Prediction of Partitioning in Location-aware Mobile Ad Hoc Networks

Bratislav Milic, Nikola Milanovic, Miroslaw Malek
Humboldt Universität zu Berlin
{milic,milanovi,malek}@informatik.hu-berlin.de

Abstract

We propose an algorithm for detection of partitioning in location-aware mobile ad hoc networks. The partitioning occurs when movement pattern of nodes is such that they separate into groups that cannot communicate with each other. We use information about node position and speed in order to build a model that is able to predict when partitioning will occur and which link is critical using properties of planar graphs that represent the network. Our algorithm is distributed and uses only local topology knowledge where nodes keep track of position and speed of their one-hop neighbors.

Indexed terms: mobile ad hoc networks, partition prediction, perimeter routing, Gabriel graph

1. Introduction

Mobile ad hoc networks (manets) are praised for their ability to adjust themselves automatically to rapidly changing conditions in the surrounding environment. However, that does not come without a price. When compared with widespread wired/wireless networks (such as the Internet) they are noticeably lacking in throughput, security, QoS, scalability. Therefore, it is logical to use them only in specific situations, such as catastrophes or battlefields when other communication infrastructure is not available or it is damaged.

In ideal case, preconfiguration before deployment of nodes is not needed, nodes are moving freely, routing of messages is done in a distributed manner. Most of the research in the area was focused on these topics, trying to improve weak points of ad hoc networks and to enforce the benefits that they offer, assuming one of the most important features of each network – connectivity. Sometimes, partitioning problem is mentioned, but it is assumed that network will regain connectivity by itself. This may be the case only if density of nodes in the observed area is relatively high. But even in the case that network gets reconnected, there could be time intervals when a node cannot communicate with all other nodes in the network, but only with the nodes from its partition.

Partitioning causes serious problems in the network: nodes from different partitions cannot communicate with each other, reducing quality of all services the network offers, or making services unavailable. If a density of nodes in the observed area is low, it is possible that network will never regain its full connectivity. The worst case is that a network can go through a series of partitioning ending up as a unconnected group of one-node subnets.

Several reasons can cause network partitioning - shutdown of a device, node movement, physical destruction of a node, energy source depletion. Our goal is to predict partitioning of a network that is caused by the node movement.

So far, existing proposals for detecting mobile ad hoc networks partitioning made wide assumptions regarding capabilities of nodes and underlying transport layer. Solutions introduced centralized servers, global network knowledge at each node, and flooding to test the network for critical links. We will use location and velocity information to predict which link could be broken, and Gabriel graph properties to test whether that link is critical in a localized and efficient way. Velocity and location information will be exchanged only with immediate neighbors keeping topology information locally and reducing the processing load of a node and bandwidth consumption (in contrast with solutions that require for each node to keep track of all node movements in the network). Also, in our solution we assume that nodes in the network can have different transmission ranges, and that the ranges are not time invariant thus introducing one more degree of freedom.

2. Related Work

There are several existing approaches for detection and/or prevention of partitioning in mobile ad hoc networks. Hauspie et al. [7] propose a model that allows prediction of partition occurrence. They construct all disjoint paths between source and destination node and calculate path robustness based on probabilities that some of the links constituting that path will break. They offer two metrics: based on the number of disjoint paths and based on the length of disjoint paths. The first metric says that if there is only one disjoint path between two nodes, it is highly probable that a graph will become disconnected if some link on that path breaks. The second metric states that the longer the path is, the weaker it is in the sense of robustness, meaning that all paths do not contribute evenly to the robustness. The problem with this approach, however, is that it does not guarantee correctness, because the network can get partitioned without this algorithm detecting it. The second problem is that it requires global knowledge of the network topology in order to generate all disjoint paths and calculate link robustness. The benefit is that it does not require any information about positions of mobile nodes.

The approach proposed by Goyal and Caffery [2] is based on the similar idea. It offers a method for detecting a critical link that will, if fails, partition an ad hoc network. The difference is that it is location based. All nodes are location aware and periodically update their neighbors with current locations. The algorithm uses depth first search (DFS) in order to find the critical separation links based on this information. After a potential partition is detected, the algorithm offers prevention mechanisms, either by changing the trajectories of critical nodes, or by adding another helper node to reinforce the link. However, in order to work, this approach requires active approach using DFS, meaning that every node must compute a DFS tree with itself as a root when detecting a partition, which induces significant overhead.

Wang and Li [5] claim that by knowing only local node positions and based on the individual node mobility model, global scale topology changes such as network partitions cannot be predicted. Therefore, they introduce a model for group mobility. Instead of grouping nodes by location, they employ a clustering algorithm that groups nodes by their speeds. A group is then characterized by mean group velocity. They argue that clustering by velocities provides a clearer characterization and separation of mobility groups. Separation of groups leads to the network partition. The problem of this model is the centralized nature of the solution. There must be a central server to which all nodes report their positions and speeds. The server then runs clustering algorithm on this data and tries to predict network partitioning. Another problem is that algorithm assumes that velocities of the mobility groups and mobile nodes are time invariant. Also, it assumes that all groups have the same circular coverage area with mobile nodes uniformly distributed inside.

The last solution that we present is proposed by Chen et al. [4]. It employs a topology prediction algorithm that tries to calculate whether a current movement pattern of two mobile nodes will lead to network partition, assuming that each node has uniform transmission range. The problem is that algorithm requires that each node maintains an update table, in which it stores updates from all other nodes in a network. Therefore, a global knowledge of the entire network is necessary at every node. After partitioning is predicted, the algorithm offers data duplication techniques that logically prevent partitioning.

We present an approach that tries to address problems stated for each algorithm described here. It offers partition prediction based on node position. However, it is local and distributed, in a sense that all calculations are done on the nodes themselves, without any central server, and that all nodes know and communicate only with their neighbors, without global network knowledge. Also, we allow for dynamic node transmission range that can span between minimum and maximum value.

3. Partition Prediction

The ability to predict network partitioning would be very valuable to both network and the application layer. In this section we introduce a model based on Gabriel graph, and then explain our prediction algorithm.

3.1. Model

A set of mobile nodes is situated in a Euclidean plane. Each mobile node is location aware, that is, it knows its x and y coordinates by means such as GPS. It can also calculate its speed, by using differential coordinates obtained in the same manner. The transmission range of each node is time variant, and is given by $r(t) = (1-\epsilon(t))R$, where $0 < \epsilon(t) < 1$ and R is the maximal node range. We assume that the speed of communication is substantially higher compared with the rate at which $\epsilon(t)$ and the velocity of nodes are changing. We also assume that links are bidirectional, meaning that the link between hosts a and b is valid if and only if b can acknowledge a message received from a. This is required for the construction of the Gabriel graph.

The network is represented by a geometric undirected graph $G = (V, E)$, where vertices represent mobile nodes and edges connect nodes that can communicate directly. Each vertex runs the algorithm for Gabriel graph extraction for the purpose of perimeter routing [8, 9]. The Gabriel graph GG is a spanning subgraph of G. It is defined as follows: for every two adjacent nodes a and b in G, the edge (a, b) belongs to the Gabriel graph if no other node of G is located in the disk which has segment (a, b) as a diameter. Otherwise, this edge is removed from the Gabriel graph. As can be seen, Gabriel graph extraction is a distributed process that requires only local topology knowledge. When a Gabriel graph is extracted, routing is performed along its edges. Gabriel graph is planar and connected, and that ensures message delivery by routing along the faces of the graph that intersect the line between the source and the destination using the right-hand rule. An example of perimeter routing using Gabriel graph is shown on Figure 1.

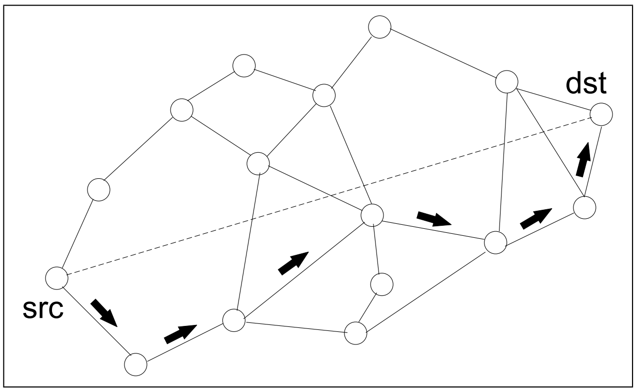

Figure 1. Perimeter routing along the faces of Gabriel graph

Although original algorithm for perimeter routing using Gabriel graphs cannot work with nodes that have variant transmission ranges, Barriere et al. [6] have recently proposed a method for Gabriel graph construction and perimeter routing in cases where ratio of maximum and minimum transmission range is at most $\sqrt{2}$. Since we will be using this scheme for partition prediction, it enforces a restriction upon $\epsilon(t)$ such that $r_{max}(t)/r_{min}(t) < \sqrt{2}$.

3.2. Prediction Algorithm

It is crucial for our algorithm that all nodes in the network have accurate information about the positions of their neighbors. To keep the information on neighbor nodes up to date, two types of notifications are used:

- *Regular updates* – each node is broadcasting regular update of its location and speed to all immediate neighbors. Frequency of regular updates is parameter of the network. Updates have to be more frequent if the ratio of maximum possible speed of nodes in the network and their maximum transmission range is large, and can be less frequent if it is low. Therefore, frequency of regular updates should be the compromise between bandwidth consumption and required accuracy of data.
- *Critical updates* – in case that a node significantly changes speed vector $\vec{v}$ or transmission range $r(t)$, it will broadcast locally that information.

This ensures that nodes will have accurate information for predicting behavior of neighbors and construction of Gabriel graph.

Each node keeps data about neighbor nodes in a table. For each neighbor, the following data is stored: *node-id*, location (x, y), transmission range $r(t)$ and current speed $\vec{v}$. Other data, such as remaining energy of surrounding nodes can be also kept in the table. After receiving an update message, node will update the corresponding entry in the table.

Each node is running local link failure prediction algorithm. Based on the information it has on neighbor nodes, a node is calculating probability that a link will be broken. We are assuming that the probability of link breakage is proportional to distance $l(t)$ of two nodes

$$P_{breaking} \sim \frac{l(t)}{r(t)}$$

where

$$l(t) = \sqrt{(\Delta x + t \cdot \Delta v_x)^2 + (\Delta y + t \cdot \Delta v_y)^2}$$

Δx is difference of x coordinates of the nodes at $t_0 = 0$, Δv_x and Δv_y are differences of x and y components of speed vectors of observed nodes.

To improve the performance, instead of regularly calculating the distance between itself and the neighbor nodes, a node can calculate the time at which the distance to a neighbor node will reach certain limit D. We can define limit D as a product of transmission range and threshold probability which is a system parameter and can be determined taking into account similar discussion as with frequency of updates.

$$D = P_{threshold} \cdot min(r_{currentnode}(t), r_{neighbornode}(t))$$

Then, the critical time in which distance will reach the limit D is given by:

$$t_{critical} = \frac{-(\Delta_x + \Delta_y)^{+}_{-}\sqrt{-\alpha^2 + D^2 \cdot \Omega}}{\Omega}$$

where Δ_x is $\Delta x \Delta v_x$, Δ_y is $\Delta y \Delta v_y$, Ω is $\Delta v_x^2 + \Delta v_y^2$ and α is $\Delta x \Delta v_y - \Delta y \Delta v_x$. This equation will be correct only if, in calculated time period, none of two nodes change their velocity vectors or transmission range. Otherwise, time $t_{critical}$ will have to be recalculated. In case that a node drastically changes its velocity, critical update will provide accurate data on the change.

If two nodes conclude that a link connecting them is about to be broken, they will initiate the algorithm for checking whether the breaking of that link will lead to network partitioning. For that purpose we use planarity of Gabriel graph representing the network.

For the purpose of testing, nodes are not using the link identified as potentially critical. That link will be used only for regular network traffic. Then they delegate which of them will start the link testing phase. For instance, a node with lower *node-id* can be responsible for starting the testing sequence, but other, more sophisticated schemes for achieving fairness can be applied. Once the decision is made, delegated node is sending Link Test message (LTEST) to the other node using the right/left hand rule along the inner side of the face that contains tested link (link A-B). Right/left hand rule is known from maze theory and it states that it is possible to visit every wall in a maze by walking forward and keeping right/left hand on the wall all the time. Note that the tested link is not used in this process. The idea is to check whether there exists an alternative path between two nodes, that is, whether the graph is still connected after the elimination of the tested link. If LTEST message reaches destination node, it replies with Link Not Critical message (LNC) using the link that is being tested. That means that the graph is still connected and the link being tested is not critical. However, if the delegated node receives LTEST again, the graph is partitioned since LTEST traveled along the inner face without reaching destination node and returned to the originating node. It then sends Link Critical message (LC) to the other node using the link being tested, thus informing it that the link connecting them is critical and can lead to network partitioning. Since we are planing to use this algorithm to prevent network partitioning in the future, we are assuming that testing of the link has started early enough and that a tested link between nodes A and B still exist for the purpose of sending LNC and LC messages.

Figure 2 shows an example of testing noncritical link. Assume that nodes A and B are testing their link and that A is delegated to start the testing algorithm. Node A is starting testing sequence by sending LTEST message to B using the right hand rule along the inner side of the face. The alternative path connecting nodes A and B exist, node B will receive LTEST message from A and it will reply with LNC to A using direct link. Now both B and A know that the link connecting them is currently not critical.

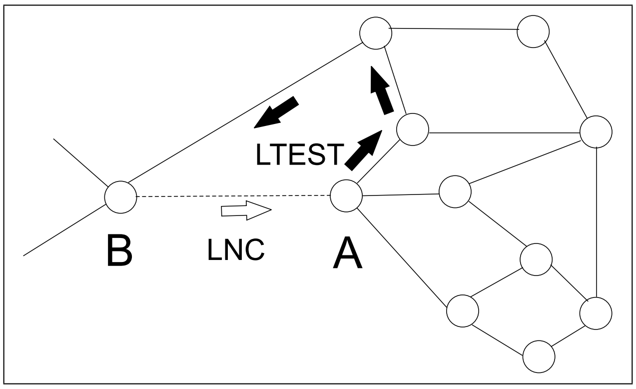

Figure 2. Observed link A-B is not critical

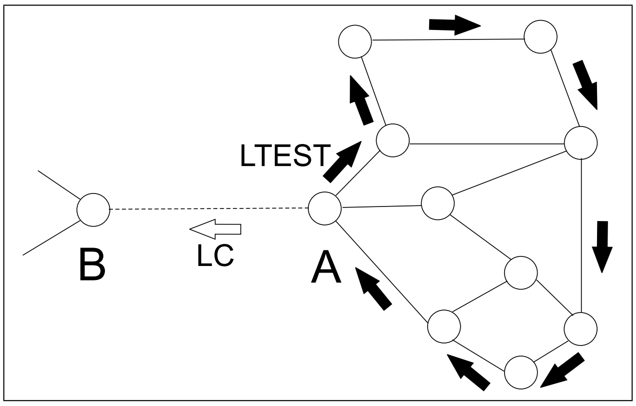

Figure 3. Observed link A-B is critical

Figure 3 presents a case where breaking of the link $A - B$ is leading to partitioning of the network. Again, node A is sending LTEST message using the right hand rule along the inner face. Since there is no alternative path to node B, message will be routed back to node A after traveling along the inner face. Node A then sends LC message to B informing it that the tested link is critical.

The algorithm is detailed on Figure 4. It shows three processes that are activated each time one of the following events is raised: an update from a neighbor node is received, local information such as $\vec{v}$ or $r(t)$ is changed, and a local link is estimated to become critical. The processes are sharing one data structure, a list containing *($t_{critical}$, node-id)* tuples sorted by ascending values of $t_{critical}$. Operation *put($t_{critical}$, node-id)* replaces an element with the same value of *node-id* and sorts the list. Operation *get()* removes the first element of the list and returns a tuple *($t_{critical}$, node-id)*.

```
process: receive
receive update from node id
calculate $t_{critical}$ for node id
put($t_{critical}$, node id)
process: local change
foreach node id from neighbors
  calculate $t_{critical}$ for node id
  put($t_{critical}$, node id)
endfor
process: critical time
($t_{critical}$, node id)=get()
if delegate() begin
  send LTEST
  wait for reply
  if reply = LTEST
     send LC
  endif
else
  wait for reply
  if reply = LTEST
     send LNC
  endif
endif
```

Figure 4. Prediction algorithm

When a node receives an update, process *receive* is activated. It calculates new value for $t_{critical}$ and replaces old entry in the list with the new entry. Process *local change* is activated when behavior of the current node is changed - its velocity $\vec{v}$ or its transmission range $r(t)$. The process recalculates $t_{critical}$ for all neighbor nodes and places new values in the list. Process *critical time* is started at time $t_{critical}$ for which it is estimated that one of the links will have probability $P_{threshold}$ of being broken. Current node removes the leading entry from the list. Then it starts delegation process and in case that it is delegated, it sends message LTEST. If message LTEST is returned to the originating node, it sends LC message to the other node, informing it that the link is critical. In case that current node was not delegated for link testing sequence, it will wait for a message from the other (delegated) node. If it receives LTEST message, it notifies the delegated node that the link is not critical by sending LNC message.

3.3. Correctness

Now we need to show that this approach is correct. For that purpose we will prove two theorems and then explain their implications to our model.

Lemma 1 *If for two nodes A and B, forming an edge in Gabriel graph, there exist alternative paths between them not containing the edge A-B, then at least one path must belong to the bounded face containing A-B.*

Proof:
We will prove this lemma by means of mathematical induction. Let us observe an alternative path connecting nodes A and B through only one node C (Figure 5). It is obvious that the alternative path is forming a bounded triangular face which contains the edge A-B.

We are adding another node so that it is forming new alternative path between nodes A and B. It can be done in three different ways as shown on Figure 5. First way to add a node is independently of the existing face (node D', forming new path $ACD'B$), second is to insert a node on existing path (node D'', altering existing path ACB to $ACD''B$)) and third is to insert it in such manner that it creates a new face (node D''', forming face $AD'''B$). So, after adding a new node, a face containing edge A-B still exists, altered or unaltered.

Let us assume that this assumption is valid for case of n nodes forming one or more alternative paths between nodes A and B. If we add one new node to existing n nodes, the node can be added in three different ways, which do not disturb existence of face containing the edge A-B as it was shown in induction hypothesis. By using mathematical induction, we have proved that this theorem is valid for every n.

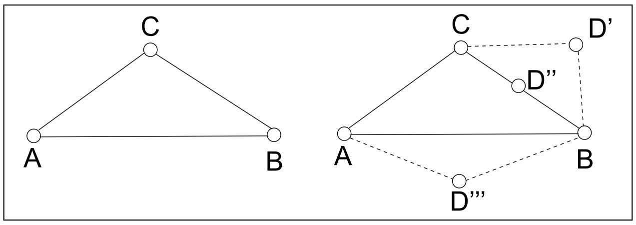

Figure 5. Alternative path forming

Lemma 2 *(Restricted Jordan Curve Theorem) A simple closed polygonal curve C consisting of finitely many segments partitions the plane into exactly two faces, each having C as boundary.* [3]

Lemma 3 *If for two nodes A and B, forming an edge in Gabriel graph, there exist alternative paths between them not containing the edge A-B, then there exist two faces containing edge A-B, one bounded and the other bounded or unbounded.*

Proof:
From Lemma 1, existence of alternative path implies existence of one bounded face to which the edge A-B belongs to. From Lemma 2, existence of one closed face implies existence of another face.

If there is more than one alternative path surrounding the edge A-B from different sides, there are two bounded faces containing the edge A-B. Otherwise, the edge is on outer, unbounded face from one and on the bounded face on the other side.

Theorem 1 *If for two nodes A and B, forming an edge in Gabriel graph, there exist alternative paths between them not containing the edge A-B, then a message can be sent from A to B using the right/left hand rule along the inner side of a face containing A-B, without using edge A-B.*

Proof:
From Lemma 3, existence of alternative path between nodes A and B implies existence of two faces containing the edge A-B. Since removing of an edge inside a face cannot create two partitions, connectivity between nodes A and B will still exist inside the face after the edge A-B is removed. So, it will be possible to send a message from A to B using right/left hand rule through a face containing the edge A-B without using the edge A-B.

Theorem 2 *If for two nodes A and B, forming an edge in Gabriel graph, there exists no alternative path between them, then a message sent from A to B along the inner side of the unbounded face containing A-B, using the left/right hand rule, without using the edge* A-B*, will return to A.*

Proof:
Since edge A-B is a bridge in the observed graph, its removal from the graph will form two partitions, each of them connected, node A belonging to one and node B to the other partition. Both of the nodes are belonging to the unbounded face as a direct consequence of partitioning. If node A sends a message to node B using the left/right hand rule on the inner side of unbounded face surrounding the partition to which the node A belongs, that message will traverse all the nodes on perimeter of partition to which node A belongs and return to node A, since node B belongs to the other partition.

In our algorithm, in order to test a link that is possibly critical, delegated node A is sending a message to the node B on the other side of the link without knowledge of global graph topology. In case that alternative paths between nodes A and B exist, by theorem 1, message from node A will reach node B in finite number of steps. In case that there are no alternative paths between nodes A and B, by theorem 2, the message sent by node A to node B will return to its sender in finite number of steps. This verifies the correctness of our algorithm.

4. Additional Issues and Extensions

We are developing a protocol based on the proposed algorithm and implementing it in ns-2 network simulator [1]. Node density will be varied and different parameters of network will be measured. One of main concerns is to determine the amount of traffic generated by our approach, frequency of regular updates and probability of link breakage $P_{threshold}$ as functions of average node movement speed. Another interesting parameter to observe is the amount of the processor time and energy spent on calculations needed for prediction of local link breakage.

Two issues are left open by the algorithm. The most important one is shown on Figure 6. Let us suppose that links A-B and C-D are declared as locally critical at the same time. Link A-B will be declared as non-critical because of existence of the link C-D and vice versa. However, the network can become partitioned because of the simultaneous breaking of both links at approximately the same time. We are considering this a pathological case because it is highly unlikely to happen in a network where nodes are having random and independent movement.

The second issue is that, because of the specific requirements for the construction of Gabriel graph, system can occasionally generate false alarms. We consider this a feature of the system, and not a problem. More important for our algorithm is to predict the partitioning always when a partitioning is going to happen, and several false alarms can be disregarded. We plan to investigate the frequency of false alarms through simulations.

After we have simulation results, we plan to solve both issues on the protocol layer.

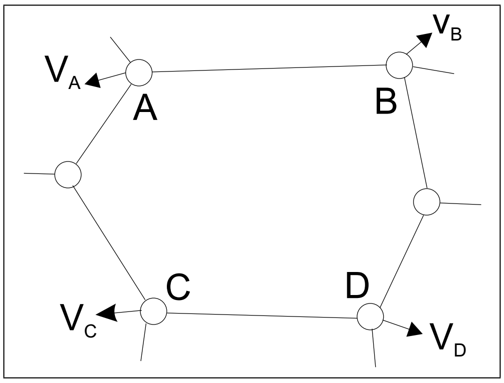

Figure 6. Pathological case of graph partitioning

The proposed algorithm will be extended to support prevention of partitioning in manets, under the assumption that movement of the nodes can be influenced. We are currently investigating several strategies for partition prevention:

- Adding helper nodes to reinforce critical links, acting as mobile routers.
- Assigning priorities to nodes, where nodes of higher priority must maintain connections and movement pattern at the expense of movement of nodes with lover priorities.
- Assigning a task (movement pattern) to the group of nodes, with the objective that at least a specified percent of nodes will always be in range, thus allowing clustering (partitioning) with predefined granularity.

5. Conclusion

The issue of partition prediction in manets is becoming increasingly important, as we develop new scenarios and applications that rely on this paradigm. Standing out among them are rescue operations, where partition of the network can lead to the loss of human lives.

The existing solutions known from the graph theory can be applied to test for partitioning of the quasi-static networks, like the Internet, but are not suitable to fast changing topologies, such as manets. The main problem is that these solutions require global topology knowledge, which is not acceptable in manets. Therefore, we are forced to look for alternative solutions.

We propose a distributed algorithm for predicting network partitioning in location-aware manets. Based on positions and speeds of one-hop neighbors, every node calculates probability that its neighbor links will become broken. Then, using planarity of Gabriel graph, the algorithm checks whether links identified as potentially critical can cause network partition. For this algorithm to work, it is enough for each node to know its one-hop neighbors only since the construction of the Gabriel graph, calculation of link breakage probabilities and critical link testing are all distributed processes that require no global network topology knowledge.

The proposed approach can predict network partitioning but it cannot prevent it. We plan to extend the algorithm with techniques to prevent partitioning once it is predicted by influencing node movement patterns.

References

[1] The network simulator ns-2. *http://www.isi.edu/nsnam/ns/*, 2004.

[2] Goyal D. and Caffery J. Jr. Partitioning avoidance in mobile ad hoc networks using network survivability concepts. In *International Symposium on Computer and Communications (ISCC)*, Sicily, 2002.

[3] West D.B. *Introduction to Graph Theory.* Prentice Hall, 1996.

[4] Chen K., Shah S.H., and Nahrstedt K. Cross-layer design for data accessibility in mobile ad hoc networks. *Wireless Personal Communications 21: pp 49-76*, 2002.

[5] Wang K.H. and Li B. Group mobility and partition prediction in wireless ad-hoc networks. In *Proc. of IEEE International Conference on Communications (ICC 2002)*, New York, 2002.

[6] Barriere L., Fraigniaud P., Narayanan L., and Opatrny J. Robust position-based routing in wireless ad hoc networks with irregular transmission ranges. *Wireless Communications and Mobile Computing 3, pp. 141-153*, 2003.

[7] Hauspie M., Simplot D., and Carle J. Partition detection in ad-hoc networks using multiple disjoint paths set. In *Proc. of the 1st Int. Workshop on Objects Models and Multimedia Technologies (OMMT)*, Geneva, Switzerland, 2003.

[8] Bose P., Morin P., Stojmenovic I., and Urrutia J. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks 7, pp. 609-616*, 2001.

[9] Karp P. and Kung H. Gpsr: greedy perimeter stateless routing for wireless networks. In *6th ACM Conference on Mobile Computing and Networking (Mobicom '00)*, 2000.