CS 204 – Advanced Computer Networks, fall 2003

A Performance Comparison of Routing Protocols for Large-Scale Wireless Mobile Ad Hoc Networks

Ioannis Broustis, Gentian Jakllari, Thomas Repantis

University of California, Riverside
Department of Computer Science & Engineering
Riverside, CA 92521

http://www.cs.ucr.edu
{broustis, jakllari, trep} @ cs.ucr.edu

ABSTRACT

It is commonly known to the wireless research community that use of efficient routing algorithms in ad hoc networks offers a number of considerable benefits. Some of them are: larger throughput, lower average end-to-end delay, decrement in the number of lost data packets and generally an improved network performance. Many routing protocols for such networks have been proposed so far, the most popular of which are the Distance Vector Routing protocol (DSR), the Ad hoc On-demand Distance Vector routing protocol (AODV), the Temporarily-Ordered Routing Algorithm (TORA) and the Location-Aided Routing protocol (LAR). In this report we demonstrate some results, derived from the extended simulations that we performed, in order to compare the efficiency of the above four protocols. We consider that wireless mobile terminals are spread in a large geographical area. For our simulations we used two simulators, the QualNet and the ns-2 simulator.

INDEX TERMS

Wireless Communication, Mobile Ad Hoc Networks (*MANET*), IEEE 802.11, Data Routing, Algorithm, Simulation, Performance Evaluation.

1. INTRODUCTION

Wireless Ad Hoc Networks are mobile or static networks in which wireless terminals cooperate to maintain network connectivity and to exchange information. WLANs are an alternative to the high temporal installation and maintenance cost incurred by traditional changes in wired LAN infrastructures. Moreover, deployment of such networks is inevitable in cases where wired network installation is sometimes impossible, such as in battlefields, old monuments and concrete buildings with no previous network cabling. Because of the lack of a centralized control (access point) in ad hoc networks, terminals have to act as routers that forward data packet from sources to destinations. In order for ad hoc networks to operate as efficiently as possible, appropriate on-demand routing protocols have to be incorporated, which can find efficient routes from a source to a destination node, taking into consideration the fact that wireless stations have the freedom of movement. Mobility affects the ongoing transmissions, since a mobile node that receives and forwards packets may move beyond the coverage range of its neighbors. As a result, some (or all) of the links with its neighbors can be broken. In that case, a new route will have to be established, so as for the data flows to be restored. A quick route recovery should be one of the main characteristics of a well-designed routing protocol.

Our contribution has to do with the fact that our performance comparison between the most popular of these routing protocols involves a large-scale ad hoc network; terminals are spread uniformly across an extended geographical region. Our motivation is the lack of studies on large-scale network routing, and the goal is to test the efficiency of the above on-demand routing protocols in scenarios that involve spreading of mobile terminals in such environments.

The rest of this report is organized as follows. In section 2 we explain the two different kinds of routing protocols and describe in detail the on-demand protocols that we simulated. In section 3 we mention the most important previous studies on the subject and explain our contribution and extension to those studies. Section 4 gives a theoretical comparison mostly for AODV, DSR and LAR, and section 5 includes a description of the simulators that we used, the simulation results and our observations on the behavior of each protocol for each one of the metrics evaluated. Finally, section 7 concludes this paper.

2. ROUTING PROTOCOLS

2.1 GENERAL

According to their characteristics, routing protocols can be divided in two different categories: table-driven *(proactive)* and on-demand *(reactive)*. Table-driven routing protocols enforce mobile nodes to maintain tables with path information from every terminal to every other terminal in the wireless network. This information is updated by transmitting messages containing network topology changes, so as for each station to have at least one possible route towards any intended receiver. The most popular table-driven protocol is DSDV (Destination-Sequenced Distance-Vector Routing protocol).

The protocols that we compare in sections 5 and 6 belong to the on-demand category. A route discovery process is initiated only when a terminal needs to send data to an intended receiver. These protocols are source-initiated, since routes are discovered when sources need them. Moreover, these protocols have route maintenance mechanisms, which store the routing information until sources do not need it anymore or until routes becomes invalid; that is, some intermediate nodes become unreachable. A node X may not reached by a node Y (meaning that the X-Y link has failed) either because the node fails (mechanical or power failure) or because the node is moving beyond the coverage region of its neighbors. In paragraphs *2.2* to *2.5* we give the basic characteristics for each protocol that we evaluate and compare.

PARAMETERS	ON-DEMAND	TABLE-DRIVEN
Availability of Routing Information	Available when needed	Always available regardless of need
Routing Philosophy	Flat	Mostly Flat except for CGSR
Periodic route updates	Not Required	Yes
Coping with Mobility	Using Localized route discovery in ABR	Inform other nodes to achieve consistent routing tables
QoS Support	Few Can Support QoS	Mainly Shortest Path as QoS Metric

Table 1

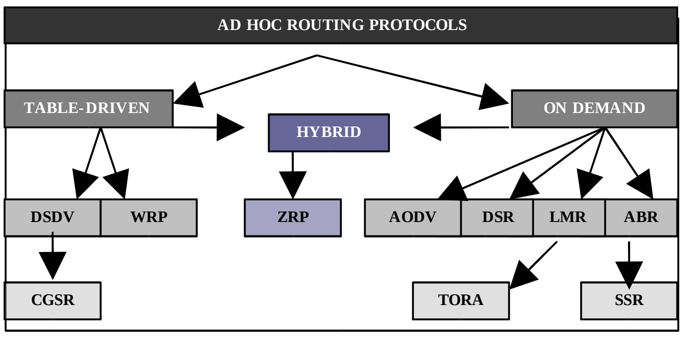


Figure 1

Both categories have their advantages and disadvantages. A new protocol called ***Zone Routing Protocol*** (ZRP) was designed to combine the advantages of both categories into a *hybrid* scheme. In takes advantage of the table-driven discovery within a node's local neighborhood and uses an on-demand protocol for communication between these neighborhoods. ZRP is actually not a distinct *protocol,* as it provides a *framework* for other protocols. The separation between a neighborhood and a global network topology gives space for different approaches - and thus taking advantage of each technique's features for a given situation. These local neighborhoods are called *zones*; each node may be within multiple overlapping zones, and each zone may be of a different size. Because we do not include ZRP in out work, will not further refer to it.

At this point we should discuss the basic characteristics that make an on-demand routing algorithm valuable. There are a number of problems that have to be taken into consideration during a routing protocol design:

- In a MANET, nodes are not constrained *"by the fetters of wires"*; mobility is the basic parameter that determines the efficiency of a routing algorithm. This is because as a node is moving, it can be placed beyond the coverage range of a neighbor with which it maintains a data transfer link. This will result in link failure. A well-designed routing algorithm has to be quick: nodes, that belong to the route that fails, have to be informed promptly about the problem, and a valid route has to be discovered as soon as possible, so as for the data flow to be continued. Thus, route recovery is an important metric for routing algorithm design.
- All modern *on-demand* routing algorithms include mechanisms for route discovery and route maintenance. Obviously this involves proper messaging with topology changes, something that requires bandwidth and increases overhead. Routing protocols should be able to constrain such message transmissions to nodes that need them. The rest of the nodes need not receive such *update* messages that will discard anyway. Broadcasting routing information, without bounding the broadcast to nodes that will actually need this information, creates other numerous problems. Some of them are: *congestion increment, bandwidth expenditure and additional energy consumptio*n. The negative consequences of the first two of them (additional data delays, overhead, etc) are obvious. For power consumption, someone could say that this actually is not a problem, since the transmitter will send the routing packet anyway - no additional power will be expended. As a matter of fact, this is true from the transmitter's perspective.

However, nodes expend power while receiving packets as well. If nodes discard packets that they receive, energy will be consumed without a purpose. Energy expenditure should be a highly considerable factor since, wireless terminals are powered by portable batteries; thus node lifetimes are limited. Many extensions to current routing protocols have been proposed, that try to reduce the power consumption in terminals [17], [18].

2.2 AODV

The Ad hoc On-demand Distance Vector routing protocol is based on the table-driven DSDV, however as an on-demand protocol, it does not maintain global routing information for the whole network. Nodes that do not belong to a route do not need to keep information about that route. Such nodes do not send or receive topology-update packets, so they have information only for their active routes; a node considers a route as active, if it sends, receives or forwards packets for that route and through which there is at least one data packet transmitted within a fixed time interval. (For some routing protocols, a node considers a route as active, if it overhears routing information that makes it realize that the route is active).

Hence in AODV, route discovery packets are initiated and broadcasted by sources, only when these sources desire to contact their intended destinations and do not have valid routes towards. Furthermore, changes in network topology need to be sent only to those terminals that will need this information, as changes in topology will occur in their local neighborhood and will not affect other distant active routes. Thus, AODV dynamically establishes route table entries, while every terminal maintains an increasing counter in order to replace unused or broken routes. The protocol includes mechanisms for both path discovery and path maintenance.

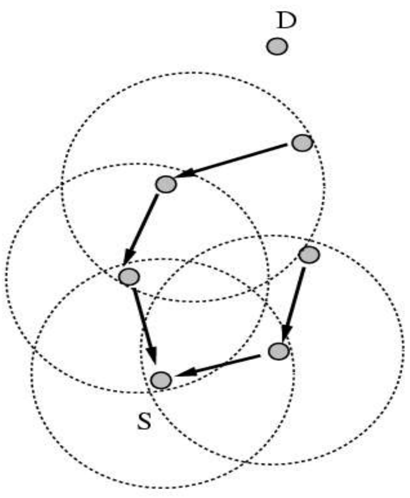


Figure 2

In a real environment there may be many different kinds of nodes, with different transmission powers. Thus it is possible that, while node A can forward packets to node B, node B may not be able to reach node A, since its maximum transmission power may be not enough. The link between nodes A and B is then called *"asymmetric"*. A large disadvantage of AODV is that it does not support asymmetric links. That is, AODV is capable of supporting only symmetric links between nodes, both of which are able to send packets to eac

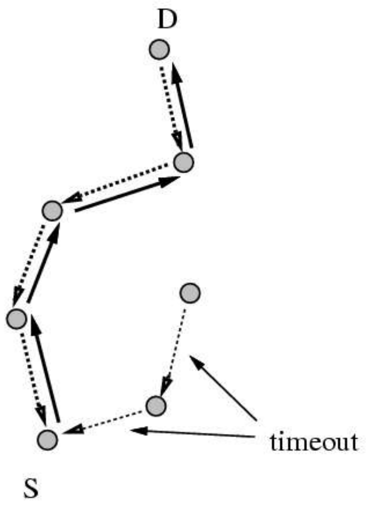


Figure 3

The route discovery procedure involves request and reply messages. Each source broadcasts a RREQ (Route REQuest) packet to all of its neighbors so as for the route discovery procedure to start. This packet contains the source and destination addresses, a hop counter, a source and a destination sequence number and a broadcast ID. The broadcast ID is increased by one, every time the source initiates a new request; it uniquely identifies a RREQ together with the source address.

If a station receiving the RREQ has a route to the destination, it will send a reply (RREP) back to the source. Else it will rebroadcast the RREQ to its neighbors after increasing the hop counter. If an intermediate node has already received a RREQ with the same source address and broadcast ID, it will discard it. The source sequence number is needed for nodes to maintain valid information about the reverse route back to the source. As the RREQ travels towards the intended destination, it sets up a (reverse) path back to the source: every node parses the RREQ and identifies the address of the neighbor that sent the first copy of that RREQ ***(figure 2)***. A node, that receives the RREQ, checks to see if it can provide a route towards destination, by reading the destination sequence number. If this number is larger than the one that the node has stored, the node will rebroadcast the RREQ. If it is smaller, and the node hasn't received a similar RREQ, then the node will send a RREP to the neighbor from which it received the RREQ. As the RREP travels back to the source, nodes set up pointers towards neighbors that forwarded this RREP packet. In addition, each of these intermediate nodes updates the timeout info for route entries towards both the source and the destination, and stores the latest destination sequence number. This process is depicted in ***figure 3***. Similar RREP packets will be forwarded back to the source, if they contain either larger destination sequence numbers, or fewer hop counts for the same destination sequence number.

Wireless mobile stations than run AODV, maintain for every route the following additional information:

- A *route request expiration timer*; it is used by nodes to delete reverse path entries that do not further belong to the source-destination path.
- The *route caching timeout*; it is the time after which the stored route is considered invalid.
- The *addresses of the active neighbors* that belong to the same route. By this way, all sources will be informed about a link failure.
- *A table with records for all the destinations of interest* is also maintained. Each record contains the destination, the next hop, the number of hops to reach that destination, the sequence number of this destination, all the active neighbors for this route and finally the expiration time for this record, which is updated every time the certain route is used for data transfer.

Mobility is the main reason that enforces sources to re-initiate the route discovery procedure. If the source, or any intermediate node, finds out that it cannot reach its next hop in the path, it will propagate a RREP with a "*fresh*" sequence number and an "*infinite*" hop count to all of its upstream neighbors. This RREP packet will be sequentially forwarded upstream until all active sources are informed about the link failure. If a source node still needs that route, it will re-initiate a new path discovery procedure, by broadcasting a RREQ with a new, incremented by one, destination sequence number. This will notify downstream nodes that a new route is needed.

The latest AODV version [3] has added a mechanism for query control optimization in the route discovery procedure. More accurately, an ***expanding ring search*** is initially followed to find routes; this process searches increasingly larger neighborhood regions to discover the intended target. A *TTL*

(Time To Leave) field has been added to the RREQ *IP* header. The source initially sets a timeout for receiving the RREP and, if this timeout is reached, the source will retransmit the RREQ with the TTL value incremented by *TTL-Increment*. Further retransmissions may occur, until the TTL reaches a maximum value; then, all the data packets destined to the target should be dropped, and a *Destination Unreachable* message should be delivered to the application. With this mechanism, unnecessary network-wide RREQ spreading will be prevented.

Finally, nodes that belong to at least one active route offer connectivity information to their neighbors by broadcasting local ***hello*** messages every *HELLO_INTERVAL* milliseconds. These *hello* messages are RREPs with *TTL=1*, and contain the node's address, the node's latest sequence number, a zero hop count and a lifetime based on the *HELLO_INTERVAL*.

Summarizing AODV, requests for route discovery are initiated only when sources need them, something that reduces drastically the packet routing overhead, and hence the overall packet delays. Moreover, because stations do not maintain global topology information, less memory space is required. In addition, "bad news", concerning link failures, travel back to sources quite quickly, so as for the new route discovery procedure to be initiated soon after the failure. This makes AODV very useful in a variety of applications, such as in battlefields, emergency services and video conferences, especially in large node populations.

2.3 DSR

The Dynamic Source Routing protocol also gives the capability to mobile sources of dynamically discovering paths towards any desired destination. Every data packet includes a complete list of nodes, which the packet has to visit in order to reach the destination. Hence, all nodes that forward or overhear these packets may store important routing information for future use. Even though nodes may move at any time and even continuously, DSR can support fast network topology changes.

Moreover, DSR can support asymmetric links; it can successfully find paths and forward packets in unidirectional link environments. What is more, like AODV, it has a mechanism for route maintenance that operates on-demand, meaning that there are no periodic topology update packets. When link failures occur, only nodes that forward packets through those links must receive proper routing advertisements.

In addition, DSR allows source nodes to receive and store more than one path towards a specific destination. Intermediate nodes have the opportunity to select another cached route as soon as they are informed about a link failure. By this way, less routing overhead is required for path recovery, something that reduces the overall data packet delay.

Let's see how DSR works. A source, that desires to send data to a destination, first checks to verify that it has a route in its cache for that destination. If it does, it will use that route by placing (in the data packet header) the sequence of hops that the packet must follow to reach the destination. If there is no such route stored in the local cache, the source will initiate a new path discovery process, by broadcasting a ***Route Request*** to its neighborhood. This message contains the source and destination addresses, a request ID and an ordered intermediate node address list, through which this message has passed. This node list is initially blank since the message is firstly sent by the source node (it has not been forwarded yet).

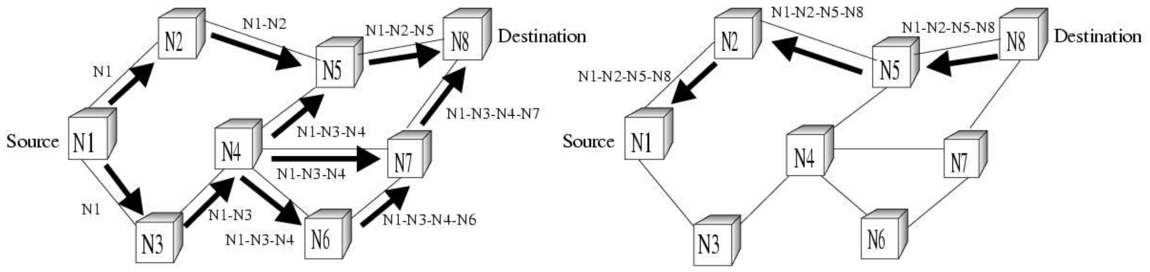


Figure 4

Every terminal, receiving the request message, parses it to see if it is the intended destination. From this point, the node may perform one of the following actions:

- •If it is the destination, it will reply with a Route Reply back to the source, after attaching the list with all intermediate nodes through which the request message passed.
- •If it is not the final target of the request, and has already received a similar request with the same ID from the same source, it will discard this request message.
- •If it is not the final target of the request and it sees that its own address is included in the message list, it will discard this request message.
- •Else it will append its own address in this list and then it will further broadcast it to its neighbors.

When the source receives a Route Reply message, it stores this route for further use. A node X, that receives a request and has a route towards destination, will typically check its local cache to find a route back to the source so as to send the Route Reply. If it has one, it will use it. Else it will initiate a new route discovery to get a path back to the source. This mechanism allows DSR to support asymmetric links among intermediate nodes. What is more, node X can decide to use the address list contained in the Route Request message and reverse it. This will immediately provide a reverse path back to the source; however this route may not be valid. This choice does not guarantee that the reply message will reach the request initiator, since the reverse path may be comprised of asymmetric-unidirectional links. In ***figure 5***, node X can reach node Y, but Y cannot reach X; the X-Y link is asymmetric. In ***figure 6***, however, all links all symmetric.

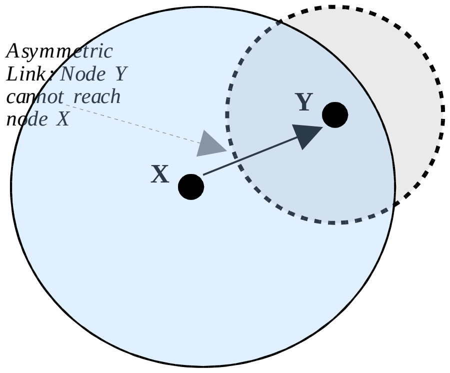
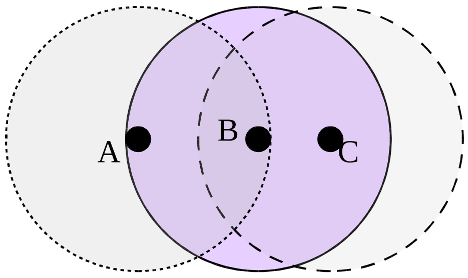


Figure 5

Every time a source transmits a Route Request, it stores a copy of this message in a *send buffer*, together with the time at which the transmission took place. After a certain timeout period, this message copy will be deleted. While this copy remains in the buffer, the source may initiate a new request for the same destination. However the source must limit the rate at which it sends similar requests, since the intended destination may not be reached anyway. Hence, this mechanism uses ***exponential backoff*** to limit the rate of similar request messages destined to the same target.

Each terminal belonging to the path, except the final target, must receive a positive acknowledgement from its next hop for successful delivery of every packet. If a node does not receive a confirmation within a time window, it will defer transmitting the next queued packet, but will retransmit the same one, until either a confirmation is received, or a maximum number of retransmissions is reached. The confirmation may not be sent as a different acknowledgement packet, since the underneath MAC layer may be taking care of it. Moreover, if e.g. node A sends a packet to node B and node B forwards it to node C, then the transmission from B to C may be overheard by node A, which then will make sure that node B received it (***figure 6***).

Figure 6



If none of the above can take place, the transmitter may set a bit (in the packet's header) that enforces the receiver to send a *DSR-specific* acknowledge to its previous hop. After a maximum number of retransmissions, without reception of an ACK, the node assumes that the link maintained with its next hop has failed, so it will generate a Route Error message that travels back to the source, pointing at the certain link that failed. The source will remove this link from its cache. If it has stored an alternative route towards the final target, it will use it. Otherwise it will re-initiate the route discovery process.

The major difference between DSR and the previously described AODV, is that DSR gathers much more routing information. Sources in DSR have complete routing information to reach any intermediate mode in a certain path. Moreover in DSR, intermediate nodes overhear routing packets exchanged between neighbors belonging to different active routes. By this way, intermediate nodes store a lot or routing information asynchronously, and may use it at any time. Another difference is that, destinations in DSR reply to all requests that they receive. Thus sources know more than one path towards a destination and may immediately use them as soon as they are informed about link failures. On the other hand in AODV, sources gather a very limited amount of routing information, as they reply only to the first RREQ and discard the rest. Furthermore in DSR, there are currently no mechanisms that either expire routes in the caches, or prefer a newer route. In AODV however, the newly discovered route is always chosen and the selection is based on the destination sequence number. Finally in DSR, a Route Error backtracks the data packet that meets a failed link and, as a result, nodes that are not on the upstream route but use that link will not be informed promptly about the problem. In AODV however, the RRER will reach all nodes that forward packets through that link. Our simulation results reveal the pros and the cons of each protocol.

2.4 LAR

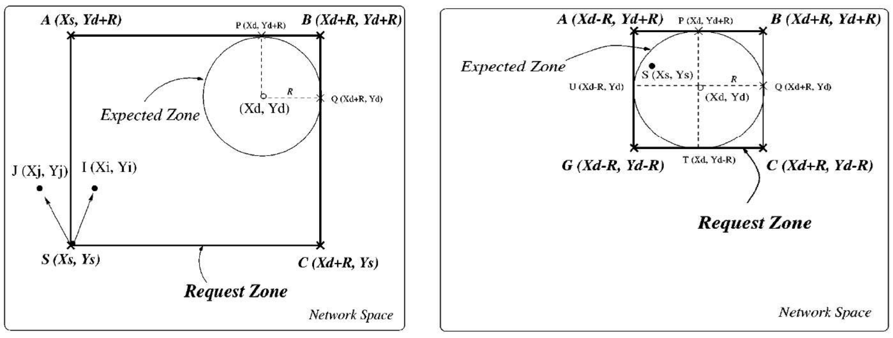
Routing overhead can be decreased, by giving location information to the mobile terminals, with use of the Global Positioning System (GPS) for route discovery. Two *Location-Aided Routing* algorithms that use location information have been proposed, showing how a route discovery protocol, based on flooding, can be improved.

Location determination, provided by GPS, includes an amount of error, of 50-100 meters, using NAVSTAR GPS [14], and 15-26meters using Differential GPS. Authors in [10] assume precise location information; however the LAR algorithms can be applied even when nodes have approximate neighbor location knowledge. If a node S wants to send data to a node D, for which it knows the previous location L at time t_0 and node D's speed *u*, then S *expects* that D will be located within an "***expected zone***" at time t_1, a circular area of radius $u(t_1 - t_0)$ and center L. If node S does not know the previous location L, then the "expected zone" for node D will be considered as the whole network geographical region, and the algorithm will follow the basic flooding as in DSR algorithm.

The LAR algorithms in [10] use flooding with one modification; the source node S defines a "***request zone***" for the route request. An intermediate node will forward the request message, only if it is located within the request zone. If the request zone includes the expected zone, the probability of finding node D will be increased. The request zone may also include other neighboring request zones. The two proposed LAR schemes in [10] *give terminals the capability of determining whether they belong to a requested zone or not*, so as to know if they should forward certain route request messages.

In the first LAR scheme, the request zone is a rectangular, and the source node S knows the target node D's average speed *u* and previous location (X_d, Y_d) at time t_0. The request zone is considered to be the smallest rectangle that includes the current source location, and the expected zone. The sides of the rectangle are parallel to X and Y axes. Hence, the source node may determine the request zone, and the route request message will additionally contain the four corners' coordinates (***figure 7***).

Each intermediate node, receiving the request message, checks whether it belongs to the rectangle; if it does not, it will discard the message, such as node *J* in the left side of the following figure. In the same left figure, node *I* will forward the message because it belongs to the rectangle. In the reply packet, node D will attach its precise location and the current time stamp, which will be stored in the source's cache for future use.



Source outside the expected zone *Source inside the expected zone*

Figure 7

In the second LAR scheme [10] the source S is assumed to know the destination's (D) coordinates (X_d, Y_d) at time t_0, and initiates the request at time t_1?t_0, in which it includes the distance between S and D and the (X_d, Y_d) coordinates. When a intermediate node *I* receives the request, it calculates its distance $DIST_I$ from D. If ?*$DIST_S$ + ? ? $DIST_I$ (?, ? are parameters) then node *I* will forward the message, in which it will have previously replaced the $DIST_S$ with $DIST_I$. Otherwise, node *I* will discard the request message. This procedure is followed by all intermediate terminals. The purpose of parameters ? and ? is discussed in [10].

To be more explanatory, in the first LAR scheme (***figure 8a***), nodes I and K will forward the request message, since they belong to the request zone. However, if node N receives the request, it will discard it. In the second scheme, (***figure 8b***), nodes I and N receive the request from S and they both forward the message, since they are both closer to D than S (assume ?=1, ?=0). However, node K will discard it. Hence, we can see that nodes K and N act differently in each scheme.

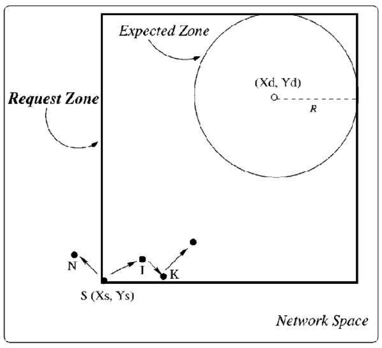


Figure 8a

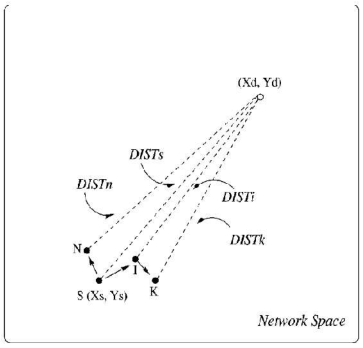


Figure 8b

In [10], both schemes are simulated and evaluated. For some of the simulations authors take into account the impact of location error. They also discuss and suggest the use of ***directional antennas*** on stations running LAR; this may decrease the routing overhead. With directional transmissions not all neighbors receive route requests that may discard anyway, but only those neighbors that lie within the directional transmission beam of each transmitter [11],[12]. This can be viewed in ***figure 9***.

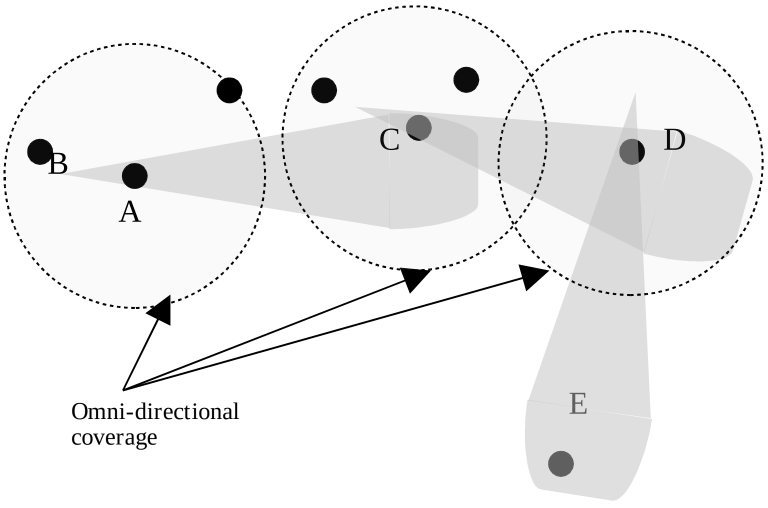


Figure 8
Figure 9

Here, all transmissions are directional. Let's assume that node B does not lie within the requested zone of node A. If node A transmitted omni-directionally, node B would receive the request and would simply discard it, because it would just realize that it does not belong to node A's requested zone. Using directional transmissions, such a situation can be avoided.

Use of directional antennas is clearly a MAC layer issue. However, this is another example of how the interconnection and collaboration between the MAC and the network layer can result to a better network performance.

2.5 TORA

Another distributed and loop-free algorithm is the Temporary-Ordered Routing Algorithm (TORA), which quickly provides multiple routes, with less routing overhead. The name of this source-initiated routing protocol arises from the fact that, the generation of routing messages, that would potentially pass though a large number of intermediate terminals, is bounded to the terminals located close to the topological changes. Each station needs information about its one-hop neighbors only. This reveals the distributed operation of this routing protocol, which provides multiple routes towards a destination. The protocol includes mechanisms for route discovery, route maintenance and route deletion.

We assume a network with N nodes represented by a graph $G = (N, L)$, where L is an initial set of undirected symmetric links *(i, j)*. Each link may be assigned one of three states: *undirected, directed from i to j, and directed from j to i.* For a node *i,* we define the neighbors $N_i \in N$, to the set of nodes *j* such that $(i, j) \in L$. Mobile nodes establish a directed acyclic graph towards destinations. When topological changes cause link failures, route re-establishment takes place through some *"temporarily-ordered"* computations, consisting of a sequence of directed link reversals. TORA discovers routes on demand; however the main target of the algorithm is to establish routes quickly, while finding the shortest path is of secondary importance. Below we give a brief description of this routing protocol. More details can be found in [15].

Every terminal has a "height" with respect to the destination, calculated by the protocol. Each time a source node desires to send data towards an intended receiver, it initiates a *Query* message in which it includes the destination address. The destination, or an intermediate receiver of this message with a route to destination, will reply with an *Update* packet listing its height. Each terminal receiving this *Update* packet sets its height to some value larger than the one contained in the packet. By this way, a set of sequential directed links is created, with edges from the source node to the node that first broadcasted the *Update* packet. If a station X finds out that a route to the destination is invalid, it will broadcast an *Update* packet with a height value that is larger than any other height value in the neighborhood. Of course there is a case where no neighbors have a finite height value for the same destination. In such case, node X will initiate a new route discovery towards that destination. In addition, if a node detects a network partition, it will broadcast a *Clear* message that deletes invalid routes from the network. Because the height metric depends on the time that a link failed, in TORA all terminals use clocks for synchronization.

One can better understand the "height" functionality in terms of hydromechanics. Let's consider water (data) flowing downhill towards a destination through a set of tubes, representing links. The edges of tubes are the network stations. Each node has a height with respect

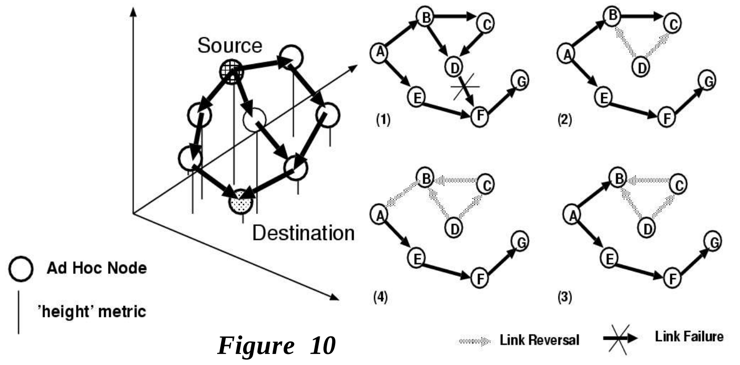


Figure 10

to the destination. If a tube between two nodes A and B stopples (link failure), water will not further flow through that tube. In that case, the protocol will set the height of A to a value greater that any height in the neighborhood in a way that water won’t flow either towards A (because of the gravity) or towards any previous terminal that forwarded packets to node A, destined to the final target.

In conclusion, TORA is a totally distributed routing protocol that quickly establishes and maintains multipath routes with the minimum routing overhead. It reacts rapidly to link failures due to changes in network topology. We simulated the TORA algorithm version that is currently implemented in the ns-2 simulator. For the simulation of this protocol we used ns-2 only and evaluated its performance in comparison to AODV, which we simulated using both ns-2 and QualNet.

3. PREVIOUS WORK

In this section we present the most valuable previous studies concerning ad hoc on-demand routing performance comparisons. Many researchers have compared and evaluated the efficiency of the most famous on-demand routing protocols for wireless mobile ad hoc networks, in the past. As we mentioned above, node mobility is the main network characteristic that reveals the vigilance of a routing protocol to respond to network topology changes, and to perform fast route establishment and recovery. Thus, most of the previous studies compare the behavior of the protocols as a function of the node mobility. More specifically, authors in [4] compare four ad hoc routing protocols using a maximum number of 50 nodes but their traffic load is relatively slow, since the data packet size is 64 bytes, the maximum number of sources is 30 and every source node transmits 4 packets / sec. Furthermore, authors do not use the latest AODV version with the expanding ring search, which has an increased performance compared to the older version. Their mobility metric involves pause times: nodes are moving towards a specific point and as soon as they reach it, they stay static for a certain amount of time, which is the pause time. Most of the previous work has taken this metric into account. Authors in [5] compare three routing protocols, AODV, DSR and STAR, for which they used two simulators as well: GlomoSim and ns-2. Their results are quite valuable; however they assume a relatively small geographical region. An interesting approach is also followed in [6], in which authors have introduced a new mobility metric: the relative terminal speeds rather than absolute pause times and speeds. An excellent work is presented in [7], in which authors have performed an extended performance evaluation between DSR and AODV, in which the basic mobility metric is the node pause times. This work however does not include large-scale networks either. Their results show that the lack of a mechanism that could expire unused routes from caches in DSR, together with the “aggressive” use of caching, are the main reasons for the large data packet delays and the small throughput at high loads. Moreover they observe that if both protocols use larger queues and mechanisms for clearing memory from old routes, then the performance can be increased, while the interaction with a suitable MAC protocol will also be beneficial.

Most of the previous work is limited on performing simulations for ad hoc networks with a limited number of nodes deployed in small geographical are. The main reason is that simulations with many nodes, spread in a large area, need too much time to be completed using common simulators such as ns-2. Nowadays, the ad hoc network technology becomes more and more popular and, as a result, large-scale ad hoc networks are often deployed in battlefields, regions of disaster and large towns. A systematic comparison of the efficiency of the current famous ad hoc routing protocols, is needed so as for the wireless community to observe their behavior and usefulness in such networks. In fact, this was our motivation: to make observations about the behavior of these protocols under a large-scale environment, and to decide about whether a new protocol needs to be designed, in order for new challenges to be addressed.

4. THEORETICAL PERFORMANCE COMPARISON

In this section based on [22] we discuss the main algorithmic differences between DSR, AODV and LAR that we simulated in QualNet, and briefly mention what we expect to see from the simulation results. We compare the protocols with respect to flexibility in topology changes, resource usage, scalability, path discovery costs, and security. Because of the fact that we did not simulate TORA using QualNet, we do not have a common basis for simulating all four protocols, since we used a different simulation tool. This is why we avoid anticipating comparative simulation results for the TORA routing protocol. Hence, for the theoretical comparison between TORA and AODV, which we both simulated in ns-2, we provide ***table 2*** below. The main algorithmic differences between DSR and AODV were mentioned in section 2.3; however we give more explanations and details later in this section. The reader may refer to the respective papers for more details.

PARAMETERS	AODV	TORA
Time Complexity (start)	O(d)	O(d)
Time Complexity (after fail)	O(d)	O(d)
Routing / Loop Free	Flat / Yes	Flat / Yes
Multicast Capability	Yes	No
Beaconing Requirements	No	No
Multiple Route Possibilities	No	Yes
Route Reconfiguration	Erase Route, Notify Source	Link reversal, route repair
Route Maintained in	Route table	Route table
Routing Metric	Fresh and shortest	Shortest

Table 2

Below, we briefly review the basic protocol characteristics and techniques that we will compare theoretically:

4.1 AODV

a. ***Flexibility:*** Link failures are detected using the timeout technique. If small timeout values are selected, "bad news" about link failures will travel fast, back to the sources. Moreover, if hello messages are not received within an appropriately predetermined "*HELLO_INTERVAL*" value, nodes will assume link failures before a packet is lost due to timeout. The link failure news reaches all member nodes of the route on the source's side of the link break. New routing information will be passed to nodes that were using the route recently. Because AODV assumes symmetric links, performance is decreased in unidirectional link environments.

b. ***Memory:*** Assuming that we have *N* nodes, the largest route table may be *O(N)*. Similarly, the maximum size of the set of active neighbors, and thus the maximum total memory usage, will be *O(N)*. However, the real route table size will be much smaller, as AODV is an on-demand protocol…
c. ***Number of packets:*** For path maintenance, each node belonging to an active route has to receive one packet only. *Hello* messages are sent during periods of network inactivity, so they do not have an impact on the network.
d. ***Path discovery cost:*** For *N* nodes in the network, the maximum discovery packets are N^2. The total time it takes for a route to be discovered in *O(N)*.
e. ***Scalability:*** We expect that AODV will perform well as scalability increases, since there are no special needs for memory usage.

4.2 DSR

- ***Flexibility:*** Every node maintains complete routing information, since the routing control packets include a list with all nodes in the path. When a route becomes unavailable, the protocol will require a route re-discovery, and all old state information will be discarded. A topology change that causes a link failure travels quickly. Invalid or old routes may be advertised, since nodes are allowed to respond authoritatively on behalf of others. DSR will not be able to keep up with changes in network topology, when nodes move very quickly.
- ***Memory:*** Every member of the active route maintains full routing information to contact any other node in the same route. As a result, each node needs enough memory to store a full network topology.
- ***Number of packets:*** As we mentioned above, there are control packets for request, reply and error messages. We expect that the number of required control packets will not affect the performance.
- ***Path discovery cost:*** In DSR there may be a case, in which many nodes know a route to the destination. If all of them replied to the request, then bandwidth would is consumed. To avoid such situation, DSR has a mechanism that *defers* transmitting a reply. The *defer* time period is: $d = H * (h - 1 + r)$, where:
  - d is the delay time,
  - H is the time needed to make one hop,
  - h is the number of hops from the current node to the destination, and
  - r is a number with a random value, $r \in [0,1]$.

  During d, the node listens to the channel; if it hears a response to the request with a *fresher* route that the one that the node has stored, then the node will not send a reply. Otherwise it will initiate its own reply.
- ***Scalability:*** In an active route, each member maintains complete knowledge about the paths through which it communicates with others. This decreases the efficiency of the protocol as scalability increases. In the worst case, a node will need to contact every other

node in the network (*$O(N^2)$*). In addition, in a dense environment, nodes overhear too much routing information. The overhead of such transmissions grows linearly as *$O(N^2)$*.

4.3 LAR

- ***Flexibility - Number of packets:*** No update packets are needed to be exchanged. We expect the routing overhead to be lower than in protocols that use flooding as their basic technique for route discoveries.
- ***Resource allocation:*** There is a smaller number of control packets received, since transmissions of such packets are bounded within (small) request zones. What is more, in LAR nodes need only store location and speed of their intended destinations, instead of detailed route information.
- ***Scalability:*** The routing overhead is the same with flooding, for small node populations. Are population increases, we should expect the routing overhead to be much lower than the one that flooding generates.

4.4 COMPARISON

After some brief observations, let's compare the protocols for each field:

- ***Flexibility:*** All of them use timeouts to detect link failures; however AODV utilizes *hello* messages and traffic monitoring on active nodes as well. LAR anticipates topology changes (expected zone), something that reduces control overhead. DSR has the advantage of quickly discovering routes in asymmetric link environments, and in addition, a Route Error message may invalidate many routes simultaneously. From these observations we expect that DSR is more vigilant to topology changes.
- ***Memory:*** In AODV, nodes need only store information about their next hop in the path. In contrast, DSR enforces nodes to store the whole path information (bounded by *$O(N^2)$*) and also in LAR nodes need to store coordinates and speed (bounded by *$O(N^2)$* as well). Thus, AODV outperforms in memory usage.
- ***Path discovery cost:*** In the worst case, the source will need to flood the whole network, for all three protocols. In that case, if *D* is the network diameter, each protocol performs in *$O(D)$* time. What is more, For flooding the whole network, *$O(N)$* packets will have to be sent.
- ***Scalability:*** As we mentioned above, AODV allocates much less memory than DSR; thus we expect AODV to be more scalable than DSR. What is more, LAR was designed to limit flooding, thus it can be used to enhance scalability.

To conclude this discussion, we admit that none of the three protocols outperforms in all cases. We believe that AODV is preferable for more static networks, while DSR seems more eligible for small and dynamic networks. For LAR, we think that it is quite beneficial, especially in mobile populations, and it gets more beneficial, as scalability increases.

5. PERFORMANCE COMPARISON THROUGH SIMULATIONS

5.1 THE SIMULATORS

As we mentioned before, at our best knowledge, there is no previous work comparing the efficiency of ad hoc routing protocols for *large-scale* wireless networks. This is why we've used the *QualNet* simulator as our primary simulation tool. QualNet has the ability to run simulations with large number of nodes spread in an extended geographical region in reasonable time. It is a commercial derivation of GloMoSim developed at UCLA for doing efficient large-scale simulations involving thousands of nodes. The commercial version includes a number of useful features:

- An extended MANET library providing wireless dynamic routing, detailed physical layer effects such as steerable directional antennas, fading, shadowing, mobility, and implementation of the most important protocols for MAC and the Routing Layer
- GUI tools for system/protocols modeling
- Standard API for developing new protocols

There are a variety of simulation parameters that can be chosen, as well a large number of available statistics collected at each layer. We were able to determine simulation time, number of nodes, mobility model and node pause time and speed, traffic load, application (e.g. CBR, FTP etc), wireless characteristics, antenna model, desired routing and MAC protocols, environmental conditions and many other features that can make simulation results correspond as closer to the reality as possible. QualNet provides even an animator, Fig 11, written in JAVA for demonstrations purposes. The animator shows the coverage range of each node, the mobility and the path that packets follow to reach their destinations. For our simulations however, we did not use the *QualNet Animator* because it is to slow specially for a large number of nodes such as in our case.

Moreover, after the end of each simulation, the *QualNet Analyzer* provides the simulation results in categories for each OSI level, separately for each terminal. Because of our problem formulation, we were mostly interested in results related to the network and the application layers. More specifically, we needed results showing the routing overhead, the average end-to-end packet delay and the fraction of delivered data packets, mostly as a function of mobility. QualNet provides *.dat* files with results, as well as graphs. We used QualNet to simulate three routing protocols: AODV, DSR and LAR.

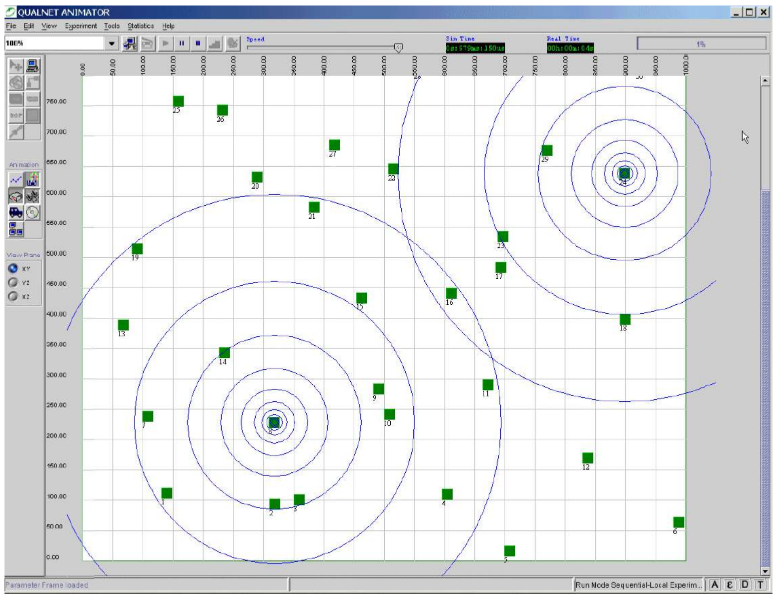

QUALNET ANIMATOR
File Edit View Experiment Tools Statistics Help
100%
Speed
Sim Time
Real Time
Animation
View Plane
Parameter Frame loaded
Run Mode Sequential-Local Experim...

Figure 11

Besides QualNet, we also performed simulations using ns-2. The basic limitation of this simulator is that it is very slow for scenarios with many nodes, and relatively difficult for users to outdistance some programming handicaps. Moreover, many researchers that have previously used ns-2, have admitted that the simulator has many bugs that can lead to confusion and wrong reality representation. We used ns-2 to simulate AODV and TORA, but with a limited number of nodes, and without having the capability of determining as many important parameters as in QualNet. Hence, out primary simulation tool was QualNet; however we used ns-2 as well to observe differences.

In section 6 we present our results for both simulators. In addition, we make observations about the different behavior of each routing protocol and try to explain this behavior, based on the algorithm that each protocol follows. Furthermore, we try to compare results from different simulators, giving a collateral evaluation for each tool.

5.2 PERFORMANCE ANALYSIS USING QUALNET

5.2.1 The Traffic and Mobility Models

We've used a similar model with [5], [7] changing only the number of nodes. We did so, in order to see the impact of using large-scale topologies in the performance of the protocols as opposed to the case when a limited number of nodes, 50-100, are used.
The traffic sources are continuous bit rate (CBR). The source-destination pairs are chosen randomly from the set of the networks nodes and are the same for all the duration of the simulation. The data packet is chosen to be 512 bytes and the channel bandwidth 2 Mbps.
The mobility model is random waypoint in a rectangular field 12000m x 6000m with 500 nodes. Each node at the beginning of the simulation remains still for "pause" time, then selects a random destination from the simulation space and moves towards that point with a randomly chosen speed (uniformly distributed between 0-20 m/s). Each simulation is run for 200s (simulation time).

5.2.2 Performance metrics

Again we've used the same performance metrics as in [5], [7]. Namely:

- Average end-to-end delay of data packets
- Normalized routing overhead – The number of routing packets per data packets delivered at the destination.
- Normalized routing load – The number of routing packets transmitted per data packet delivered to the destination.

5.2.3 Simulation Results

For our simulation we've used 20 sources generating packet with a fix rate of 4 packets/seconds.
In Fig. 5.1, is depicted the Packet Delivery Fraction (PDF) for the three routing protocols upon investigation. As we can see there's an important degradation of PDF for the AODV as opposed to that of LAR1 and DSR. What is most important is that there is a non-trivial difference between the PDF of AODV measured for 500 nodes and that measured in [7], Fig. 5.2, for 50 nodes. A possible reason could be that the route discovery process of AODV causes very long delays for

large scale networks, due to the amount of control packets transmitted. These delays result in packets waiting in the queues being dropped. One should not be surprised by the fact that the end-to-end average delay of AODV appears to be small, as it refers only to delivered packets.

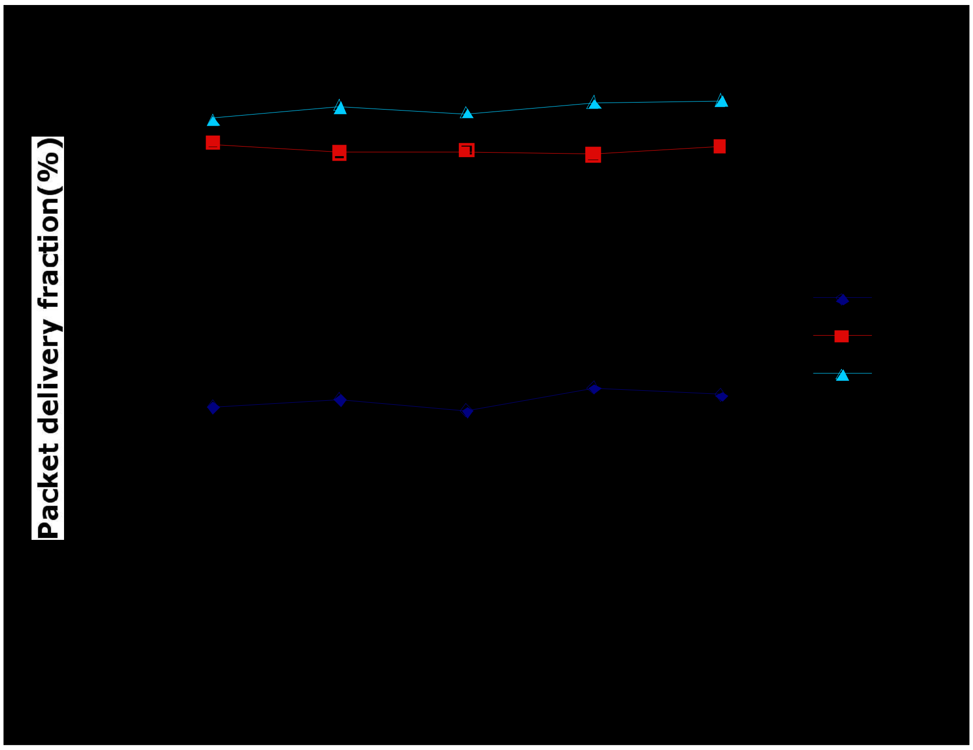


Fig. 5.1

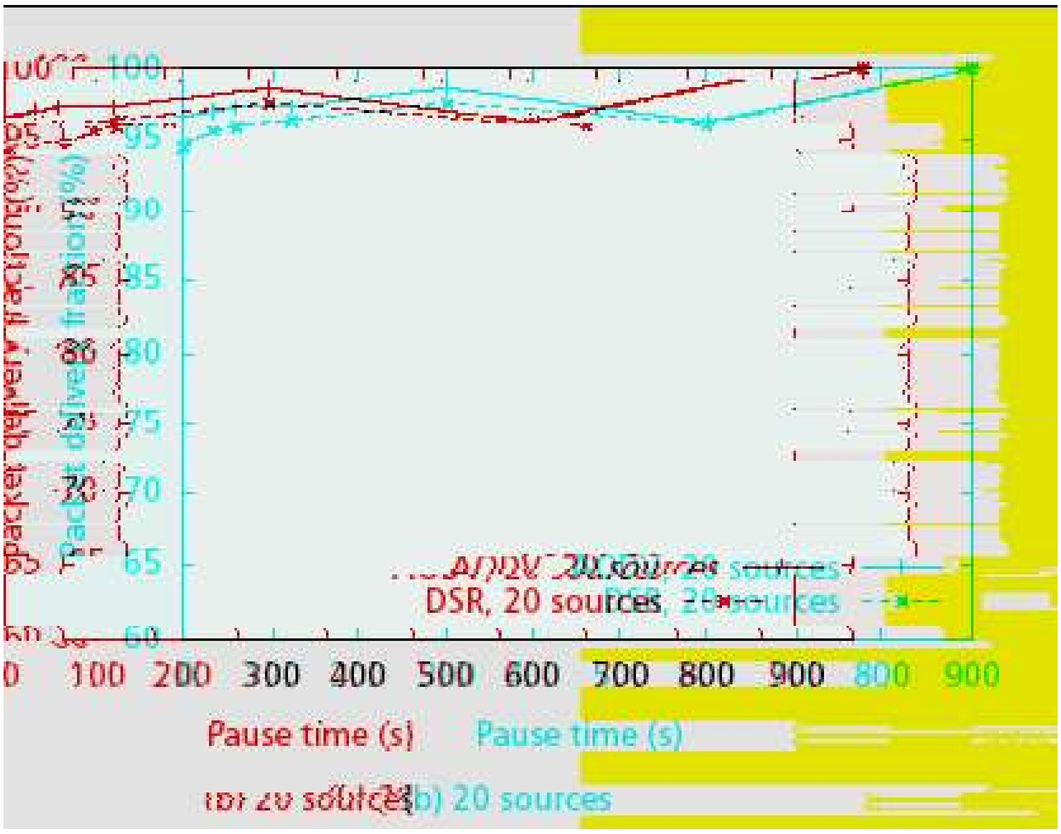


Fig. 5.2 from [7]

In Fig. 5.3 is depicted the Average delay in seconds for LAR, DSR and AODV. For this metric DSR is demonstrating a bad performance as opposed to that achieved for a 50 nodes topology [7] Fig. 5.4. A possible explanation for this result could be the aggressive use of route caching in DSR. For a large number of nodes the cache size can increase significantly resulting to increase in delay. Furthermore choosing stale routes can further increase the delay.

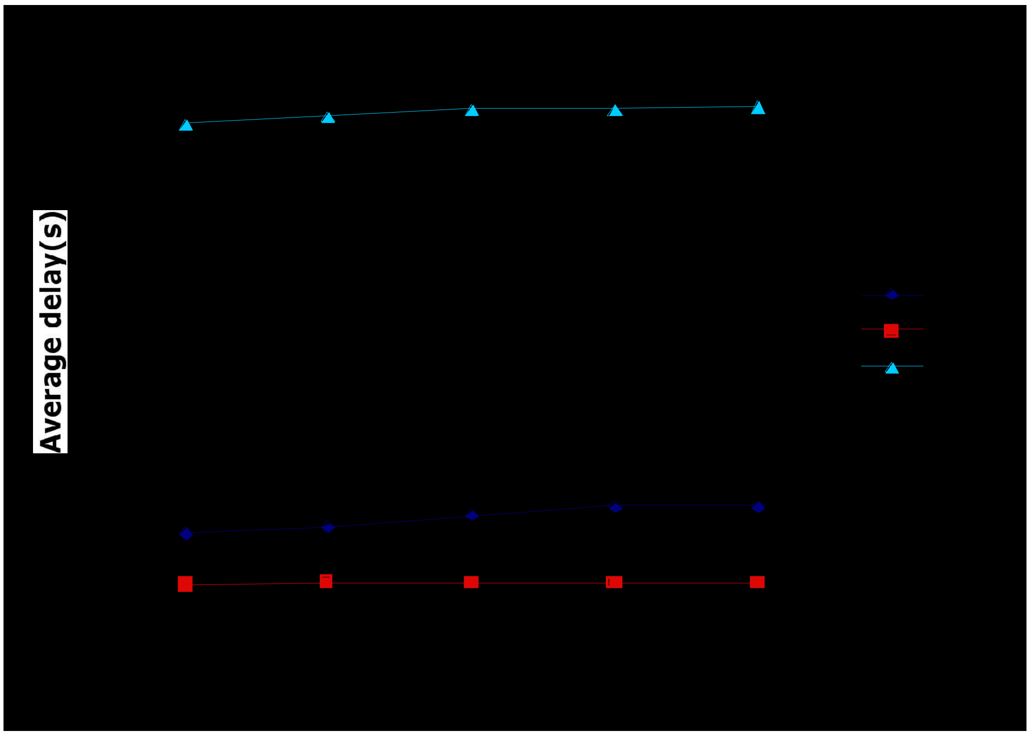


Fig. 5.3

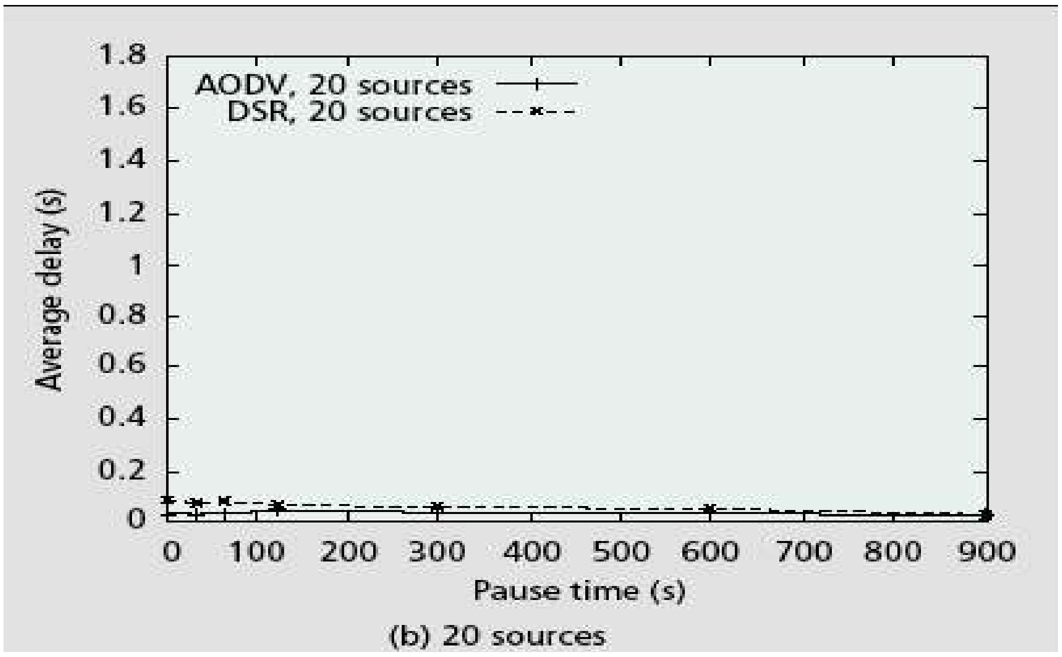


Fig. 5.4 from [7]

For the last metric investigated in our analysis, normalized routing overhead, the results are depicted in Fig. 5.5. There is a dramatic increase in routing overhead for both DSR and AODV compared to the 50 nodes topology [7], Fig. 6. This is expected, as many more packets are needed for the route discoveries, especially for AODV, where each of its route discoveries typically propagates to every node. DSR limits the amount of routing packets by making use of cached routes. Another observation is that LAR performs much better than the other two since it makes use of the nodes location, decreasing in this way the number of routing packets broadcasted.

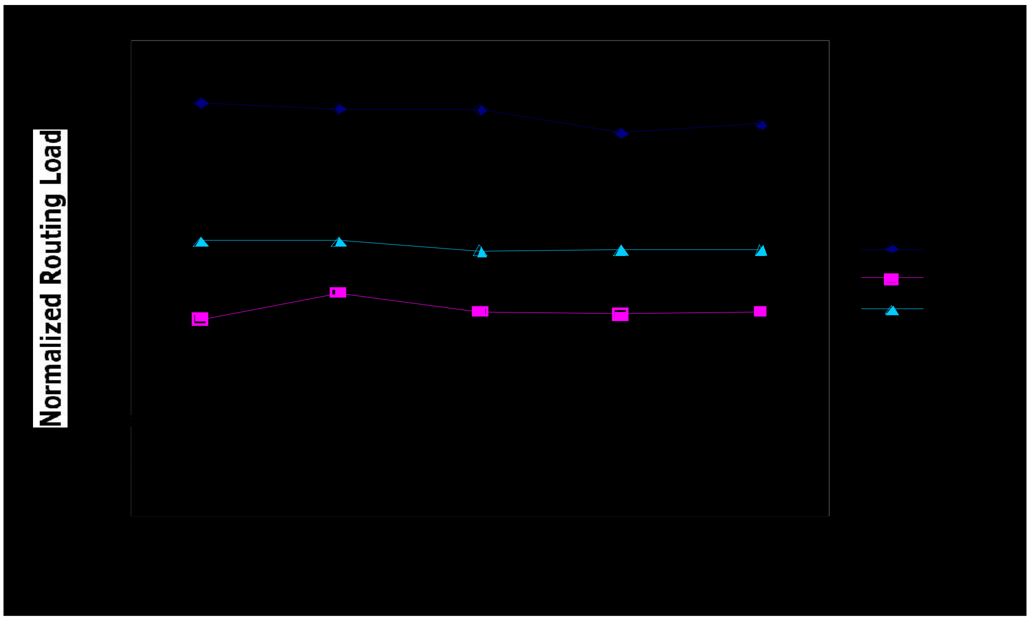


Fig. 5.5

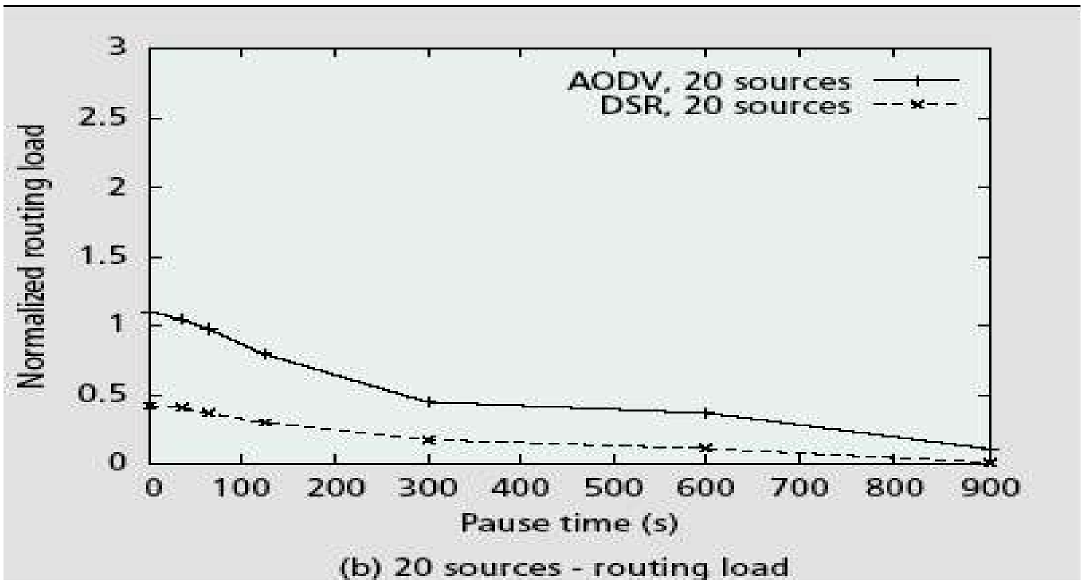


Fig. 5.6 from [7]

5.3 PERFORMANCE ANALYSIS USING NS-2

5.3.1 SIMULATION MODEL

The simulation model we used was based on the Monarch Project's extensions to ns-2, to support multi-hop ad hoc wireless networks [4]. These include physical, data link, and medium access control layer models. The Distributed Coordination Function (DCF) of IEEE 802.11 is used to model the contention of nodes for the wireless medium. The radio model uses characteristics similar to Lucent's WaveLAN direct sequence spread spectrum radio.

The protocols maintain a send buffer of 64 packets, which contains the data packets waiting to be routed. Those are dropped if they wait in the send buffer for more than 30s. All the packets are queued in the interface queue, until the MAC layer can transmit them. The interface queue can hold 50 packets at most.

5.3.2 TRAFFIC MODEL

The source-destination pairs were spread randomly over the network. Constant bit rate (CBR) traffic sources were used. We experimented for different offered loads, by varying the number of source-destination pairs (10 and 20), while keeping the size of the packets and the packet sending rate constant, at 512 bytes and 4 packets/s respectively.

5.3.3 MOBILITY MODEL

We simulated 50 wireless nodes forming an ad hoc network, moving over a rectangular 1500x300 flat space, with a maximum speed of 20 m/s (average speed 10 m/s). The movement of the nodes was based on the random waypoint model [19]. Each packet starts its journey from a random location to a random destination with a seed of 1 (randomly chosen and uniformly distributed between 0-20 m/s). Once the destination is reached, another random destination is chosen after a pause. The pause time, which affects the relative speed of the nodes is varied, from 0 (constant motion) to the length of the simulation (no motion). We ran this scenario for both 200s and 900s of simulated time.

5.3.4 METRICS

Three performance metrics were evaluated:

End-to-end average delay of data packets. This includes the propagation and transfer times, delays at the MAC due to retransmission, and delays at the interface queue and the send buffer.

Packet delivery fraction. The ratio of the packets received by the CBR sinks at the destinations over the packets generated from the CBR sources. The packet delivery fraction describes the loss rate, which shows the maximum throughput the network can support.

Routing overhead. The total number of routing packets transmitted. The routing overhead does not include MAC or ARP packets, since each routing protocol could be run over different medium access or address resolution protocols, each having different overhead. The routing overhead measures the degree to which the protocol will function in networks with many nodes, under heavy load, or in low-bandwidth environments. Large numbers of routing packets can increase the delays in the network interface transmission queues, the probability of packet collisions, and the power consumption in the nodes.

5.3.5 IMPLEMENTATION DETAILS

Two different versions of ns were compiled and used, the latest (2.26), and the one before that (2.1b9a). Unfortunately DSR simulations dumped core on both versions, and therefore we were only able to use AODV and TORA as our on-demand routing protocols.

Random traffic connections and node movements were generated using the cmu-scen-gen scripts. Specifically, we used *cbrgen* to create files describing traffic connections, giving as input the type of the connections (CBR), the number of nodes (50), the seed (1), the rate (4 packets/s), and the maximum number of connections (10 and 20). We also used *setdest* to create files describing node movements, giving as input the number of nodes (50), the different pause times, the maximum speed (20 m/s), the simulation time (200 s and 900 s), and the x and y coordinates (1500m and 300m respectively).

We wrote the tcl scripts to carry out the simulations. In them the various wireless simulation characteristics were defined, including the type of the antenna, the radio-propagation model, and of course the type of the ad-hoc routing protocol. We also wrote python scripts to parse the trace files that are produced by the simulations, and extract the information used for the performance metrics. The new trace format of ns was used. It includes information such as packet id, source port, destination port, time, source, destination, x,y,z coordinates, and energy level of nodes.

5.3.6 RESULTS

In order to test the ability of the protocols to successfully deliver data packets, while adapting to network topology changes, we varied the workload, by using 10 and 20 maximum connections. By experimenting with different pause times, we were able to measure the performance of the protocols for different degrees of mobility.

To compare the two routing protocols fairly, identical mobility and traffic scenarios were used for both of them. In order to achieve that, each run of the simulator was given two scenario files, describing the exact motion of each node and the exact sequence of packets originated by each node, together with the exact time at which each change in motion or packet origination occurs. We generated 21 scenario files altogether.

We also run the simulations for 900s of simulated time, apart from 200s, to make sure that this does not greatly affect the results.

EFFECT OF MAXIMUM CONNECTIONS ON AODV

We determined the behavior of AODV when doubling the number of maximum connections, hence increasing the network load. As we can see in figure 5.3.1, the average delay did –as expected– increase, but to a reasonable extent. This increase can be justified by the additional bandwidth consumed by the data packets that are dropped, as well as by the extra routing and MAC control packets. MAC control packets (RTS, CTS, etc.) have also to be retransmitted often, due to collisions or link loss.

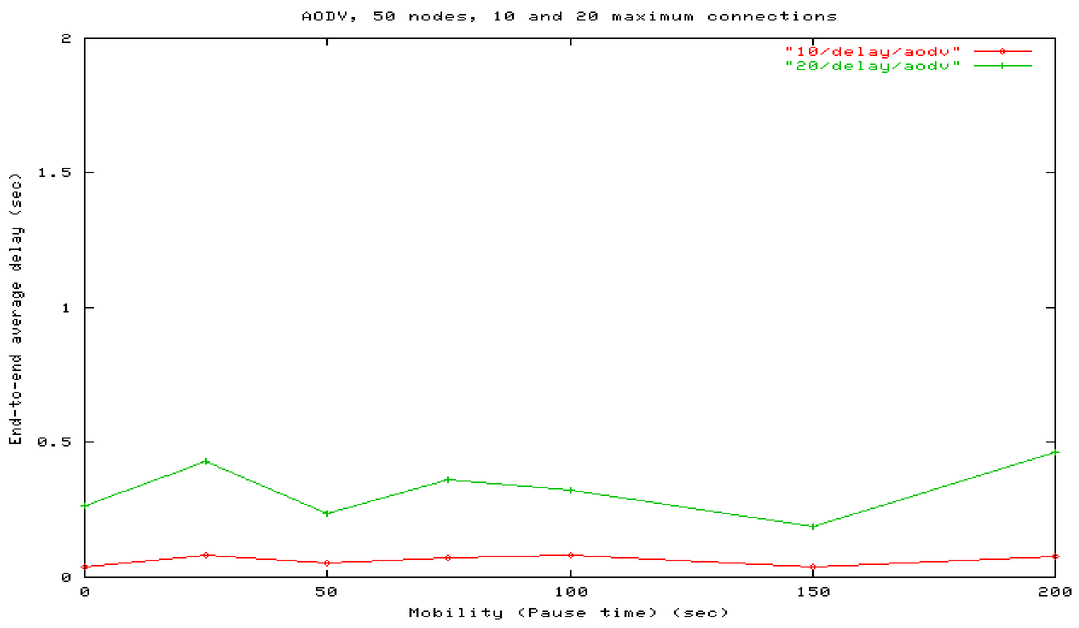


Fig 5.3.1

Figure 5.3.2 shows the drop in the packet delivery fraction, when doubling the maximum connections. The amount of packets received has decreased significantly, especially for low pause times, (higher mobility). These results agree with the results presented in [20].

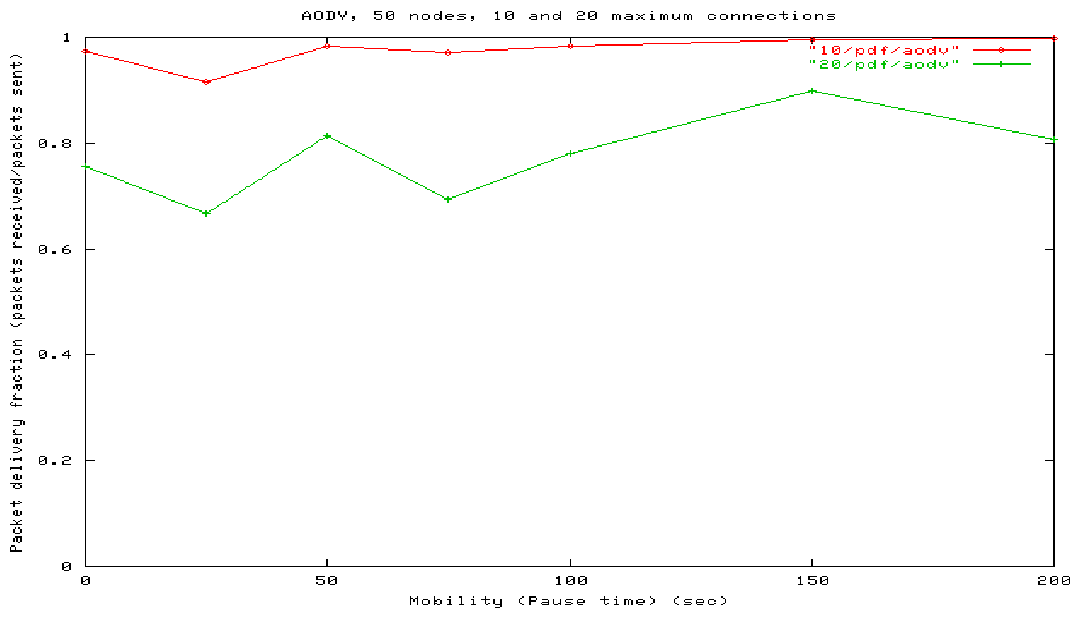


Fig. 5.3.2

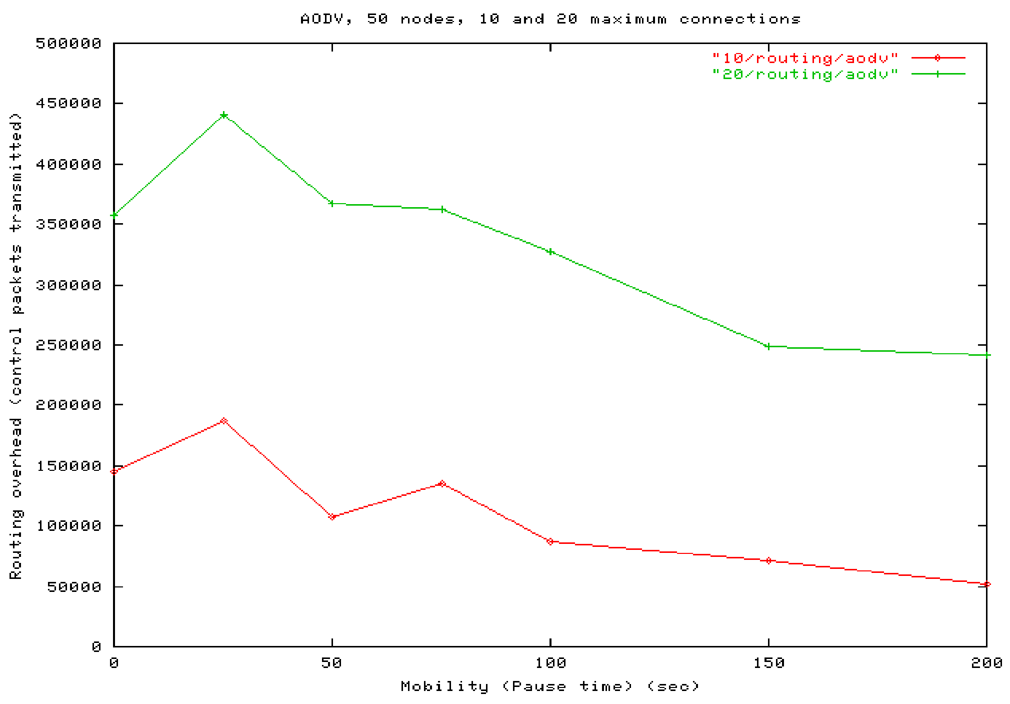


Fig. 5.3.3

Figure 5.3.3 Shows the significant increase in routing packets when the maximum connections double. This is to be expected, since AODV is an on-demand routing protocol and as the number of sources increases, more routing packets have to be transmitted, for working routes to more destinations to be maintained. The results agree with those presented in [19], even though the number scales are different, since there 64- instead of 512-bytes packets are used.

EFFECT OF MAXIMUM CONNECTIONS ON TORA

The effect of maximum connections was more severe on TORA. Taking into account the packet size (512 bytes), TORA seemed unable to route that amount of traffic, and dropped the major part of the packets, as shown in **figure 5.3.5**. This is an extreme case of the phenomenon described in [19], happening for 30 sources and only 64-bytes packet size. TORA fails to converge, because of increased congestion. TORA is layered on top of IMEP, the Internet MANET Encapsulation Protocol [21], which is required to provide reliable, in-order delivery of all routing messages from a node to each of its neighbors, as well as notification to the routing protocol whenever a link to one of its neighbors is created or broken. The congestive collapse observed is most probably happening due to a positive feedback loop developed in TORA/IMEP, wherein the number of routing packets sent cause numerous collisions in the MAC-layer, which in turn cause data, *ACK*, and *HELLO* packets to be lost. The loss of these packets cause IMEP to erroneously believe that links to its neighbors are breaking. TORA reacts to the perceived link breakages by sending more *UPDATE* messages, which in turn cause more congestion. Moreover each *UPDATE* requires reliable delivery, which increases the exposure to additional erroneous links failure detections, since the failure to receive an *ACK* from retransmitted *UPDATEs* is treated as a link failure indication.

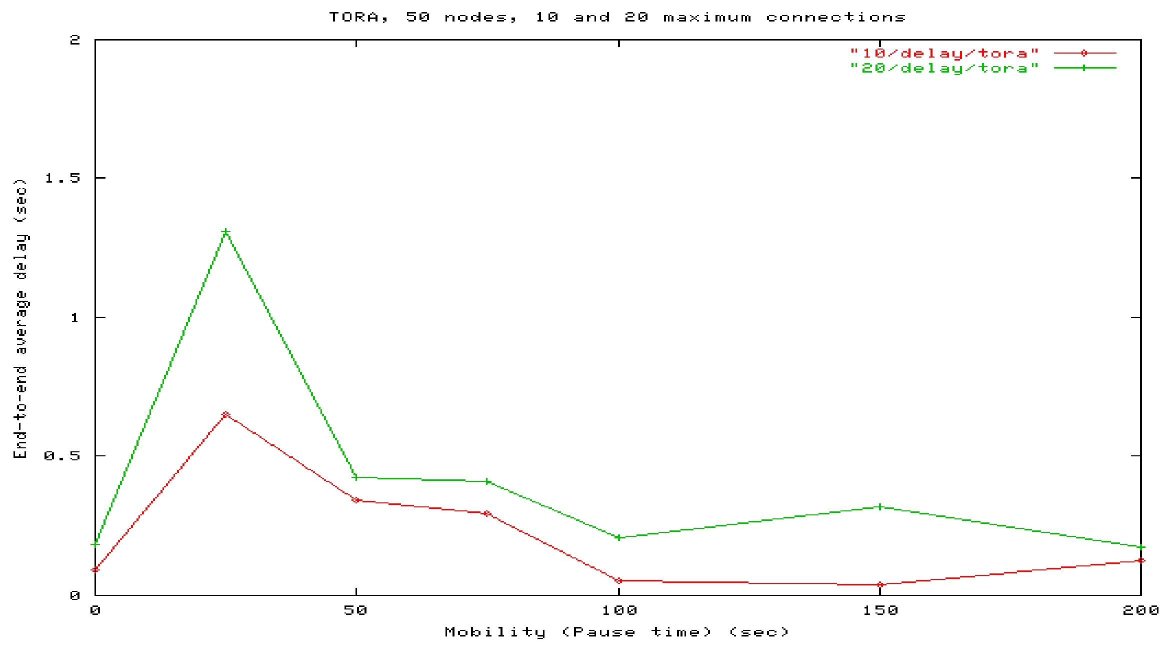


Fig. 5.3.4

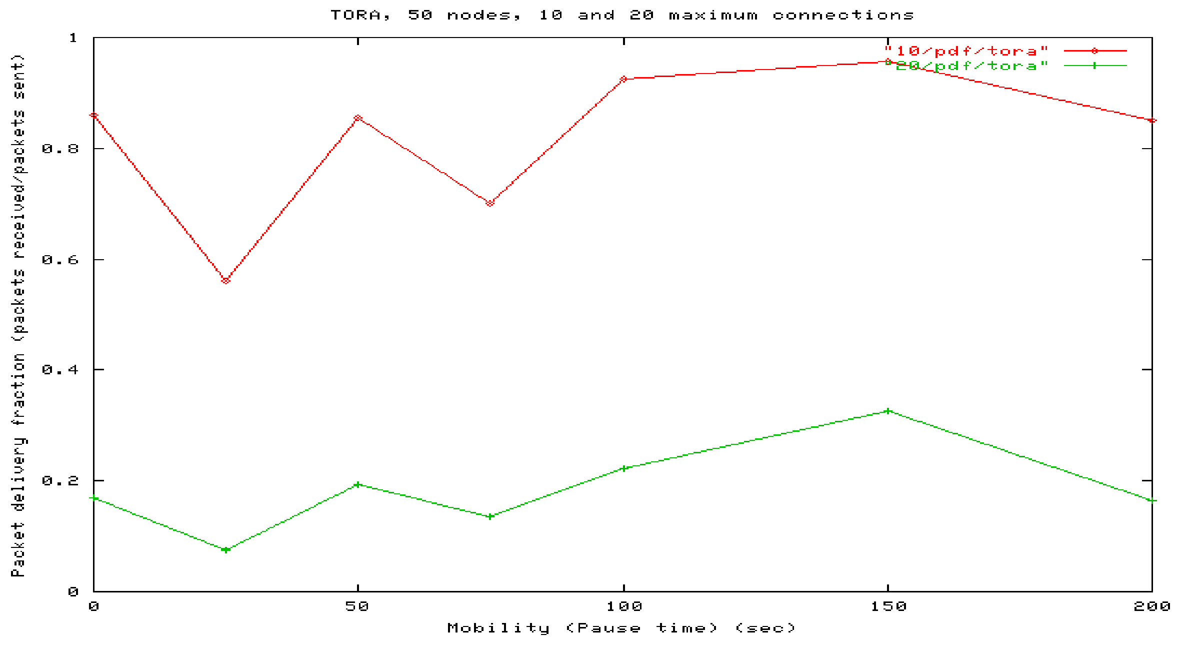
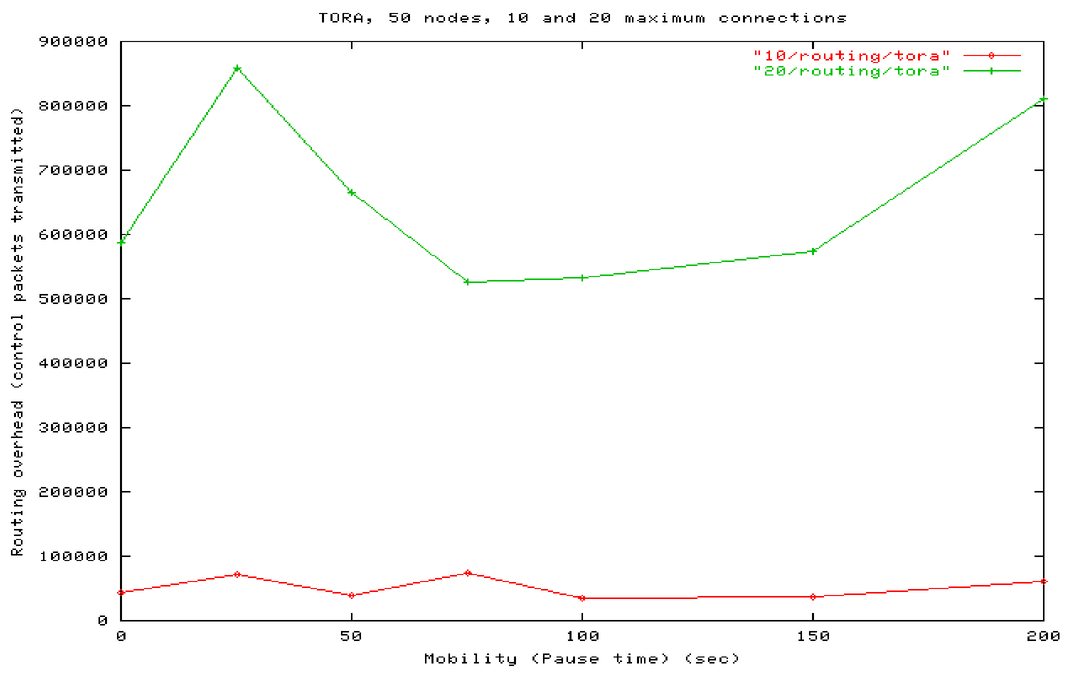


Fig. 5.3.5

Figure 5.3.6 shows the tremendous increase in routing packets, which is also responsible for the congestion. These packets are the ones used to create and maintain routes, multiplied by the number of retransmission and acknowledgement packets IMEP uses to ensure reliable and in-order delivery. To that amount of packets is also added a substantial amount of traffic generated as a result of IMEP's neighbor discovery mechanism, which requires each node to transmit at least 1 *HELLO* packet per *BEACON* period.

Fig. 5.3.6



COMPARISON OF AODV AND TORA

AODV provides less end-to-end average delay compared to TORA. The difference is however bigger, when taking into account the reaction of TORA to congestion, which causes it to drop a major amount of traffic. Therefore the average delay presented for TORA (especially in **figure 5.3.8**) is not accurate, as a lower delivery fraction means that the delay metric is evaluated with fewer samples. The longer the path lengths, the higher the probability of a packet drop, and thus with a lower delivery fraction, samples are usually biased in favor of smaller path lengths and therefore have less delay.

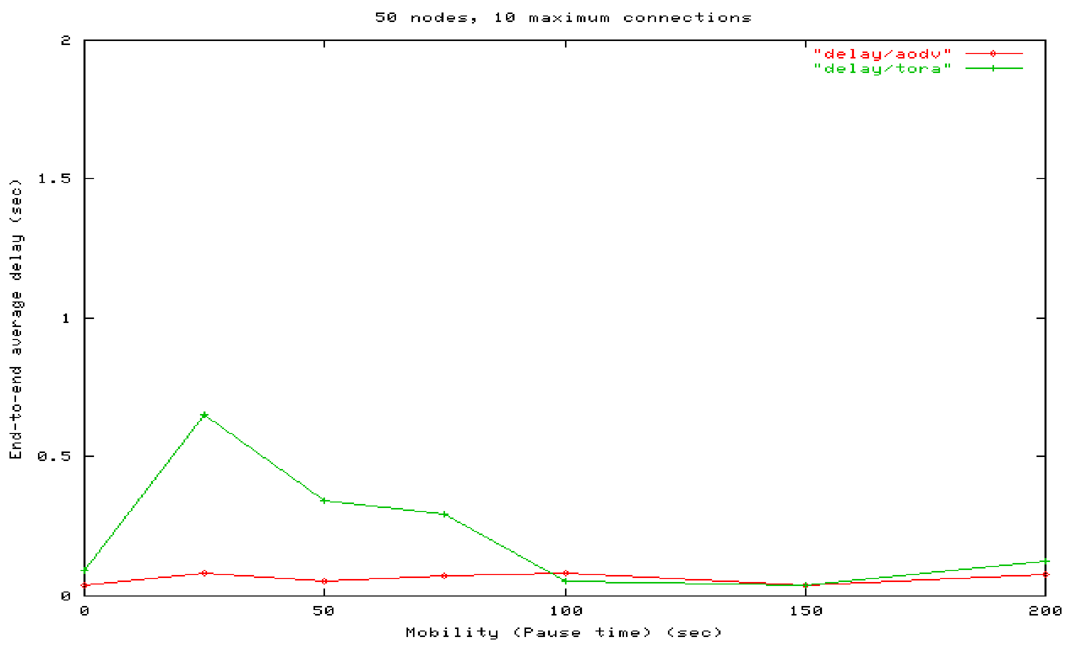


Fig. 5.3.7

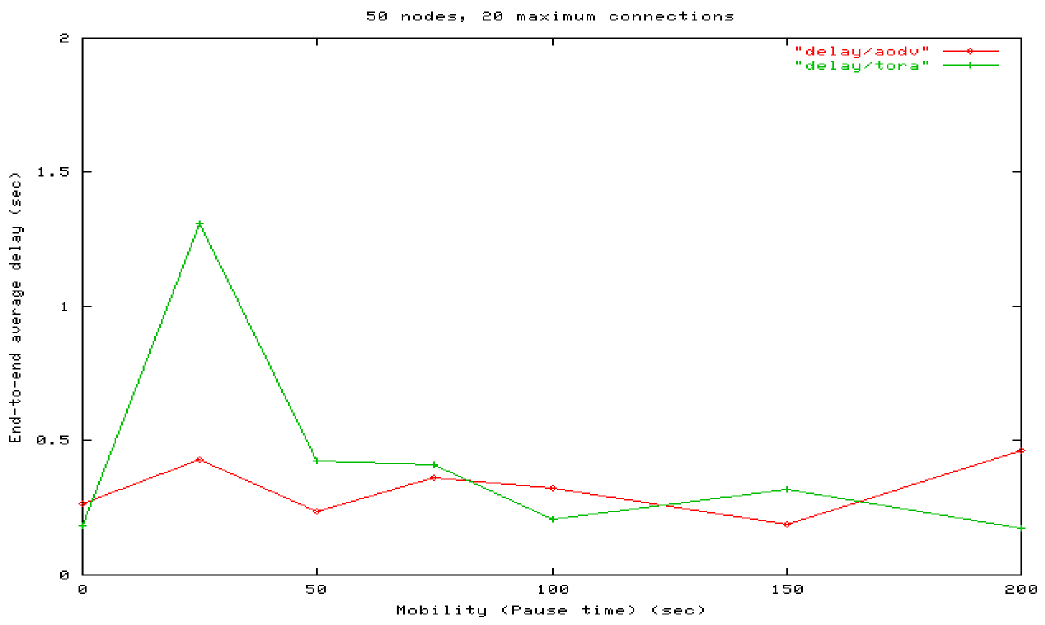


Fig. 5.3.8

Again AODV outperforms TORA in terms of packet delivery. For 10 maximum connections the packet delivery fraction is approaching 1 and is in accordance with results presented in [19]. The size of the packets (512 bytes) does not allow AODV to reach maximum packet delivery for 20 maximum connections, which is the case in [19], where the packets are only 64 bytes long. For 10 maximum connections TORA has relatively lower packet delivery fraction than that presented in [19], due to the bigger packet size, and of course the situation gets much worse for 20 maximum connections, as described earlier. For bigger pause times (less mobility), the packets delivered are -as expected- more, for both protocols. However TORA is not able to recover from the positive feedback loop happening for 20 maximum connections, even when all nodes are stationary.

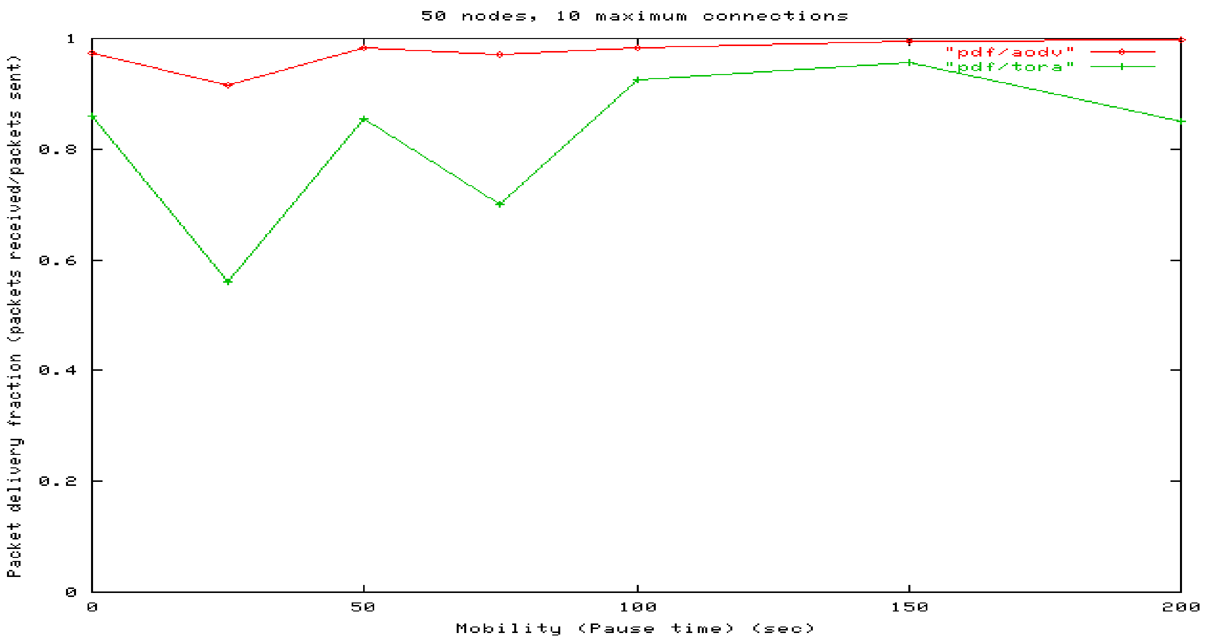


Fig. 5.3.9

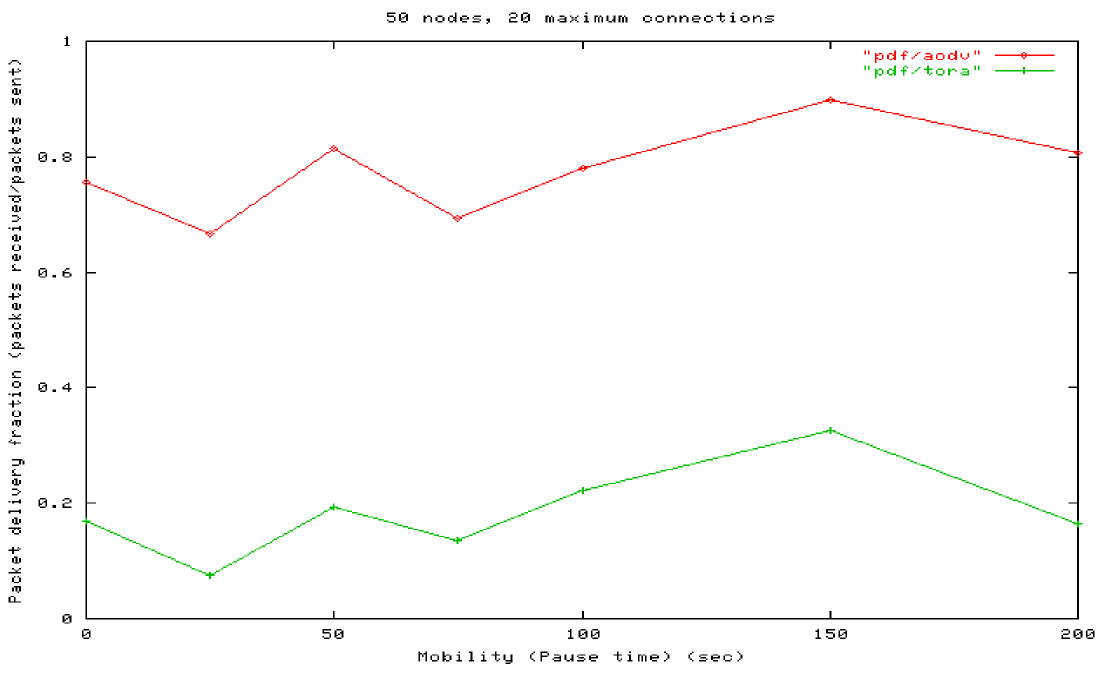


Fig. 5.3.10

The routing packets transmitted give us information regarding the ability of the protocols to function in networks with many nodes, heavy load or low-bandwidth. **Figure 5.3.12** shows that TORA is not suitable for such environments. For high degrees of mobility, both protocols produce a significant amount of control packets, especially for 20 maximum connections, where there are many working routes to be maintained (**figure 5.3.12)**. For TORA the situation then is extreme, as already described. TORA produces less packets than AODV for 10 maximum connections, in contrast to what presented in [19]. This is explained, if we take into account that we use 512-bytes packets, instead of 64, and that IMEP aggregates many TORA and IMEP control messages together into a single packet before transmission,

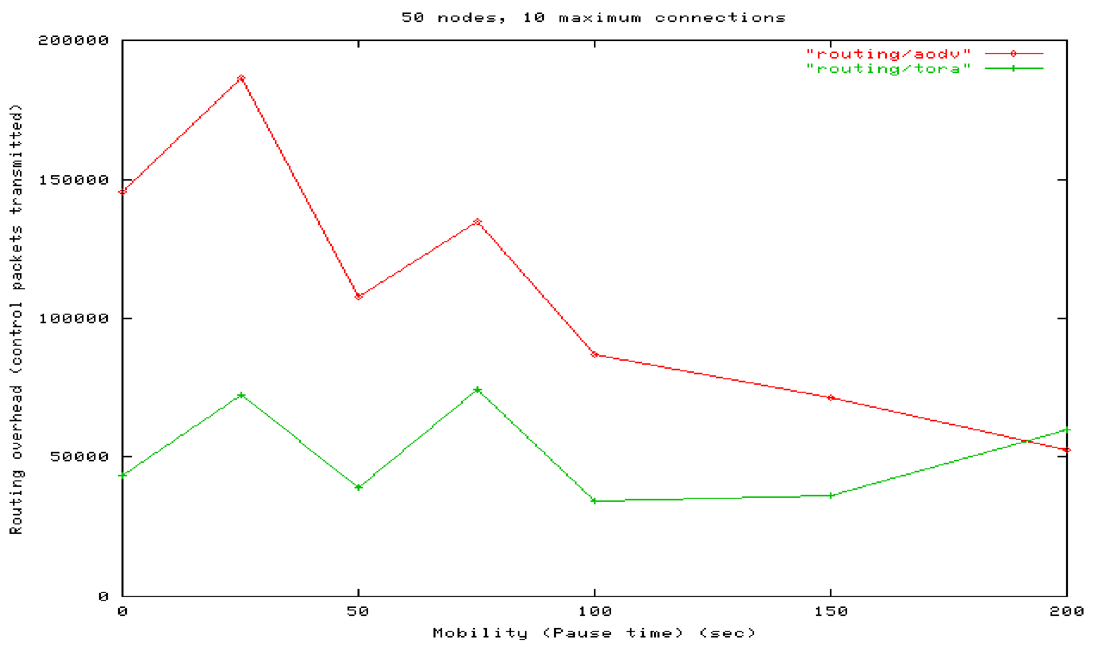


Fig. 5.3.11

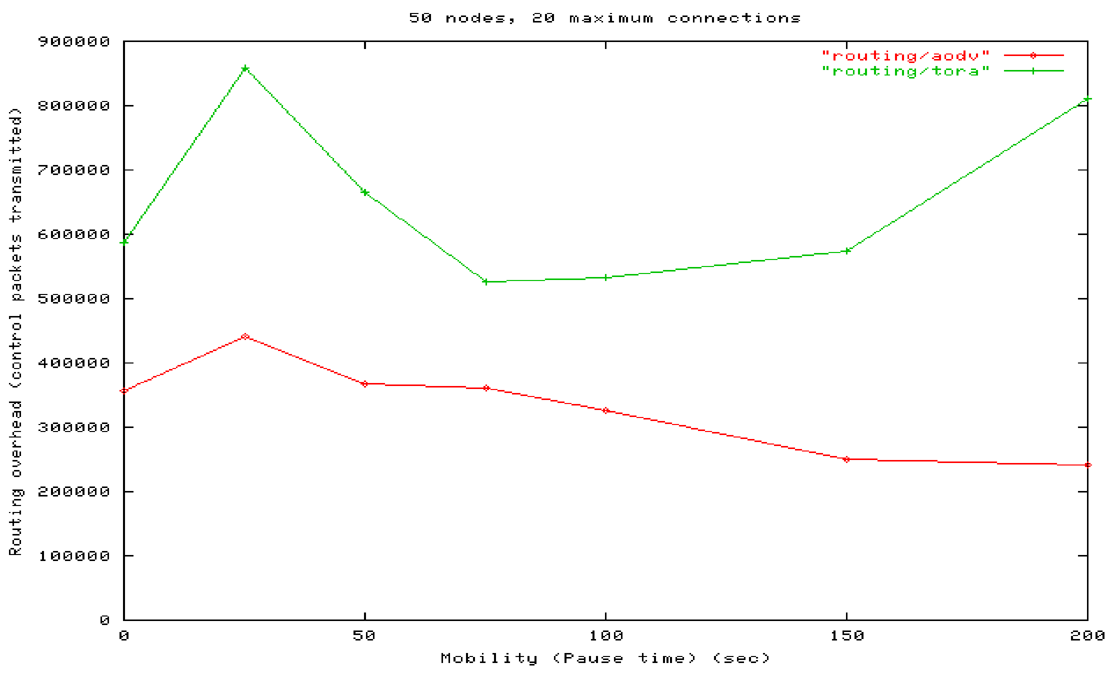


Fig. 5.3.12

5.4 COMPARISON OF SIMULATIONS OF QUALNET AND NS-2 FOR AODV

In order to evaluate both simulators, we also present comparative results of a simulation of a network of 50 nodes, for 10 flows, with the previous setup. The graphs in figures 5.4.1, 5.4.2, and 5.4.3 show the comparative results for the packet delivery fraction, average end-to-end delay, and the number of routing packets respectively.

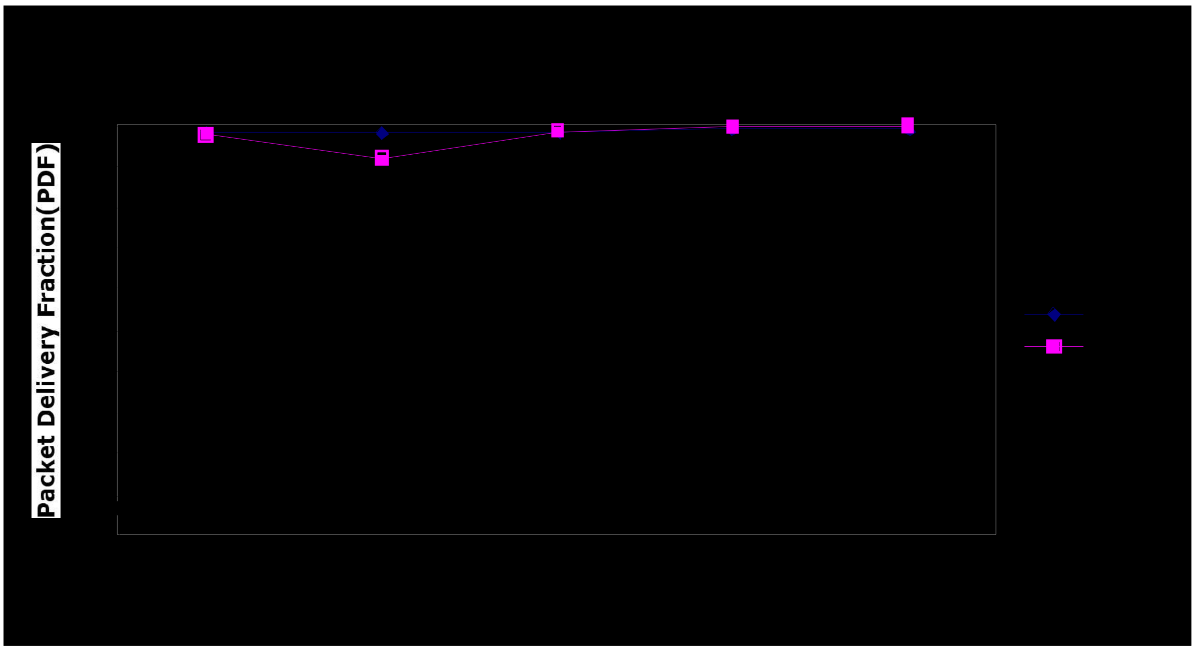


Fig. 5.4.1

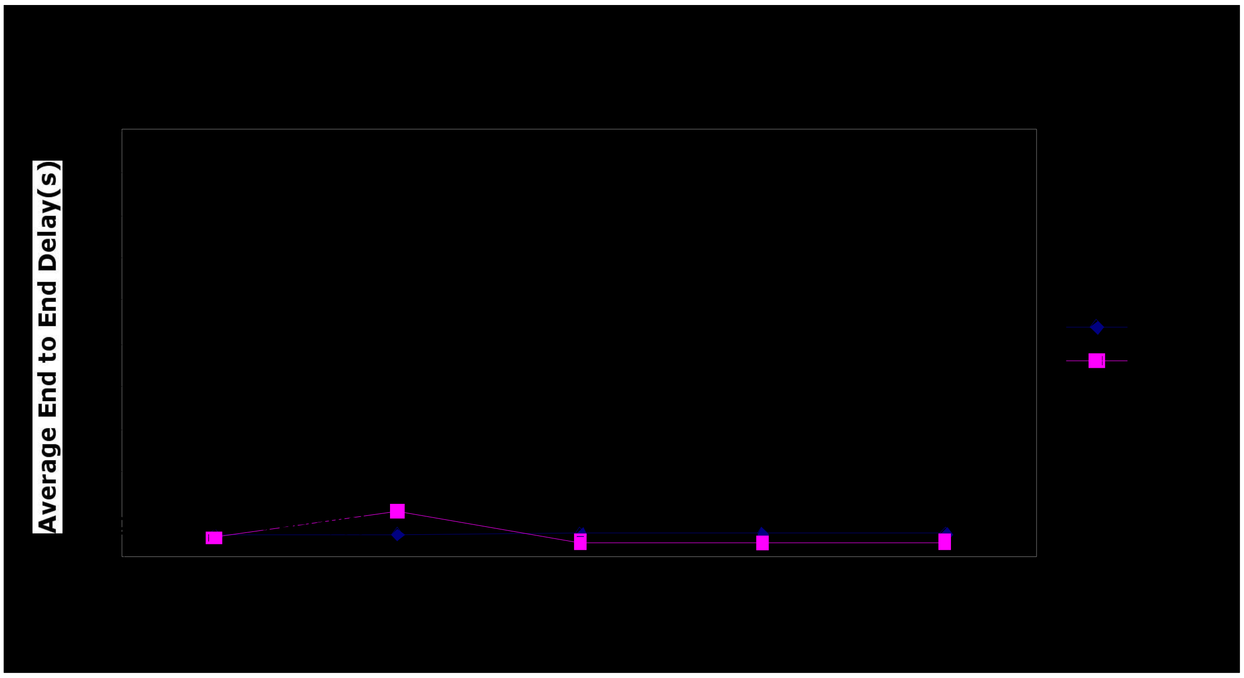


Fig. 5.4.2

As we can see, the results are very similar, proving the simulators to be relatively reliable.

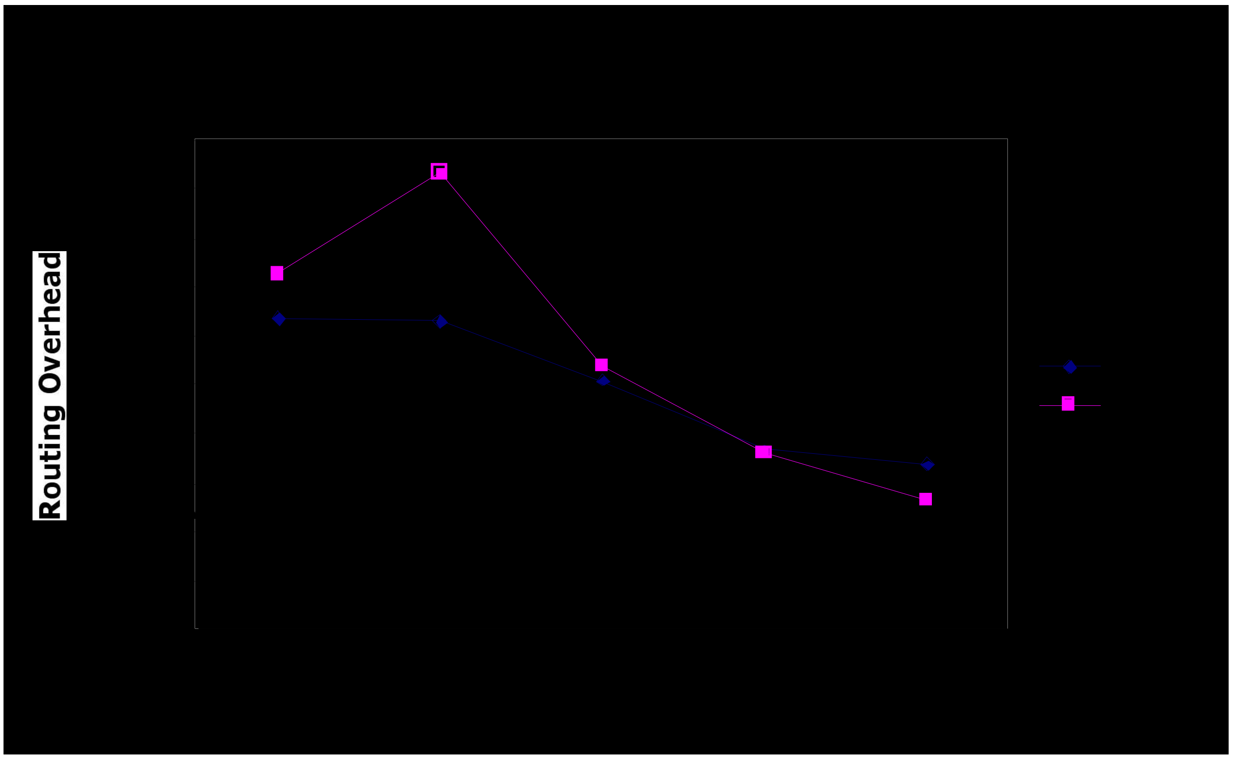


Fig. 5.4.3

The minor differences, such as the ones of figure 5.4.3, can be explained by the fact that the simulations are based on different random scenarios for traffic and topology

6. CONCLUSIONS

In this work we have presented a detailed description and performance comparison of major routing protocols for mobile ad hoc wireless networks. The routing protocols we evaluated were AODV, DSR, LAR and TORA.

AODV, DSR and TORA are on-demand routing protocols. A route discovery process is initiated only when a terminal needs to send data to an intended receiver. These protocols are source-initiated, since routes are discovered when sources need them. All the protocols have some kind of route maintenance mechanisms, which store the routing information until sources do not need it anymore or until routes becomes invalid; that is, some intermediate nodes become unreachable. LAR extends the on-demand approach making use of physical location of the nodes provided by global positioning systems (GPS). This way a significant decrease in routing overhead is achieved.

Using ns-2 we simulated wireless ad hoc networks of 50 nodes, using AODV and TORA as the routing protocols. In order to test the behavior of the two protocols under increased workload, we performed simulations with 10 and 20 maximum connections. AODV managed to handle the increased load, even though more packets are dropped and more routing packets are generated. TORA on the other hand was unable to route that amount of traffic, and dropped the major part of it, while producing a tremendous amount of routing packets. The cause of the congestion collapse lies most probably in a positive feedback loop between the loss of data packets and the creation of routing packets. This observations lead us to conclude that TORA most probably would not be suitable for networks with many nodes, heavy load, or low-bandwidth.

Using QualNet we were able to analyze the performance of AODV, DSR (both of them are Internet drafts) and LAR using large-scale topologies with 500 nodes. At our best knowledge in all previous studies the performance evaluation has been limited to a small number of nodes, usually 50.

The results of the simulations yield some interesting conclusions: AODV suffers in terms of packet delivery fraction (PDF) but scales very well in terms of end-to-end delay. DSR on the other hand scales well in terms of packet delivery fraction (PDF) but suffers an important increase of end-to-end delay, again as compared to the performance achieved in small-scale topologies. The last protocol we evaluated, LAR, seems to scale very well in terms of all metrics used but it requires additional hardware for getting the nodes location.

From the results obtained one can come to the conclusion that both major routing protocols, AODV and DSR, have important drawbacks when it comes to scalability. Therefore this work can motivate further research on improving the current protocols and/or create new ones to meet the challenges of large-scale wireless networks.

REFERENCES

[1] Charles E. Perkins, Elizabeth M. Royer. *"Ad-hoc On-Demand Distance Vector Routing." Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*,

New Orleans, LA, February 1999, pp. 90-100.

[2] Elizabeth M. Royer and C.-K. Toh. *"A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks." IEEE Personal Communications Magazine*, April 1999, pp. 46-55.

[3] C. E. Perkins, E. M. Royer, and S. R. Das, *"Ad Hoc on Demand Distance Vector (AODV) Routing"*, http://www.ietf.org/internet-drafts/draft-ietfmanet-aodv-06.txt, IETF Internet Draft, July 2000.

[4] J. Broch *et al.*, "A Performance Comparison of Multihop Wireless Ad Hoc Network Routing Protocols", *Proc. IEEE/ACM MOBICOM '98*, Oct.1998, pp. 85–97.

[5] Hong Jiang, J. J. Garcia-Luna-Aceves, "Performance Comparison of Three Routing Protocols for Ad Hoc Networks", Computer Communications and Networks, 2001. Proceedings, Tenth International Conference on, 2001; Page(s): 547V554

[6] P. Johansson *et al.*, "Routing Protocols for Mobile Ad-hoc Networks – A Comparative Performance Analysis," *Proc. IEEE/ACM MOBICOM '99*, Aug. 1999, pp. 195–206.

[7] Samir R. Das, Charles E. Perkins, Elizabeth M. Royer and Mahesh K. Marina. "Performance Comparison of Two On-demand Routing Protocols for Ad hoc Networks." *IEEE Personal Communications Magazine* special issue on Ad hoc Networking, February 2001, pp. 16-28.

[8] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva. "Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols". *In Proceedings of the Fourth Annual International Conference on Mobile Computing and Networking (MobiCom'98)*, ACM, Dallas, TX, October 1998.

[9] David B. Johnson, David A. Maltz, and Josh Broch. DSR: "The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks". in Ad Hoc Networking, edited by Charles E. Perkins, Chapter 5, pp. 139-172, Addison-Wesley, 2001.

[10] Young-Bae Ko, Nitin H. Vaidya, "Location-Aided Routing LAR in Mobile Ad Hoc Networks" Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking, Dallas, Texas, Pages: 66 – 75, ISBN:1-58113-035-X

[11] R.R. Choudhury, X. Yang, R. Ramanathan, N. H. Vaidya "Using Directional Antennas for Medium Access Control in Ad Hoc Networks", ACM MobiCom 2002, September 2002.

[12] Thanasis Korakis, Gentian Jakllari, Leandros Tassiulas, "*A MAC protocol for full exploitation of Directional Antennas in Ad-hoc Wireless Networks*", ACM MobiHoc 2003.

[13] V.D. Park and S. Corson, Temporally-ordered routing algorithm (TORA) *version 1* functional specification (Internet-draft), in: *Mobile Ad-hoc Network (MANET) Working Group, IETF* (1998).

[14] B.W. Parkinson and S.W. Gilbert, NAVSTAR: Global Positioning System – Ten years later, Proceedings of the IEEE 71(10) (1983) 1177–1186.

[15] V. Park and M. S. Corson, "*A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks*", *Proc. IEEE INFOCOM '97,* Kobe, Japan (1997).

[16] V. Park and M. S. Corson, "*A Performance Comparison of the Temporally Ordered Routing Algorithm and Ideal Link-State Routing*", Proceedings of IEEE Symposium on Computers and Communication '98, Athens, Greece (June 1998).

[17] Ye, Z., Krishnamurthy S.V., and Tripathi, S.K., "A Framework for Reliable Routing in Mobile Ad hoc Networks", IEEE INFOCOM 2003.

[18] Vikas Kawadia and P. R. Kumar, ``Power Control and Clustering in Ad Hoc Networks." INFOCOM 2003, San Francisco, March 30 - April 3, 2003.

[19] David B. Johnson and David A. Maltz. "Dynamic source routing in ad hoc wireless networks". in Mobile Computing, edited by Tomasz Imilienski and Hank Korth, Chapter 5, pp. 153-181, Kluwer Academic Publishers, 1996.

[20] Samir R. Das, Charles E. Perkins and Elizabeth M. Royer. "Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks". INFOCOM 2000, Tel Aviv, March 26 -30, 2000.

[21] S. Corson, S. Papademetriou, P. Papadopoulos, V. Park, and A. Qayyum. "An Internet MANET Encapsulation Protocol (IMEP) Specification". draft-ietf-manet-imep-spec02.txt. IETF, August 1999.

[22] John Bellardo Jenny Fang Greg Hamerly, "Comparison of Ad Hoc Network Routing Protocols", UCSD, AI lab, November 1999.