JAIST Repository
https://dspace.jaist.ac.jp/

Title	A lazy routing protocol for large-scale mobile ad hoc networks
Author(s)	Cartigny, Julien; Defago, Xavier
Citation	Research report (School of Information Science, Japan Advanced Institute of Science and Technology), IS-RR-2005-008: 1-10
Issue Date	2005-04-05
Type	Technical Report
Text version	publisher
URL	http://hdl.handle.net/10119/4788
Rights	
Description	リサーチレポート（北陸先端科学技術大学院大学情報科学研究科）

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

A Lazy Routing Protocol for Large-Scale Mobile Ad Hoc Networks

Julien Cartigny[1], Xavier Défago[1,2]

[1]*School of Information Science, Japan Advanced Institute of Science and Technology (JAIST)*
[2]*PRESTO, Japan Science and Technology Agency (JST)*

April 5, 2005

IS-RR-2005-008

Japan Advanced Institute of Science and Technology (JAIST)

School of Information Science
1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan

http://www.jaist.ac.jp/

ISSN 0918-7553

A Lazy Routing Protocol for Large-Scale Mobile Ad Hoc Networks

Julien Cartigny* (contact author) Xavier Défago*,†

*School of Information Science
Japan Advanced Institute of Science and Technology (JAIST)
1-1 Asahidai, Nomi-shi, Ishikawa-ken 923-1292, Japan
†PRESTO, Japan Science and Technology Agency (JST)

Tel.: +81-761-51 1254 - Fax.: +81-761-51 1149
E-mail: {cartigny,defago}@jaist.ac.jp

Abstract

1. Introduction

An ad-hoc network consists of mobile hosts with a wireless radio interface. Because of the radio nature of the communication, a node can directly communicate with its neighbors (a message is received simultaneously by all neighbors within the radio transmission radius). To reach farther nodes, other nodes act as relays (which forward the message to the addressee). So, ad hoc networks are totally decentralized (without the aid of any centralized infrastructures) and form a temporary network only supported by the users.

Routing protocols have been shown to work well with a limited number of nodes. But scalability requires new methods, as bandwith overhead and MAC access are too high if many nodes are involved in the ad hoc network. For instance, the reactive routing protocol AODV [8] broadcasts a route request message to the whole network, even if the destination is close. An other example is the proactive routing protocol DSDV [7] which also has a high routing overhead, because of its regular diffusion to the neighbors of the whole routing table. More recent protocols like OLSR [3] try to reduce the bandwidth consumption, using information from local topology at two hops to reduce the number of rebroadcasts. But this protocol also cope with a high routing overhead when the the density and/or size of the network increase.

All theses protocols have a major drawback: they have been designed to have the most accurate information possible about the network topology. In this paper, we propose in this paper new ideas to cope with routing overhead in the context of very large ad hoc networks. The goal is to reduce the number of messages during network discovery (using only the small ammount of information disseminated in HELLO messages) and reducing the cost of the route research broadcasts (which do not cover all the network but only nodes between the source and the destination).

Hence, LRP (Lazy Routing Protocol) is an hybrid protocol (both reactive and proactive) based on the following properties:

Avoid redundancy between neighbours: If nodes are close, it is more interesting that they do not diffuse the same routing information, to maximize the routing information received by the neighbors. In LRP, nodes listen to the neighbouring communications, to select which route information is pertinent to diffuse.

Limit the size of the route research: During route research, it is also interesting to have several routes to the destination (for more robust communication). But in the same time, it is better to not cover the whole network when looking for a route. LRP, using proactive information disseminated in HELLO messages, can limit the route research broadcast to a geographical zone between the source and the destination.

Because the protocol has a relaxed approach about the knowledge of the network, LRP is more adapted for disconnected applications (which are aware that the knowledge of the network is limited, so communication can be delayed until a route is discovered). The protocol offers good reachability for a minimal number of routing and discovery messages.

The major contribution is the use of a non-redundant broadcast-limited approach for the diffusion of information. We show that accurate information about the topology is not necessary, as LRP uses simple and shorter HELLO messages to reduce the cost of route research. The algorithm LRP is a hybrid routing algorithm composed of two parts:

HELLO Announcements: Each node adds to each HELLO message a subset of its routing table (called "lazy table" in this paper). The information to be added in the packet depends on various policies.

Route Discovery: When a route request is sent, each node rebroadcasts the packet if the the node is closer to the source. The protocol exploits the information diffused by the HELLO messages to limit the size of the broadcast in the network.

The paper is organized as follows. A partial review of previous work on routing protocols is presented in Section 2. Section 3 is an overview of our lazy routing protocol. Simulation results are presented in Section 4. Finally, Section 5 concludes the paper.

2. Previous Works

Several routing protocols exist for ad hoc networks. Among them, various approaches are used to manage the route discovery. The proactive approach uses frequent messages to construct route information. An example of such a protocol is DSDV, where all the nodes regularly exchange their routing tables to construct accurate route information in the same manner as Internet routers do[1]. So, route research is fast but the protocol overhead is important and constant (especially in dense networks, as complexity grows in $\Theta(n^2)$).

The reactive approach is the opposite: broadcast route request messages when a route is needed, giving high bandwidth overhead when simultaneous route requests occur. For instance, using AODV, a source node floods a request message in the whole network (every node rebroadcasts the message one time) to find a route to the destination node. Such event can paralyze the whole network, because of the

[1]The DSDV doesn't really send the whole routing table regularly. Usually, it sends frequent updates of its routing table and less often a full dump of the routing table. But in the context of a large number of highly mobiles nodes the data sent for route discovery grows linearly with the number of nodes.

important number of accesses to the MAC layer. This situation is called a "broadcast tempest" and is described by [6].

There also exist also hybrid approaches using a mix of both behaviors (offering better flexibility, depending of the network topology). For instance ZRP [5] uses a proactive approach for local neighbouring (called zones, and limited to a certain number of nodes) and a reactive protocol for route research between such "zones". Because of the dual nature of this approach, hybrid protocols offer the flexibility needed for scalability. In fact, the protocol presented in this paper is a hybrid protocol.

Of course, the routing overhead is different depending of the situation. For AODV, bandwidth consumption is high during a route discovery, as the route request is broadcasted to the full network using a flooding algorithm. For DSDV, the higher cost is related to the constant updating of the route topology, but there is a low overhead for a route discovery (depending of the number of nodes on the route between the source and the destination).

Nevertheless, the high cost of complexity (approximately $\Theta(n^2)$ for DSDV and $\Theta(n * r)$ for AODV, with r the number of route request) is not adapted for the context of scalability in large ad hoc networks.

OLSR is a proactive algorithm designed to reduce the routing overhead. Each node regularly sends a list of all its neighbors in HELLO messages. Regularly, each node broadcasts to the whole network its identity (with several of its neighbors, known as MPR selectors) using TC messages in a proactive way to help other nodes to update their routing tables. For this operation, OLSR uses a broadcast reduction algorithm called MPR (Multi-point relay algorithm), which selects the minimal number of one-hop neighbors to cover every two-hop neighbors. These selected one-hop neighbours (known as a MPR set) are the only neighbours which forward the messages (acting as relays).

The complexity for route announcement is lower than in the previous algorithms, because the number of rebroadcasts from one-hop nodes to cover two-hop nodes are limited by the heuristic algorithm (as the average number of rebroadcasts to cover a bounded space stays stable when increasing the number nodes inside). While this method is efficient for high density, it is less efficient when the diameter of the network becomes important.

3. Lazy Routing Protocol

The Lazy Routing Protocol (or LRP) is a hybrid protocol composed of two parts: a proactive network discovery algorithm and a reactive route research algorithm. LRP does not use the topology approach (like DSDV, AODV, OLSR). Instead, the proactive network discovery regularly selects a set of identifiers from its lazy table and broadcasts it to the neighbours following a policy given by the function *selection*(). When a route is needed, the reactive route research algorithm broadcasts a request packet. The packet is forwarded only by nodes closer to the destination (using information from the proactive network discovery algorithm). Hence, the broadcast is limited to a subset of the network (usually nodes located geographically between the source and the destination) and the number of nodes involved is linear to the distance between the source and the destination. Furthemore, the reactive nature of the algorithm offers better handling of high mobility.

3.1. Proactive Network Discovery

Each node i has a lazy table (called *lazy*). An entry $lazy[j]$ contains $lazy[j].id$ the identifier about the node j, $lazy[j].sn$ the last sequence number seen from j and $lazy[j].hop$ the estimated number of hop between i and j. The node also has a sequence number (called *sequence_number*) initialized to zero and increases for each HELLO message sent. The role of the sequence number is to help other nodes to find the most recent information.

A node i regularly sends (using an interval *hello_interval* between each HELLO messages) a *hello* message containing its identifier. Hence, each node is able to know its one-hop neighbours. More precisely, if $hello[l]$ with $l = 0, 1, 2...size\_hello$ are the fields available in a HELLO message, then $hello[0].id$ contains the sender identifier, $hello[0].sn$ contains the sequence number of the node and $hello[0].hop$ is set to zero.

Furthemore, in the payload of each HELLO message, the node adds a subset of its lazy table $lazy[j]$ (*i.e.* $hello[1...size\_hello]$ contains a copy of some entries from $lazy[j]$). The choice of which entries are to be inserted depends of the selection function *selection*(). Hence, the field selection process is an important parameter. This function is responsible for good diffusion of the knowledge of the network, and the helpfulness of this diffusion for route construction.

In this paper, we have implemented the function *selection*() with an OLDER policy: selects the entries in *lazy* which have been heard from HELLO message sent by the neighbours least recently. Thus, we insure that every entry in the lazy table will be diffused in a bounded time and that the information in the network will be regularly updated. This function is clearly non-optimal, but still presents interesting results.

When a node j receives a HELLO message, it updates the entries in its lazy table and discards the message. For each field $hello[l]$ with $l \in [1...size\_hello]$ in the message, the update is done as following:

- If $hello[l].id \notin lazy$ (*i.e.* $\forall x, lazy[x].id \neq hello[l].id$), a new entry is created with a copy of the field $hello[l]$, and the variable *hop* os this new entry is incremented.
- If there exists an entry $lazy[k]$ with the same identifier (*i.e.* $lazy[k].id = hello[l].id$), then the entry is updated if:
  - the sequence number of the message field is more recent than the one in the lazy table entry (*i.e.* $lazy[k].sn < hello[l].sn$);
  - or if the sequence number of the message field is equal to the one in the lazy table entry (*i.e.* $lazy[k].sn = hello[l].sn$) AND the number of hops in the message field is lower than the number of hops in the lazy table entry (*i.e.* $lazy[k].hop > hello[l].hop$).

The update operation consists of copying the sequence number and the hop distance from the field to the entry: $lazy[k].sn$ is set to $hello[l].sn$ and $lazy[k].hop$ is set to $hello[l].hop + 1$ (as the distance has increase by one hop).

LRP sends only a subset of its lazy table because when two closest nodes broadcast to their neighbours in formations about their lazy table, it is more interesting to avoid redundancy of data between them (because redundancy creates a high overhead, especially for dense networks). Minimizing redundancy has the advantage of reducing the cost of the routing protocol. Because neighboring nodes are close, they have a greater chance of receiving the same information about a node far away in the network, so it is efficient that only one node diffuses the information about farther nodes.

Avoiding redundancy has two major drawbacks: the information about the network is incomplete and inconsistent, and the routes found are not necessary optimal. But the task of the proactive network discovery algorithm is not to find shortest and available routes. In a sense, the proactive diffusion of information is here to help the reactive route discovery algorithms to find the "direction" of the destination node and to limit the size of the broadcast to avoid a full coverage of the network by the broadcasted route request messages.

An other important parameter here is *size_hello*, the number of fields in each HELLO message. It represents the number of fields inserted in each hello message. This parameter depends on the *selection*() function, but also on the size of the network, the density, the mobility of each node, the frequency of

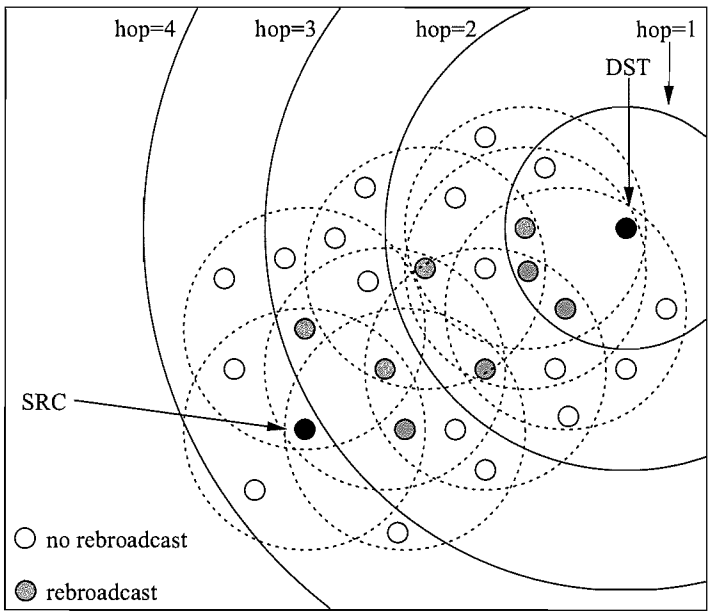

Figure 1. Example of route discovery between src and dst

HELLO messages and some characteristics of the MAC layer (for instance, the maximal size of a HELLO message). The tradeoff of this parameter is to give a good diffusion in the network to avoid failure during route research while keeping a low overhead and so reducing MAC access and bandwith overhead over the network.

Hence, information about the topology of the network is diffused using HELLO messages in an economic way for the bandwidth. This information is not accurate (as good as other routing protocol) so the route construction algorithm uses limited broadcasts to establish a path between the source and the destination.

3.2. Reactive Route Construction

When a node src needs to construct a route to a node dst, it sends a REQUEST message (called $request$) to its neighbors. The message includes a copy of the entry $lazy[dst]$: the identifier ($lazy[dst].id$), the last sequence number known ($lazy[dst].sn$) and the estimated number of hops between the src and the dst ($lazy[dst].hop$). The estimated number of hops is just an indication, not precise and accurate information. It is added in the REQUEST message because it helps each node decide if it has to rebroadcast it.

When a node receives the request message, it rebroadcasts it if an entry with the same identifier exists in its lazy table ($\exists k$ tq $lazy[k].id = request.id$) and if the number of hops of the entry is inferior to the number of the message ($lazy[k].hop < request.hop$). The rebroadcast message is a copy of the one received, except that the number of hops is updated to the hop distance to the entry from the lazy table ($request.hop$ is set to $lazy[k].hop$). Hence, the broadcast message is not sent to every node, but diffused only by the nodes closest to dst. Fig. 1 shows an example of route research. The large circles (shown as hop=1, hop=2...) represent the distance in hops from the dst diffused in the HELLO message. The route research broadcast between src and dst is relayed only by nodes inside a limited geographical zone between src and dst. In this zone, not all the nodes forward the route request message, because

Number of nodes (NN)	50, 100, 150, 200
Size of the simulation	1000mx1000m
Number of route request	NN
Simulation Time	250s
Radius	210m
Speed of nodes	1.5 m/s

Figure 2. General parameters

size_hello	8
HELLO interval	2s
Lazy Information Hold Time	3 * HELLO interval

Figure 3. Parameters for LRP

nodes rebroadcast the message if and only if the distance to the *dst* is inferior to the previous. So nodes receiving route request messages from nodes situated at the same hop distance will not rebroadcast the message.

The route discovery protocol uses the ability of the network to maintain persistant but not accurate information about the topology network. Of course, the size of the limited broadcast depends o nthe distance between two nodes. Furthermore, it is possible to discover several routes between the source and the destination. Even if the broadcast is limited to a geographical zone, all the nodes will repeat the message, so the destination will receive several route request messages and can choose multiple ways to reach the source. In this paper, the protocol chooses only the shortest path.

4. Experimental Results

We use the discrete-event simulator OMNET++ [9] with the mobility framework [4]. We compare LRP with the OLSR protocol because we want to compare our lazy-based approach compared to the full topology knowledge approach used by OLSR.

Each node moves with the random waypoint model, used in most ad hoc networks simulations. The algorithm is simple: each node randomly chooses a new destination in the area and moves to this point using a constant speed. The process is repeated until the end of the simulation. We present in Table. 2 the common parameters for the simulation, in Table. 3 the LRP parameters, and in Table. 4 the OLSR parameters. About the number of route requests, each one randomly chooses another node in its lazy table and sends a route request to it. Hence, the simulation tries to create a number of routes equal to the number of nodes in the simulation. The Lazy Information Hold Time is the delay of valiging an entry in the lazy table. For OLSR, the neighbour hold time and topology information hold time is the validity delay of an entry in, respectively, the neighbour and topology information tables.

Fig. 5 presents the reachability of the two protocols (*i.e.* the percentage of route success). LRP clearly has better results than OLSR, but neither protocol succeeds in reaching 100%. For LRP, the reason is the choice of the policy and *size_hello*. First, the policy OLDER does not follow the principle described in DREAM [1]: "the farther away a node, the lower the needed information". So, the OLDER policy ensures that the whole network is covered, but it considers close and far nodes at the same level. Second, the parameter *size_hello* is fixed for every experimental configuration, and we can see that the reachability is better as density increases. This is because each node receives in one second $size\_hello \times local\_density$, and so receives more information when density is high.

HELLO interval	2
TC interval	5
Neighbour Hold Time	3 * HELLO Interval
Topology Information Hold Time	3 * TC interval

Figure 4. Parameters for OLSR

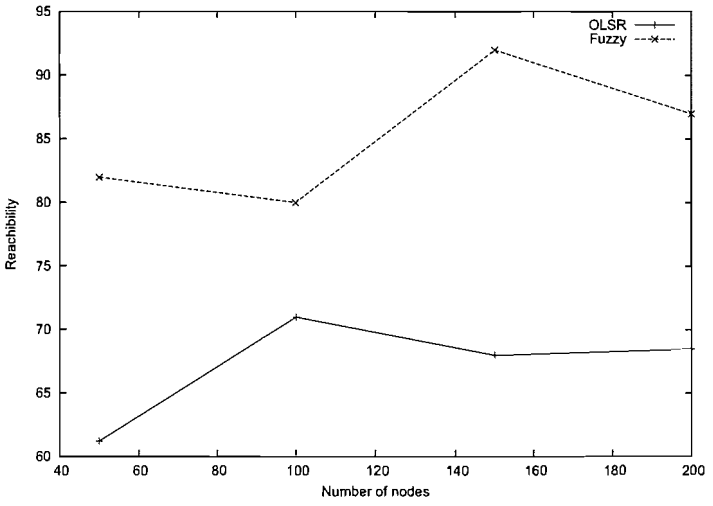

Figure 5. Reachability

OLSR doesn't cope well with mobility for two reasons. First, MPR, the broadcast reduction algorithm used during diffusion of TC message, can become inconsistent when a MPR node leaves. When a node a as selected a node b as MPR, and that B has leaved the communication radius of A, then A does not know the departure before the expiration of the TC information hold time. Furthermore, A has to exchange several HELLO messages with its neighbours before the MPR becomes valid again. Second, MPR is a heuristic algorithm which selects the shortest subset of one-hop neighbours which can cover all the two-hop neighbours. Using such a selection, the probability that the MPR nodes are close to the border of the communication zone is high, so disconnection and/or change of topology can happen quickly. The authors have proposed Fast-OLSR [2] which uses frequent and slighter Fast-Hello messages for nodes with high mobility. But such scheme works only if only a minor set of the networks has high mobility.

Fig. 6 presents the number of routing messages. The gap between OLSR and LRP is because OLSR tries to keep the most accurate information about the topology of the network. LRP instead uses information disseminated with the proactive network discovery, and so can reduce the size of the broadcast. The gap is also present in Fig. 8, which presents the quantity of routing data sent in the network (in octets). OLSR needs to send in each of its HELLO message the complete list of its neighbors, and TC messages are regularly sent inside the network. Even with the MPR algorithm, the quantity of data is important, compared to LRP which uses only HELLO messages and size-limited broadcast.

But the behavior of LRP can also be explained by the nature of the random route distribution. Fig. 7 presents the distribution of length of successful route (the bar) with the average size of the broadcast (the line). Routes between close nodes have more chance to appear, and such routing does not require an important number of relays for the route request broadcast. This behavior is due to the propriety of

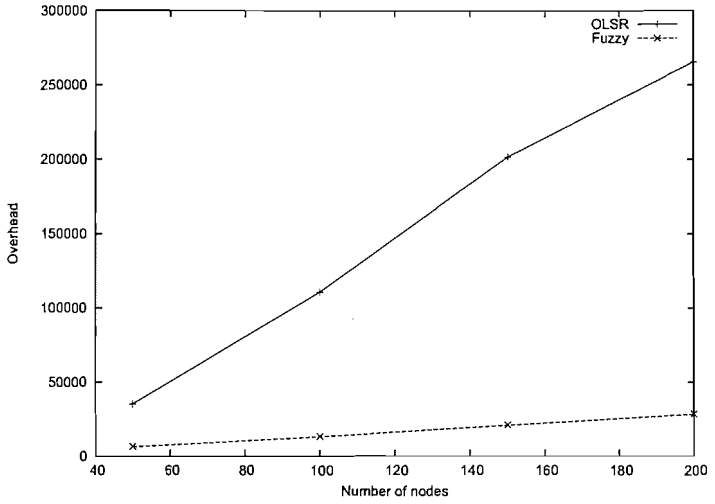

Figure 6. Routing message overhead

the random selection of route in a defined graph.

5. Conclusion

The LRP has been develop for low routing overhead in the case of dense networks. In fact, due to the non-redundancy of HELLO packets and the size-limited broadcast, it is a viable solution in the case of dense networks with a low number of route requests. Because of the size-limited broadcast, it is efficient when route requests between closes nodes are important. Furthermore, the computation cost for each node is very low (only indexed requests in the lazy table and conditional statements) compared to OLSR (using an heuristic algorithm with a complexity increasing as a function of the density). In this paper, we have present a basic version of the protocol. More work is needed about the choice of the policy, the *size_hello* parameter. Nevertheless, the results are promising and can offer a new way to deal with dense and large networks.

References

[1] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. Woodward. A distance routing effect algorithm for mobility (dream). In *Fourth ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom'98)*, pages 76–84, Dallas, Texas, October 1998.

[2] M. Benzaid, P. Minet, and K. Al Agha. Integrating fast mobility in the olsr routing protocol. In *IEEE MWCN*, Stockholm, Sweden, September 2002.

[3] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, A. Qayyum, and L. Viennot. Optimized link state routing protocol. In *Proc. IEEE INMIC*, Pakistan, 2001.

[4] W. Drytkiewicz, S. Sroka, V. Handziski, A. Koepke, and H. Karl. A mobility framework for omnet++. In *3rd International OMNeT++ Workshop*, 2003 January.

[5] Z.J. Haas. The routing algorithm for the reconfigurable wireless networks. In *ICUPC'97*, San Diego, CA, October 1997.

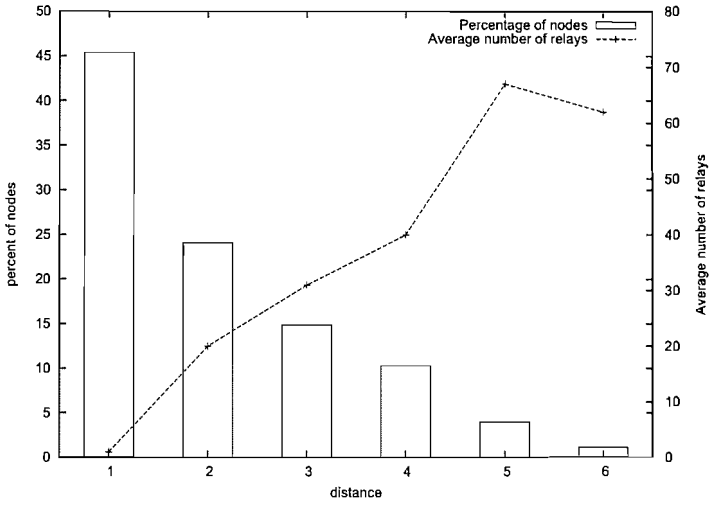

Figure 7.

[6] W. Peng and X.C. Lu. On the reduction of broadcast redundancy in mobile ad hoc networks. In *Proc. Annual Workshop on Mobile and Ad Hoc Networking and Computing (MobiHOC 2000)*, pages 129–130, Boston, Massachusetts, USA, August 2000.

[7] C.E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance vector (DSDV) for mobile computers. *ACM SIGCOMM '94 Computer Communications Review*, 24(4):234–244, October 1994.

[8] C.E. Perkins and E.M. Royer. Ad-hoc on-demand distance vector routing. In *Proc. Second Annual IEEE Workshop on Mobile Computing Systems and Apllications*, pages 90–100, February 1999.

[9] A. Varga. The omnet++ discrete event simulation system. In *Proc. of ESM*, Prague, Czech Republic, June 2001.

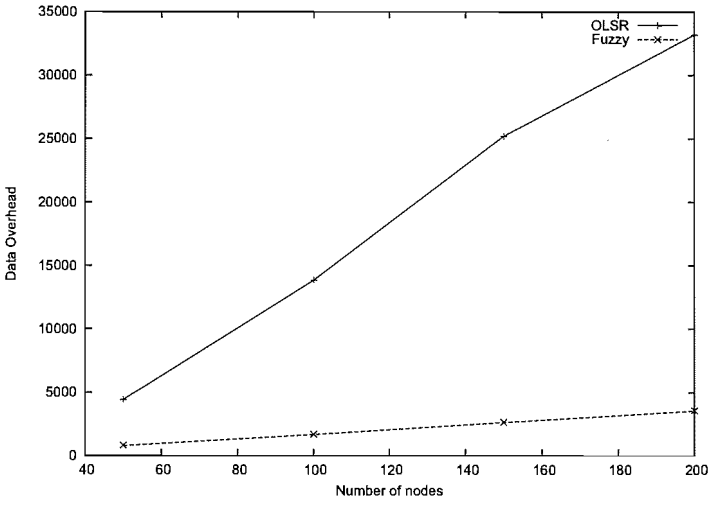
35000
30000
25000
20000
15000
10000
5000
0
40
60
80
100
120
140
160
180
200
Data Overhead
Number of nodes
OLSR
Fuzzy

Figure 8.