hence the link is declared as critical. Nevertheless, the *3-top_CL* algorithm correctly declares the link *XY* as not being critical because the 3-hop neighbors of *X* (*H, I, O, T, S,G, F*) and *Y* (*D, C, A, B, E, F*) are not disjoint. If we examine the link *KL* with the *3-top_CL* algorithm, we declare it as critical, because the 3-hop neighbors of *K* (*O, P, M, T, S, I, H*) and *L* (*R, Q, N*), both enclosed in quadrilaterals in Fig.3, are disjoint. On the other hand, the 3-*pos_CL*

algorithm correctly detects that the link *KL* is not critical. Even though the neighbors of *K* and *L* are disjoint, the algorithm detects that the link between *M* and *N* (which are 3-hop neighbors of *K* and *L* respectively) exists and declares the link *KL* as not being critical.

3.3 Localized criteria for detection of critical links (pos/top_LC criteria)

In this section, we introduce critical link definitions which are based on finding the length of the shortest loop between two link endpoints. A link *UV* is *k-LC* if the hop distance between *U* and *V* in the given graph, with only edge *UV* being eliminated, is $>p$. There are several possible implementations of this definition, since a common decision should be made between two nodes and/or link endpoints. In general, we can consider the kU-hop neighborhood of *U* and the pV-hop neighbourhood of *V*, where $pU + pV = p$. If topological information is used, the algorithm is defined as follows. A link *UV* is *(pU,pV)-top_LC* if the sets of *kU* -hop neighbors of *U* and *pV* –h op neighbors of V are disjoint. It follows then that a link is *p-top_LC* if and only if it is *(p,p)-top_LC* (or *2p-LC* in a more general definition). Because of such equivalency, and expectation that $pU = pV$ is reasonable to assume, we did not implement this definition. If positional information is used, the corresponding algorithm applies the following test. A link *UV* is *(pU,pV)-pos_LC* if the sets of *pU* -hop neighbors of *U* and *pV* -hop neighbors of *V* are disjoint, and there are no two nodes, one from each of these sets, that are neighbours. It follows then that a link is *p-pos_LC* if and only if it is *(p,p)-pos_LC* (that is, *2p-LC*). Note that our initial definition of a *p-LC* link is equivalent to the new definition of a *(p,0)-LC* link.

Figure 4 illustrates these definitions. For example, after applying *1-top_LC* and *2-top_LC* algorithms on link *AB*, we declare the link as critical because from node *A* we can not reach node *B* with 2 hops. However, the link will not be detected as critical with *2-pos_LC* algorithm, since 2-hop neighbour node *D* of *A* is also a neighbour of node *B*, based on their geographical positions. The *3-top_LC* algorithm detects the link *XY* as critical, but the *3-pos_LC* correctly detects the link as not being critical, since 3-hop neighbour *F* of node *Y* is a neighbour of *X*. The link *JK* is the link where the *3-pos_LC* algorithm wrongly declares it as critical. The loop is too wide, and more hops are needed in order to correctly declare the link as not being critical. The *1-top_LC* algorithm is very week, which is illustrated on link *MN*. This algorithm only detects the node *O*, which obviously does not form a loop from *M* to *N*, therefore, all the links are declared critical with the *1-top_LC* algorithm. However, *1-pos_LC* correctly declares the link *MN* as not being critical.

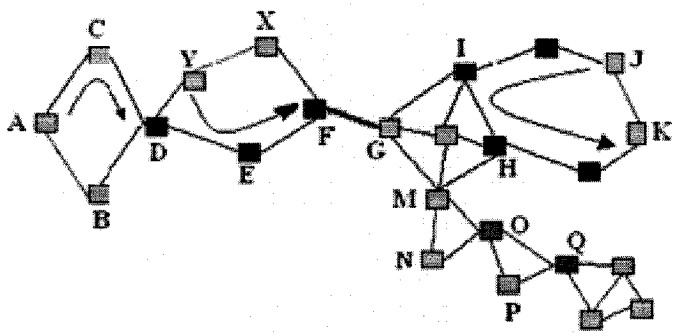

Figure. 4. Loop length based critical links *AB, XY, MN* and *JK*

Localized algorithms for detection of critical links based on loop length, falsely declare many links as critical.

3.4 Localized criteria for detection of critical links (pos/top_LNC criteria)

Perhaps the simplest definition of critical nodes is by using prior recognition of critical nodes, as follows. A link *AB* is declared *p-pos_LNC* (*p-top_LNC*) if both *A* and *B* are declared as *p-pos_critical nodes (p-top_critical nodes,* respectively). The advantage of this definition over other two is that its implementation does not require two nodes to exchange their neighbourhood information beyond the already exchanged. Instead, they need to exchange merely their decisions about being critical nodes.

Alternatively, each node may decide which of its links are *p-hop* critical by using its (*k+1*)-hop information, which involves more communication overhead for collecting than the suggested decision exchange method. The communication overhead involved with the detection of critical links via critical nodes is the smallest among the mentioned definitions and corresponding implementations. However, as already observed, this algorithm is not as accurate for detection of critical links as the *k-top/pos_CL* algorithm. Two critical nodes may be connected with more than one link. From Figure 4 we see that nodes *O* and *Q* are critical nodes and that they have a direct link between them, yet they also have a common neighbour *P*, which creates a path *OPQ*. Therefore, the link *OQ* is in fact not critical, which can be easily verified with the *1-top_CL* algorithm. These cases are present in our simulations; hence, due to the inaccuracy of this algorithm, we did not perform experiments on it.

Chapter 4

Localized criteria for detection of *k*-connectivity

An ad hoc network is *2-connected* if it cannot be broken into disconnected pieces by deleting any single node and its incident edges. If we consider any path between *B* and *F* (see Figure 5), it will be disconnected if *C* is removed. Hence this graph is 1-connected but not 2-connected.

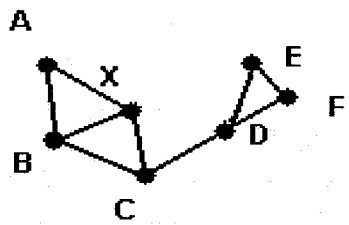

Figure 5. Example of a 1-connected graph, which is not 2-connected

An ad hoc network is *3*-connected if it cannot be broken into disconnected pieces by deleting any two single nodes and their incident edges. Consider the source node *K* and destination node *E* in Figure 6. For example, if we remove 2 nodes *C* and *O*, the graph will be broken into 2 components and there will be no path from *K* to *E*. Therefore, the graph in Figure 6 is 2-connected but not 3-connected.

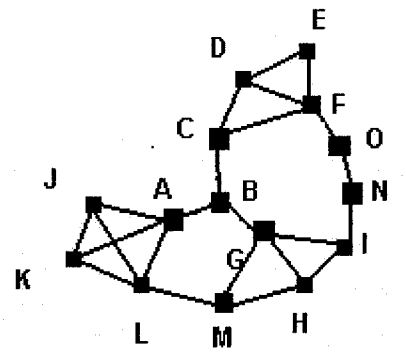

Figure 6. 2-connected ad hoc network which is 3-disconnected

We now describe three localized algorithms for detecting if a given network is *k*-connected: LND (local_neighbour_detection), LSCD (local_subgraph_connectivity detection), and LCND (local_critical_node detection). These localized algorithms will be addressed with respect to 1-, 2-, and 3-connectivity in this thesis.

4.1 Local neighbour detection (LND) criterion

The algorithm is based on verifying whether nodes in the local neighbourhood of the given node all have degree at least *k*. In other words, every node verifies whether each node up to *p* hops away (each *p*-hop neighbour) has degree at least *k*. This test can be performed in straightforward way. After learning about 1-hop neighbors, each node may include the number of them in their next 'hello' messages. In this way, nodes learn how many neighbors each *p*-hop neighbor has, and can run this test. Since (as already commented), all nodes conclude that they are 1-connected, we only consider *k*-connectivity for $k > 1$. More precisely, we experimented only for cases $k=2$ and $k=3$.

For k=2, for each node we check if its p-hop neighbours have at least 2 neighbours. For example, if we test node B (see Figure 7) we first check if its p-hop neighbours have at least 2 neighbours. Therefore, for p=1 and p=2 hops, node B declares the graph as 2-connected. After 3-hops we find the node E that has only 1 neighbour; therefore, node B declares correctly that the graph is not 2-connected with 3-hop information. If we perform the same experiment in Figure 5, node B declares that the graph is 2-connected because all its p-hop neighbours have at least 2 neighbours. When verified with the global algorithm we declare the graph as 2-disconnected because the removal of B disconnects the graph.

For k=3, for each node we check if its p-hop neighbours have at least 3 neighbours. For example, node B (see Figure 7), declares the graph as 3-disconnected, because all B's p-hop neighbours have less than 3 neighbours. In Figure 8, node H believes that graph is 3-connected for p=1, but recognizes that graph is 3-disconnected for p=2.

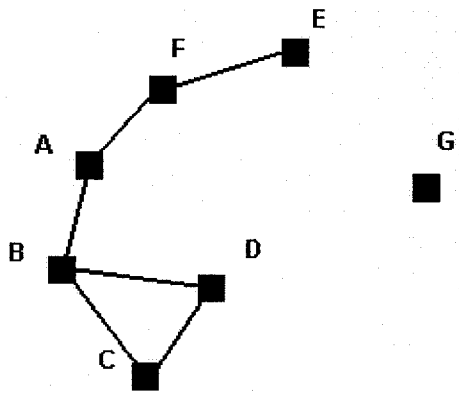

Figure 7. B declares this network as 2-connected with 2-hop and 2-disconnected with 3-hop knowledge.

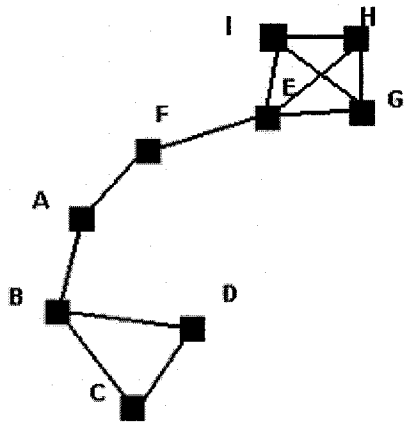

Figure 8. B declares this network as 2-connected graph although globally it is 2-disconnected.

In Figure 6, if we test node *A* for 3-connectivity we get the following results. For p=1, all neighbours of *A (J, K, L, B)* have at least 3 neighbours; therefore, node *A* declares the graph to be 3-connected. For p=2, all 2-hop neighbours of *A (K, L, J, M, B, C, G)* have also at least 3 neighbours; therefore, node *A* declares the graph to be 3-connected. In fact, all nodes in Figure 6 have at least 3 neighbours, except nodes *O* and *N*. When p=4, node *N* is reached. Node *N* has only 2 neighbours, and node *A* declares the graph to be 3-disconnected.

4.2 Local critical node detection (LCND) criterion

This algorithm is based on exchanging decisions with local neighbors to arrive at a consensus. All nodes declare themselves locally 1-connected. Each node then applies algorithm *p-top_CN* or *p-pos_CN*, depending on whether 'topological only' or 'positional as well' information is available to decide whether or not it is locally 2-connected or critical (that is, 2-disconnected). The decisions are propagated to p-hop neighbors. If any of the p-hop neighbors declared themselves as being critical then the graph is declared as 2-disconnected. Otherwise each node tests whether the graph consisting of its p-hop neighbors (excluding itself) is 2-connected. If so, it declares its neighborhood to be 3-connected, otherwise it is 3-disconnected. This definition can be further generalized to test k-connectivity. Iteratively, each node decides locally whether it is k-connected or k-disconnected, based on the information from p-hop neighbors. If node *A* is *(k-1)*-connected, all its p-hop neighbors are *(k-1)*-connected, and the graph consisting of p-hop neighbors of *A* (excluding *A*) is *(k-1)*-connected; then node *A* declares its neighbourhood as k-connected. Otherwise it declares its neighbourhood as k-disconnected.

We will now explain how the algorithm works using the graph in Figure 9 as an example. First we test if node *A* is critical by the *p-top_CN* or *p-pos_CN* algorithm, where *p=3*. Node *A* is not declared critical, hence we consider the subgraph of *p*-hop neighbours of *A* excluding *A*. If we run *3-top_CN* or *3-pos_CN* on this graph where node *A* is excluded, we will notice that it is not 2-connected because of critical node *B*. Hence the graph will be declared as 3-disconnected, because the removal of *A* and *B* will disconnect the graph into two separate components.

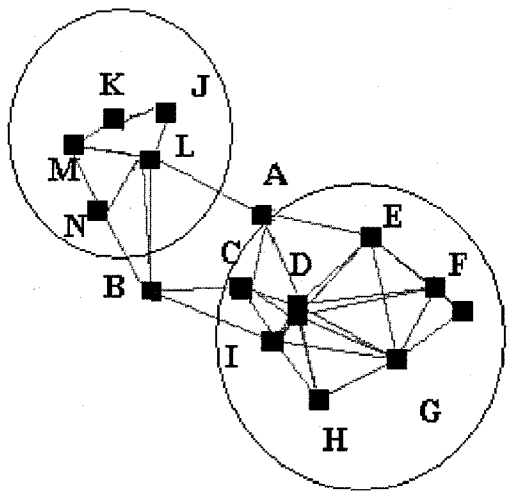

Figure 9. A 3-disconnected graph

4.3 Local subgraph connectivity detection (LSCD) criterion

In this algorithm, we also test if the subgraph of *p*-hop neighbours of a given node is *k*-connected. A node *A* declares that the graph is *k*-connected, if and only if the following two conditions are both satisfied:

(ii) each of the *p*-hop neighbours of *A* has at least degree *k*, and the

(ii) subgraph of *p*-hop neighbours of *A*, including *A* itself, is *k*-connected.

For example, if we run for $k=2$ the LSCD algorithm on node B in Figure 7, the first condition will be satisfied, but the second condition will fail for any p. Thus the graph is 2-disconnected for any p.

Consider the example in Fig. 10 for $k=3$ for node H. For $p=1$, we will conclude that the graph is 3-disconnected for the following reason. All direct neighbours of H (A, G, E) have at least 3 neighbours which satisfies the first condition. When the local_connecitivy_detection algorithm is run on subgraph A, H, G and E, it concludes that the subgraph is 2-disconnected, therefore 3-disconnected. The same decision will be made for $p=2$. Only for $p=3$, the algorithm correctly detects that the graph is in fact 3-connected.

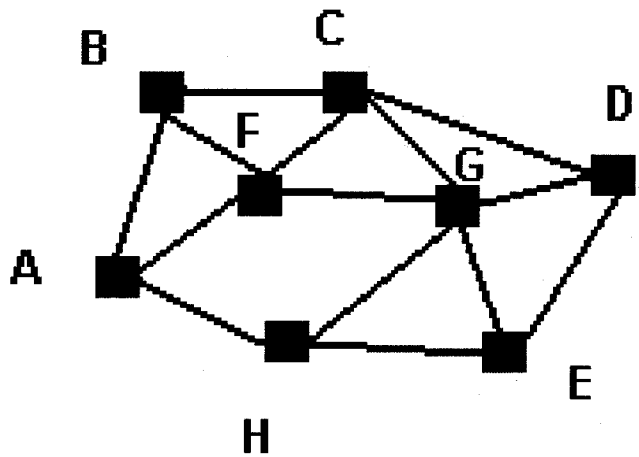

Figure 10. A 3-connected graph.

Chapter 5

Performance evaluation for detection of critical nodes and links

We have measured the accuracy of the proposed localized critical nodes and links detection algorithms by comparing them with the corresponding globalized algorithm, that is, with the correct conclusion. We will describe here the experimental results obtained by experimenting with connected random unit graphs.

To generate graphs ranging $d= 3$ to 15 we used standard methods for generating random unit disk graphs, as described in [SL1]. An approximate transmission radius r is obtained from the formula $d=(n-1)\pi r^2/a^2$, where a is the length of the area where nodes are located. We find the exact value r in the following way. After sorting all $n(n-1)/2$ possible edges, the desired radius r is the $(n*d/2)$-th edge in sorted list. The unit graph is then defined using this value of R (that is, two nodes are neighbors if and only if the distance between them is at most R). Once a graph is generated, we verify if it is connected by running Dijkstra's shortest path algorithm. Only connected graphs are considered for the experiments. Our experiments are performed with $n=500$ nodes for several densities, ranging from 3 to 15.

The proposed method tends to group the nodes around the first generated point, with distribution from there inside a square region. We believe that such generation may indeed reflect better networks which are created by members joining the already created group. For instance, a new laptop entering a conference room is likely to take a position so that it is connected to at least one laptop already there. We have performed also experiments with the classical random unit

graph generation model, with and without enforcing connectivity. The data obtained do not significantly differ from those reported here for the novel unit graph generation method.

In our tests, the main measure considered is the *detection ratio*, which is the probability that a node or link will be declared as critical by considered localized algorithm is indeed critical when verified by global algorithm. We also measured the average number of critical nodes and links detected in the network. We only report detection ratios for *n=500,* which are overall somewhat better (by as much as 10%) than for *n=100,* which was counterintuitive but encouraging for the scalability of our approach.

5.1 Localized algorithms for detection of critical nodes

Tables 5-1 and 5-2 show the detection ratios for critical nodes, using the *p-top_CN* and *p-top_CN* algorithms, for *k*=1, 2 and 3, obtained after 20 simulations for different values of *d* for random connected unit graphs with n=500 nodes.

Detection ratios are generally over 50%, meaning that over half of locally estimated critical nodes were indeed globally critical even for *p=1* (the accuracy increases to over 70% for *k=2* and over 80% for *p=3*), for average number of neighbours ranging from 3 to 15. As expected the *p-pos_CN* algorithm performs better than the *p-top_CN* algorithm.

Tables 5-3 and 5-4 present the average numbers of detected critical nodes for each algorithm. The third column shows the average number of critical nodes detected by the global algorithm. The fourth and fifth columns show the number of critical nodes detected by local algorithms for *k=2* and *3*.

It can be observed that there are not many critical nodes in these graphs, especially for graphs with medium density and dense graphs. Localized algorithms do not declare too many nodes as critical that are in fact not globally critical.

Average number of neighbours (d)	Detection Ratio 2-hop algorithms (%)	Detection Ratio 3-hop algorithms (%)
15	50.000	71.429
11	64.151	79.487
10	73.016	90.196
9	70.400	97.703
8	73.333	88.709
7	64.828	82.456
6	67.879	83.582
5	71.500	78.571
4	73.504	91.489
3	66.67	86.486

Table 5-1. Detection ratios for the *p-top_CN* algorithm on connected graphs with 500 nodes.

Average number of neighbours (d)	Detection Ratio 1-hop algorithms (%)	Detection Ratio 2-hop algorithms (%)	Detection Ratio 3-hop algorithms (%)
15	50.000	71.430	83.330
11	64.151	82.930	94.440
10	70.769	85.190	97.870
9	55.346	74.576	96.778
8	52.133	75.862	90.909
7	64.828	82.456	90.385
6	61.538	78.873	86.154
5	49.310	76.064	85.119
4	51.343	80.374	92.473
3	56.140	69.565	86.486

Table 5-2. Detection ratios for the *p*-pos_CN algorithm on connected graphs with 500 nodes.

Average number of neighbours (d)	Average number of critical nodes (global alg)	Average Number of critical nodes (2-hop alg)	Average Number of critical nodes (3-hop alg)
15	3.0	6.2	4.2
11	6.8	10.6	7.8
10	9.2	12.6	10.2
9	8.8	12.5	9.1
8	11.0	15.0	12.4
7	9.4	14.5	11.4
6	11.2	16.5	13.4
5	14.3	20.0	18.2
4	17.2	23.4	18.8
3	19.2	28.8	22.2

Table 5-3. Average numbers of detected critical nodes by the *p-top_CN* and global algorithms.

Average number of neighbours (d)	Average number of critical nodes (global alg)	Average Number of critical nodes (1-hop alg)	Average Number of critical nodes (2-hop alg)	Average Number of critical nodes (3-hop alg)
15	3.0	6.0	4.2	3.6
11	6.8	10.6	8.2	7.2
10	9.2	13.0	10.8	9.4
9	8.8	15.9	11.8	9.0
8	11.0	21.1	14.5	12.1
7	9.4	14.5	11.4	10.4
6	11.2	18.2	14.2	13.0
5	14.3	29.0	18.8	16.8
4	17.2	33.5	21.4	18.6
3	19.2	34.2	27.6	22.2

Table 5-4. Average numbers of detected critical nodes by the *p-pos_CN* and global algorithms.

5.2 Localized algorithms for detection of critical links

Tables 5-5 and 5-6 show the detection ratios obtained after 20 simulations on the *p-top_CL* and *p-pos_CL* algorithms, for connected random unit graphs with $n=500$ nodes.

Average degree of neighbours (d)	Detection Ratio 1-hop algorithm(%)	Detection Ratio 2-hop algorithm(%)	Detection Ratio 3-hop algorithms (%)
15	31.250	50.000	71.429
11	39.081	64.151	79.487
10	50.549	73.016	90.196
9	43.750	71.186	100.00
8	41.667	78.125	90.909
7	44.00	67.692	84.615
6	49.039	72.857	85.000
5	45.890	72.043	80.723
4	47.619	76.923	93.023
3	48.924	67.407	86.667

Table 5-5. Detection ratios for the *p-pos_CL* algorithm on connected graphs with 500 nodes.

Average degree of neighbours (d)	Detection Ratio 1-hop algorithm (%)	Detection Ratio 2-hop algorithms (%)	Detection Ratio 3-hop algorithms (%)
15	50.000	71.429	83.333
11	64.151	82.927	94.444
10	70.769	85.185	97.872
9	57.534	76.364	100.000
8	55.555	76.923	92.593
7	67.692	84.615	93.617
6	64.557	80.952	86.441
5	52.756	77.907	89.333
4	56.338	83.333	97.561
3	57.232	71.654	86.667

Table 5-6. Average number of detected critical links by the *p-top_CL* and global algorithms.

From these tables we conclude that the localized algorithms give excellent results. In particular, the 3-hop localized algorithms have the accuracy greater than 70% for any d, while 2-hop localized algorithms have accuracy greater than 70% in most cases. Even 1-hop localized topological criteria declare correctly more than 50% of links.

Average number of neighbours (d)	Average Number of links	Average number of critical links (global alg)	Average Number of critical links (1-hop alg)	Average Number of critical links (2-hop alg)	Average Number of critical links (3-hop alg)
15	5420	1.5	4.8	3.0	2.1
11	4523	3.4	8.7	5.3	3.9
10	4306	4.6	9.1	6.3	5.1
9	4039	4.2	9.6	5.9	4.2
8	3785	5.0	12.0	6.4	5.5
7	3365	4.4	10.0	6.5	5.2
6	3015	5.1	10.4	7.0	6.1
5	2785	6.7	14.6	9.3	8.3
4	2555	8.0	16.8	10.4	8.6
3	2115	9.1	18.6	13.5	10.5

Table 5-7. Average number of detected critical links by the *p-pos_CL* and global algorithms.

Average number of neighbours (d)	Average Number of links	Average number of critical links (global alg)	Average Number of critical links (1-hop alg)	Average Number of critical links (2-hop alg)	Average Number of critical links (3-hop alg)
15	5420	1.5	3.0	2.1	1.8
11	4523	3.4	5.3	4.1	3.6
10	4306	4.6	6.5	5.4	4.7
9	4039	4.2	7.3	5.5	4.2
8	3785	5.0	9.0	6.5	5.4
7	3365	4.4	6.5	5.2	4.7
6	3015	5.1	7.9	6.3	5.9
5	2785	6.7	8.6	7.5	12.7
4	2555	8.0	14.2	9.6	8.2
3	2115	9.1	15.9	12.7	10.5

Table 5-8. Average number of detected critical links by the *p-pos_CL* and global algorithms.

The average number of critical links detected by local the *p-top_CL* and *p-pos_CL* algorithms and the number of critical links detected by the global algorithm are recorded in Tables 5-7 and 5-8, respectively.

From the tables we conclude that local algorithms for detection of critical nodes have approximately the same accuracy as the local algorithms for detection of critical links. This behavior is well expected because if a link is declared as critical, then the two nodes making this link are also declared critical. So, we can expect to have at least two times more critical nodes than links. Some critical nodes are not a part of a critical link. The number of these nodes is not high. For example, no critical nodes that are not part of a critical link were found for d between 10 and 15.

We will now give some insight in order to explain the obtained results. Figure 11 shows one of the random connected unit graphs with 500 nodes, using our described procedure. The average density is high, d=15. The main problem that lowers the detection ratio of localized algorithms can be associated with creation of ring structures shown in Figure 11. Ring structures (those producing some critical nodes or links are marked by 'R' in Fig. 11) are very common in these graphs and they are the focal problem in detection of critical links/nodes with local algorithms. The critical links detected by the *1-top_CL* algorithm are drawn in black in Fig. 11. Critical nodes detected by the *1-top_CN* algorithm are drawn in green in Fig. 11. Rings are often too wide to be detected by 3 hops and most of the nodes/links that are part of a specific ring are then declared as critical. These rings can be detected if we increase the hop count, but then the nature of the local algorithms is lost.

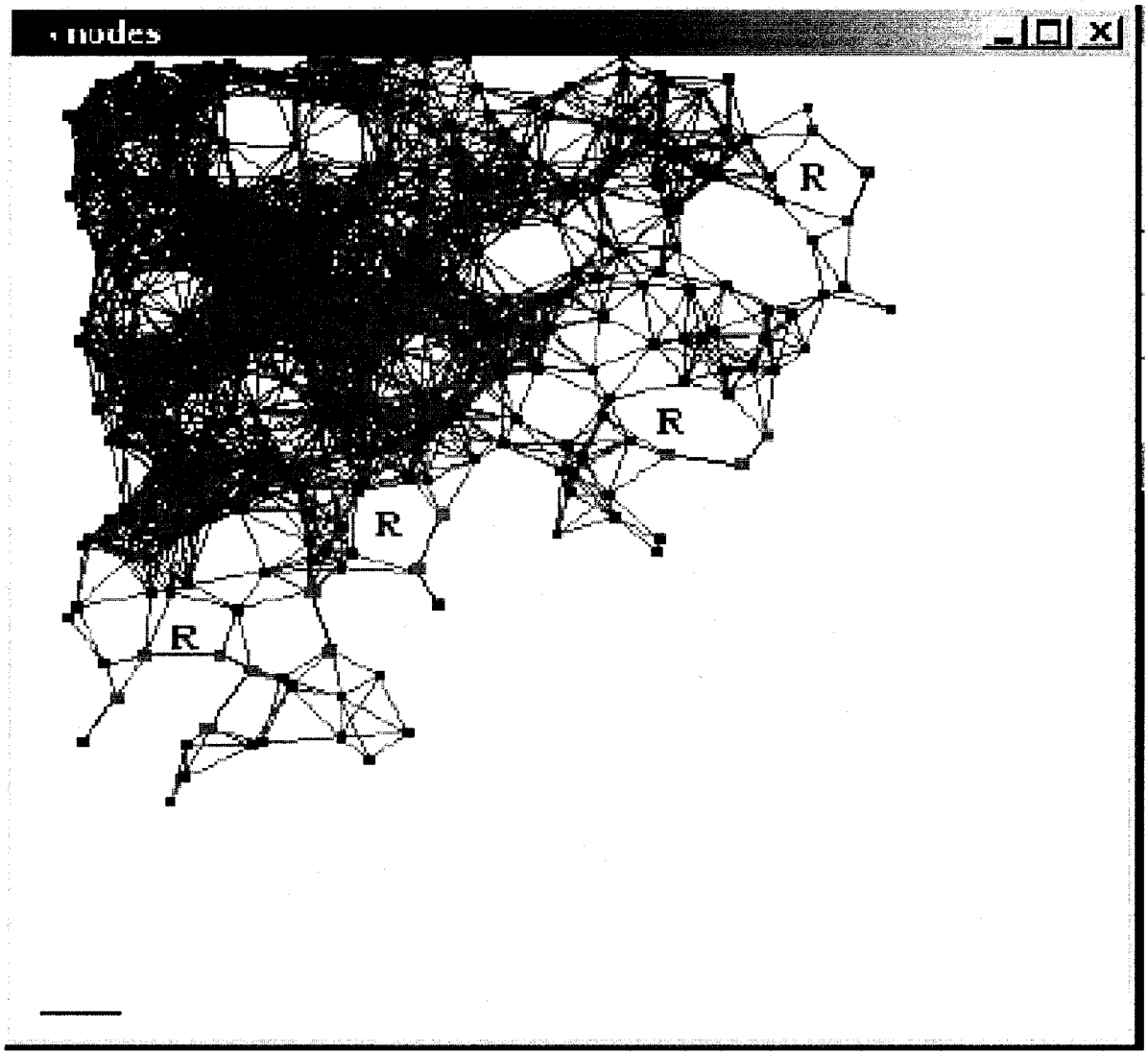

Figure 11. Ad-hoc network for d=15.

Chapter 6

Performance evaluation for detection of k-connectivity

We considered large-scale wireless ad hoc networks whose nodes are distributed in a two-dimensional unit square region. We assume the node density (average number of neighbours) to be d. Each node can directly communicate only with neighbors located within a circle of radius R, which is called the transmission radius. An edge connects two nodes that can communicate directly. Furthermore, we assume n nodes are uniformly distributed over the area of size that depends on selected n, d and R. For this simulation, we experimented with 50, 100, 250 and 500 uniformly distributed nodes. We performed tests for p = 1, 2, 3 and connectivity k = 2, 3 for different average degree of neighbours d, ranging from 3 to 10. For each pair (p, k) there are four possible outcomes with localized tests and the global test.

For graph generation we used 2 different techniques. To generate graphs ranging d= *6* to *10* we used the standard method for generating random unit disk graphs, as described in Chapter 5. In order to generate graphs for d <6 we used MAX_DPA [AS] because it is almost impossible to generate connected graphs with the standard graph generation method. Graphs generated with MAX_DPA are generated in the following manner:

A new node cannot be placed to increase the degree of any existing node over a pre-specified parameter limit and must be in the proximity of previously generated nodes.

We call the placement procedure of each new node as a 'round' and the set of nodes placed by the end of round j as S_j. At round j, $2 \leq j \leq N$, first a random position is generated. To be accepted, the position has to pass the proximity test and the maximum degree test. To check the

latter, the approximate degrees (that is, currently considered degrees) of node j and all nodes in S_{j-1} are calculated assuming that node j were placed to this new position. If none of these degrees is greater than or equal to the maximum degree allowed $d_{\max}$(we used $d_{max}=d+3$), the position is accepted. Otherwise, a new position is generated.

Number of nodes/ avg. number of neighbors	n=500	n=250	n=100	n=50	n=25
d=10	78%	82%	90%	97%	95%
d=9	67%	74%	85%	85%	96%
d=8	47%	53%	63%	84%	84%
d=7	22%	26%	46%	58%	69%
d=6	68%	65%	54%	41%	31%
d=5	65%	62%	51%	37%	27%
d=4	58%	55%	48%	33%	24%
d=3	44%	41%	39%	28%	19%

Table 6-1. Probability UDG is 1-connected for pairs (n, d).

Number of nodes/ avg. number of neighbors	n=500	n=250	n=100	n=50	n=25
d=10	90%	87%	85%	87%	83%
d=9	88%	82%	80%	80%	78%
d=8	75%	69%	65%	64%	69%
d=7	69%	61%	57%	58%	55%
d=6	43%	38%	33%	33%	27%
d=5	22%	22%	20%	10%	8%
d=4	11%	9%	5%	5%	5%
d=3	0%	0%	0%	0%	0%

Table 6-2. Probability that a connected UDG is 2-connected for pairs (*n, d*).

Number of nodes/ avg. number of neighbors	n=500	n=250	n=100	n=50	n=25
d=10	90%	87%	85%	87%	85%
d=9	88%	82%	80%	80%	82%
d=8	74%	65%	63%	60%	64%
d=7	61%	57%	54%	50%	55%
d=6	29%	26%	20%	18%	17%
d=5	6%	5%	0%	0%	0%
d=4	0%	0%	0%	0%	0%
d=3	0%	0%	0%	0%	0%

Table 6-3. Probability that a connected UDG is 3-connected for pairs (*n, d*).

The number of neighbors of each node is part of our criteria. If $d>5$ then it is highly likely that almost all nodes have at least 3 neighbors, which in fact is maximum tested, and directly part of LND and LSCD.

We considered k-connected and k-unconnected graphs separately. For each graph we calculated how many nodes made a false or correct decision.

In our experiments we considered 100 graphs (for each p) that are 2 and 3-connected. The same was done for 2 and 3-unconnected graphs. We only considered connected graphs.

Our localized algorithms were used to test the k-connectivity of the network. Global algorithms were used to check the correctness of the localized algorithms. We used a modified DFS (depth first search) algorithm to test k-connectivity of the UDG. To test 1-connectivity we ran a DFS search on a the UDG and marked nodes that were visited. If all nodes were visited, we declared that the graph is 1-connected. For detection of 2-connectivity we used the following algorithm: LND, LSCD and LCND.

6.1 Local neighbour LND criterion

The *local_neighbour_detection* (LND) crierion detects the nodes that declare the graph as being k-connected or being k-unconnected. Let us consider a 2-connected graph. This means that each node in a 2-connected graph has at least 2 neighbours. If it didn't have 2 neighbours, the graph wouldn't be 2-connected. Therefore, if a given graph is in fact 2-connected, the LND algorithm should always correctly detect it. If a graph is 3-connected we will also always detect this property since in a 3-connected graph there can't be any nodes with less than three neighbors.

If a given graph is 2-unconnected; it can happen that a given node has less than 2 neighbours. The percentages of nodes that falsely classify a 2-unconnected graph have been recorded in Table 6-4 and the percentages of nodes that falsely classify a 3-unconnceted graph are recorded in Table 6-5.

n=500

Density (d)	detection ratio (p=1)	detection ratio (p=2)	detection ratio (p=3)
10	0.4400	1.6800	4.2200
9	0.8800	2.2400	7.2200
8	1.2600	3.3600	11.7000
7	2.1200	3.8800	12.7200
6	2.1600	4.2600	15.9400
5	4.3000	7.7600	18.3000
4	8.3400	13.7400	21.9600
3	N/A	N/A	N/A

Table 6-4. Percentage that a node, in a 2-unconnected graph, correctly classified the graph as 2-unconnected

Density (d)	detection ratio (p=1)	detection ratio (p=2)	detection ratio (p=3)
10	1.1400	3.5000	4.3600
9	1.5600	4.5600	7.3600
8	1.4600	5.7200	9.6800
7	2.2200	11.8400	12.7000
6	3.2800	12.2800	20.4800
5	13.6800	18.2800	23.9400
4	27.780	33.7400	39.9600

Table 6-5. Percentage that a node, in a 3-unconnected graph, correctly classified the graph as 3–unconnected by LND.

After analysis we conclude that the LND algorithm gives unclear results for the detection of *k*-connectivity. Even though the results seem to be satisfactory for detection of *k*-connectivity by this algorithm, it does not correctly detect due to the following reason. The local algorithm tests if each node has at least *k* neighbors up to 3 hops. For detection of 2-connectivity, each node tests if there exits a node up to 3 hops away with the degree at least 2. Hence, this algorithm will only

detect the graph as not being 2-connected if there exists a node that has degree 1. The nodes that have degree 1 can only be found as the outer nodes that are connected to the rest of the network. Consequently, this algorithm does not handle many cases that should be considered for detection of *k*-connectivity. In Figure 12, LND didn't detect any nodes whose removal would disconnect the graph into components, however the red node X is a true critical node (detected by *pos_CN* algorithm) whose removal would separate the graph into components, hence due to the node X, we don't have 2-connected graph.

For detection of 3-connectivity, this algorithm also does not handle all the cases; hence, this algorithm is not reliable for detection of *k*-connectivity. The description of this algorithm and its false detection is described on the next page.

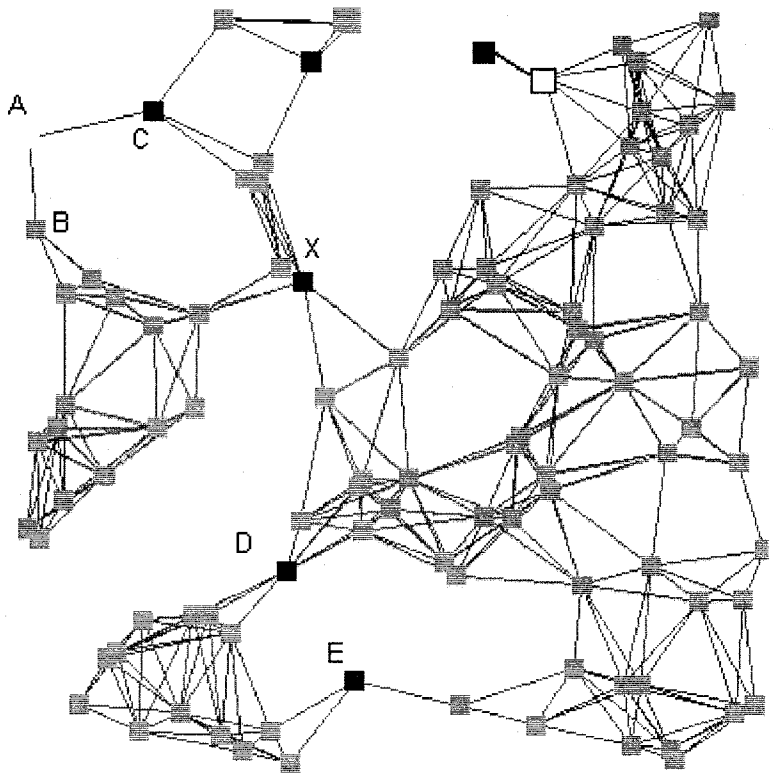

Figure 12. An instance of an ad-hoc network for d=9

The yellow node A in Figure 12 is the only node that has less than 3 neighbours. So, all the nodes up to 3 hops away from the node A will detect the yellow node and declare this graph as not being 3-connected.

This graph is globally declared as 3-unconnected because the removal of nodes D and E as well as B and C would separate the graph into components. Please note that there are more pairs of nodes whose removal would disconnect the graph into components

The definition of 3-connectivity states that a graph is 3-connected if the removal of any 2 nodes will not partition the graph into components. The main reason behind the *LND* algorithm is

that if node A has only 2 neighbours B and C, then the removal of B and C would partition the graph into components. Overall, since A is disconnected from the rest (by removing B and C), by definition, it means it is not 3-connected because of B and C (and some other pairs). Basically, the local_neighbour_detection algorithm should be modified in such a way so that it finds 2 nodes whose removal would partition the graph into components but without using global algorithms. Therefore, we proposed the *local_critical_node_detection* algorithm (see section 6.3). This algorithm will improve the performance and at the same time give accurate results.

6.2 Local subgraph connectivity detection (LSCD) criterion

The LSCD criterion detects the nodes that declare the graph as being k-(un)connected. If a given node declares the graph as being k-connected, there is a certain probability that this declaration is correct. The number of nodes that falsely classify a graph as a k-(un)connected by the LSCD algorithm is shown in tables 6-6, 6-7, 6-8 and 6-9 for n=500. For n=250, n=100 and for n=50 tables can be found in Appendix A.

Results for n = 500

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	2.0400	1.4600	0.7200
9	2.2200	1.6600	1.0400
8	2.9400	2.2800	1.1000
7	3.0200	2.4200	1.8800
6	3.1000	2.7800	2.0800
5	5.8000	3.7600	3.3600
4	6.7000	4.2800	3.7200

Table 6-6. Probability that a node, in a 2-connected graph, falsely classified the graph as 2–unconnected by LSCD.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	2.6200	1.2400	0.9800
9	3.1400	1.8200	1.2800
8	3.6400	1.9400	1.5200
7	5.3600	3.7200	2.8200
6	6.2800	5.3000	3.9600
5	8.5600	7.9000	7.5600

Table 6-7. Probability that a node, in a 3-connected graph, falsely classified the graph as 3–unconnected by LSCD.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	2.7200	3.3800	6.6600
9	3.0400	4.7400	7.9200
8	6.1800	7.3200	13.0600
7	8.3200	9.4600	14.1800
6	11.5400	14.8800	17.7400
5	22.3400	26.7400	32.2600
4	24.2800	28.9800	35.5400

Table 6-8. Probability that a node, in a 2-unconnected graph, correctly classified the graph as 2–unconnected by LSCD.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	5.740	11.140	14.360
9	9.620	17.660	21.980
8	17.080	19.920	26.540
7	20.620	22.700	29.340
6	26.740	29.360	34.580
5	30.360	36.460	42.580
4	37.580	42.840	47.580

Table 6-9. Number of nodes in a 3-unconnected graph which correctly classify the graph as 3–unconnected by LSCD.

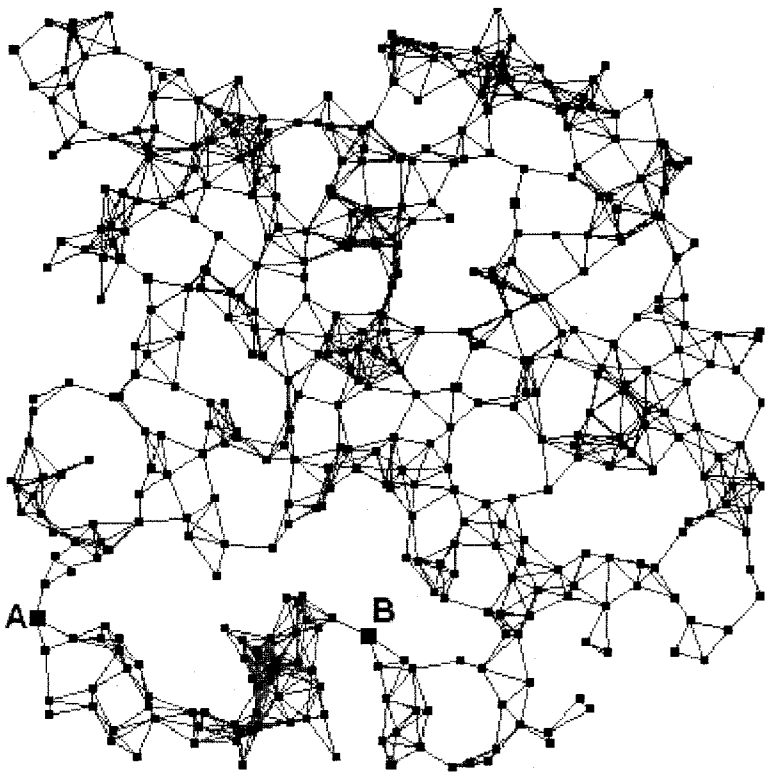

Figure 13. An instance of a 2-connected graph for d=6.

Let us discuss Figure 13, which is 2-connected, but 3-unconnected (detected globally). When we apply the first condition of LSCD we will not find any nodes with less than 2 neighbours so we can proceed with verifying the second condition of the algorithm. If we consider node A, the subgraph of p-hop neighbours of A will be declared as 2-unconnected hence, 3-unconnected. The classification of this graph as 2-unconnceted is false since the removal of A will not affect the graph's connectivity; however the removal of A and B will disconnect the graph into separate components. So node A will falsely declare the graph as 2-unconnected.

In this graph there are quite a few nodes which will declare the graph as 2-unconnected, thus 3-unconnected. Some of the nodes which declare the graph as 2-connected but 3-unconnected are shown in blue.

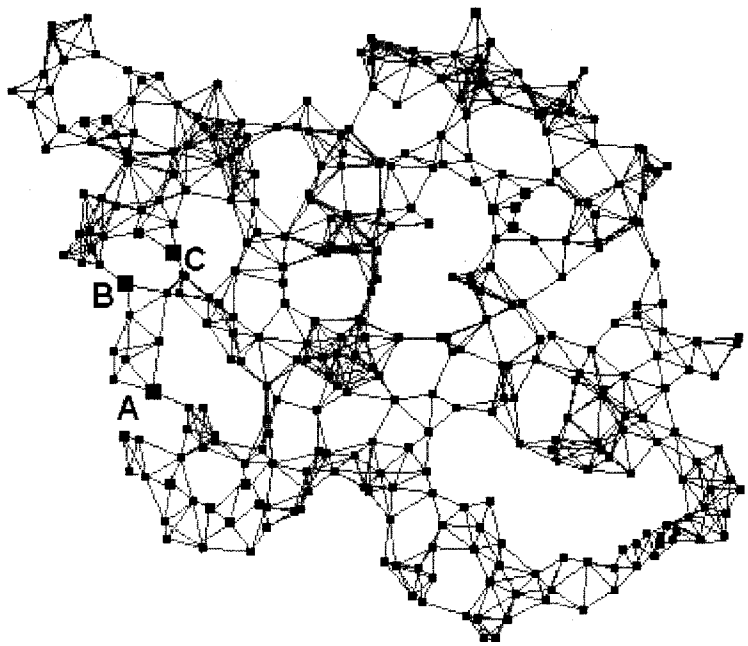

Figure 14. An instance of a 3-connected graph for d=6.

The graph in figure 14 is declared 3-connected when running the global algorithm. With *LSCD* only a few nodes falsely declare the graph as 2 and 3-unconncted. If we consider node *A*, the first condition will be satisfied but second one will fail because the subgraph of *p-hop* neighbours of *A* is not 2-conneceted. So node *A* will classify this graph as 2-unconnected. If we consider node B, the first condition is satisfied but the subgraph of *B* is 2-connected but not 3-connected since the removal of *C* would separate the subgraph into components. Thus node *B* correctly declares the graph as 3-unconnected.

6.3 Local critical node detection (LCND) criterion

The *local_critical_node_detection* criterion detects the nodes that declare the graph as being k-(un)connected. If a given node declares the graph as being k-connected, there is a certain probability that this declaration is correct. The number of nodes that falsely classify a graph as a k-(un)connected by the *local_critical_node_detection* algorithm is shown in tables 6-1, 6-2, 6-3, 6-4.

Results for n = 500

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	2.040	1.460	0.720
9	2.560	1.820	1.080
8	3.100	2.480	1.140
7	3.140	2.540	2.020
6	3.220	2.860	2.320
5	6.240	3.856	3.580
4	7.700	5.880	5.520

Table 6-10. Probability that a node, in a 2-connected graph, falsely classified the graph as 2–unconnected by LCND

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	2.660	1.240	1.080
9	3.380	1.860	1.360
8	3.880	2.080	1.760
7	5.420	3.920	2.860
6	6.660	5.460	4.080
5	10.420	9.100	8.060

Table 6-11. Probability that a node, in a 3-connected graph, falsely classified the graph as 3–unconnected by LCND.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	2.720	3.380	6.960
9	3.040	4.740	7.620
8	6.180	7.320	12.060
7	9.320	11.460	13.180
6	10.540	12.880	13.740
5	21.360	24.660	25.500
4	23.780	25.320	26.960

Table 6-12. Probability that a node, in a 2-unconnected graph, correctly classified the graph as 2–unconnected by LCND.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	4.740	11.140	10.360
9	10.620	11.660	12.980
8	12.080	17.920	23.540
7	15.620	19.700	25.340
6	21.740	25.360	26.580
5	23.280	26.580	27.280
4	25.080	27.460	29.560

Table 6-13. Probability that a node, in a 3-unconnected graph, correctly classified the graph as 3–unconnected by LCND.

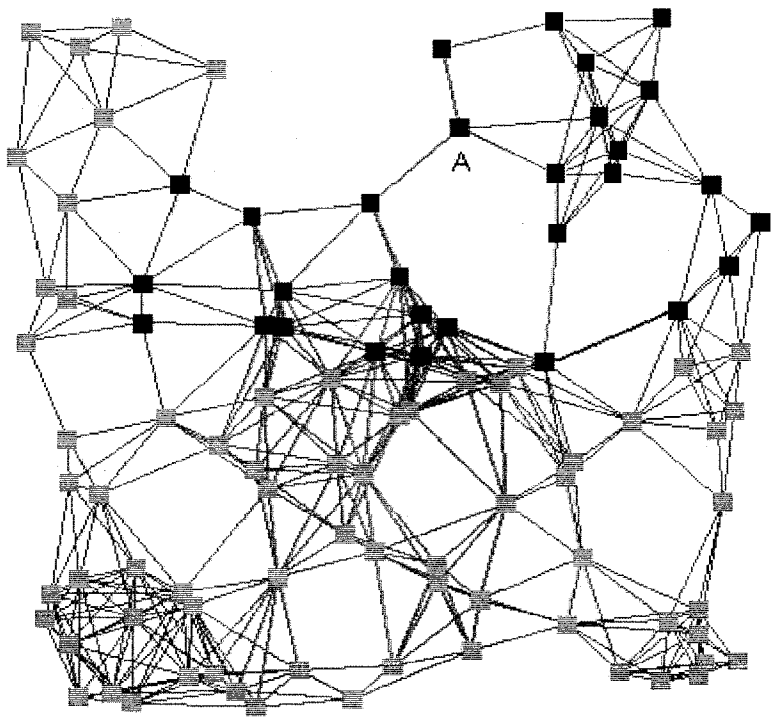

Figure 15. An example of an ad-hoc network for n=100, d=10.

Figure 15 has 2 and 3-conectivity. If we run the *LCND* on the node *A*, with p =1, 2 we declare the node *A* as critical, hence the graph as 2 and 3 unconnected. For p=3 the algorithm doesn't not declare the node *A* as critical, so we can proceed with detection of 3-connectivity. Now we consider a subgraph of 3-hop neighbours of *A*, excluding *A*, and run the algorithm for detection of critical nodes (i.e., the *pos_CN* algorithm*)* on every node in the subgraph. Blue nodes represent the subgraph of *A*. In this subgraph (excluding *A*) we didn't find any critical nodes, thus removal of *A* and any other node in its subgraph would not affect the subgraph's connectivity; so node *A* declares the graph as 3-connected.

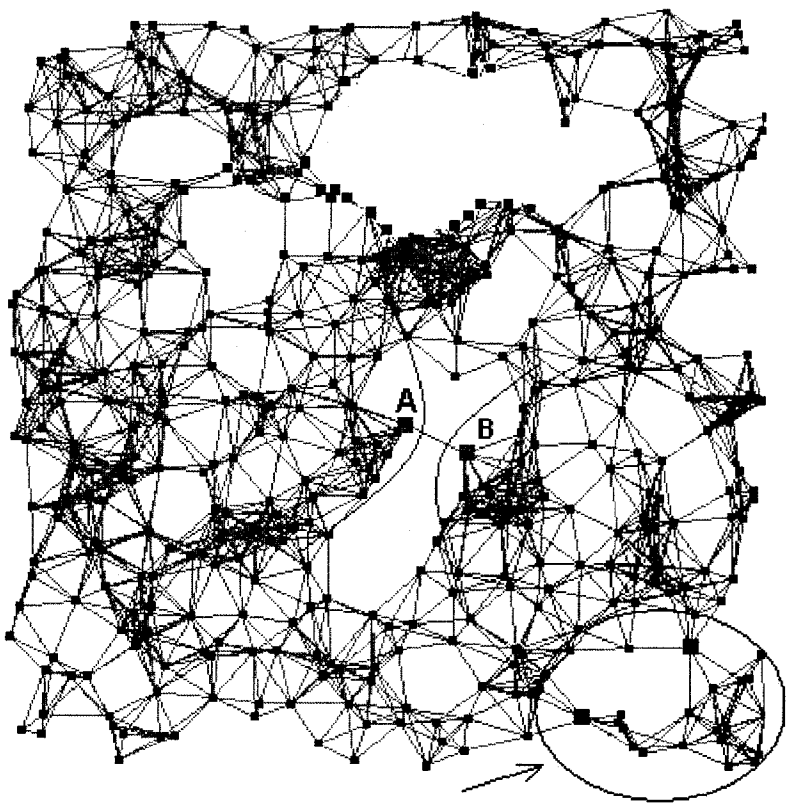

Figure 16. An example of 3-unconnected network n=500 d=10.

In Figure 16, with the global algorithm we declare the graph as 2-connected. However due to the red nodes in the encircled area, the graph is declared as 3-unconnected. The removal of red nodes will disconnect the graph into separate components. If we run the *LCND* on each node in the graph, blue nodes and red nodes will declare the graph as 3-unconnceted. The rest of the nodes will declare the graph as 3-connected.

Let us now consider Figure 17 where n=500 and d=6.

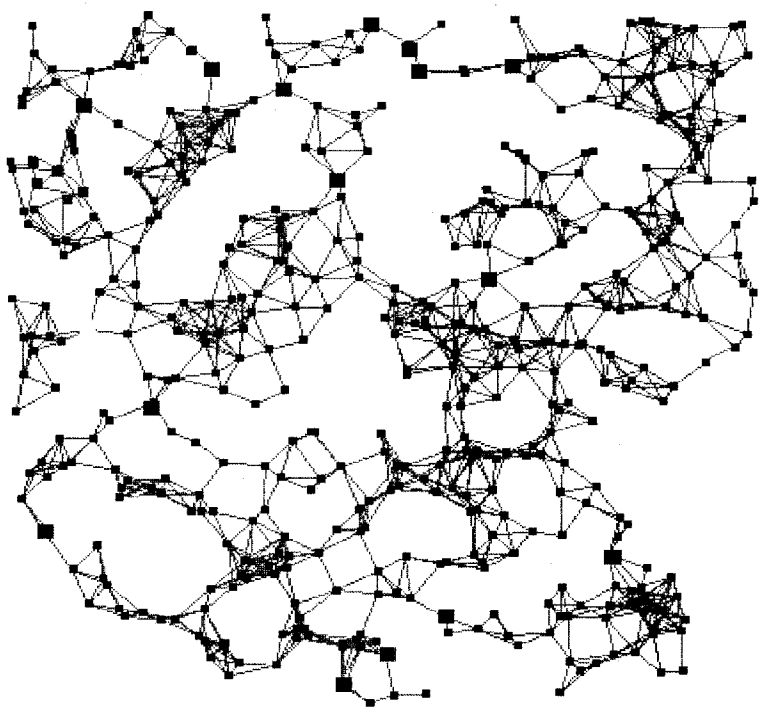

Figure 17. An example of 3-unconnected network n=500 d=6.

The global algorithm classifies this graph as 2-unconnected, hence 3-unconnected. The graph is declared as 2-unconnected because the removal of yellow node will separate the graph

into components. The *local_critical_node_detection* criterion will declare quite a few nodes will classify this graph as 2 and 3-unconnected. Red and blue nodes represent some of the nodes that make false detections for 2 and 3-connectivity respectively.

Chapter 7

Conclusions, future work, and open problems

We described several localized protocols for fast and mostly accurate detection of critical links and nodes in ad hoc networks. Existing algorithms for detection of critical links, nodes and k-connectivity have a high complexity and require knowledge of the topology of a network. This thesis showed that the localized criteria could successfully be used in order to detect critical links/nodes and for detection of k-connectivity. Our experiments show that the localized algorithms are reasonably accurate.

The proposed localized algorithms for detection of critical links and nodes are very beneficial for service replication. In this scenario, a particular route between a client and a server node is monitored for the presence of critical links or nodes. If such node or link is detected, an alternate service in the network is searched, or service is replicated.

The proposed localized partition detection schemes may be applied to sensor network scenarios. Sensors that detect their criticality or existence of critical links may report to monitoring center, asking for deployment of additional sensors in the area, or wakening up some nearby sleeping sensors. We intend to consider this application of localized partition detection for designing sensor activity scheduling protocols.

Detection of k-connectivity is an important property in mobile ad hoc, sensor and actuator networks. Mobility may separate a network into components. Since the network should always be connected, the need for fast detection of 2 and 3-connectivity is extremely important so that the separation of the network can be avoided. For example if we are trying to send a message from a

source to a destination we may need to traverse a lot of nodes in the network. Each node makes a localized decision if the network is 2 or 3-connected. If a node declares locally that observed movements could separate the graph into components, the node may initiate certain appropriate response, depending on the task being executed. For example, it may try to move closer towards neighbouring nodes which have at least k neighbours. This way there is a high probability that node's movement would not have any affect on networks separation. The localized algorithms are fast and there is no need for position information of all the nodes in a network. Whereas globalized algorithms demand time and communication resources. By the time a base station completes those algorithms and detects the graph's connectivity, the topology of the network may significantly change, hence the information may not be helpful anymore. That is the reason why localized algorithms perform better, albeit they sometimes make false declaration.

For detection of critical nodes the best results gives the CN criterion with topological information and the best results for detection of critical links are obtained by CL criterion also with topological information. Half of the locally estimated critical nodes and links were indeed globally critical even for p=1 (the accuracy increases to over 70% for p=2 and over 80% for p=3), for an average number of neighbours ranging from 3 to 15. The errors mostly occur when alternative routes exist but are relatively long, and therefore may not provide satisfactory service in applications

All three proposed algorithms for detection of k-connectivity give excellent accuracy for local confirmation of global k-connectivity. However, they all have problems confirming k-disconnectivity. The LCND criterion appears to be the most accurate. The percentage of nodes in a k-connected graph, falsely declaring a graph to be k-disconnected is well under 10%, however the

percentage of nodes in a k-disconnected graph that correctly classify the graph as k-disconnected is not that high. It ranges from approximately 3% to 30%.

This study may provide grounds for further investigation in this direction. This includes the properties of unit disk graphs that are actually created by coordinated movements of robots, actuators, human, and vehicles in a wireless network. The experimental data presented here depends on the nature of graphs generated, and may not truly reflect the nature of practical networks constructed by coordinated nodes. New localized protocols may be needed for certain applications. Applications of presented protocols are also subject of further investigation.

Another interesting problem is to assign, in localized manner (as opposed to globalized solution [HIM]), transmission radii to each node so that the network is k-connected with high probability.

Chapter 8

References

[ABS] Duque Anton, M., Bruyaux, F., Semal, P.: Measuring the survivability of a network: connectivity and rest-connectivity; European Transactions on Telecommunications, 2000.

[AS] Furuzan Atay, Ivan Stojmenovic. Generating Random Graphs for Wireless Actuator Networks, manuscript, 2006.

[B2] S. Baase, Computer Algorithms, Introduction to Design and Analysis, Addison Wesley, 1988, 184-191.

[BHM] M. Bahramgiri, M. Hajiaghayi, V.S. Mirokni, Fault-tolerant and 3-dimensional distributed topology control algorithms in ad hoc networks, IEEE ICCCN, 2002.

[B] C. Bettstetter, On the minimum node degree and connectivity of a wireless multihop network, Proc. ACM MobiHoc, 2002.

[CYD] G. Cao, L. Yin and C.R. Das, Cooperative cache-based data access in ad hoc networks, IEEE Computer Magazine, February 2004, 32-39.

[GDS] D. Goyal and J. Caffery, Partitioning avoidance in mobile ad hoc networks using network survivability concepts, Proc. IEEE Int. Symp. Computers and Communications ISCC, Taormina, Italy, July 2002, 553-558.

[H] T. Hara, Replica allocation methods in ad hoc networks with data update, Mobile Networks and Applications 8, 2003, 343-354.

[HIM] M. T. Hajiaghayi, N. Immorlica, V.S. Mirrokni, Power optimization in fault-tolerant topology control algorithms for wireless multi-hop networks, Proc. ACM MOBICOM, Sept. 2003.

[HSC] M. Hauspie, D. Simplot, and J. Carle. Partition detection in mobile ad-hoc networks. *Proc. 2nd IFIP Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET 2003)*, (Mahdia, Tunisia, 2003).

[HSJS] M. Hauspie, D. Simplot-Ryl, M. Jorgic, I. Stojmenovic, Localized algorithms for service replication in ad hoc networks, in preparation.

[JS] L. Jia and C. Scheideler, On local algorithms for topology control and routing in ad hoc networks, *Proceedings of the 15th Annual ACM Symposium on Parallel Algorithms and Architectures,* pages 220-229, June 2003.

[JGKNS] Milenko Jorgic, Nishith Goel, Kalai Kalaichevan, Amiya Nayak, Ivan Stojmenovic, Localized detection of k-connectivity in wireless ad hoc, actuator and sensor networks, 16th International Conference on Computer Communications and Networks ICCCN, Hawaii, Aug. 2007, to appear..

[JSHS] M. Jorgic, I. Stojmenovic, M. Hauspie, D. Simplot-Ryl, Localized algorithms for detection of critical nodes and links for connectivity in ad hoc networks, *The Third Annual Mediterranean Ad Hoc Networking Workshop Med-Hoc-Net,* Bodrum, Turkey, June 27-30, 2004, 360-371.

[K] H. Koskinen, Statistical model describing connectivity in ad hoc networks, Proc. Modeling and optimization in mobile, ad hoc, wireless networks WiOpt, INRIA, Sophia-Antipolis, March 2003.

[KMP] G. Karumanchi, S. Muralidharan, R. Prakash, Information dissemination in partitionable mobile ad hoc networks, Proc. IEEE Symp. Reliable Distributed Systems, October 1999.

[LR] Q. Li and D. Rus, Communication in disconnected ad hoc networks using message relays, ACM MOBICOM 2000; J. Parallel and Distributed Computing, 63, 2003, 75-86.

[LWWY] X.-Y. Li, Y. Wang, P.-J. Wan, and C.-W. Yi, Robust Deployment and Fault Tolerant Topology Control for Wireless Ad Hoc Networks, *Wiley Journal on Wireless Communications and Mobile Computing*, Volume 4, Issue 1 (Feb. 2004), pp. 109- 125.

[M] U. Manber, Introduction to Algorithms, A Creative Approach, Addison Wesley, 1989, p. 217-225.

[PC] V.D. Park and M. S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. *IEEE INFOCOM '97*, Kobe, Japan, April 1997.

[RWS] H. Ritter, R. Winter, J. Schiller, A partition detection system for mobile ad-hoc networks, IEEE SECON, 2004

[SCN] S.H. Shah, K. Chen, and K. Nahrstedt. Cross-layer design for data accessibility in mobile ad hocnetworks. In *Proc. of 5th World multiconference on systemics, cybernetics and informatics (SCI 2001), Orlando, Florida*, July 2001; Wireless Personal Communications, vol. 21, pp. 49-76, 2002.

[SRV] I. Stojmenovic, M. Russell, and B. Vukojevic, Depth first search and location based localized routing and QoS routing in wireless networks, Computer Science, SITE, University of Ottawa, TR-00-01, January 2000; Computers and Informatics, Vol. 21, No. 2, 2002, 149-165.

[SL1]. Ivan Stojmenovic and Xu Lin, Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks, IEEE Transactions on Parallel and Distributed Systems, Vol. 12, No. 10, October 2001, 1023-1032.

[T] R. Tarjan, Depth first search and linear graph algorithms, SIAM J. Computing, 1, 2, 146-160, 1972.

[VB] A. Vahadat, D. Becker, Epidemic routing for partially connected ad hoc networks, Technical Report CS-200006, Duke University, April 2000.

[WL] K.H. Wang, B. Li, Efficient and guaranteed service coverage in partitionable mobile ad hoc networks, INFOCOM, 2002.

[WL1] K.H. Wang and B. Li, Group mobility and partition prediction in wireless ad hoc networks, IEEE ICC, April 2002.

[XK] F. Xue, P. R. Kumar, The number of neighbors needed for connectivity of wireless networks, Wireless Networks, Vol. 10, No. 2, pp. 169-181, March 2004

[ZH] H. Zhang, J. Hou, On the critical total power for asymptotic k-connectivity in wireless networks, IEEE INFOCOM, 2005.

Chapter 9

Appendix A

Results for n = 250 using LSCD

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	2.450	1.820	1.040
9	2.850	2.240	1.220
8	4.080	2.980	2.020
7	4.680	4.080	2.240
6	4.920	4.520	3.640
5	8.640	7.720	6.780
4	9.120	8.040	7.620

Table 9-1. Probability that a node, in a 2-connected graph, falsely classified the graph as 2–unconnected by LSCD.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	3.120	2.960	2.140
9	3.620	3.780	2.080
8	4.440	4.160	2.280
7	6.820	5.980	3.120
6	8.920	7.690	4.040
5	11.320	10.060	8.050

Table 9-2. Probability that a node, in a 3-connected graph, falsely classified the graph as 3–unconnected by LSCD.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	2.240	3.120	4.840
9	5.880	6.440	7.420
8	6.620	7.380	8.080
7	9.040	10.450	12.880
6	11.880	13.640	16.840
5	26.860	29.940	31.960
4	27.860	30.880	33.640

Table 9-3. Probability that a node, in a 2-unconnected graph, correctly classified the graph as 2–unconnected by LSCD.

Density (d)	detection ratio in % (p=1)	detection ratio in % v(p=2)	detection ratio in % (p=3)
10	5.88	11.080	14.520
9	9.840	16.921	20.020
8	15.980	18.850	23.850
7	21.850	24.160	27.480
6	23.860	29.800	31.920
5	28.450	35.020	38.020
4	33.040	37.420	41.080

Table 9-4. Probability that a node, in a 3-unconnected graph, correctly classified the graph as 3–unconnected by LSCD.

Results for n = 100 using LSCD

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	6.600	4.700	2.400
9	7.100	5.400	3.700
8	7.900	6.100	4.800
7	9.300	7.800	6.300
6	9.900	8.900	7.100
5	12.100	11.200	10.100
4	14.200	13.700	12.900

Table 9-5. Probability that a node, in a 2-connected graph, falsely classified the graph as 2–unconnected by LSCD.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	7.100	5.700	4.100
9	9.400	7.900	4.900
8	11.100	9.100	5.100
7	14.800	11.900	8.900
6	17.800	14.300	11.300
5	20.400	18.900	17.400

Table 9-6. Probability that a node, in a 3-connected graph, falsely classified the graph as 3–unconnected by LSCD.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	2.400	4.900	5.800
9	5.900	5.700	7.300
8	6.100	7.200	8.400
7	9.800	10.300	12.600
6	12.500	14.200	17.100
5	26.300	29.100	31.700
4	27.870	30.100	33.800

Table 9-7 Probability that a node, in a 2-unconnected graph, correctly classified the graph as 2–connected by LSCD.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	5.100	11.700	15.100
9	9.900	16.900	20.400
8	15.100	18.100	23.100
7	21.900	24..900	27.800
6	23.300	29.300	31.800
5	28.900	35.100	35.600
4	33.140	36.480	41.180

Table 9-8. Probability that a node, in a 3-unconnected graph, correctly classified the graph as 3–connected by LSCD.

Results for $n = 50$ using LSCD

Density (d)	detection ratio % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	6.400	5.800	2.800
9	8.200	6.800	5.600
8	9.800	8.200	7.200
7	10.800	9.800	8.200
6	13.600	11.800	10.400
5	15.400	14.100	12.500
4	17.600	16.800	15.700

Table 9-9. Probability that a node, in a 2-connected graph, falsely classified the graph as 2–unconnected by LSCD.

Density (d)	detection ratio % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	8.800	7.800	4.200
9	9.600	8.600	6.600
8	10.400	9.600	8.200
7	11.600	11.200	9.800
6	14.400	12.800	12.200
5	17.800	15.600	14.100

Table 9-10. Probability that a node, in a 3-connected graph, falsely classified the graph as 3–unconnected by LSCD.

Density (d)	detection ratio in % (p=1)	detection ratio % (p=2)	detection ratio in % (p=3)
10	19.800	20.800	22.400
9	25.600	26.800	28.200
8	27.260	28.200	31.800
7	29.280	31.800	35.800
6	44.400	48.800	51.600
5	58.250	60.400	65.700
4	61.400	63.100	67.400

Table 9-11. Probability that a node, in a 2-unconnected graph, correctly classified the graph as 2–unconnected by LSCD.

Density (d)	detection ratio in % (p=1)	detection ratio % (p=2)	detection ratio in % (p=3)
10	20.200	21.800	23.800
9	25.600	26.600	29.600
8	27.200	29.600	32.400
7	29.800	33.200	36.600
6	45.200	49.800	55.400
5	61.700	62.400	67.400
4	63.550	68.540	71.450

Table 9-12. Probability that a node, in a 3-unconnected graph, correctly classified the graph as 3–unconnected by LSCD.

Results for n = 250 using LCND

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	2.840	2.120	1.040
9	3.720	2.440	1.680
8	4.280	3.080	2.320
7	5.080	4.280	2.840
6	5.320	4.520	3.800
5	9.040	7.720	7.080
4	10.120	9.840	8.320

Table 9-13. Probability that a node, in a 2-connected graph, falsely classified the graph as 2–unconnected by LCND.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	4.120	3.160	1.640
9	4.720	3.920	2.160
8	5.400	4.360	2.280
7	7.480	6.160	4.400
6	8.920	7.800	4.600
5	12.720	11.640	8.360

Table 9-14. Probability that a node, in a 3-connected graph, falsely classified the graph as 3–unconnected by LCND.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	2040	3.120	5.840
9	2.80	4.440	6.720
8	6.320	7.080	11.280
7	8.840	11.280	13.080
6	9.800	12.520	13.320
5	20.2440	23.840	24.680
4	21.120	24.500	25.840

Table 9-15. Probability that a node, in a 2-unconnected graph, correctly classified the graph as 2–unconnected by LCND.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	4.640	10.480	10.120
9	9.120	11.720	11.720
8	12.280	17.360	22.400
7	15.440	20.160	26.480
6	20.400	25.800	26.920
5	23.120	26.020	27.020
4	24.440	27.320	28.880

Table 9-16 Probability that a node, in a 3-unconnected graph, correctly classified the graph as 3–unconnected by LCND.

Results for n = 100 by using LCND

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	6.800	4.900	2.400
9	7.300	5.700	3.900
8	8.400	6.200	5.100
7	9.600	8.300	6.800
6	10.100	9.200	7.500
5	12.400	11.600	10.400
4	14.800	14.200	13.200

Table 9-17 Probability that a node, in a 2-connected graph, falsely classified the graph as 2–unconnected by LCND.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	7.100	5.700	4.100
9	9.400	7.900	4.900
8	11.100	9.100	5.100
7	14.800	11.900	8.900
6	17.800	14.300	11.300
5	20.400	18.900	17.400

Table 9-18 Probability that a node, in a 3-connected graph, falsely classified the graph as 3–unconnected by LCND.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	4.300	9.700	9.800
9	9.500	11.900	11.300
8	12.900	17.600	22.400
7	15.800	21.000	24.600
6	20.500	24.200	25.100
5	23.300	28.100	27.700
4	24.700	26.900	28.700

Table 9-19. Probability that a node, in a 2-unconnected graph, correctly classified the graph as 2–connected by LCND.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	4.100	9.700	10.100
9	9.900	11.900	11.400
8	11.900	18.100	23.100
7	16.000	20.900	25.800
6	20.300	24.300	26.800
5	22.900	25.900	27.600
4	25.100	26.900	29.100

Table 9-20. Probability that a node, in a 3-unconnected graph, correctly classified the graph as 3–connected by LCND.

Results for $n = 50$ by using LCND

Density (d)	detection ratio % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	6.400	5.800	2.800
9	8.200	6.800	5.600
8	9.800	8.200	7.200
7	10.800	9.800	8.200
6	13.600	11.800	10.400
5	15.400	14.100	12.500
4	17.600	16.800	15.700

Table 9-21. Probability that a node, in a 2-connected graph, falsely classified the graph as 2–unconnected by LCND.

Density (d)	detection ratio % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	8.800	7.800	4.200
9	9.600	8.600	6.600
8	10.400	9.600	8.200
7	11.600	11.200	9.800
6	14.400	12.800	12.200
5	17.800	15.600	14.100

Table 9-22. Probability that a node, in a 3-connected graph, falsely classified the graph as 3–unconnected by LCND.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	12.800	25.800	26.400
9	15.600	26.800	28.200
8	17.200	28.200	29.800
7	18.200	29.800	30.800
6	20.400	31.800	33.600
5	24.200	35.400	36.700
4	29.400	30.100	41.400

Table 9-23. Probability that a node, in a 2-unconnected graph, correctly classified the graph as 2–unconnected by LCND.

Density (d)	detection ratio in % (p=1)	detection ratio in % (p=2)	detection ratio in % (p=3)
10	14.200	27.800	28.800
9	16.600	28.600	29.600
8	18.200	29.600	30.400
7	19.800	31.200	31.600
6	22.200	32.800	34.400
5	29.700	36.400	38.400
4	31.240	35.450	45.440

Table 9-24. Probability that a node, in a 3-unconnected graph, correctly classified the graph as 3–unconnected by LCND.