Shortcut Detection and Route Repair in Ad hoc Networks

Manel Guerrero Zapata
Technology Department
Universitat Pompeu Fabra
Passeig de Circumval·lació 8, 08003 Barcelona
Email: manel.guerrero@upf.edu

Abstract

When a routing protocol for manet networks (mobile and ad hoc networks) does a route discovery, it does not discover the shortest route but the route through which the route request flood traveled faster. In addition, since nodes are moving, a route that was the shortest one at discovery time might stop being so in quite a short period of time. This causes, not only a much bigger end-to-end delay, but also more collisions and a faster power consumption.

In order to avoid all the performance loss due to these problems, this paper develops a technique to periodically discover shortcuts to the active routes that can be used with any destination vector routing protocol. It also shows how the same mechanism can be used as a bidirectional route recovery mechanism.

1 Introduction

One of the main consequences, if we are using IEEE 802.11b wireless cards, of the so-called "gray zones" defined and studied in [4], is that the shortest path is not always a good path. "Gray zones" are zones in which a node can receive short broadcast messages but not reasonably large data messages from a certain node.

This means that finding a method to discover the shortest path (or a shorter path) is not necessarily a very good idea. Since there is nothing that tells us that this newly discovered route will be better than the original one.

Nevertheless, if we use one of the proposed solutions mentioned in [4] we can be quite sure that the routes will not contain any "gray link". And then, the shortest path will be, with high probability one of the best path (if not the best).

Section 2 describes the mechanism used to perform shortcut detection. Section 3 explains how the same techniques can be applied to perform route repair. Section 4 gives a short overview of AODV needed to understand the following section. Section 5 studies how to use the mechanisms described in this paper in conjunction with AODV as base routing protocol. Finally, section 6 takes a look at related work.

2 Shortcut Detection

In order to detect possible shortcuts in active routes, source and destination of any active route will periodically (with jitterized periods) start a shortcut discovery by issuing a "Shortcut Request" (SREQ). The SREQ will be a routing message that will be broadcast with TimeToLive equal to zero.

Therefore, all the neighbors will receive it (without the need to be in promiscuous mode) but will not re-broadcast it. If any of the nodes that receive the request knows a shortcut it will reply with a "Shortcut Reply" (SREP) that will contain information about that shortcut. The node (or nodes) that will receive a SREQ, are part of the current route and have not generated SREQ yet, will also generate another SREQ.

This will be done until the other end of the bidirectional route receives an SREQ and also originates one. This last SREQ is sent because shortcuts that use a different link to that end of the route would not be discovered otherwise.

One could think that generating SREQs with TimeToLive equal to 1 would make more shortcuts to be discovered. Nevertheless, this would generate much more network traffic in the zone.

2.1 Shortcut Routes

Upon receival of SREQs, nodes will create "shortcut routes" with a shorter live time than the other routes. And, in a very similar manner to how reverse routes work, they will be deleted quite soon if the node does not receive a SREP confirming that there is a shortcut. The difference is that shortcut routes cannot be used until they are not confirmed by the corresponding SREP, since we do not know if, in effect, there is a shortcut nor the resulting distance in hops of using that shortcut.

2.2 Shortcut Requests

Shortcut Requests will contain the both ends of the bidirectional current route with the corresponding distance in hops to them. In the case the routing protocol uses sequence numbers, it will also include the corresponding sequence numbers of the routes to both ends.

2.3 Shortcut Replies

Shortcut Replies might just be normal route reply messages, maybe with a flag that indicates that they are a shortcut reply.

2.4 One-Hop Shortcuts

One-hop shortcuts occur when two non-neighbor nodes that were part of a route discover that they are now neighbors. They are quite easy to discover: When a node receives a SREQ, it can check if the node can be part of a one-hop shortcut by doing the following:

- Verify that the sequence numbers of the route to each of the end points of the bidirectional route are the same in the received SREQ and in its routing table.
- Verify that the hop count of the bidirectional route (sum of the hop counts to each of the end points of the received SREQ or of its own routing tables) is bigger than the hop count of the possible shortcut route (sum of the smallest hop count of the received SREQ with the hop count of the route to the other end point registered in the routing table plus one).

Figure 1 shows an example. There is a bidirectional route from $'S'$ to $'D'$. $'A'$ and $'C'$ move towards each other until they become neighbors. If $'S'$ starts a shortcut discovery, $'A'$ will send a SREQ that $'C'$ will receive. Then, it will see that the current bidirectional route ($1 + 3$ or $3 + 1$) is longer than the possible shortcut ($1 + 1 + 1$) and it will send SREPs to both end points through $'A'$ and $'C'$.

2.5 Two-Hop Shortcuts

Two-hop shortcuts occur when a node is neighbor of two nodes that are part of a route in which they are three or more hops away from each other.

In order to be able to detect two-hop shortcuts, each node needs to keep track of the SREQs that it has received recently (just in the same way that it needs to keep track of

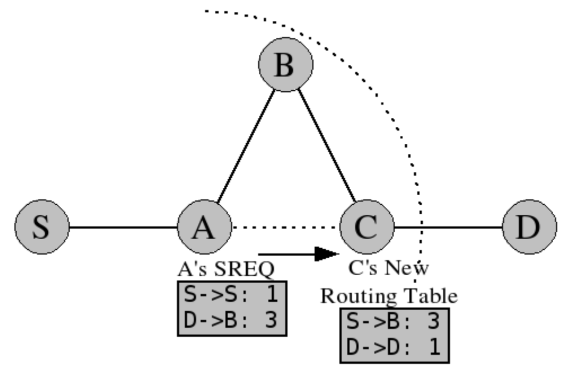

Figure 1. One-hop Shortcut Detection

the rebroadcast route requests to avoid rebroadcasting the same route request more than once).

If a node receives a SREQ for a route which has already received another SREQ, it will check if it is part of a two-hop shortcut route by doing the following:

- Verify that the sequence numbers of the route to each of the end points of the bidirectional route are the same in both SREQs.
- Verify that the hop count of the bidirectional route (sum of the hop counts to each of the end points of any of the two SREQs) is bigger than the hop count of the possible shortcut route (sum of the smallest hop count of one SREQ with the smallest hop count of the other SREQ plus two).

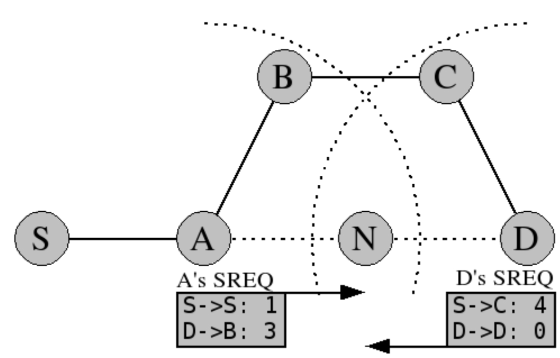

Figure 2. Two-hop Shortcut Detection

Figure 2 shows an example. There is a bidirectional route that goes from $'S'$ to $'D'$. Nodes move, and later on $'N'$ is a new neighbor of $'A'$ and $'D'$. Either $'S'$ or $'D'$ decide that it is time for a new shortcut discovery. This shortcut discovery will be propagated through the route. Therefore, $'N'$ will receive SREQs from both $'A'$ and $'D'$. It will check their contents and it will see that the hop count of the current bidirectional route ($1+3$ or $4+0$) is bigger than the hop count of the shortcut route ($1+0+2$). And it will send SREPs to both end points of the bidirectional route through $'A'$ and $'D'$.

In the case $'N'$ would be neighbor of only $'A'$ and $'C'$, it will see that the current route ($1+3$ or $3+1$) is not longer than the possible shortcut ($1+1+2$).

2.6 Other Kind of Shortcuts

Figure 3 shows an example of a three-hop shortcut that cannot be shortcut in any way by the method presented in this paper. This kind of situation will happen seldomly in networks with certain density of nodes.

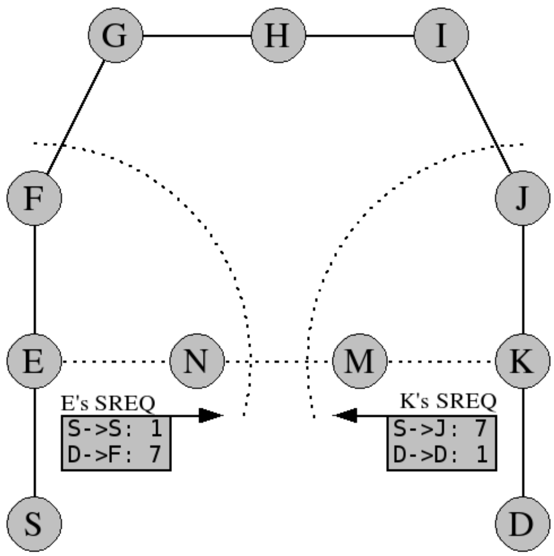

Figure 3. Three-hop Shortcut

However, in a network with certain density of nodes it is highly probable that routes that are not the shortest ones can be shortcut by one-hop and two-hop shortcuts.

A method could be designed to also detect three-hop shortcuts, but it would imply too much traffic. Basically, it could consist in that receivers of the shortcut requests from the nodes that are in the current route would also send the request to its neighbors.

3 Route Repair

Some routing protocols in manet networks have a mechanism to try to repair a broken route (due to a link breakage) that does not imply a complete route discovery. An example would be the "local repair" in AODV [6] in which when a link used to send data packets breaks, the node upstream of the link that got broken may (if it was close to the destination) do a route discovery of the destination broadcasting the route request with a TimeToLive that is assumed to be enough to reach the destination.

This method has the problem that it only repairs the route in one direction. Chances are, that the route is used in both directions. Therefore, if it only repairs the route in one direction, another route discovery will be needed to repair the route in the other direction.

A possible solution, would be to use the shortcut discovery method described in this paper to do the route repair. To do so, when a link breakage occurs, the two nodes that were connected through that link will initiate a "repaired route discovery". This repaired route discovery will consist of sending a SREQ to the end of the route to which they are still connected. The differences with a normal SREQ message will be:

- The message will be flagged as repair route SREQ.
- The hop count to the endpoint that is not available anymore will be set to infinity (typically indicated by the value 255).
- Optionally, the original SREQ (the one originated by one of the two nodes that were connected through that link) might be also forwarded by all their immediate neighbors that were not part of the original route. Of course, if they forward it, the forwarded SREQ should have increased the hop count that is not set to infinity in the SREQ (to account for the new hop that has been done).

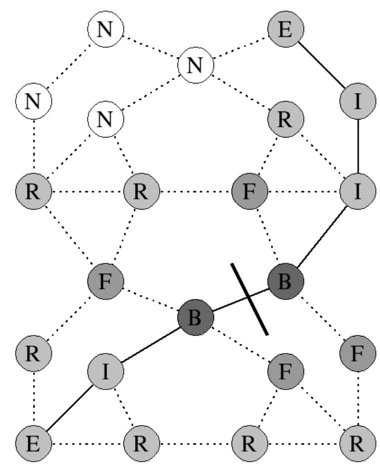

Figure 4. Propagation of SREQs

Figure 4 shows how SREQs are propagated. End points of the previous route are marked as $'E'$. The two nodes that where connected through the link that has just broken are mark as $'B'$ and intermediate nodes that where part of the route as $'I'$. The neighbors of the $'B'$ nodes that are not part of the route but will forward the SREQ are marked as $'F'$ nodes. The rest of the nodes that will receive a SREQ are marked as $'R'$ nodes. Finally, the other nodes are marked as $'N'$.

Due to the fact that the neighbors of the $'B'$ nodes (the $'F'$ nodes) forward the SREQs, there will be a broader diffusion of the SREQs in the zone nearby the link breakage.

Figure 5 shows how routes get repaired. When $'N'$ receives both SREQs and updates its routing table, it will send both SREPs to $'S'$ and $'D'$ and the route will be repaired. The routing table of $'N'$ is updated in the following manner: Since it receives two SREQs for the route between $'S'$ and $'D'$, one of them with a finite hop count to $'S'$ and the other with a finite hop count to $'D'$, it can deduce that it can do a shortcut route between $'S'$ and $'D'$ incrementing the finite hopcounts that were in the SREQs by one (to account for the last hop the messages did to arrive to $'N'$).

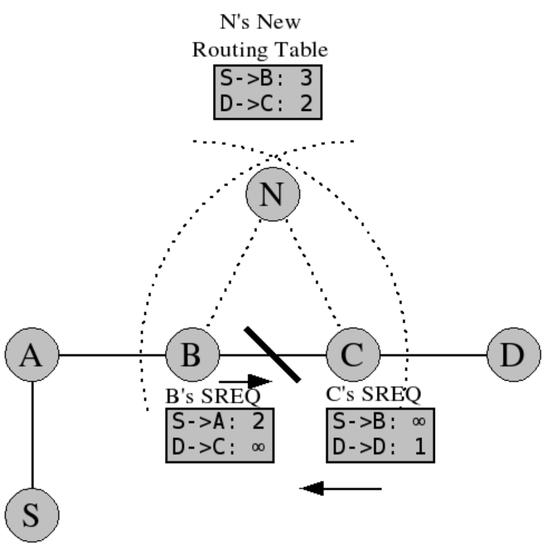

Figure 5. Simple Repaired Route Discovery

Figure 6 shows a more complicated example, in which the neighbors of the nodes that were connected through the link that broke will also forward the SREQ. This increases the chances of getting the route repaired, but implies a little bit more traffic.

4 AODV

This section gives an introduction to AODV, necessary to understand how the techniques explained in this paper can be applied to it.

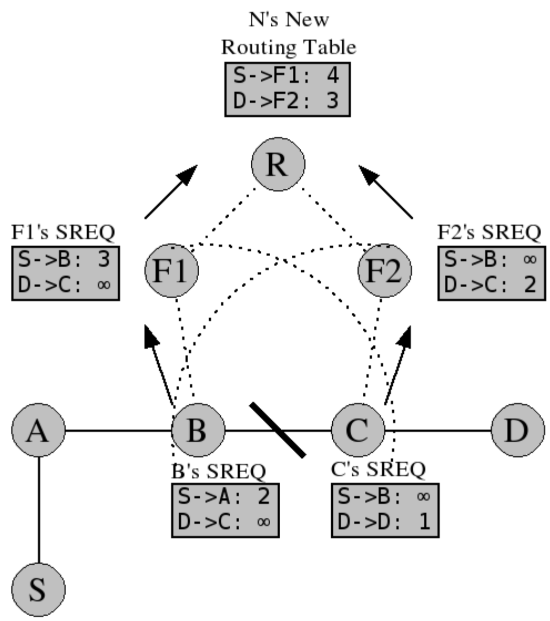

Figure 6. Repaired Route Discovery with Additional Forwarding by neighbors

Ad Hoc On-Demand Vector Routing (AODV) protocol [6] is a reactive routing protocol for ad hoc and mobile networks that maintains routes only between nodes which need to communicate. The routing messages do not contain information about the whole route path. Instead, they only contain information about the source and destination nodes. Therefore, routing messages do not have an increasing size. It uses destination sequence numbers to specify how fresh a route is (in relation to another), which is used to grant loop freedom.

Whenever a node needs to send a packet to a destination for which it has no ‘fresh enough’ route (i.e., a valid route entry for the destination whose associated sequence number is at least as great as the ones contained in any RREQ that the node has received for that destination) it broadcasts a route request (RREQ) message to its neighbors. Each node that receives the broadcast sets up a reverse route towards the originator of the RREQ (unless it has a ‘fresher’ one). When the intended destination (or an intermediate node that has a ‘fresh enough’ route to the destination) receives the RREQ, it replies by sending a Route Reply (RREP). It is important to note that the only mutable information in a RREQ and in a RREP is the hop count (which is being monotonically increased at each hop). The RREP travels back to the originator of the RREQ (this time as a unicast). At each intermediate node, a route to the destination is set (again, unless the node has a ‘fresher’ route than the one specified in the RREP). In the case that the RREQ is replied to by an intermediate node (and if the RREQ had set this option), the intermediate node also sends a RREP to the destination. In this way, it can be granted that the route path is being set up bidirectionally. In the case that a node receives a new route (by a RREQ or by a RREP) and the node already has a route 'as fresh' as the received one, the shortest one will be updated.

In addition to these routing messages, Route Error (RERR) messages are used to notify the other nodes that certain nodes are not anymore reachable due to a link breakage.

5 AODV-SDR

AODV-SDR (AODV with shortcut discovery and route repair) incorporates two new types of messages: Shortcut REQuest (SREQ) and Shortcut REPly (SREP). The format of these messages is shown in figures 7 and 8, and its fields are specified in the tables1 and 2.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |R|  Reserved   |  Hop Count 1  |  Hop Count 2  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            SREQ ID                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    End Point 1 IP Address                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     End Point 1 Next Hop                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  End Point 1 Sequence Number                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    End Point 2 IP Address                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     End Point 2 Next Hop                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  End Point 2 Sequence Number                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7. Shortcut Request (SREQ) Message Format

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |R|A|S|   Reserved      |Prefix Sz|   Hop Count   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Destination IP address                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Destination Sequence Number                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Originator IP address                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Lifetime                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 8. Shortcut Reply (RREP) Message Format

SREQs have a "R flag" that is set if the SREQ is used to do a route repair. They also contain a "SREQ ID", that is a sequence number that identifies uniquely the SREQ with

Table 1. Shortcut Request Message Fields

Field	Value
Type	64
R flag	The flag that indicates that this is a repair route SREQ.
Reserved	Sent as 0; ignored on reception.
Hop Count 1	The hop count to the end point 1.
Hop Count 2	The hop count to the end point 2.
SREQ ID	A sequence number uniquely identifying the SREQ with then end point 1.
End Point 1 IP Address	The IP address of the end point 1.
End Point 1 Next Hop	The next hop in the route to the end point 1.
End Point 1 Sequence Number	The sequence number of the route to the end point 1.
End Point 2 IP Address	The IP address of the end point 2.
End Point 2 Next Hop	The next hop in the route to the end point 2.
End Point 2 Sequence Number	The sequence number of the route to the end point 2.

Table 2. Shortcut Reply Message (RREP with the 'S' flag) Fields

Field	Value
Type	2
R flag	Repair flag. It is used for multicast.
A flag	Acknowledgment required flag.
S flag	The flag that indicates that is a shortcut reply.
Reserved	Sent as 0; ignored on reception.
Prefix Size	Specifies the prefix size of the address.
Hop Count	The hop count from the originator to the destination.
Destination IP Address	The IP address of the destination.
Destination Sequence Number	The destination sequence number associated to this route.
Originator IP Address	The IP address of the node for which the route is supplied.
Lifetime	The lifetime in milliseconds of the route.

the end point that originated the SREQ. In case this SREQ was originated due to a route repair both nodes that where connected through the link that broke will generate SREQs that will probably have different sequence numbers.

SREQs also contain the following information about both end points of the route: IP address , the next hop of the route that goes to the end point, the sequence number of that route, and the hop count to the end point.

SREPs are basically AODV's "Route Reply" (RREP) messages with a flag set to indicate that they are SREPs. Once the shortcut is discovered they propagate back the shortcut route. Therefore they contain all the information about that route: hop count, IP address, lifetime, etc.

6 Related Work

In discussions in the manet mailing list was argued that distance vector routing protocols where not discovering the shortest route, but the one through which the route requests where broadcasted faster.

"Coping with Communication Gray Zones in IEEE 802.11b based Ad hoc Networks" by Lundgren et al. [4] (published in 2002) was a paper that came out as a result of physical experimentation with real ad hoc networks. In the paper they find out different solutions to cope with the "gray zones" (zones through which short broadcast packets are received but not data packets).

Therefore, it was clear that, on one hand, distance vector routing protocols did not discover the shortest path and that, on the other hand, the shortest path was not always the best path (not if one of the links is a "gray link").

In 2003, some publications tried to address this problem. Like SHORT [2], that also tries to find shortcuts, but in order to do that all data packets must carry certain information (like a "hop count"). Which is a very strong requirement that would, arguably, render it unfeasible for a lot of scenarios. It is always tempting to add control information in data packets. It simplifies the problem you are trying to solve, but it complicates extremely the problems in the rest of the system.

In addition, SHORT fails to find shortcuts in a very simple scenarios (when the shortcut involves the source node, when the short cut involves the destination node, ...). Finally, it also fails to look at the routes as bidirectional routes.

A paper by De Couto et al. [1] defines a metric to measure the throughput of a multi-hop route to be used with DSDV [5] and DSR [3]. It expects that nodes will calculate the throughput using dedicated link probe packets. But, it is not clear how a node collects information of the links that are far away from him to do all the statistics.

There are other papers, that try to address route efficiency by finding minimum energy disjoint paths, like this one by Srinivas and Modiano [7].

7 Conclusion

This paper provides a technique that can be used with any destination vector routing protocol in a manet network to periodically discover shortcuts to the active routes. Therefore, making the network communications much more optimal. In addition, it also shows how the same mechanism can be used as a bidirectional route recovery mechanism. Finally, it specifies how to implement this techniques on top of the AODV routing protocol.

Acknowledgment

This work was supported in part by CYCIT TIC2003-09279-C02-01 and by the I2CAT Foundation.

The author wants to thank his new colleagues at the Universitat Pompeu Fabra for their support, friendship and help (specially with all those little things that might end up burying you into a red tape mountain).

References

[1] Douglas S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 134 – 146, September 2003.

[2] C. Gui and P. Mohapatra. SHORT: self-healing and optimizing routing techniques for mobile ad hoc networks. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking and computing*, pages 279 – 290, June 2003.

[3] D. B. Johnson, D. A. Maltz, and Y.-C. Hu. The dynamic source routing protocol for mobile ad hoc networks (DSR). INTERNET DRAFT, MANET working group, Apr. 2003. draft-ietf-manet-dsr-09.txt.

[4] H. Lundgren, E. Nordstrm, and C. Tschudin. Coping with communication gray zones in IEEE 802.11b based ad hoc networks. In *Proceedings of the 5th ACM international workshop on Wireless mobile multimedia*, pages 49 – 55, September 2002.

[5] C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, pages 234–244, 1994.

[6] C. E. Perkins, E. M. Belding-Royer, and S. R. Das. Ad hoc on-demand distance vector (AODV) routing. Internet Request for Comments RFC 3561, Nov. 2003.

[7] A. Srinivas and E. Modiano. Minimum energy disjoint path routing in wireless ad-hoc networks. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 122 – 133, September 2003.