MOBILE AD HOC NETWORK'S ROUTE MAINTENANCE ANALYSIS IN REACTIVE ROUTING PROTOCOLS

By
Enas Khaled Al-Tarawneh

Supervisor
Dr. Wesam Al-Mobaideen

تعتمد كلية الدراسات العليا
هذه النسخة من الرسالة
التوقيع ... التاريخ ٢٠٠٥/١٠/٢٧

Co – Supervisor:
Dr. Emad Qaddoura

This Thesis was Submitted in Partial Fulfillment of the Requirement for the Master's Degree of Science in Computer Science

Faculty of Graduate Studies
The University of Jordan

December, 2004

This Thesis (Mobile Ad Hoc Network's Route Maintenance Analysis in Reactive Routing Protocols) was successfully defended and approved on --------------------

Examination Committee	**Signature**
Dr. Wesam A. Al-Mobaideen, Chairman Assist. Prof. of Wireless Mobile Networks	------------------------------
Dr. Emad A. Qaddoura, Member Assist. Prof. of Wireless Mobile Networks	Emad ------------------------------
Dr. Ahmad A. Sharieh, Member Assoc. Prof. of Parallel Processing	------------------------------
Dr. Imad Kh. Salah, Member Assist. Prof. of Complex Systems and Networks	------------------------------
Dr. Aymen I. Zreikat, Member Assit. Prof. of Wireless Mobile Networks (Mutah University)	------------------------------

تعتمد كلية الدراسات العليا
هذه النسخة من الرسالة
التوقيع .... التاريخ ٢٧/٨/٢٠٠٥

ACKNOLEGMENTS

First of all, I thank God for his grace and blessings.

I would like to thank my supervisors, Dr. Wesam Al-Mobaideen and Dr. Emad Qaddoura for their guidance, supervision, and the many insightful discussions we had that helped me clarify my thoughts and present this thesis in a suitable way.

I would like to thank the teachers in the King Abdullah the second for Information and Technology Department for the many things I've learned from them. I'd like to thank my class mates for their support and helpful conversations.

Finally, I would like to thank my parents, my brother and my sisters for their support, care and understanding.

List of Contents

List of Tables

List of Figures

List of Abbreviation

ACK	ACKnowledgment
AODV	Ad-hoc on-demand Distance Vector Routing
API	Access Point Interface
ARM	Anticipated Route Maintenance
CBR	Constant Bit Rate
DCM	Distributed Coordination Function
DSDV	Destination-Sequenced Distance-Vector Routing
DSR	Dynamic Source Routing
ERU	Early Route Update
GloMoSim	Global Mobile system Simulator
IP	Internet Protocol
LAN	Local Area Network
LL	Link Layer
MAC	Media Access Control
MANET	Mobile Ad Hoc Network
NPDU	Network Protocol Data Units
NS2	Network Simulator
OSI	Open System Interconnections
PEH	Promiscuous Error Handling
PARSEC	PARallel Simulation Environment for Complex Systems
RRQ	Route Request Packet
RRP	Route Reply Packet

ERR	Route Error packet
SHARP	Sharp Hybrid Adaptive Routing Protocol
TCP	Transmission Control Protocol
TORA	Temporally Ordered Routing Algorithm
WEN	Wider Error Notification
WMP	Without Multiple Paths
VINT	Virtual InterNetwork Testbed
UCLA	University of California, Los Angeles

ERR	Route Error packet
SHARP	Sharp Hybrid Adaptive Routing Protocol
TCP	Transmission Control Protocol
TORA	Temporally Ordered Routing Algorithm
WEN	Wider Error Notification
WMP	Without Multiple Paths
VINT	Virtual InterNetwork Testbed
UCLA	University of California, Los Angeles

MOBILE AD HOC NETWORK'S ROUTE MAINTENANCE ANALYSIS IN REACTIVE ROUTING PROTOCOLS

By
Enas Khaled Al-Tarawneh

Supervisor
Dr. Wesam Al-Mobaideen

Co – Supervisor:
Dr. Emad Qaddoura

ABSTRACT

The recent advances in the area of mobile computing present wireless networks that can be deployed instantly without any fixed infrastructure. Such networks, also referred to as ad hoc networks, can be utilized in military, education, business and many other environments. Reactive routing protocols are Ad-Hoc protocols that consist of two basic mechanisms, route discovery and route maintenance. These protocols employ caching mechanisms that cache routes to arbitrary destinations in the network, for future use. When a stale cached route (invalid route) is used, route maintenance eventually occurs. During route maintenance, packets may drop causing throughput degradation. The time taken to maintain the route is unnecessary lost time that increases latency.

DSR relies mostly on route maintenance to improve the validity of cached routes. On the other hand, AODV mostly relies on route discovery for the same purpose. Because of this, AODV outperforms DSR with small transfers. In order to reduce the effect of route maintenance caused by invalid routes on DSR's throughput and delay, two solutions are proposed. One solution is to increase the validity of multiple routes using WEN, another is to apply one factor of the two factors of AODV's caching mechanism (one entry per destination, preferring fresher routes) on DSR by disabling the multiple path feature of DSR. In addition to that, we study the suitability of integrating a caching mechanism within route maintenance.

The results show that applying a caching mechanism which mostly relies on route maintenance, such as DSR, is not suitable for all environments, rather, environments with long-lived connections. A caching mechanism that places more effort on route discovery, such as AODV, is a more suitable caching mechanism with small transfers. Applying one factor of the AODV caching process, preventing multiple paths, on DSR provides an improvement on throughput and a 30% improvement on delay caused by invalid routes on all small-lived connections. The remaining factor effecting DSR's performance is its inability to prefer fresher routes. It is highly probable that DSR would further improve if it preferred fresher routes.

Introduction

Mobile computing has become extremely popular over the recent years. Improved computer-based applications are being offered to an increasing part of the population due to the nonstop miniaturization of mobile computing devices and the surprising rise of processing power existing in mobile laptop computers.

A Mobile Ad Hoc Network (MANET) is an autonomous system of mobile nodes (simultaneously hosts and routers) connected by wireless links. The mobile nodes are free to move randomly and organize themselves arbitrarily. Thus, the topology of the network may change rapidly and unpredictably. Such a network may operate in a standalone fashion, or may be connected to the larger Internet [12]. Developing a dynamic routing protocol that is capable of finding routes between two communicating nodes efficiently is a basic concern in designing ad hoc networks. This routing protocol must adapt to the reasonably high degree of node mobility that frequently causes significant and unpredictable change in the network topology.

Example applications of ad hoc network applications include students using laptop computers to participate in an interactive lecture, business associates sharing information during a meeting, home networking, embedded computing applications, soldiers relaying information on the battlefield, and emergency services such as disaster relief personnel coordinating efforts after an earthquake [15].

The ad hoc routing protocols can be divided into two categories: proactive and reactive routing based on when and how the routes are discovered. Further elaboration of these two categories can be found in Section 2. In proactive routing protocols, each node maintains one or more tables containing routing information to every other node in the network. It has the advantage of communication with arbitrary destinations experience minimal initial delay, but suffers from additional control traffic to constantly update stale route entries. Destination-Sequenced Distance Vector (DSDV) [22] is an example of a proactive protocol.

Reactive protocols take a lazy approach to routing. The routes are created as and when required. These protocols consist of two basic mechanisms: route discovery and route maintenance. These two mechanisms are described in detail in Section 2.2. Nodes in reactive protocols only maintain info about active routes to specific destinations. Reactive protocols may use far less bandwidth for maintaining the route tables at each node, but the latency for many applications will increase because a route will have to be acquired before communication can begin. Ad Hoc On-demand Distance Vector (AODV) [20] and Dynamic Source Routing (DSR) [7] are examples of reactive protocols.

Previous studies [2][5][13] have shown that reactive (on-demand) routing protocols are more efficient than proactive protocols in terms of high throughput and low overhead especially with node mobility. Reactive protocols (e.g. DSR, AODV) robust route maintenance mechanisms that deal with frequent route breaks and avoid unnecessary route discovery floods by employing efficient route caching mechanisms [30]. A caching mechanism may reduce latency and overhead by providing immediate routes to arbitrary destinations across the network. This is only true if the routes in the caches actually reflect the network topology.

1 Motivation

Researchers of [14] [30] show that the performance of a reactive protocol can highly depend on the employed caching mechanism. The efficiency of the caching mechanism is an essential factor. One other factor is the suitability of the basic mechanism (i.e.

relationship also causes DSR to outperform AODV with long-lived connections. We apply a partial imitation of the AODV caching mechanism on DSR for this purpose.

This thesis will show the effect of caching mechanisms on the route maintenance mechanism and the effect of route maintenance on caches by applying modifications to DSR, one of two well known reactive routing protocols, AODV and DSR.

This thesis provides the following contributions:

1. A study defining the suitability of applying a caching mechanism on a reactive routing protocol by integrating that mechanism within the route maintenance process. A caching mechanism known as Wide Error Notification (WEN) [14] is used for this purpose. The results show that applying a caching mechanism with the specifications of WEN is not suitable for all environments, rather, environments with long-lived connections.

2. A comparative analysis featuring the effect of the caching mechanism "Wide Error Notification" on the DSR's route maintenance process. This analysis will compare the results of DSR, the modified DSR and AODV. The results show that WEN has no effect on improving the performance of original DSR with short-lived connections. WEN had a very similar behavior to original DSR on route maintenance in regard to small transfers. WEN's effect on route maintenance caused DSR to have better performance than AODV with long-lived connections, and caused DSR to have a worse performance than AODV with small lived connections.

3. A comprehensive evaluation of a DSR protocol that prevents multiple cached paths to the same destination, partially imitating the AODV caching mechanism, in terms of a specific set of performance metrics related to route maintenance. This evaluation is compared to a similar evaluation of the AODV protocol in terms of the same set of metrics. Preventing multiple paths in DSR helped improve DSR's performance with

small-lived connections by reducing the number of invalid routes (route maintenance). However, this improvement is small. There is an improvement on throughput and a 30% improvement on delay caused by invalid routes. Our study also shows that the greater effect is due to another factor involved in the caching mechanism; DSR's inability to prefer fresher routes.

3 The GloMoSim simulation package (GloMoSim 2.03)

The GloMoSim (for Global Mobile system Simulator) simulation package provides a number of functional capabilities in order to simulate the behavior of the AODV and DSR protocols to adapt to specific scenarios. It also provides means to retrieve and analyze the simulation results, and to modify these protocols to reflect additional specifications. Our choice of GloMoSim was based on a number of characteristics;

- Extensible: the simulation package must be easy to extend if we are to add new functionality.
- Flexibility: the simulation package must be easy to modify if we are to manipulate its functionality.

- Scenario generation: the simulation package can easily create traffic patterns, topologies and dynamic events.
- Protocol availability: the original AODV and DSR protocols must have been already implemented in the simulation package, so we can immediately apply our modifications without having to build them ourselves.
- Scalability: A simulation package using parallel simulation to simulate a model can significantly reduce execution times even with large network models.

GloMoSim is a scalable simulation library for wireless network systems built using the PARSEC simulation environment [34]. Table 1, as presented in [1], illustrates the GloMoSim models available at each major layer. Unlike existing network simulators such as OPNET [36] and The Network Simulator (NS) [3], GloMoSim has been developed and designed with a goal of simulating very large network models. Network models in GloMoSim can scale up to a million nodes using parallel simulation to significantly reduce execution times of the simulation model.

GloMoSim is designed using a layered approach as most network systems adopt a layered architecture. This layered approach is similar to the Open System Interconnections (OSI) seven layer network architecture. Between different simulation layers, simple Access Point Interfaces (APIs) are defined. This layered design doesn't only provide easy integration of actual operational code, but is also ideal for a simulation model as it has already been validated in real life and no abstraction is introduced. This layered design also allows the rapid integration of models developed at different layers by different people.

Table 1: Models currently in the GloMoSim library.

Layer:	Models:
Physical (Radio propagation)	Free space, Rayleigh, Ricean, SIRCIM
Data Link(MAC)	CSMA, MACA, MACAW, FAMA, 802.11
Network (Routing)	Flooding, Bellman-Ford, OSPF, DSR, WRP,AODV,ZRP
Transport	TCP, UDP
Application	Telnet, FTP, CBR

PARSEC (for PARallel Simulation Environment for Complex systems) developed by the Parallel Computing Laboratory at UCLA (for University of California, Los Angeles) is a C-based simulation language, for sequential and parallel execution of discrete-event simulation models [1]. It can also be used as a parallel programming language.

OPNET Modeler from OPNET Technologies Inc. [36] is a network development environment tool with object-oriented modeling approach and graphical editors. Its Model Library has dozens of models of network protocols, technologies and applications including Wireless Local Area Networks (LANs) (IEEE 802.11).

The Network Simulator is a discrete event simulator developed by the University of California at Berkeley and the Virtual InterNetwork Testbed (VINT) project [3]. It provides substantial support for simulation of TCP, routing, and multicast protocols over wired and wireless (local and satellite) networks. The disadvantage of NS2 is that it is a large system with a relatively steep initial learning curve.

4 The Organization of Thesis

This thesis focuses on the route maintenance process, its relationship with performance and invalid cached entries. Chapter 2 is a survey about existing wireless ad hoc networks routing protocols. Related work are presented in Chapter 2 and a brief discussion of this work. Chapter 3 presents the simulation problem in the small transfer environment. Two proposed solutions, results of simulating suitable simulation scenarios, and discussion and analysis of the presented results are discussed. Chapter 4 proposes further analysis of the relationship between route maintenance and invalid cache entries with longer transfers. Simulation, results and discussion are also presented in this chapter. Chapter 5 draws the conclusions and suggests directions for future research.

Background & Related Work

1 Overview

Mobile wireless networks are becoming increasingly popular as Mobile hosts and wireless networking hardware have become highly obtainable. Mobile wireless networks can be categorized into two main categories. The first is known as infrastructure based networks. These networks have fixed and wired gateways. A mobile host communicates with a fixed bridge in the network (base station) within its communication range. During communication, the mobile unit is free to move geographically. When a mobile host goes out of range of one base station, it can connect with another base station. This is called handoff [28] [35].

The second category is known as infrastructureless based networks also referred to as ad hoc networks. In these networks nodes are mobile and can be connected dynamically. All nodes of these networks act as routers and take part in discovery and maintenance of routes to other nodes in the network [23].

In wired networks, routing protocols generally use distance vector or link state algorithms, and both require a periodic broadcasting of routing advertisements [28]. Distance vector routing operates by having each router maintain a table giving the best known distance to each destination and which line to get there. These tables are updated by exchanging information with the neighbors. On the other hand, link state routing operates by having each router discover its neighbors, measure the metric, such as cost and delay, to each neighbor, send this info to all other routers and finally compute the shortest path to other routers. Each router has a complete view of the network formed from the link information obtained by all routers.

Conventional routing protocols pass detailed topology information between hosts, thus, they are not effective in an ad hoc network due to the high rate of topology change.

These protocols have not been designed specifically to provide the kind of dynamic, self-starting behavior needed for ad hoc networks. In ad hoc networks, the information about topology maintained in the routing tables is out dated early after it being stored and the propagation of this information is too slow to be accurate [28].

In ad hoc wireless network, a routing protocol must store and forward packets over multiple-wireless-hop paths [23] [15]. A node may not have direct access to another node in MANET due to power constraints that limit the nodes transmission range. Frequency reuse [28] and channel effects and frequent topology change in ad hoc networks would require storing packets waiting for a suitable transmission time. MANETs have three main salient characteristics: Dynamic topologies, Energy-constrained operation and Bandwidth-constrained [25].

2 Ad Hoc Routing Protocols

Several ad hoc routing protocols have been developed due to the inability of conventional routing protocols to address ad hoc conditions. Mobility, device heterogeneity, bandwidth and limited battery power make the design of adequate routing protocols a major challenge [4]. A variety of wireless ad hoc routing protocols have been proposed in the recent years [15].

2.1 Proactive Routing Protocols

Proactive (also known as table-driven) are routing protocols that attempt to maintain up-to-date routing information from each node to all other nodes in the network. To store routing information these protocols require each node to keep one or more tables that maintain a consistent view of the network. These routing protocols differ in the technique used to distribute the topology change information across the network and the number of routing-related tables preserved [13]. The following section will discuss one of the existing table-driven ad hoc routing protocols.

2.1.1 Destination-Sequenced Distance-Vector Routing (DSDV)

The Destination-Sequenced Distance-Vector Routing (DSDV) [22] is based on the classical Bellman-Ford routing algorithm [22]. The additional feature in DSDV over traditional distance vector protocols is that it stamps each route table entry with a sequence number to order the routing information and guarantee loop-free routes.

Each node in DSDV maintains a routing table containing all available destinations. Each entry defining a destination has the following attributes: the next hop, the number of hops to the destination, and a sequence number, originated by the destination node.

To maintain table consistency DSDV uses both time-driven and event-driven routing updates. When network topology changes are detected event-driven routing updates are used in order to quickly propagate the routing information. To ease the potentially large amount of network traffic the routing table updates are sent in two ways: a "full dump" or an incremental update.

The "full dump" type of packets may need multiple Network Protocol Data Units (NPDU) to hold all the available routing information; to reduce the amount of traffic generated, the incremental type of packets only carries the information changed after the last full dump and should require only one NPDU. If a link to the next hop is broken, due to mobility or any other reason, infinity metric and an updated sequence number is assigned to every route through that next hop.

Sequence numbers are usually assigned by the source nodes and are even numbers; odd numbers are used for sequence numbers that define infinity metrics. A broadcast route update is triggered when a node receives infinity metric, and it has an equal or later sequence number with a finite metric. The route with infinity metric will soon be replaced by the new route.

The route labeled with the highest sequence number is used. If two routes have the same sequence number then the route with the best metric is used. For newly received routes the metrics are incremented by one hop since packets require one more hop to reach their destination. DSDV uses settling time to prevent fluctuations of routing table updates caused by many independent nodes transmitting these updates asynchronously. The settling time decides how long to wait before advertising new routes.

The DSDV protocol keeps track of routes very close to optimal and guarantees loop-free routes to each destination in the network. Nodes are required to transmit periodic routing update packets which are broadcast all over the network. The size of the routing tables and the bandwidth required to update them grows as the number of nodes in the network increases, causing extremely large communication overhead. In a number of simulations done by [2], it is obvious that DSDV delivers all data packets when node mobility rate and speed are low, as node mobility increases it fails to converge.

2.2 Reactive Routing Protocols

Reactive routing is the freshest candidate in the set of scalable wireless routing schemes [24]. The dominant candidates of reactive protocols are Dynamic Source Routing (DSR) [8], Ad hoc On-demand Distance Vector Routing (AODV) [19] and Temporally Ordered Routing Algorithm (TORA) [17]. This kind of routing is based on a query-reply approach. This approach is formulated into two fundamental mechanisms on which reactive routing stands [24] [28]. The first mechanism, known as the route discovery process, is initiated only when a node requires a route to a destination and doesn't have one in its cache. The objective of this process is to find a route or all possible routes to a destination. The second mechanism, the route maintenance process, maintains the route once it has been established until either the route is no longer desired or the destination becomes unreachable along every path from the source. These two fundamental mechanisms are further described as follows:

A. Route Discovery

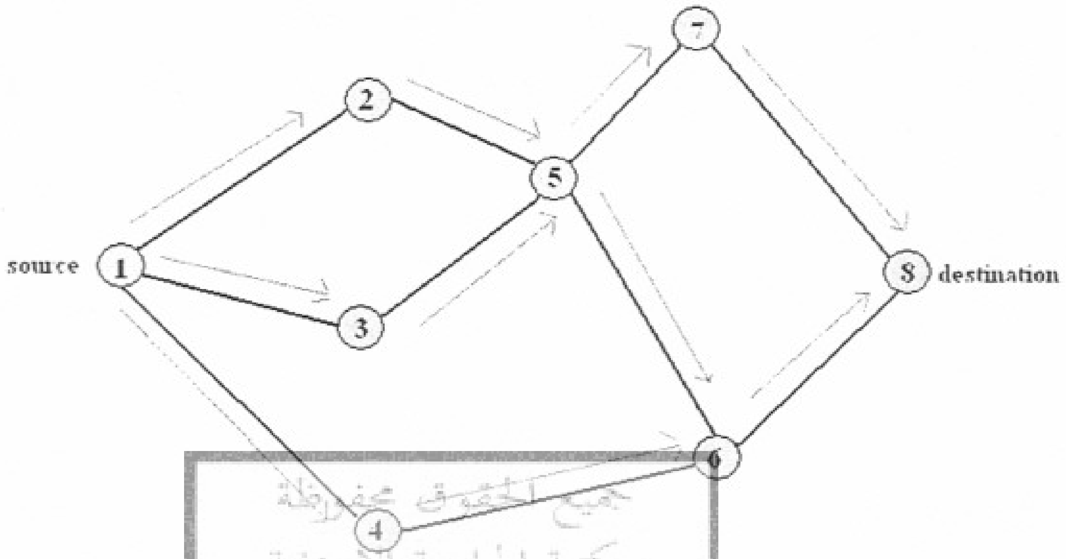
In the Route discovery phase, a broadcast packet known as the route request packet (RRQ) is sent by the source node to locate the destination node in the network. Nodes that have a valid route for the destination node initiate a route reply packet (RRP) back to the source node after receiving a (RRQ). The RRP usually contains a list of nodes along the path from the source node to the destination node [24].

B. Route Maintenance

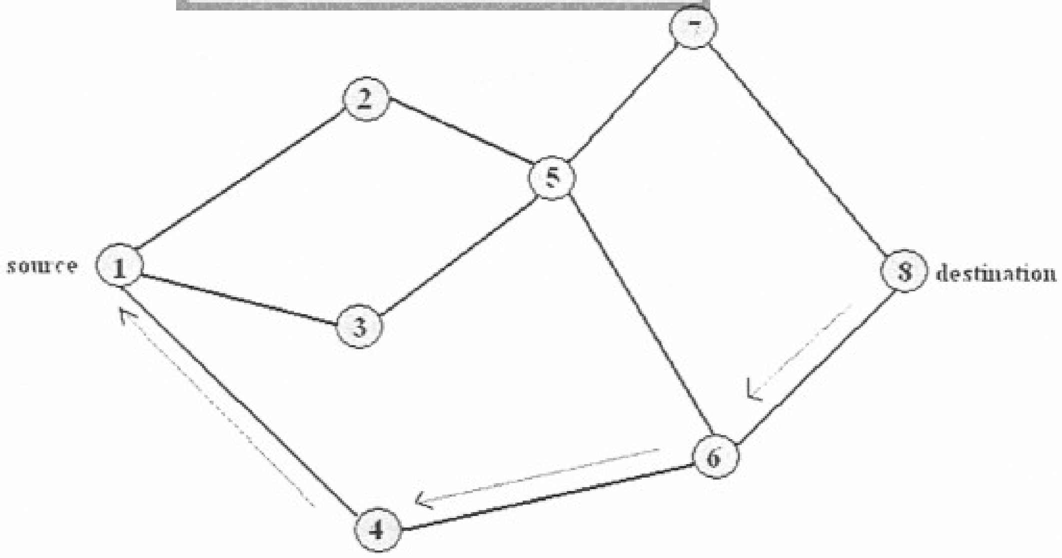
As described in [24], the Route Maintenance phase ensures that the routes stored in the Route Cache are valid. The node initiates a route error packet (ERR) and sends it to the source after the data link layer (LL) of a node detects a broken link. Several acknowledgement mechanisms may be used for error (link break) detection, such as the Acknowledgement (ACK) packet for successful packet transmission, link detection mechanism in 802.11 and others. The source searches its route cache and deletes the routes containing the broken link after receiving an ERR. The source will then either attempt to use other alternate routes in its cache if the protocol supports multi-path routing or invoke another route discovery if the protocol supports single-path routing. Section 3 provides more elaboration on route maintenance.

2.2.1 Ad hoc On-demand Distance Vector Routing

Ad hoc On-demand Distance Vector Routing (AODV) [19] is a development on the DSDV algorithm discussed in section 2.1.1. The goal of AODV was to reduce the number of broadcasts by creating routes on-demand as opposed to DSDV that maintains the list of all the routes. AODV utilizes a route table that is used to store the destination address and the next hop address as well as a destination sequence number. The objective of the sequence number is illustrated later in this section.



a) Route Request Packet propogation



b)Route Reply reverse path

Figure 1: Route Discovery in AODV

Similar to all reactive protocols a source node in AODV finds a path to the destination by broadcasting a route request packet. Each neighbor in turn will broadcast the packet to its neighbors till it reaches the destination (as shown in Figure 1 part a) or till it reaches an intermediate node that has route information about the destination in its route table. A route request packet that has already been seen by a node will be discarded.

When a node forwards a route request packet, it also records the node from which the first copy of the request came in its route table. A reverse path for the route reply packet is constructed using this information. AODV uses only bi-directional links because the route reply packet follows the reverse path of route request packet. Each node down the path enters the forward route into its route table as the route reply packet follows the tracks back to the source (as shown in Figure 1 part b). Sequence numbers are used in the route request packet to ensure that the routes are loop free and to make sure that the intermediate nodes prefer fresher routes by replying to route requests with the latest information only.

Route maintenance in AODV normally requires that each node periodically transmit a HELLO message, with a default rate of once per second. Failure to receive three successive HELLO messages from a neighbor is taken as an indication that the link to the neighbor in question is broken. The AODV specification alternatively suggests that a node may use physical layer or link layer methods to detect link breakages to neighbor nodes [19]. When a link goes down, any upstream node that has recently forwarded packets to a destination using that link is notified via an UNSOLICITED ROUTE REPLY containing an infinite metric for that destination. Upon receipt of such a ROUTE REPLY, a node must acquire a new route to the destination using Route Discovery [15] [19] [21].

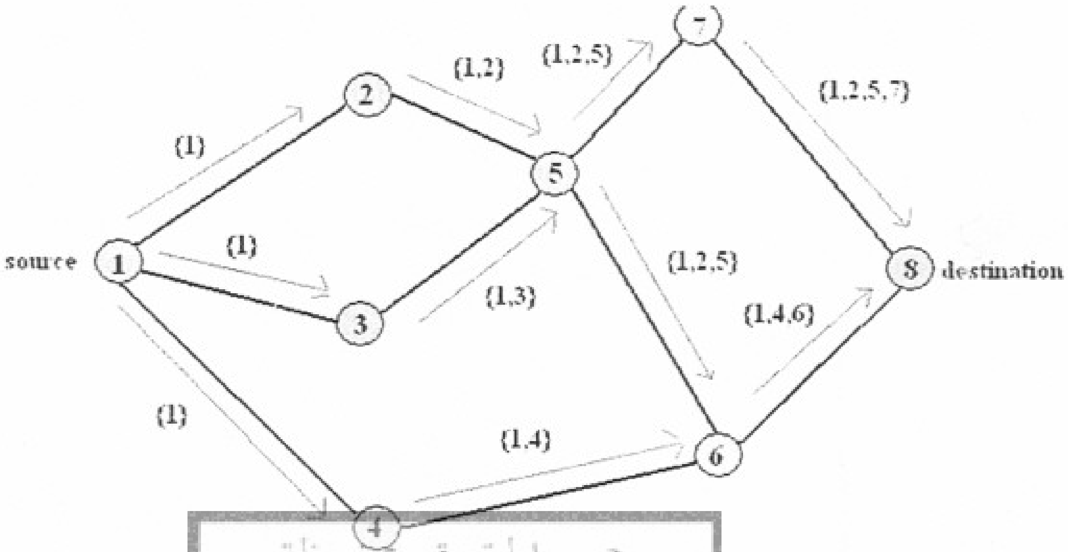
2.2.2 Dynamic Source Routing Protocol

The Dynamic Source Routing Protocol [7] is a source-routed on-demand routing protocol. Nodes maintain route caches that contain the source routes that it knows of. The route cache entries are updated as and when the node learns about new routes.

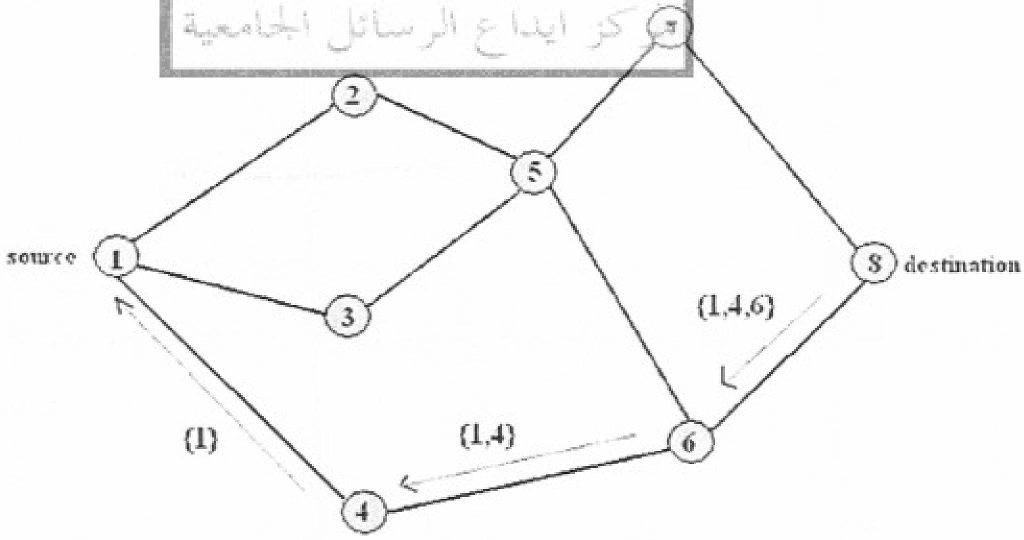
When a node wishes to send a packet to a destination, it checks its route cache for a route to the destination. If a route to the destination exists, then it uses this route to send the packet. However, if the node does not have such a route, then a route discovery process is initiated by broadcasting a route request packet.

The route request packet contains the address of the source and the destination, and a unique identification number. When receiving the request packet each intermediate node checks to see if it knows of a route to the destination. If it does not, it adds its address to the route record of the packet and broadcasts the packet to its neighbors. A node only processes the route request packet if it has not previously seen the packet and its address is not present in the route record of the packet to limit the number of route requests propagated.

When the destination or an intermediate node with current information about the destination receives the route request packet a route reply is generated [6]. A route request packet contains, in its route record, the sequence of hops taken from the source to the destination node. The route record is formed as the route request packet propagates through the network (as shown in Figure 2 part a). If the route reply is generated by the destination then it places the route record from route request packet into the route reply packet (as shown in Figure 2 part b). However, if the node generating the route reply is an intermediate node then it attaches its cached route to destination to the route record of route request packet and puts that into the route reply packet.



a) Route Request Packet propogation



b)Route Reply reverse path

Figure.2: Route Discovery in DSR

To send the route reply packet, the responding node must have a route to the source. If it has a route to the source in its route cache, it can use that route. If bi-directional links are supported the reverse of the route record can be used. In case bi-directional links are not supported, the node can initiate route discovery to source and piggyback the route reply on this new route request.

To maintain a route in DSR when originating or forwarding a packet using a source route, each node transmitting the packet is responsible for confirming that data can flow over the link from that node to the next hop. An acknowledgement can provide confirmation that a link is capable of carrying data.

After the acknowledgement request has been retransmitted the maximum number of times, if no acknowledgement reply has been received, then the sender treats the link to this next-hop destination as currently "broken". It should remove this link from its Route Cache and should return a "Route Error" to each node that has sent a packet routed over that link since an acknowledgement was last received. For sending a retransmission or other packets to this same destination, if the source has in its Route Cache another route to the destination, it can send the packet using the new route immediately. Otherwise, it should perform a new Route Discovery for this target [6] [8] [15]. Further detail on the DSR route maintenance process is provided in Section 3.1.1.

2.3 Hybrid Protocols

An adaptive hybrid protocol is desirable since requirements for network performance vary among applications. Multi-media application can tolerate high loss rates, but sensitive to variations in delay. TCP traffic is sensitive to loss in the network. Devices running on battery power are concerned with routing overhead. An adaptive hybrid routing protocol requires the following three properties

- Adaptive: The protocol should be applicable to wide range of network characteristics.
- Flexible: The protocol should enable applications to optimize for different application-specific metrics at the routing layer.
- Efficient and Practical: The protocol should achieve better performance than pure, non-hybrid, strategies without invoking costly low-level primitives.

2.3.1 Sharp Hybrid Adaptive Routing Protocol (SHARP)

SHARP is an analytically-driven hybrid routing protocol that finds the optimal mix of proactive route dissemination and reactive route discovery [29]. Sharp uses an analytical model to make the tradeoff optimally. Each node can direct the routing layer to optimize for a different metric of it's choice, such as overhead, latency or jitter, for routes targeting that node.

SHARP utilizes this fundamental trade-off between proactive versus reactive routing to find a good balance between route information propagated proactively and route information that is left up to on demand discovery. SHARP utilizes both proactive protocol and reactive protocol to perform routing by creating proactive zones automatically around hot destinations. Nodes within the proactive zone maintain routes proactively only to the central node. SHARP creates relatively large zones around popular destination and relatively small proactive zones around nodes that get little traffic[29].

Increasing the radius, SHARP can decrease the loss rate and the variance in delay, but will pay more in packet overhead. Decreasing the radius, SHARP can reduce routing overhead , pay more in delay jitter and experience higher loss rate.

SHARP's reactive routing protocol is based on AODV. If the source is outside the proactive zone, route request are broadcast by AODV. Nodes in the proactive zone of a destination generate route replies to response. The proactive zone act as a collective destination for data packets from the source. SHARP nodes continually monitor network characteristics. Nodes in a proactive zone:

1. measure and maintain the average link lifetime of immediate links
2. Measure the average node-degree (number of immediate neighbors)
3. Aggregate the average link lifetime and the average node-degree with the values received from other upstream nodes as part of the update protocol and broadcast the aggregated-values periodically along with the update packet.

Destination nodes maintain statistics about the data traffic that it has received.

1. Identity of the source
2. Current distance to the source
3. Average performance in terms of loss rate and delay jitter over a period of time

combine the network characteristics and traffic metrics, the destination can compute the optimal radius for its proactive zone.

The SHARP Proactive Routing protocol (SPR) is an efficient protocol engineered with techniques borrowed from different routing algorithm such as Destination Sequenced Distance Vector (DSDV) and Temporally Ordered Routing Algorithm (TORA). SPR performs proactive routing by building and maintaining a directed acyclic graph (DAG) rooted at the destination[29].

3 Related Work

3.1 The Route Maintenance process

The importance of route maintenance and studying the routing protocols vulnerability to node movement lays in the unstable nature of the ad hoc environment. Research is going towards optimizing routes by prediction as to prevent link breakage in the case of intermediate-node or destination movement [18] [25] [31]. A number of these optimizations are presented in Section 3.1.2. Such research is necessary to allow for the understanding of the stability of routes and paths since they are very essential due to their direct influence on the on-going communication between the two end nodes. A better and more stable route maintenance mechanism in different mobility and speed rates generally means a better routing protocol.

3.1.1 The DSR Route Maintenance process.

The DSR protocol places a little more effort on route maintenance than AODV. DSR has a number of additional route maintenance features [8]. Most of these features take advantage of the Source routing nature of the DSR protocol. In order to provide these features DSR utilizes a number of data structures described in section 3.1.1.2. The following are a number of these additional route maintenance features related to our study.

3.1.1.1 Additional Route Maintenance Features

A. Packet Salvaging

When an intermediate node forwarding a packet detects that the next hop along the route for that packet is broken, packets destined over this link are dropped. If the node has another route in its Route Cache to the packet's destination, the node could "salvage" the packet rather than drop it. To be able to salvage a packet, the node needs to replace the original source route in the packet with the route from its Route Cache. The node then forwards the packet to the next node indicated along this source route

A count is maintained in the packet of the number of times that it has been salvaged, to prevent a single packet from being constantly salvaged. Otherwise, there is a possibility that the packet could enter a routing loop if different nodes repeatedly salvaged the packet and replaced the source route on the packet with routes to each other.

If the node sends a Route Error as described in Section 2.2.2, it should originate the Route Error before salvaging the packet.

B. Queued Packets Destined over a Broken Link

When an intermediate node detects through Route Maintenance that the next-hop link along the route for a packet is broken, the node should handle any pending queued packets that are destined over this new broken link similar to the way it handles the packet detecting the link break as defined for Route Maintenance. Specifically, the node should search its Network Interface Queue and Maintenance Buffer (Section 3.1.1.2) for packets for which the next-hop link is this new broken link. The node should process each queued packet as follows:

- Remove the packet from the node's Network Interface Queue and Maintenance Buffer.
- Originate a Route Error for this packet to the original sender of the packet, as if the node had already reached the maximum number of retransmissions for that packet for Route Maintenance. The node should send no more than one Route Error to each original sender of any of these packets.

- If the node has another route in its Route Cache to the packet's Internet Protocol (IP) Destination Address, the node should salvage the packet as described in Section 3.1.1.1 part A. Otherwise, the node will drop the packet.

C. Increased Spreading of Route Error Messages

When a source node receives a Route Error for a data packet that it originated, it can propagate this Route Error to its neighbors by piggybacking it on any subsequent Route Request. Using this method stale information in the caches of nodes around this source node will not generate Route Replies containing the same invalid link for which this source node received the Route Error.

3.1.1.2 DSR Route Maintenance structures

A packet being transmitted could be queued in a variety of ways depending on the structure and organization of the operating system, protocol stack implementation, network interface device driver, and network interface hardware. Before transmission by the network interface outgoing packets from the network protocol stack might be queued at the operating system or link layer. The network interface may also provide a retransmission mechanism for packets, such as in IEEE 802.11 [28]; in the DSR Route Maintenance mechanism only limited buffering is required for packets already transmitted for which the reachability of the next-hop destination has not yet been determined. DSR maintains two conceptual data structures that together provide the necessary queuing behavior.

A. The Network Interface Queue

The Network Interface Queue is an output queue of packets from the network protocol stack waiting to be transmitted by the network interface; this queue is used to hold packets while the network interface is in the process of transmitting another packet.

primarily define regions for transmission management[27].There has been work enhancing AODV to provide more stability in the case of cluster merging. Clusters allow auto configuration and reuse of IP addresses. The merging of clusters could cause IP conflict. Solutions for broken fabrics and on-going communication have been presented in [35].

3.2 Ad Hoc Caching Mechanisms

In an on-demand routing protocol a newly discovered route should be cached, so that it may be reused the next time that the same route is requested. Two cases of route caching exist [4].

In the basic case, *source route caching,* a source node caches routes so that a route is available when an application, running within the same node, demands it. The other case is an extension to the above; this case is called *intermediate route caching*. Many on-demand routing protocols, such as AODV and DSR, allow an intermediate node that has a cached route to the destination to reply to the source with the cached route. These on-demand protocols also allow an intermediate node to cache info not destined to it.

The benefit of using route caches in the on-demand approach is two-fold. First and most importantly, when a route request arrives, a route is immediately available, leading to significantly smaller routing latency if the cached route is not obsolete. This is specifically important in audio and video transmissions, since the successful play-back of the received information is delay sensitive. Second, route caching reduces the control traffic required to search the network for a new route.

A serious challenge to such protocols is presented regarding *cache staleness.* Caches represent learned portions of the network topology; however, cache entries may become invalid due to topology changes. A topology change could be two nodes moving out of wireless transmission range of each other. Nodes are only notified that one of its cache entries has become invalid when the node itself actually attempts to use the cache entry to route a packet. Although periodic routing protocols such as a distance-vector or link-state routing protocol can distribute updated information in a logically timely manner, periodic protocols have been shown to have higher overhead in a number of studies [15][28]. Periodic protocols also take some amount of time to detect a link failure and to distribute this information.

The cache staleness problem is compounded when a node uses information from its route cache that was learned from overheard packets. Stale information could circulate in the network indefinitely [4]. For example, one node may use stale information to route a packet, allowing a number of nodes to overhear that packet and to also cache that stale routing information. Those nodes will be left with a stale link in their route cache If they do not subsequently overhear the corresponding route breakage notification. The nodes may later use this stale routing information to route other packets starting another life cycle of the same problem.

3.2.1 Caching Optimization

All optimizations in this section are presented as they are presented from their original source. We applied little rephrasing so that the information is presentable. A reference to each original source is placed after the optimization headline.

A) Epoch numbers [4]: This paper presents a new mechanism, called *epoch numbers*, for reducing the amount of stale information in each node's route cache. It presents an cache management that prevents a node from re-learning stale information about a link after having earlier heard that this link has broken. This scheme does not rely on ad hoc mechanisms such as short-lived negative caching of routing information; rather, this scheme allows a node, having heard that a link has broken and having heard a discovery of the same link, to sequence the two events and determine whether the link break or the link discovery occurred more recently than the other.

B) Wider Error Notification [14]: In order to increase the speed and the extent of error propagation, this method transmits route errors as *broadcast* packets at the Medium Access Control (MAC) layer. Initially, the node that determines the link breakage (via a link layer feedback, e.g.) broadcasts the route error packet containing the broken link information. When a node receives a route error, it updates its route cache so that all source routes containing the broken link are truncated at the point of failure.

A node receiving a route error further propagates it only if cached route containing the broken link exists and that route was used before in the packets forwarded by the node. Using this method route errors reach all the sources in a tree fashion starting from the point of failure. So, route error information is efficiently sent to all the nodes that forwarded packets along the broken route and to the neighbors of those nodes that may have cached the broken route by overhearing it.

C) Timer-based Route Expiry [14]: Link breakage is detected only by a link layer feedback when an attempted data transmission fails. If there is no attempt to use this route the loss of a route will go undetected. The timer-based expiry approach described in [14] will be able to clean up such routes. It is based on the hypothesis that routes are only valid for a specific amount of time ΔT (timeout period) from their last use. Each node has an associated timestamp of the last use of a cached route. This timestamp is updated each time the cached route or part thereof is “seen” in a unicast packet being forwarded by the node. Portions of cached routes unused in the past ΔT interval are pruned.

The benefit of this approach depends on the selection of the timeout period ΔT. A small value for the timeout may cause many unnecessary route invalidations, while a very large value may defeat the purpose of this technique. Although well-chosen static values a given network can be obtained, a single timeout for all the nodes may not be appropriate in all scenarios and for all network sizes. Therefore, a Adaptive mechanism that allows each node to choose timeout values independently based on its observed route stability is desired.

However when many route breaks occur in short bursts with a large separation in time, the average route lifetime does not accurately predict ΔT during the periods of no route breaks.

D) Negative Caches [14]: To improve error handling in DSR, caching of negative information has been suggested in [14]. Every node caches the broken links seen recently via the link layer feedback or route error packets. Within a Δt interval of creating this entry if a node is to forward a packet with a source route containing the broken link, (1) the packet is dropped and (2) a route error packet is generated. In addition, the negative cache is *always* checked for broken links before adding a new entry in the route cache.

E) Path validation: [16]: This is an active mechanism for verifying path validity. A path is validated by sending a test routing packet to the destination. If the destination receives the packet, it replies to it or simply discards it. However, if the path is invalid, the test packet will be dropped and a route error packet will be sent to the source. The source can then remove this stale path from its cache. Thus, the technique requires no support from intermediate nodes and generates overhead only for paths that the source is interested in validating.

As described in [14], If a large number of cached routes are in the source nodes cache, validating all of them at the same time may lead to significant overhead on the network, especially if these paths are close to each other. Thus, validation should be applied judiciously.

F. Scoped Route Searches [16]: This mechanism discovers new routes in the network that have become available since the last search was applied. To limit the effect of using low quality paths when shorter paths are available in the network, [16] introduces periodic scoped route searches. These searches are similar to floods, but they are only allowed to propagate X hops around the source. The value of X must be smaller or equal to the best currently available path. In effect, this technique can discover any shorter paths that have appeared in the network since the last full search.

As described in [16], although the scoped route searches do not search the entire network, they are expensive and should be used judiciously.

G. Path Pruning [16]: In a route request process many routes can be found; Both short, high quality routes and long routes will be returned. The node will go through the list of routes in its cache in the order of hop count; eventually it will reach and attempt to use these low quality long routes. Path pruning prunes out low quality paths from the cache to avoid using them. While other studies have ways to expire older cache paths, such as timer based expiry described in this section part C, this method takes a risk-reward based approach to expiring paths.

Protocols would like the paths in the cache to be high quality (short, and reasonably fresh). Path pruning can help limit the amount of paths to be validated, as described in this section part E, by removing the low quality paths. This method investigates criteria

If the route was to remain valid throughout the connection lifetime, then the connection would have minimum delay. However, due to link breakage and route maintenance, further delay is added to the life time of the connection. The overall connection delay is proportional to the number of route maintenance mechanisms. Route maintenance must be taken in consideration to improve the performance of any ad-hoc protocol.

5.2 The Route maintenance-Caching Mechanism Relationship

Caching mechanisms and the route maintenance mechanism that are used in reactive routing protocols, share a direct relationship with one another. Invalid cache entries represent incorrect information about the network topology. A packet destined over a stale or invalid route will eventually drop which will invoke a route maintenance process. Route maintenance may be invoked in one other situation. When a broken link is detected while routing data packets, as a result of a sudden change in topology, a route maintenance process occurs.

A caching mechanism that increases the validity of cached routes will reduce the number of link breakages (route maintenance) detected due to invalid cache entries. This will eventually reduce the number of data packets dropped and delay caused by route maintenance, and the overall performance will improve.

5.3 Comparing Caching Mechanisms

As discussed earlier, Ad-Hoc protocols rely on a caching mechanism that involves caching information about the network topology, which incurs a substantial amount of effort in research, development and design in order to reduce the effects of stale routes. Different caching mechanisms come with different advantages and limitations. Table 2 lists some of the basic characteristics of caching mechanisms discussed in this thesis.

It is obvious in Table 2 that most caching mechanisms are applied on the route maintenance process and do not have the ability to prefer fresher routes. In Chapter 3 we will show unsuitable such characteristics can be in a specific environment, such as an environment of small transfers.

5.4 Validity of Cached Routes vs. Link Prediction

Link Predication has taken a large amount of researcher's interest. Reducing the effect of route maintenance on throughput and delay is the basic interest of these predication algorithms. Link predication algorithms target route maintenance caused by mobility [18] [25] [31]. Route maintenance caused by a link breakage due to the use of an invalid cache entry can't be predicted because the breakage occurred before the route is used. As a result, Link predication cannot reduce the effect of route maintenance caused by invalid cached entries on throughput and delay. Increasing the validity of cached entries is the only way to reduce the effect of route maintenance caused by invalid cache entries.

When attempting to reduce the effect of route maintenance on the performance of a routing protocol, the life time of the connection should be taken in consideration. With small transfers the life time of the connection is too short to apply link predication; assuming a moderate mobility rate, the connection usually closes before any mobility occurs. Thus, the only remaining factor is the invalid cache entry route maintenance.

5.5 Comparing Ad-Hoc Routing Protocols

As discussed earlier, table-driven routing relies on a routing table update mechanism that involves the constant propagation of routing information, which incurs substantial signaling traffic and power consumption. Since both bandwidth and battery power are scare resources in mobile computers, this becomes a serious limitation. In on-demand

Table 2: Comparison of the Characteristics of Caching Mechanisms

Caching mechanism	Fresher route preference	Use of Ad-hoc mechanism	Accumulated info	Sequence numbers	Additional features	Higher quality (hop count)
Epoch numbers	yes	none	none	yes	Epoch no.	no
Wide-error notification	no	Route Maintenance	none	no	none	no
Base DSR	no	Route Maintenance	none	no	______	yes
Base AODV	yes	Both mechanisms	none	yes	______	yes
Timer-Based Expiry	yes	Route Maintenance	yes	no	none	no
Negative caches	no	Route Maintenance	none	no	none	no
Path validation	no	Route Maintenance	none	no	Active Packets	no
Scoped Searches	no	none	yes	no	Scoped Requests	yes
Path pruning	yes	none	none	no		yes

routing, when a route to a new destination is needed, it will have to wait until a route is discovered, but in table-driven protocols, a route to every node is always available. Hybrid protocols have also been introduced to utilizes this fundamental trade-off between proactive versus reactive routing. Table 3 lists some basic differences between the classes of protocols [25]. We added the hybrid column in Table 3.

Table 3: Comparison of Ad-Hoc Routing Protocol Categories

Parameter	On-demand	Table driven	Hybrid
Availability of routing info	Available when needed	Always available regardless of need	Available according to destination popularity
Routing philosophy	Flat	Mostly flat	Flat
Periodic route update	Not required	Required across entire network	Required in proactive zones
Coping to mobility	Localized route discovery	Inform other nodes to achieve consistent routing table	Maintain info about proactive zone central node and localized route discovery outside proactive zone
Signaling traffic generation	Grows with increasing mobility	Greater than on-demand	Balance according to targeted advantage

Route Maintenance in On-Demand Protocols and Small Transfers

Route maintenance has significant effect on throughput and delay. Reducing the number of route maintenance processes must be taken in consideration to improve the performance of on-demand protocols.

Depending on the caching mechanism applied in an on-demand protocol in an environment of mostly small transfers, the protocol may suffer from performance degradation because of route maintenance caused by invalid cache entries.

Small transfers may constitute a significant portion of traffic in future ad hoc networks [30]. These transfers may include resource discovery, text messaging, object storage/retrieval, queries and transactions. Most performance studies of ad hoc routing use long-lived randomly assigned connections that usually last throughout simulation duration [30]. These connections emulate File Transfer protocol (ftp) or user-specific Constant Bit Rate (CBR) applications. However, for short-lived small transfers routing protocols may exhibit significantly different behavior [30].

The delay caused by a route maintenance process invoked during a small transfer is expensive due to the short life time of these transfers. The suitability of integrating a caching mechanism within the route maintenance process is questioned. Analyzing the effect of such integration on throughput and delay is essential since many DSR caching mechanisms are applied to the route maintenance process [14]. On the other hand, AODV applies its caching mechanism on the route discovery process. In this Chapter, we apply a study on AODV and DSR in regard to this context.

The rest of this chapter is organized as follows. Section 1 provides further elaboration on the specified simulation problem. Section 2 defines the simulation model. Section 3 introduces the performance metrics. Section 4 shows the results of simulations and the last two sections discuss these results and conclude a solution for the simulation problem, respectively.

1 The simulated problem

According to [30] AODV outperforms DSR in an environment with low traffic and small transfers. DSR heavily depends on caching route information. Trusting in such route information would be an adverse affect in throughput in very light traffic scenarios since cached routes are often out-dated. Researchers of [30] also suggest a possible explanation in regard to the performance of DSR with small transfers. They suggest that the multiple routes to the same destination maintained in DSR's caches are what cause this DSR performance. Early packet delivery failure involved with the detection of a broken link in a specific route only suggests the invalidation of such route entries. Other alternative routes can be found in the same cache. However, the probability of that route being valid is the same as all other routes. AODV's caching mechanism includes only one cache entry per destination. Therefore, if the preceding packets in a stream get a valid route to a destination, the chances of successful delivery of subsequent packets to that destination are almost guaranteed. Therefore, AODV seems to outperform DSR in such short grouped traffic patterns [30].

1.1 Increasing the validity of routes

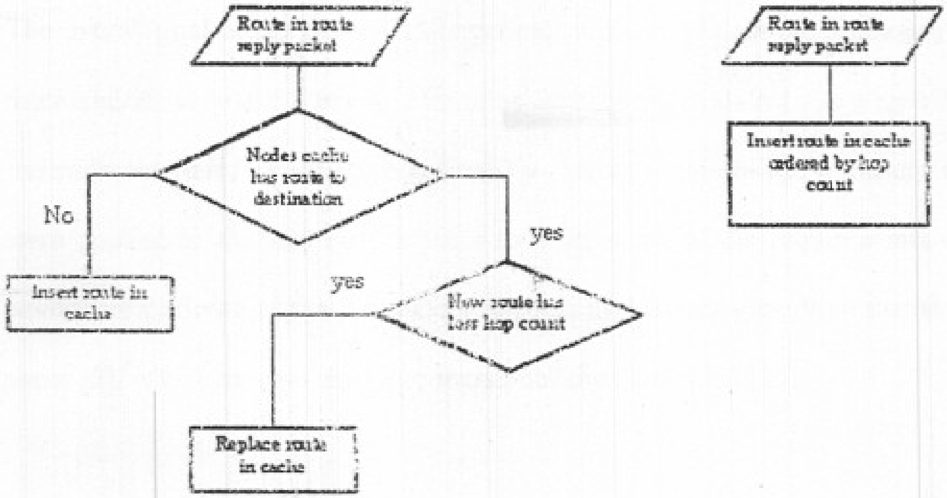
Caching information is a basic strength of DSR in many cases presented in many studies [2] [25] [24]. Optimizing the cached route validity in order to address the simulated problem in this chapter is one of the most feasible approaches. This will increase the validity of the Cached routes. Consequently, route maintenance invoked due to invalid route entries may reduce, increasing throughput and reducing delay.

To provide a solution we need to rephrase the problem. Here are the basic problems in the DSR caches as presented in paper [14]:

1- Incomplete error notification: route error packets are only unicast to the source [14]. This was solved by using piggybacking with route discovery packets, however, intermediate node route replies prevent error propagation network wide.
2- No expiry mechanism to expire stale routes.
3- Quick pollution: subsequent "in flight" packets can put stale routes right back in cache (snooping cache helps for quick pollution).

Wide Error Notification, described in chapter 2 section 3.2.1, is a method deployed on the original route maintenance mechanism which is triggered by broken links [14]. If all connections started at the beginning of the simulation when short lived connections are used, broken links (route maintenance) hardly ever occur. In this case, Wide Error Notification is not effective in increasing the validity of cached routes because it is rarely invoked. However, in a more realistic communication model invalid routes may already be placed in the route cache. These invalid routes when used will cause broken link detection. Thus the route maintenance mechanism will be invoked. Wide Error Notification will solve the first problem of DSR caches. And will thus increase the validity of cached routes. This should reduce the number of route maintenances invoked. Consequently increasing throughput and reducing delay.

According to the specifications WEN, presented in [14], we applied a number of modifications on the original DSR implementation in the GloMoSim simulation package. Such implementation required a great amount of understanding of the DSR prefer fresher routes). This is true because these two factors are the only factors involved in the caching process. Figure 3 illustrates the algorithm used for this imitation process .



a) After imitation b) before imitation

Figure 3: Disabling Multiple Paths in DSR

It would require deep understanding of the DSR and AODV protocol implementation to allow for the implementation of this partial imitation. Studying the protocol functions, how they relate to one another and how caches are structured helped in producing an efficient implementation of this imitation. We also applied an enormous amount of testing to ensure that the newly produced code actually performed as desired.

We are trying to study and evaluate the route maintenance mechanism in both AODV and DSR. We are also trying to figure out if these two routing protocol are using this mechanism properly. In an ad hoc network where mostly small transfers are used it's preferable that the route maintenance mechanism is used as rarely as possible. Because the delay caused by this mechanism is very significant relative to the life time of such short-lived connections. Having multiple routes in DSR increases the possibility of "pause time" seconds, and then it moves to a randomly selected destination in the 1200m x 1200m area at a speed distributed uniformly between 0 and "maximum speed". When the node reaches the destination, it stops moving for "pause time" seconds, randomly selects another destination, and then repeats the previously described process for the duration of the simulation.

In our simulation, we use small transfers. If the nodes were to be stationary for any time more than 0 pause time, most of the packets would be sent before any movement takes place. As a result, the movement scenario files are generated for a pause time: 0 only. We use a maximum speed of node movement: 20 meters per second.

2.3 Communication Model

To be able to successfully study the behavior of the route maintenance process in an environment of small transfers, we must generate a specific communication pattern. We choose the traffic sources to be constant bit rate (CBR) sources. A data sending rate of 10 packets per second is used with 512-byte data packet size. A communication pattern based on 8 bursts with 30 seconds between each burst and another is chosen corresponding to 30 traffic sources at each burst.

The reason for choosing this specific communication pattern is that if all sources of traffic were to start at the beginning of the simulation, most connections would be closed before any link break occurs. Such a communication pattern is highly unsuitable to our study since we are studying the route maintenance mechanism which is invoked by link breakage.

We did not choose TCP sources because TCP adapts to the load of the network. If TCP was used, the time when a node sends a packet would be different for different protocols, even though they have the same data traffic and node movement scenario.

Then it would be difficult to compare the performance of the original and modified protocols [25].

3 Performance metrics

3.1 The Route Maintenance Ratio

The Route Maintenance Ratio reflects the weight of the route maintenance process to all processes for a specific protocol; in addition it shows the suitability of applying methods on this process to improve the routing protocol. This suitability can be inferred from the relationship between route maintenance and performance described in Chapter 2 (Section 5.1).

As for the importance of this metric to DSR it should reveal the effect of using the two different methods used to improve DSR, described in Section 1, on the number of route maintenances. Any change in the number of route maintenances will have an effect on the number of dropped packets due to route maintenances which is directly related to throughput. The following is the formulated definition of this metric.

Route Maintenance Ratio = (Number of route maintenances / (number of route maintenances+ number of route discoveries))

In our study, we provide the following metric which is much more specific than the earlier metric. The Invalid Route Maintenance Ratio shows the exact affect of any one of the two methods, described in Section 1, applied on DSR to increase the validity of the routes used by a specific source to an arbitrary destination. It also directly reflects the relationship between the validity of routes and route maintenance.

Invalid Route Maintenance Ratio = (The number of route maintenances caused by invalid cache hit / the number of route maintenances)

Both of the pervious metrics have their own individual importance. They also complete one another.

3.2 Throughput

The importance of the following, the Invalid Packet Drop Ratio, is its direct relation to the actual performance of the routing protocol. When a route maintenance process occurs it is suspected that a number of packets are dropped. An increasing number of route maintenances imply an increasing number of dropped packets. This metric is formulated as follows:

Invalid Packet Drop Ratio = (number of packets dropped due to invalid routes/ number of dropped packets)

To back up this metric, we have chosen a definition of "Packet Delivery Ratio" that has already been used in other papers [2] [25]. We also present this metric to show the actual relationship or weight of dropped packets caused by invalid Routes to the actual performance of the protocol. The following is the presented definition of “packet delivery ratio”.

Packet Delivery Ratio = (the percentage of total successfully delivered CBR packets: Total CBR packets- total dropped packets)/ total generated CBR packets.

We also introduce the Route Validity Ratio. This metric shows the number of invalid routes to all routes used by a source node throughout the simulation. This ratio is proportional to the Invalid Route Maintenance Ratio. So, reducing the number of invalid routes implies reducing invalid route maintenances and so an increase in throughput. This metric is formulated as follows.

Route Validity Ratio = (the number of invalid routes/ the overall number of routes used by a source node).

3.3 Delay

Delay is an important metric in relation to route maintenance. The DSR protocol actually has presented the alternative cached routes used in the route maintenance process as a method to improve the end-to-end latency. However, if that route was an invalid cached route, it would cause more delay. The following is the definition of delay caused by invalid routes.

Invalid Route Delay= the average (of the accumulated delay from when an invalid cached route was chosen up to when another route is chosen in a specific connection).

The overall delay also requires a definition. A definition of overall delay used in [14] defines Average end-to-end delay of data packets to include all possible delays buffering route discovery, queuing delay at interface, retransmission delay, propagation and transfer times…

3.4 Normalized routing overhead

Routing overhead is an important performance metric; it shows how much control packets a protocol uses to achieve its functionalities. A protocol would like to use less control packets to achieve the same functionalities. Routing overhead may increase the load on the network which in turn may increase the number of packets being dropped. The following is the definition of normalized routing overhead used in this thesis.

Normalized routing overhead= the number of control packets handled by a node throughout the simulation.

3.4 DSR and AODV Simulation Results

The AODV and DSR simulations share the same data traffic and mobility scenario files in order to allow for a fair comparison. A Visual Basic program was written to read the simulation trace files and extract the metrics we need. All simulations are done with the GloMoSim simulation Package on windows XP professional.

Each simulation is repeated 10 times with different seed values. Each trace file maintains the states of 100 nodes. This provides, in the best case, 1000 samples for each value extracted for metrics integrated in the nodes state. The nodes state is maintained in the GloMoSim Network Layer.

Each source node has a number of connections through the simulation time. Each simulation has 240 connections over the life time of the simulation. Since each simulation is repeated 10 times, this gives us a maximum of 2400 samples for metrics integrated in the connection state. The connection state is maintained in the GloMoSim Application Layer. More on Glomosim Layers can be found in Chapter 1 Section 3.

We apply these simulations on four different protocols:

A) The original protocols, DSR and AODV:

1) DSR with Promiscuous Error Handling (DSR_PEH): This is the current implementation of DSR in the GloMoSim package. This implementation follows the specification of DSR Internet Draft (03). It applies error handling to error packets over heard by a peek function when the protocol is under promiscuous mode.
2) Ad-Hoc On-Demand Distance Victor (AODV): This is the current implementation of AODV in the GloMoSim package. This implementation follows the specification of AODV Internet Draft (03).

B) The DSR protocol with two different modifications:

1) DSR with Wide Error Notification (DSR_WEN): This DSR protocol has all the specifications of the current implementation of DSR in the GloMoSim package with a different Error handling algorithm. This algorithm follows the specification of a caching mechanism described in Chapter 2 Section 3.2.1 part B. This algorithm was first introduced in [14].
2) DSR without Multiple Paths (DSR_WMP): This DSR protocol has all the specifications of the current implementation of DSR in the GloMoSim package with a different route reply caching mechanism. This algorithm follows the specification of a caching mechanism described in Section 1.2. According to our knowledge this algorithm has been introduced in this thesis for the first time individually in such a context. In [16], this algorithm has been presented as part of a caching mechanism that applies pruning on caches. In a certain situation, caches may have one entry per destination. More on this pruning caching algorithm in this thesis can be found in Chapter Section 3.2.1 part G.

In order to retrieve the required information to calculate the results of the metrics, described in Section 3, a great amount of effort and time is required. For each performance metric a retrieval algorithm was designed and implemented in the GloMoSim simulation package. These retrieval algorithms are designed to calculate the required metric values and place them in a trace file produced at the end of the simulation. To ensure that our retrieval algorithms work as desired, a large amount of testing was applied.

4.1 The Route Maintenance Ratio

Figure 4 shows the Route Maintenance Percentage as a function of traffic load (number of data packets sent by a traffic source). For DSR and all the modifications on DSR, the Route Maintenance Percentage is independent of the traffic load, having a high percentage ranging from 88% to 95% in all cases. There is a significant difference between AODV and DSR. For AODV, the Route Maintenance Percentage is much lower for all traffic loads ranging from 47% to 63%. The confidence interval is hardly noticeable exhibiting a 90% confidence in the results shown in this figure.

DSR_WMP has a slightly less Percentage, because DSR_WMP increases the number of route discoveries and reduces the number of invalid routes (see Figure 6) and therefore reduces the number of invalid route maintenances.

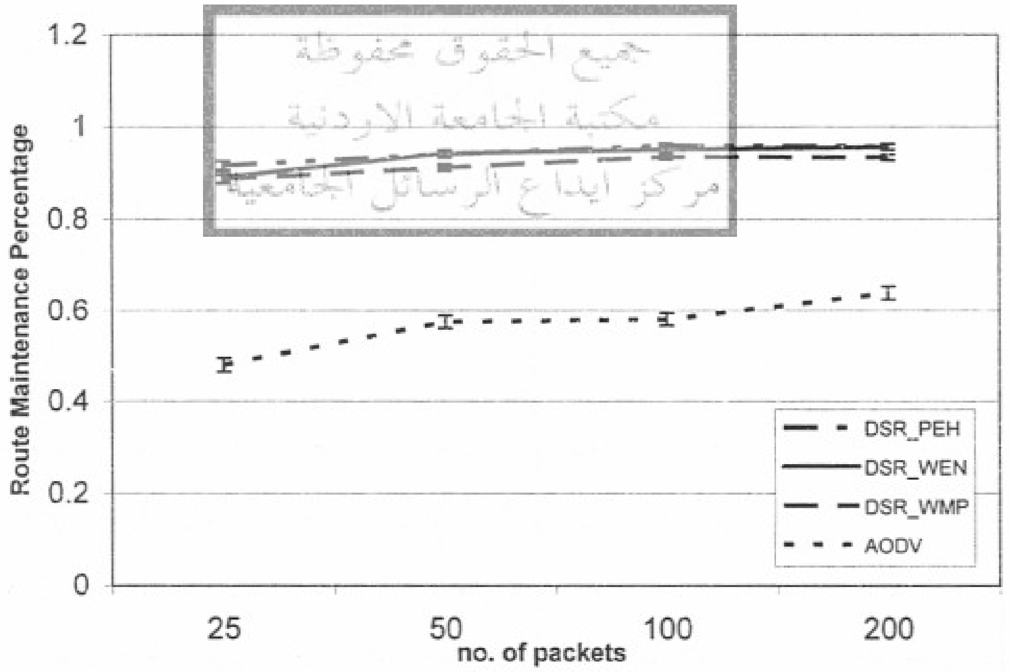


Figure 4: Route Maintenance Percentage as function of traffic load

In Figure 5, we present the results of the Invalid Route Maintenance Ratio. For DSR, 71% to 81 % of route maintenances are actually invalid. This ratio is high for all traffic loads. For AODV, the value of this ratio is lower than DSR ranging from 26% up to 42%. Unlike DSR, this ratio seems to increase for AODV with the increasing of the number of packets being sent by the traffic sources. The confidence interval obviously illustrates the accuracy of the results exhibited.

DSR_WMP has less invalid route maintenances than DSR_WEN due to applying one entry per destination which has a better affect than trying to increase the validity of multiple paths with small transfers. We conclude that optimizing the caching mechanism applied to the route discovery process, like DSR_WMP, has better effect than optimizing the caching mechanism applied on the route maintenance process, like DSR_WEN.

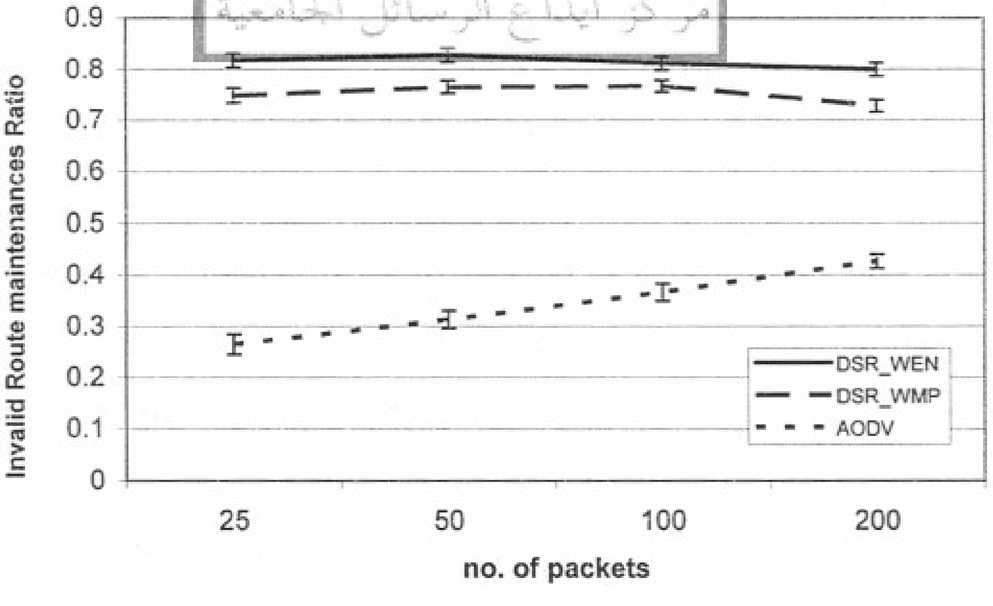


Figure 5: Invalid Route maintenances Ratio as function of traffic load

AODV prefers fresher route and applies one entry per destination. As a result, it will highly depend on route discovery to get a route to a destination. DSR_WMP's caching mechanism is a partial imitation of AODV's caching mechanism which only applies one entry per destination. Thus, the only logical explanation to the difference between DSR_WMP and AODV is the other factor, preferring fresher routes. Route discovery combined with preferring fresher routes reduces the number of route maintenances and the number of invalid route maintenances. The two Figures in this section clearly illustrate that.

4.2 Throughput

Figure 6 shows the Route Validity Ratio as a function of traffic load. For DSR_PEH and DSR_WEN, we have an almost identical result where the route validity ranges from79% to77%. For DSR_WMP, the route validity decreases to range from 70% to 74%. DSR in all cases has a similar reaction towards any change in the number of packets being sent.

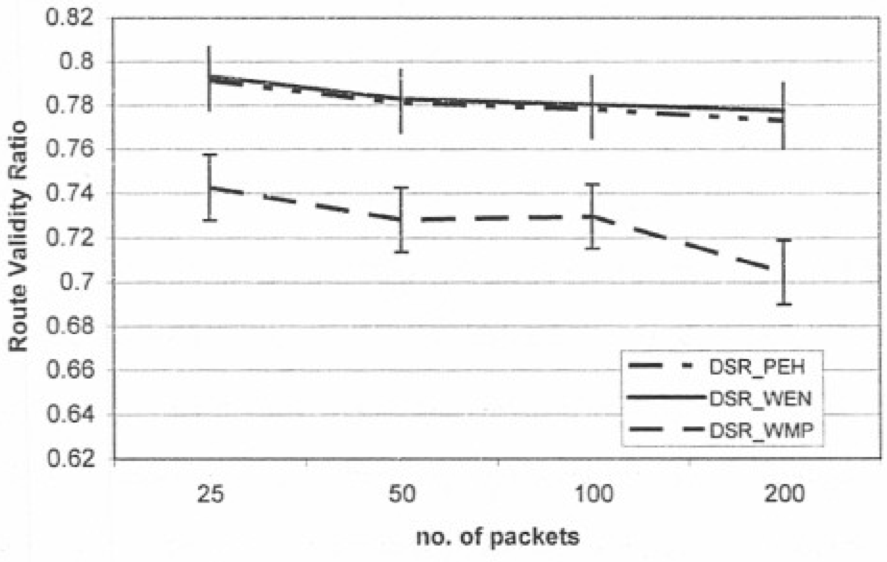


Figure 6: Route Validity Ratio as function of traffic load

Figure 7 shows the Packet Delivery Ratio for DSR and AODV as a function of traffic load. With 25 packets being sent by the sources of traffic AODV shows a higher packet delivery rate than DSR. For all protocols, the more data packets being sent the less packet delivery ratio. AODV severely steeps as more data packets are being sent; on the contrary DSR has an easier fall. As a result, DSR without multiple paths and AODV meet with a value close to 20% delivery when sending 100 data packets. Again, DSR_WEN and DSR_PEH have almost identical results. DSR without multiple paths has a better performance than other DSR protocols.

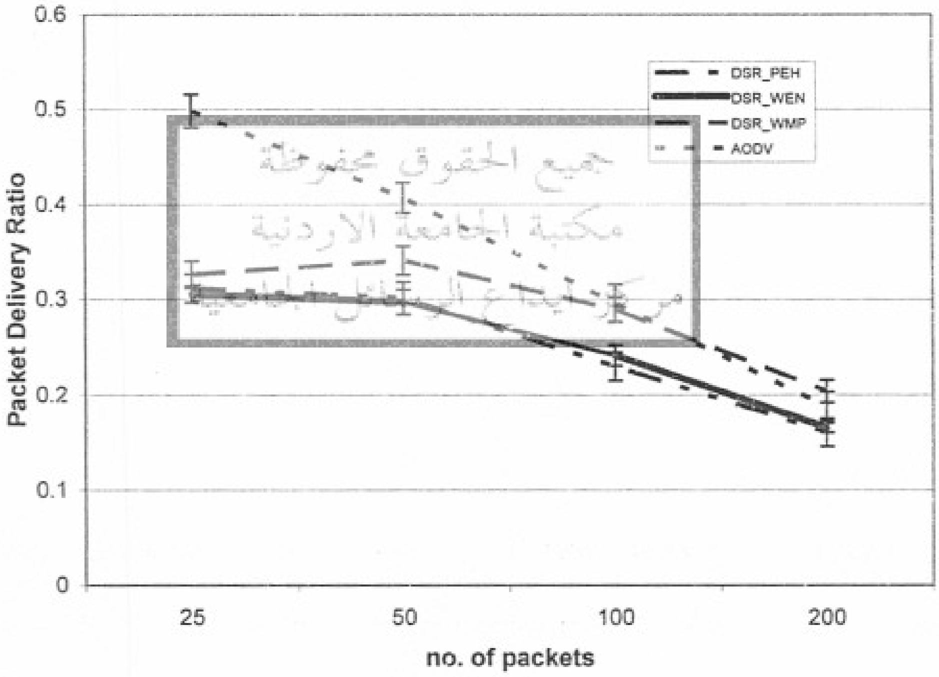


Figure 7: Packet Delivery Ratio as function of traffic load

It is obvious from these results that disabling multiple paths in DSR helped increase DSR_PEH throughput (Figure 7). Figure 6 confirms these results; it shows that there is a decrease in the number of invalid routes to the number of routes selected. Reducing the number of invalid routes implies reducing invalid route maintenances and so an increase in throughput

AODV only had better results in the first two cases of traffic load in Figure 7 because it uses fresher routes and one entry per destination. AODV takes advantage of being able to prefer fresher routes in these two cases only; because of this ability, AODV uses route discovery more often increasing the overhead on the network. This overhead was tolerable with very small transfers; however with more traffic this overhead places too much load on the network causing throughput degradation. DSR_WMP managed to reduce the difference between DSR and AODV in these two cases. This reduction was not full because DSR_WMP does not prefer fresher routes.

Figure 8 represents the Invalid Packet Drop Ratio from the source node point of view.

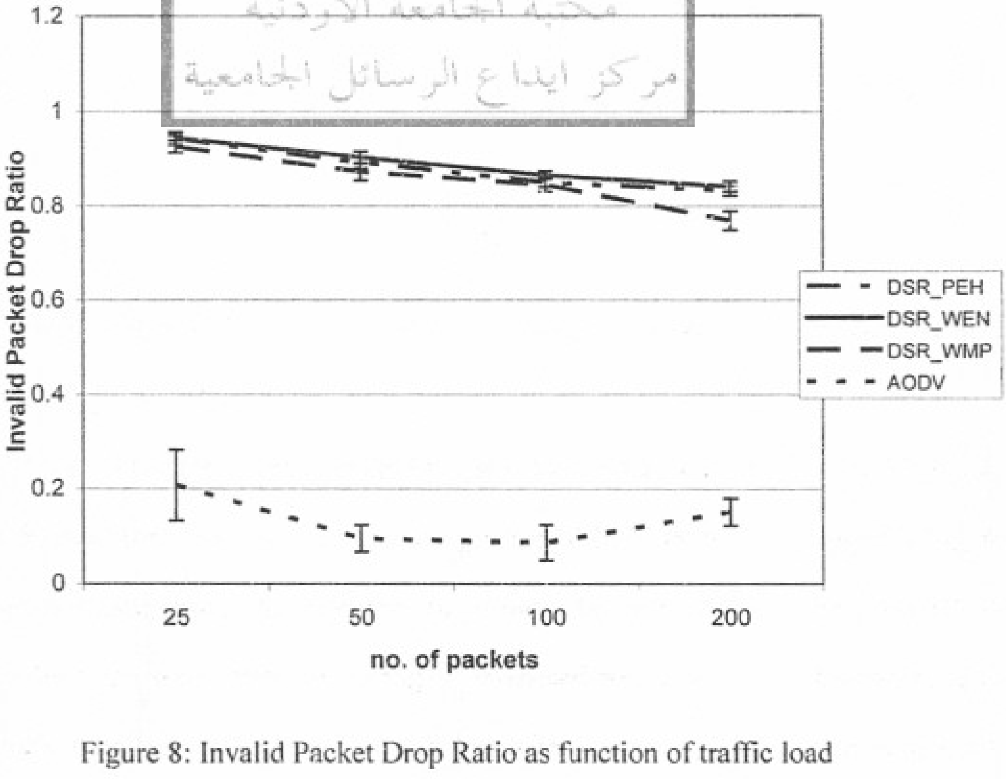


Figure 8: Invalid Packet Drop Ratio as function of traffic load

The number of invalid dropped packets for all DSR protocols is very high in all cases with respect to the overall number of dropped packets. DSR has an Invalid Packet Drop Ratio ranging from 77% to 94% indicating a strong relationship between packet delivery ratio and invalid route entries. The Invalid Packet Drop Ratio shows slight improvement after disabling the multiple path features in DSR_PEH, especially in the last case where the number of packets increase (Figure 8).

AODV has a much lower ratio than DSR, ranging between 10% and 20% confirming the result presented in Figure 6, that traffic load is the determining factor in AODV's throughput performance; Figure 8 shows that most of the packets being dropped by AODV are because of some other factor, other than invalid routes. The major factors involved in packet dropping are: route maintenance and load on the network.

An interesting observation is that the behavior of AODV's Invalid Packet Drop Ratio is similar to DSR in the first 3 cases (25, 50, 100 packets) in keeping a steady light decrease in the ratio value, however it then diverts and increases in the last case (200 packets).

4.3 Delay

Figure 9 shows the Invalid Route Delay as a function of traffic load. For DSR with WEN and DSR with PEH the delay caused by invalid routes is very similar at all cases. DSR without multiple paths has less delay caused by invalid routes than the other DSR protocol. Disabling the alternative path service in DSR does help in reducing the delay in such an environment. However, this reduction does not fall anywhere close to the performance of AODV. AODV has the best performance of all protocols in regard to this metric. The different protocols follow a similar behavior as the delay caused by invalid routes seems to increase as the number of packets increase for all loads of traffic.

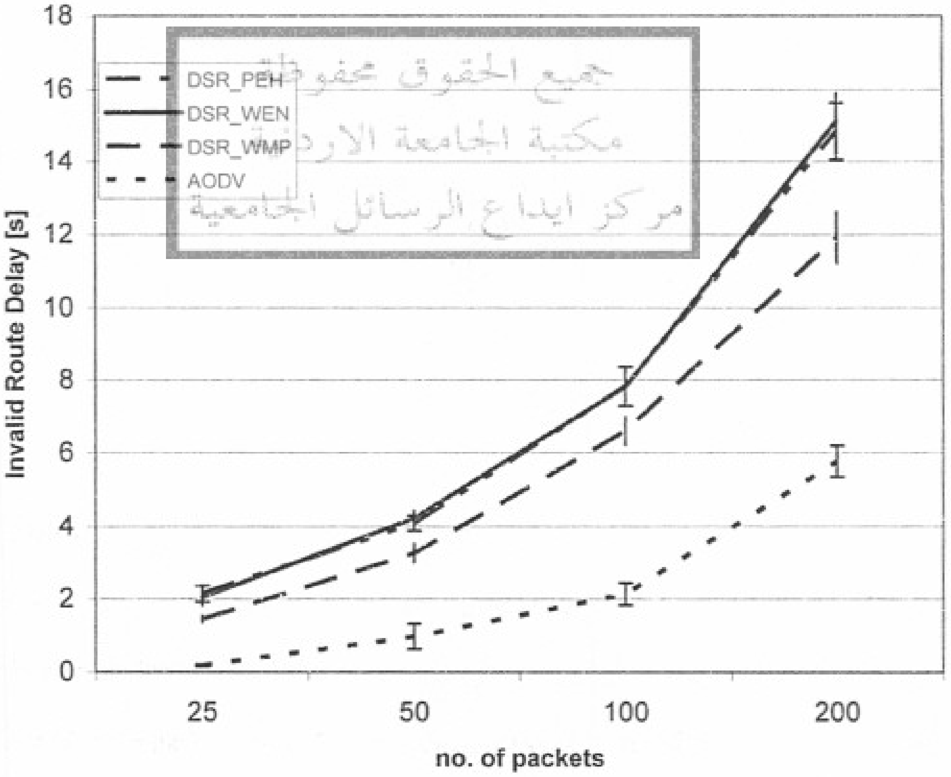


Figure 9: Invalid Route Delay as function of traffic load

In Section 1.2, we have described the imitation of the AODV route reply caching mechanism to develop the version of DSR without multiple paths. We have also illustrated in that section the two different factors involved in this process: the one entry per destination factor, and the fresher route preference factor. Figure 9 has been used to extract the weight of each one of those factors on the delay caused by invalid routes. Figure 10 is a demonstration of the result of that extraction. There is a higher weight to the fresher route preference factor with approximately 70% of the weight. However, there is significance to the 30% weight of the one entry per destination factor.

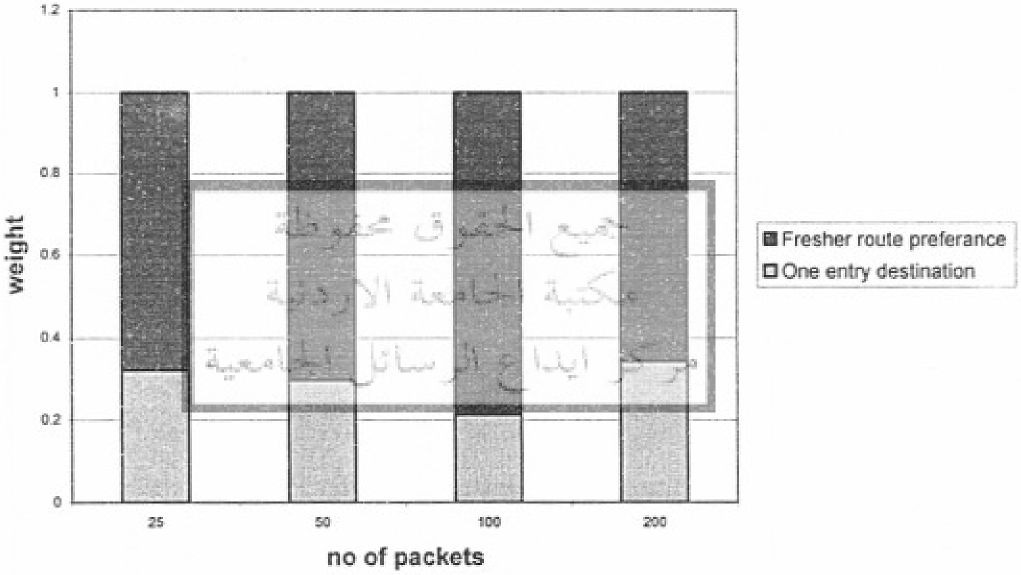


Figure 10: Weight of Caching Process Factors on Invalid Delay

In our simulations, we use CBR connections. The CBR connections provide sources of traffic, since the packets are sent through the network, this functionality is achieved whether these packets arrive at the destination or not. The CBR traffic sources do not retransmit the data packets. So, a CBR connection is established if at least one of the specified packets arrives to the server (destination). However, if none of the packets arrive at the destination, then the connection is never established.

In Figure 11, the average delay for the established connection is shown. The different protocol had different number of established connections. This Figure shows that the average connection delay for DSR protocols is similar at all cases. AODVs curve starts at the beginning of the DSR curve then DSR slowly moves away as it has a higher elevation. In the last two cases with 100 and 200 packets being sent by the traffic sources, AODV has some significant difference in delay compared to the DSR protocols.

The following results will explain why the reduction in invalid delay caused by disabling multiple paths in DSR, shown in Figure 9, was not confirmed by a reduction in the overall delay of established connection.

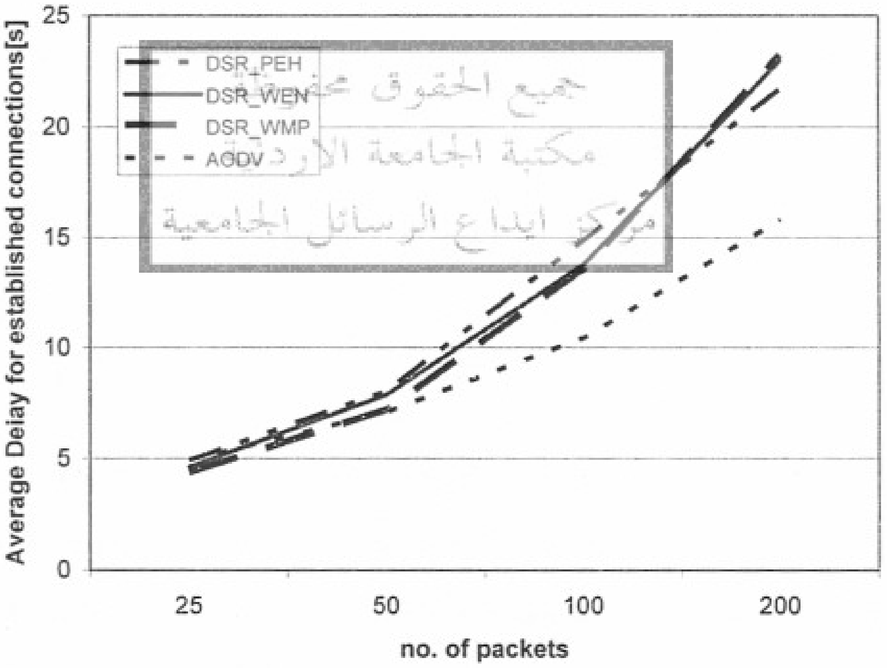


Figure 11: Average Delay for established connections
as function of traffic load

Figure 11 does not reflect the actual delay caused by all connections, rather, the established ones. The Invalid Route Delay captures the delay caused by any invalid route used through out the simulation for established and unestablished connections. It is highly probable that the difference in the behavior of the protocols in this figure and Figure 9, showing invalid route delay, is because invalid routes had a greater effect on unestablished connections. This is confirmed in Figure 13 which shows that the reduction in invalid delay caused by disabling multiple paths in DSR_WMP caused an increase in the number of connections actually established.

The reduction in unestablished connections means reducing the delay caused by trying to reestablish them. So, the decrease in DSR'S invalid delay, caused by disabling multiple paths, does have an affect in reducing overall delay by reducing the number of unestablished connections. The small improvement on the number of established connections caused be applying one entry per destination on DSR only confirms the results shown in Figure 10 that the factor having the greater affect on performance is the preferring fresher route factor.

4.4 Normalized routing overhead

Figure 12 represents the Normalized Routing Overhead. DSR with WEN and DSR with PEH have the exact same curve. The performance of DSR according to this metric degrades after disabling the multiple path service starting with 150 control packets and ending with 350, nevertheless any DSR protocol has a much better performance than AODV at any time. In the first three cases AODV has a smooth elevation, at the last case AODV suffers from a rough increase in the number of control packets. AODV starts its curve with an average of 750 control packets to end it with 2300 control packet for each node.

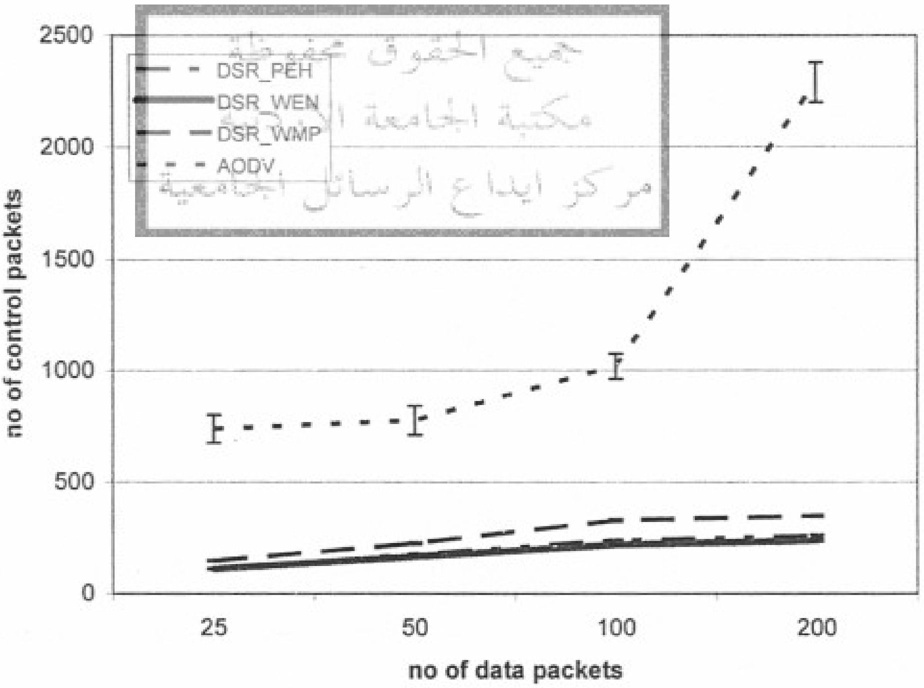


Figure 12: Normalized Routing Overhead as function of traffic load

In the next figure, Figure 13, we illustrate the difference between AODV and DSR without multiple paths in regard to the number of established connections. The number of established connections for AODV is higher than DSR without multiple paths. In this figure we only present the results of DSR _WMP and DSR_PEH since DSR_PEH and DSR_WEN have similar results and DSR without multiple paths has the best of all results comparing to DSR protocols.

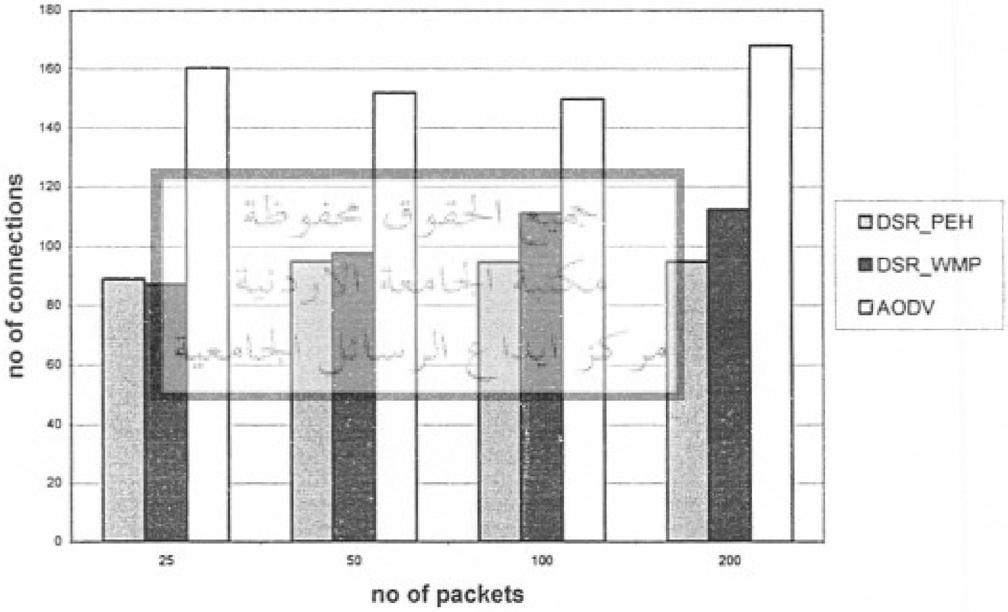


Figure 13: Number of connections established as function of traffic load

5 Observations and Discussions

The simulation results bring out several important characteristic differences between the simulated routing protocols.

•The number of route maintenances

DSR with all it's modifications always has a high number of route maintenances comparing to the number of route requests. Most of these route maintenances are caused by invalid routes. The Multiple path feature in DSR does not affect this metric. A possible explanation is that even when one entry per destination is applied on DSR it still uses intermediate caching of routes more often than AODV; such caching allows dynamic change in the route, so no route request is made. AODV has a stricter route reply caching mechanism which leads us to conclude that AODV will use the route discovery process more often. According to the definition of the Route Maintenance Ratio discussed in this section, an increase in the number of route requests will decrease this metric.

DSR_WMP has less invalid route maintenances than DSR_WEN indicating that applying one entry per destination has better effect than increasing the validity of multiple paths with small transfers. DSR_WEN is applied on the route maintenance mechanism whereas the DSR_WMP caching mechanism is applied on the route discovery mechanism. Applying a caching mechanism on the route discovery process in small transfers is more suitable. The performance of AODV also confirms this conclusion.

Since AODV prefers fresher route and applies one entry per destination it will highly depend on route discovery to get a route to a destination. DSR_WMP's caching mechanism is a partial imitation of AODV's caching mechanism which only applies one entry per destination. Thus the only logical explanation to the difference between DSR_WMP and AODV is the other factor, preferring fresher routes. Route discovery combined with preferring fresher routes reduces the number of route maintenances and the number of invalid route maintenances. A full imitation of AODV's caching mechanism would cause DSR to act like AODV with small transfers.

•Throughput:

DSR has a very low throughput in our chosen environment because DSR uses multiple paths and because DSR doesn't have a way to prefer fresher routes. DSR _WMP has a higher throughput than all the other DSR protocols. This is because it applies route requests more often, increasing the possibility of using more valid routes. The route invalidity ratio shows that DSR_WMP has less invalid routes than all the other DSR protocols. DSR_WMP has a steady invalid route ratio yet its packet delivery ratio falls as more packets are being sent. This is expected since we apply bursts in our traffic pattern, so the network will be overloaded with packets at a certain instant increasing the possibility of packets dropping due to network load. The difference in the number of packets between the applied cases may be to low to have such a fall in the packet delivery ratio; however, the combination of the number of packets being sent at each burst may cause significant difference. DSR_WEN has no significant improvement in throughput possibly indicating that applying a caching mechanism on the route maintenance process is not suitable.

AODV has a very interesting behavior in regard to the packet delivery ratio. It is only logical for AODV to behave in such a way since AODV has an obvious rise in the Invalid Route maintenances Ratio. Invalid Route maintenance is not the only factor, like DSR, AODV is effected by the load on the network even worse AODV increases this load with it's own overhead (see figure 13).

The Invalid Packet Drop Ratio indicates that the invalid routes had the higher weight in determining the behavior of DSR protocols in regard to throughput. This indicates that route maintenances caused by invalid routes is the determining factor in the Packet Delivery Ratio. It was expected that this ratio would be a straight line yet it slightly falls as more packets are being sent by the sources of traffic. This is because the more packets the more load, giving an increasing weight to that factor in regard to packet dropping. As for AODV the Invalid Packet Drop Ratio is very low which is expected since AODV has a lower Invalid Route maintenances Ratio. Similar to DSR, AODVs Invalid Packet Drop Ratio in the first three traffic load cases slowly decreases. At the last case, this ratio suddenly increases implying an abnormal behavior in regard to the balance between packets dropped due to invalidity and those dropped due to load. This abnormal behavior favors this particular ratio (invalidity).

Comparing to [30] we have similar throughput results. AODV outperforms DSR and DSR keeps a steady performance on all traffic load cases even though we have relatively different traffic patterns. However, we suggest a different explanation in regard to DSRs steady throughput performance. Our explanation is based on studying actual simulation results. On the other hand, their explanation is based on speculation.

Paper [16] studies DSR with one cache entry per destination. Similar to our results this paper concludes that DSR_WMP has better throughput then the original DSR protocol. This paper studies DSR_WMP in a completely different context yet the fact remains that DSR_WMP has better performance than Base DSR according to this metric.

•Delay:

The Delay results presented in the last section exhibit AODV's advantage over DSR. AODV has the best performance in regard to the Delay caused by invalid Routes as well as the best overall delay performance. Both protocols start with the same overall delay for all established connections. With more traffic load AODV seems to outperform DSR. The reason behind this performance is that DSR uses packet salvaging which increases the overall delay. Even though salvaging is used in all traffic load cases it has a higher probability when more packets are being sent.

Furthermore, AODV has a larger number of established connections in all cases. A possible explanation for this behavior is the short Invalid Route Delay (Figure 9).

The best performing DSR protocol is DSR_WMP. This protocol managed to reduce 30% of the difference in invalid delay between DSR and AODV. This protocol reduces the number of invalid routes used and so reduces the number of invalid route maintenances causing this improvement in invalid delay. This has no effect on the overall delay for established connections. However, it may have helped increase the number of established connections. If the unestablished connections were to be re-established, a reduction in the number of unestablished connections would reduce the delay required to re-establish them.

The slight improvement caused be applying one entry per destination on DSR only indicates that the factor having the greater affect on performance is the preferring fresher route factor.

Similar results have been published in paper [16]. The researchers of this paper study DSR with one cache entry per destination. DSR_WMP according to this paper has better latency than the original DSR protocol. The Author of this paper did not highlight the relationship between the route maintenance process and caches as we do. As a result, our metrics and the way they are conducted tend to differ according to the context.

Researchers of [14] study DSR_WEN and presents results on the delay using this protocol. Delay is studied as a function of pause time and traffic load. The traffic load studied in that paper does not relate to the traffic load in our study.

Similar results have been presented in [9]. In this paper researchers only study AODV's performance over different connection lengths. This paper provides an general explanation of AODV's performance with small transfers.

•Normalized routing overhead:

DSR outperforms AODV when it comes to normalized routing overhead. The Multiple path feature in DSR does not affect this metric. A possible explanation is that even when one entry per destination is applied on DSR, it still uses intermediate caching of routes more often than AODV; such caching will allow dynamic change in the route, so no route request is made. AODV has a stricter route reply caching mechanism which leads us to conclude that AODV will use the route discovery process more often. Normalized routing overhead is proportional to the number of route requests.

The number of connections has a direct relationship with Normalized routing overhead in this context. From the DSR Route Validity Ratio, we infer that most of the routes taken to a destination are invalid. Consequently, all packets may drop due to invalid route maintenances causing an unestablished connection. For DSR with multiple paths most of these invalid routes are alternative route cache hits. For DSR without multiple paths most of these invalid routes could be routes dynamically stored by intermediate caching. This is highly probable since we apply a bursty traffic pattern. As a result, for all DSR protocols, relatively a small number of route requests are made. Route requests have the highest weight in overhead since they flood the network. This explains the low overhead caused by all DSR protocols comparing to AODV.

In many papers presented, AODV always has more overhead than DSR [2] [25] [30]. A variety of studies have been made on the Normalized Routing Overhead regarding DSR and AODV. This metric has been studied as a function of pause time and traffic load. The results always show DSRs advantage over AODV.

Paper [14] studies DSR_WEN and presents results on Normalized routing overhead using this protocol. In this paper, Normalized routing overhead is also studied as a function of pause time and traffic load. However, the traffic applied in our study is lower than the traffic applied in that paper.

•Burst Traffic Pattern

Applying bursts to the traffic pattern had a great effect on all the metrics presented. If all connections attempted to be established at the same time, the load on the network in that instant would be very high comparing to a more steady traffic pattern. This increases the possibility of packets being dropped due to load, affecting the Packet Delivery Ratio and the Invalid Packet Drop Ratio.

If all route requests where to be sent in a tight time interval (Burst) the route replies will also come back in a tight interval. All route replies will have the same probability of being valid since they reflect the network topology for that same tight interval. Between every burst and another there is a specified silence period. During this period and according to the mobility rate chosen in our environment, these routes will simply become invalid.

6 Conclusion

Even though the performance evaluations of ad-hoc routing protocols with small transfer traffic has been investigated in a previous study [30], they present a possible explanation on DSR's throughput behavior with no further elaboration on that matter. In this Thesis, we simulate DSR, DSR with modifications and AODV with short stream traffic according to a specific traffic pattern. This pattern was designed specifically in order to evaluate the relationship between small transfers and route maintenance. In this small transfer traffic pattern, DSR's aggressive caching mechanism increases the number of route maintenances caused by invalid cache entries. Such an increase has a negative effect on delay and throughput. However, aggressive caching (Multiple paths) is not the only factor involved in this negative effect on performance. DSR also has no way to prefer fresher routes. Because of this, DSR will maintain an invalid route and use it even if another route is fresher. Using these invalid routes will also increase the number of route maintenances caused by invalid cache entries. This has a larger effect on delay and throughput than the earlier explanation. Based on a caching mechanism that can prefer fresher routes and allows one entry per destination, DSR will have better throughput and delay performance with small transfers.

AODV's throughput decreases as traffic increases. AODV has better delay results as traffic increases, in the context of small transfers, since most of its routes are valid. As a result, unnecessary route maintenances caused by invalid cached entries and the degradation in performance caused by them is avoided. AODV often has better performance than DSR when it comes to small transfers. This performance comes with a cost of much larger overhead than DSR as it often invokes expensive route discovery phases.

DSR_WEN has no significance in increasing the throughput or reducing delay. According to our results, this silence can only indicate the unsuitability of applying a caching mechanism on Route Maintenance with small transfers. Furthermore, although the caching mechanism was applied on Route Discovery in DSR_WMP it was not intensely effective since it did not prefer fresher routes.

Route Maintenance in DSR and AODV with longer Transfers

Many caching mechanisms and optimizations on caching mechanisms are applied on the route maintenance process, as shown in Chapter 2 Section 5.3. Applying a caching technique on one of the two basic mechanisms of a reactive protocol, route discovery and route maintenance, is usually based on a functionality achieved by that basic mechanism. When choosing one of the basic mechanisms, the functionality and the suitability of that basic mechanism must be taken into consideration.

In Chapter 3, we have shown that AODV's caching mechanism, which mostly relies on the route discovery process, has much better performance with small transfers than the DSR caching mechanism which relies mostly on the route maintenance process.

In order to apply a fair study, the suitability of applying a caching technique on the route maintenance mechanism must be studied over a different range of connection lengths. In this chapter, we study the suitability of integrating a caching mechanism in the route maintenance process. In Section 1, we provide a description of the simulated problem. Section 2 describes the communication pattern applied on the protocols. Section 3 illustrates the simulation results and brief discussion, further observations and discussion is presented in Section 4. In section 5, we show our conclusions.

1 The Simulated Problem

The simulated protocols:

1- AODV: implements a caching mechanism on the route discovery process with one entry per destination and uses a sequence number to prefer fresher routes. It also implements the basic method of deleting a route after receiving the unicast route error packet submitted to inform the source that the link is broken [21].

2- DSR_WEN: implements the basic DSR mechanism that relies on caching multiple paths to a destination. The basic DSR caching mechanism does not have a way to expire old routes and has no way of preferring fresher routes. DSR_WEN is an optimization on the basic route maintenance process that applies an improved error handling algorithm [7].

This chapter contains a comparative analysis, studying the DSR_WEN and AODV protocols in regard to the suitability of applying the caching mechanism on a route maintenance process. This is done by studying the behavior of a protocol with such characteristics, DSR_WEN, according to metrics that reflect the basic relationship between the caching mechanism and the route maintenance process.

WEN has a specific characteristic over all other optimizations presented in Chapter 2 Section 3.2.1 of this thesis. This mechanism is applied solely on the route maintenance process and it is invoked with the route maintenance process as it is integrated in the way the route error packet is handled [14]. This specific character is what defined our choice in studying this caching optimization in particular.

We also study the behavior of the two protocols according to the communication pattern we used in Section 2.3, but with longer transfers. Such a study will give us a better idea on the actual effect of such a pattern in a more generic sense.

The simulation model is identical to that presented in Chapter 3. we only present in this chapter some variations on the communication pattern to suit the specification of the higher traffic load and longer connection length.

2 Communication Model

In Chapter 3, to be able to successfully study the behavior of the route maintenance process in an environment of small transfers, we generated a specific communication pattern. In Chapter 4, we present further analysis of the route maintenance caching relationship in longer transfers according to the same environment. However, some adjustments needed to be made. Longer periods needed to be placed between one burst and another so that the traffic load actually reflected the number of packets in study .A communication pattern based on 8 bursts with 300 seconds between every burst and another is chosen corresponding to 30 traffic sources at each burst.

3 Simulation results

3.1 Route Maintenance Percentage

Figure 14 shows the Route Maintenance Percentage as a function of traffic load (number of data packets sent by a traffic source). For DSR the Route Maintenance Percentage is independent of the traffic load, having a high percentage, 98% in all cases. There is a significant difference between AODV and DSR, for AODV the Route Maintenance Percentage is much lower for all traffic loads ranging between 61% to 53% .The confidence interval is hardly noticeable exhibiting a 90% confidence in the results shown in this figure.

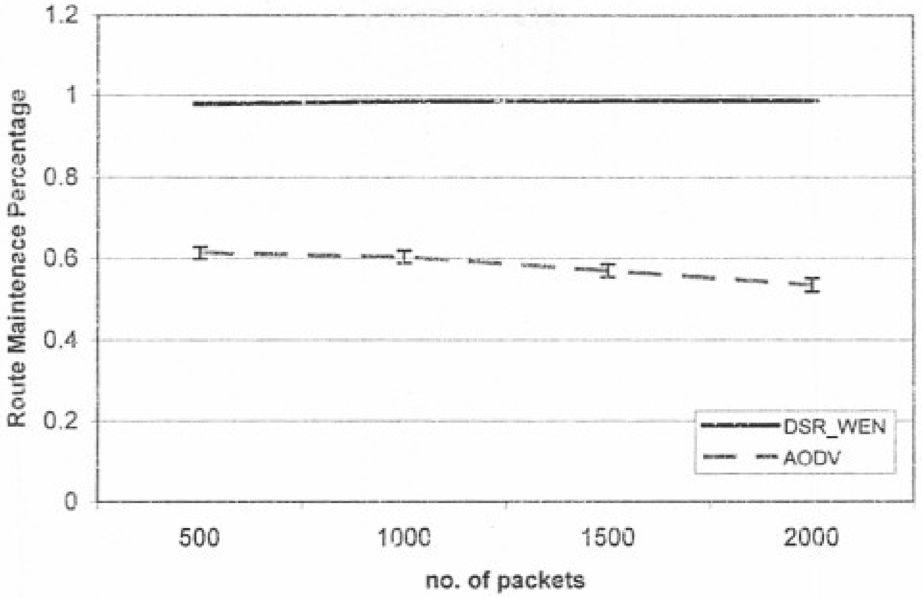


Figure 14: Route Maintenance Percentage as function of traffic load

It was expected that the behavior of AODV in this figure would follow its behavior in Figure 4, in Chapter 3; however, it does not. This can only confirm [30] suggestion, that these protocols have completely different behavior when it comes to small transfers. The AODV protocol seems to divert its logical behavior somewhere between 200 and 500 packets. This behavior depends exactly on the number of route discoveries. A possible explaination is that the number of route maintenances to route discoveries increases with small transfers in AODV up to a certain point then decreases. The DSRs route maintenance percentage increases with both small transfers and longer transfers.

3.2 Throughput

Figure 16 shows the Packet Delivery Ratio for DSR and AODV as a function of traffic load. Even though AODV has a higher packet delivery ratio than DSR with small transfers, as shown in chapter 3, AODVs packet delivery ratio severely falls as more data packets are being sent. DSR seems to outperform AODV with longer transfers in regard to throughput. Again AODV has extremely different behavior with smaller and longer transfers. DSR has a continuous steady behavior in all cases. Indicating that if a node is in a busty-mixed environment and had a choice of only one protocol, DSR would be a more realistic protocol.

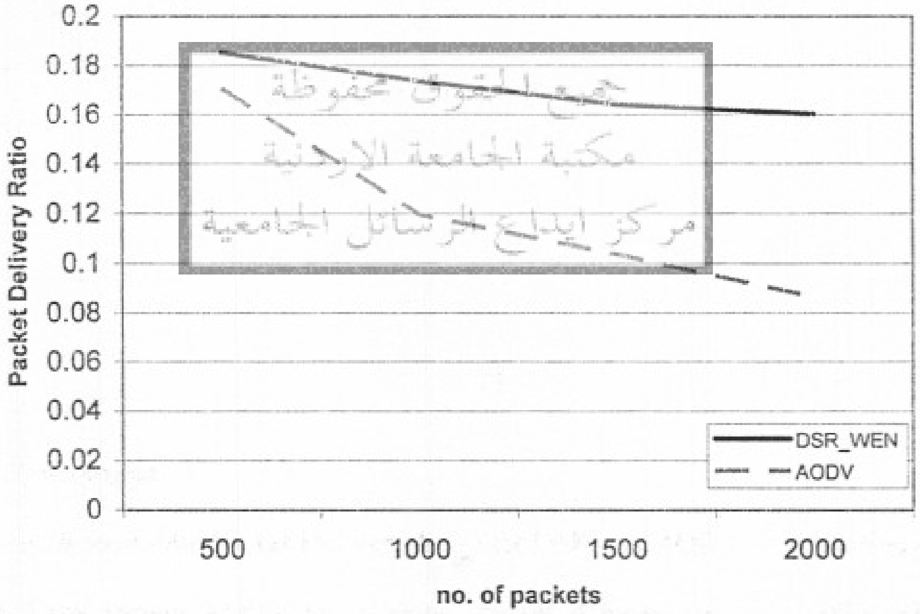


Figure 16: Packet Delivery Ratio as function of traffic load

Figure17 represents the Dropped Packet Invalidity Ratio from intermediate nodes' point of view. The number of invalid dropped packets for all DSR protocols is very high in all cases with respect to the overall number of dropped packets. DSR has an Invalid Packet Drop Ratio ranging from 89% to 93% indicating a strong relationship between packet delivery ratio and invalid route entries. AODV has a much lower ratio from the intermediate nodes perspective. In Chapter 3, AODV with small transfers has similar behavior except when 200 packets were being sent. This shows that route invalidity seems to affect more at this specific point. It then diverts again and decreases with lower ratio at 500 packets logically fitting the behavior of AODVs Route Maintenance Ratio.

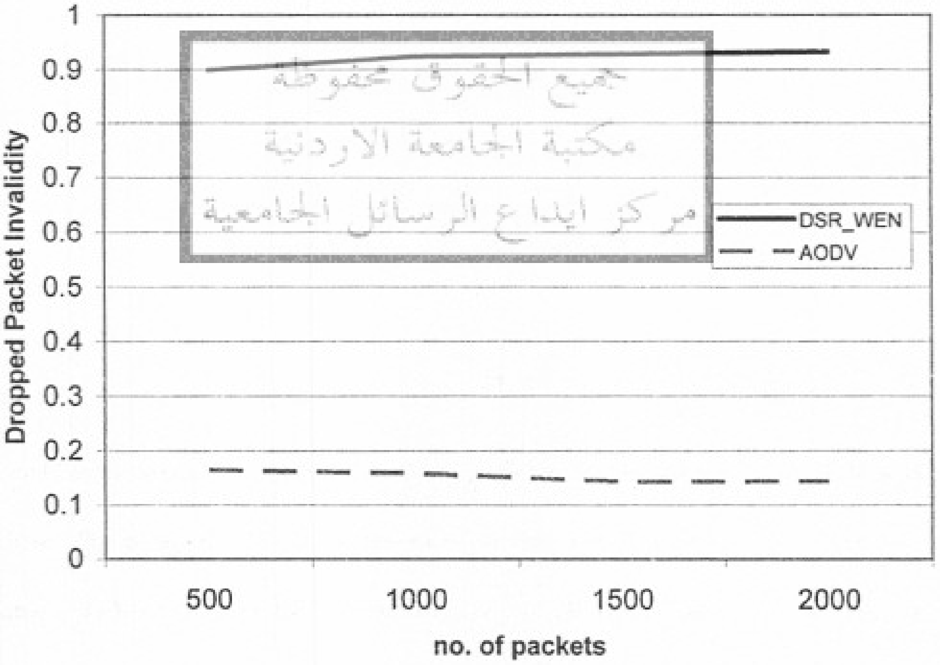


Figure 17: Dropped Packet Invalidity Ratio (intermediate) as function of traffic load

The previous analysis indicates that only with 200 packet traffic load does route invalidity have a higher weight than usual in determining AODV's packet delivery Ratio, as shown in chapter 3. As a result, we may conclude that in every other case, the load on the network was a strong factor in determining AODV's packet delivery Ratio behavior. AODV is much more vulnerable to traffic load because of the additional routing overhead it produces, see section 3.4. DSR is vulnerable to invalid routes when it comes to the delivery ratio. This definitely explains AODV's severe change as traffic load increases and the steady change in DSRs throughput performance.

3.3 Delay

Figure 18 shows the Invalid Route Delay as a function of traffic load. For DSR with WEN the delay caused by invalidity is high. AODV has better performance in regard to this metric. The different protocols follow a similar behavior as the traffic load increases. These results follow the results presented in Chapter 3, with small transfers. Figure 18 also shows that the difference in DSR's behavior and AODV's behavior becomes more obvious as traffic load increases. The results shown if Figure 18 confirms the following analysis on Figure 19.

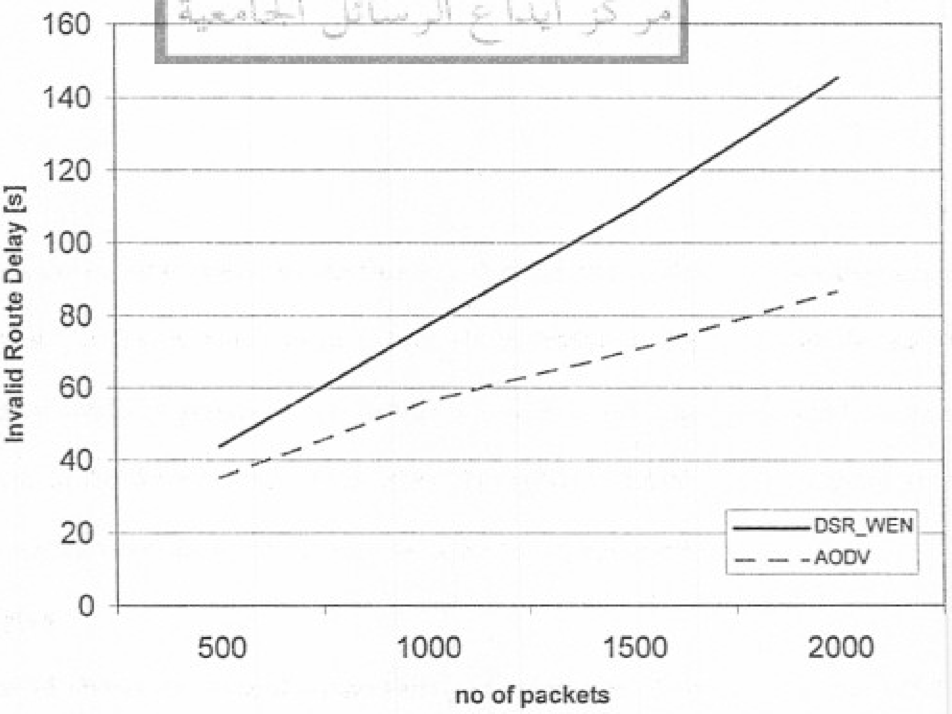


Figure 18: Invalid Route Delay as function of traffic load

In Figure 19, the average delay for the established connection is shown. AODV has less Delay than DSR in all cases. The difference in AODV's behavior and DSR's behavior become more obvious as load increases. Such results confirm the previous results on delay caused by invalid routes.

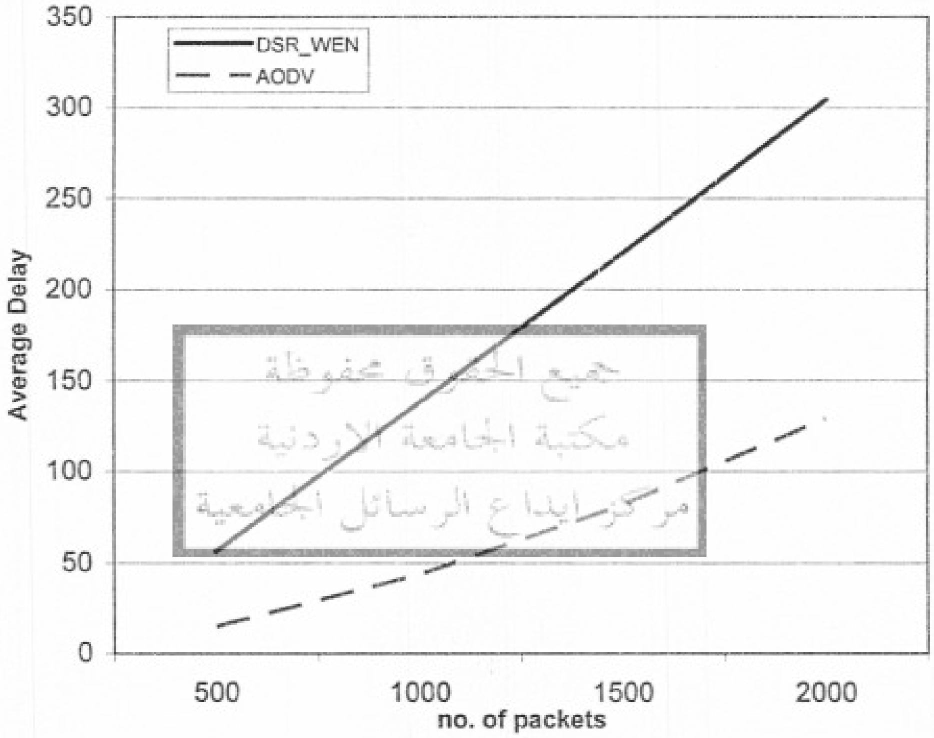


Figure 19: Average Delay for established connections
as a function of traffic load

In Figure15, we present the results of the Invalid Route Maintenance Ratio. For DSR, 87% to 93 % of route maintenances are actually invalid. This ratio is high for all traffic loads. For AODV , the value of this ratio is always lower than DSR ranging from 42% up to 50%. Both protocols have a steady increase at the beginning to later settle at a specific ratio at the end of the curve. The confidence interval obviously illustrates the accuracy of the results exhibited.

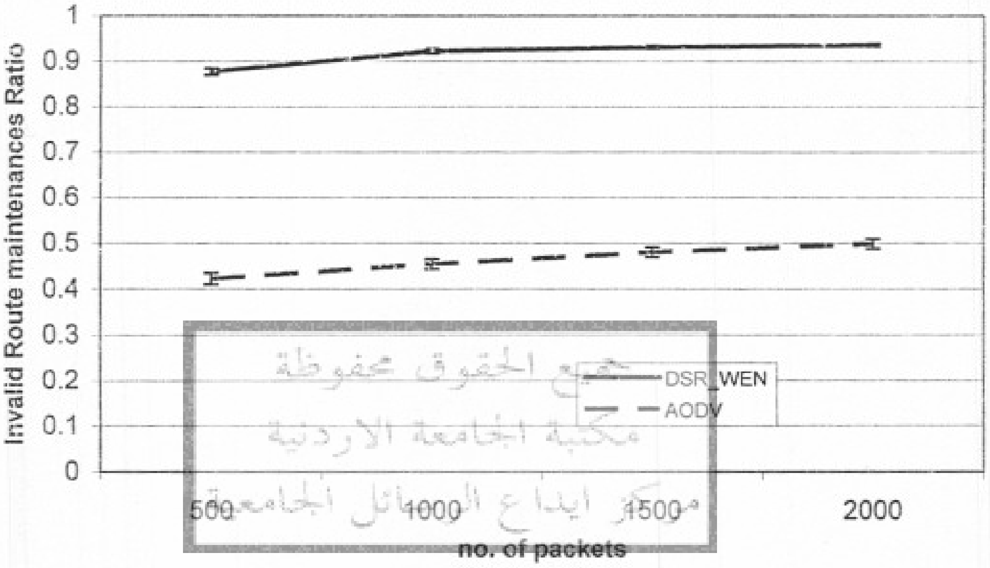


Figure 15: Invalid Route maintenances Ratio as function of traffic load

In Chapter 3, AODV's Invalid Route Maintenance Ratio has a 20% rise between 25 and 200 packets with small transfers. In this chapter, the invalid Route Maintenance Ratio has a 10 % rise over a much larger difference in the number of packets with longer transfers. Therefore, invalid route maintenance ratio has a higher rise with smaller transfers indicating a higher weight on the route maintenance ratio. With larger transfers more and more route discovery processes are invoked possibly increasing the weight of load on AODV's behavior. This explains why AODV has different behavior with small and long transfers.

In chapter 3, we presented the effect of unestablished connections on the delay results. The two protocols presented in this chapter, AODV and DSR_WEN, have different number of established connections with long transfers too; these results are exhibited in Figure 20. Again AODV has the higher number of connections established, however, with more traffic load DSR seems to catch up with AODV in regard to the number of established connections.

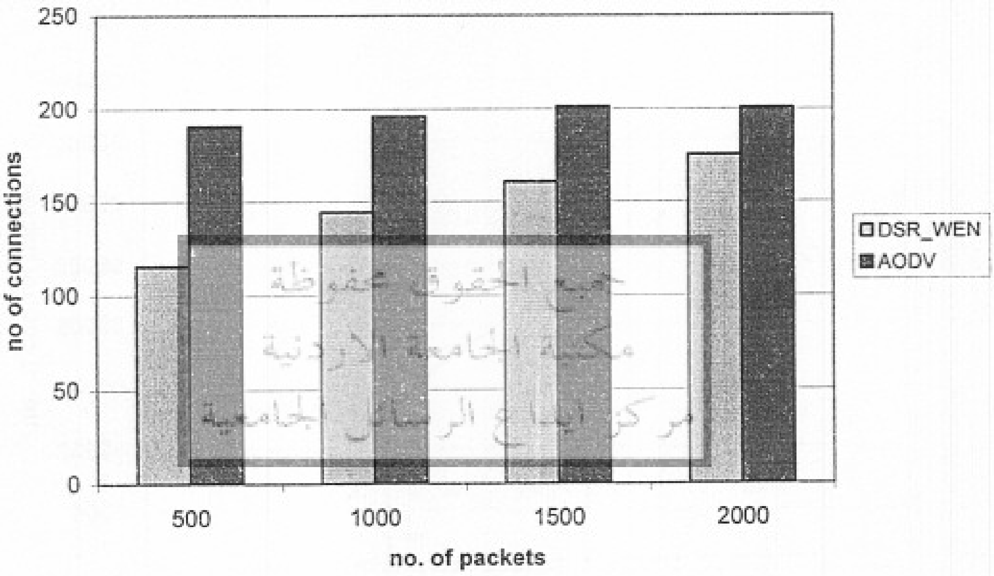


Figure 20: Number of connections established as function of traffic load

The difference in the number of connections established between AODV and DSR was more noticeable with smaller transfers. Therefore, the difference in the number of connections actually established with long transfers has less effect on the delay results than it did with small transfers. Thus, with more traffic load the invalid delay presented in figure 18 will be more involved with the overall delay results presented in figure 19.

4 Observations and Discussions

The simulation results bring out several important characteristic differences between the simulated routing protocols.

•The Route Maintenance Percentage

For DSR, the number of route maintenances is very high most of which are invalid. This is true for both long and small transfers. The more traffic the more route maintenances and invalid route maintenances. This implies that the dominant factor in DSR_WEN is the validity of routes. This is exactly why DSR has a steady performance according to all metrics.

For AODV, the number of route maintenance to all mechanisms rises with small transfers and has a corresponding rise in invalid route maintenances. As more traffic load is applied on the network AODV has a different behavior, the number of route discoveries reduce the route maintenance ratio as it becomes the dominant factor. This explains the divergence in AODVs behavior after 200 packet traffic load according to the throughput metrics.

•Throughput:

For DSR, the throughput metrics show that the dominant factor on throughput is Invalid Dropped Packets. The invalidity of routes is a general characteristic of DSR protocols, this is why DSR has a steady decrease in throughput as the traffic load increases. DSR_WEN has better performance than AODV in regard to this metric with longer transfers. This indicates that having multiple paths and no way to prefer fresher routes and using the WEN optimization is more suitable than the caching mechanism applied in AODV. This is true because the AODV caching mechanism is the exact opposite of DSR_WEN caching mechanism.

For the same exact reason, AODV's caching mechanism is more suitable than DSR_WEN's caching mechanism with small transfers. The number of route discoveries produced by AODV, because of the nature of its caching mechanism, with small transfers is tolerable. As a result, it did not have that much effect on throughput . With more traffic load the routing overhead causes a significant degrade in AODV's throughput performance. These suggestions are confirmed by the Invalidity Dropped packet ratio for both small and longer transfers (see figure 8 and figure 17). This ratio shows in general that load is the dominant factor on AODV's throughput behavior.

Similar conclusions on the behavior of protocols in regard to longer transfers have been presented in other studies [25] [5] [2] [9]. Researchers of [9] also investigate the behavior of only AODV in regard to small transfers. However, the conclusion in paper [9] only indicated a strong relationship between this performance and caching efficiency, which has already been presented in [30]. Both these studies do not present an exact explanation for this behavior as presented in our thesis.

•Delay

Regardless of the traffic load and connection length, AODV always has better performance when it comes to delay. For AODV, there are two major elements effecting delay: the average number of invalid routes taken in a specific connection, and the delay caused by route discovery.

The route discovery latency will be much higher than DSR since the DSRs route maintenance ratio is much higher than AODVs. However, this latency will have less effect than expected since we applied the bursty nature in our communication environment. With such traffic, most of the route discoveries will be in a relatively short interval of time, which gives a higher chance for constant intermediate caching.

Intermediate caching will change the route dynamically, possibly even before a route error packet is handled by the source. Such dynamic change will reduce the number of route discoveries.

Unlike DSR, AODV has a controlled caching mechanism that relies on fresher routes; this will reduce the number of intermediate caching applied.

As a result, DSR will apply more intermediate caching than AODV. This feature in DSR compound with the multiple Path feature, will reduce the Route Discovery latency. So, the major factor effecting DSR connection Delay is the average number of invalid routes for a specific connection. Another factor affecting the delay on DSR is packet salvaging which helps in increasing the average connection delay. Similar behavior in Figure 18 and 19 confirm these suggestions.

The load on traffic does not have that much effect on delay. This is why AODV has the best performance in small and longer transfers. We also conclude that the number of route maintenances are the determining factor in delay. This is why DSR has longer delay in all cases with both small and longer transfers. The invalidity in routes has a strong effect on delay since most of the route maintenances are actually caused by invalid routes. According to this metric, it would be more suitable to apply a caching mechanism similar to AODV's caching mechanism that places most of the caching process on the route discovery process.

- **Normalized routing overhead**

AODV has more overhead than DSR. These results are similar to many papers that studied the performance of these two protocols [2] [5] [9] [25]. The results presented also confirm the results and observation in all the other metrics presented in this thesis.

5 Conclusion

In this chapter, in order to apply a fair study, we simulate DSR_WEN and AODV with longer transfers according to the same specific traffic pattern used in Chapter 3. With longer transfers, DSR's aggressive caching mechanism increases the number of route maintenances caused by invalid cache entries. Such an increase has a negative effect on delay and throughput. However, the effect on DSR's throughput is not as large as the effect of the additional load caused by AODV's route discovery process on AODV's throughput. Thus, the conclusion we reached in Chapter 3 that says: "Based on a caching mechanism that can prefer fresher routes and allows only one entry per destination, DSR will have better throughput and delay performance" is only true with smaller transfers. DSR's caching mechanism, with its flaws, has better effect on the performance of the protocol than AODV's caching mechanism in regard to the throughput metric only. However, this conclusion stands for the delay metric.

Since the caching mechanism has so much effect on latency and throughput and there are two factors involved in the caching mechanism(one entry per destination factor and the preferring fresher route factor), we can't conclude that multiple path is what decreases or increases the route discovery latency.

Conclusion and Future Work

In this Thesis, we simulate DSR, DSR with modifications and AODV with short stream traffic according to a specific traffic pattern. We also simulate DSR_WEN and AODV with longer transfers. The pattern was designed specifically in order to evaluate the relationship between small transfers and route maintenance. To compare the results of small transfers with longer transfers, the same pattern was applied on the longer transfers communication model. The most important results are presented in Table 4.

With small transfers, AODV outperformed DSR. An explanation of this behavior was presented in [28]. This explanation suggested that the reason behind this degradation in DSR's performance is due to multiple paths. After disabling this feature in DSR, the improvement in DSR's performance was not extremely noticeable. Because of the nature of our implementation of DSR_WMP, we concluded that the only remaining factor effecting DSR's performance is its inability to prefer fresher routes.

Furthermore, we conclude that the reason behind AODV's performance with small transfers is not having one entry per destination as presented in [28], rather, its ability to prefer fresher routes.

The ability to prefer fresher routes is a feature applied on the route discovery process and so is the ability to maintain multiple routes to the same destinations. These are the major factors affecting the performance of protocols in small transfers. DSR_WEN is a method that provides wider propagation of route errors which eventually increases the validity of routes. This method caused no significant improvement in the protocol performance. This method is integrated in the route maintenance process. Therefore, it is more suitable to place larger weight on specifically designing a caching mechanism in order to integrate that caching mechanism on the route discovery process with small transfers.

Table 4: Simulation Results of Long and Short Transfers

Enviroment	Protocol	Route Maintenance Ratio	Packet Delivery Ratio	Invalid Delay[second]	Normalized Routing Overhead(control messages)
Long Transfers	AODV	97%-98%	17%-8%	14-128	5648-38304
	DSR_WEN	61%-53%	18%-16%	56-304	483-1100
Small Transfers	AODV	47%-63%	49%-18%	0.1-5	737-2289
	DSR_PEH	91%-95%	31%-16%	2-14	112-258
	DSR_WEN	89%-95%	30%-16%	2-15	111-237
	DSR_WMP	88%-93%	32%-20%	1-11	148-348

With small and longer transfers, DSR has a steady performance; this is because DSR only relies on caching optimizations applied on the route maintenance process. The number of route maintenances is high and the number of invalid route maintenances is high. The invalidity of routes seems to be the dominant factor on the performance of DSR. This is why DSR is open for improvement. Increasing the validity of routes in caches or applying a caching mechanism similar to AODV's is a choice DSR can make. As for AODV such a caching mechanism is partially enforced; AODV can only have one entry per destination. However, since our results show that the factor causing AODV's performance degradation was its ability to prefer fresher routes, AODV is also open for improvement.

DSR had better throughput performance than AODV with longer transfers. This is because AODV produces so much routing overhead that it increases the load on the network affecting its own throughput. The affect of route maintenance in general and invalid route maintenance in particular on DSR's performance was not as strong as the affect of the AODV's routing overhead on its performance.

A protocol would have the best performance if it applied a caching mechanism similar to AODV's with small transfers, and a caching mechanism similar to DSR's with longer transfers. In this case, the dominant factor on performance with small transfers would be the load on the network, which is very low, and the dominant factor on performance with longer transfers would be invalid route maintenances, which depends on the efficiency of the caching mechanism in reducing the number of invalid routes.

Since the caching mechanism has so much affect on latency and throughput and there are two factors involved in the caching mechanism (one entry per destination factor and the preferring fresher route factor), we can't conclude that multiple path is what decreases or increases the route discovery latency.

1 Future Work

We have planned two future work:

- Applying a caching mechanism on AODV that does not prefer fresher route. Study the affect of such an implementation on the number of route maintenances and the number of invalid route maintenances. It is expected that such an implementation will show degrade in AODV's performance with small transfers due to an increasing number of invalid route maintenances. It is also expected that this implementation will improve the performance of AODV with longer transfers since it will reduce the number of route discoveries and so reducing the routing overhead.

- Applying two kinds of implementations on DSR. The first is to allow DSR to prefer fresher routes with one entry per destination, and study the protocol behavior with both small and long transfers. The second is to allow DSR to prefer fresher routes with multiple paths. This will give a better indication to the weight of the two factors (one entry per destination , ability to prefer fresher routes) effecting the performance of the protocols. We will also study the effect of such implementations on the number of route maintenances and invalid route maintenances.

References

Lokesh Bajaj, Mineo Takai, Rajat Ahuja, Ken Tang, Rajive Bagrodia, and Mario Gerla;"**GloMoSim: A Scalable Network Simulation Environment"**, UCLA Computer Science Department Technical Report 990027, May 1999.

Josh Broch, David A. Maltz, David B. Johnson ,Yih-Chun Hu and Jorjeta Jetcheva, "**A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols"**, ACM Press, 2002.

Kevin Fall and Kannan Varahan, editors. "**NS Notes and Documentation"**. The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC, November 1997.

Yih-Chun Hu, David B. Johnson," **Ensuring cache freshness in on-demand ad hoc network routing protocols"**. POMC 2002: 25-30

Per Johansson, Tony Larsson, Nicklas Hedman, Bartosz Mielczarek, and Mikael Degermark. **"Scenario-based Performance Analysis of Routing Protocols for Mobile Ad-hoc Networks"**. In Proceedings of the Fifth Annual International Conference on Mobile Computing and Networking (MobiCom 1999), pages 195ñ206, August 1999.

David B. Johnson and David A. Maltz, "**Dynamic Source Routing in Ad Hoc Wireless Network"**, In Mobile Computing, edited by Tomasz Imielinski and Hank Korth, Chapter 5, pages 153-181, Kluwer Academic Publishers, 1996.

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.

David B. Johnson, David A. Maltz, Yih-Chun Hu, **"The Dynamic Source Routing Protocol"**, IETF MANET Working Group, INTERNET-DRAFT 15 April 2003 .

Tuulia Kullberg. **"Performance of the Ad-hoc On-Demand Distance Vector Routing Protocol"**. CiteSeer, 2004

B. Liang and Z. J. Haas, "**Optimizing Route-Cache Lifetime in Ad Hoc Networks**" , Proceedings of the 22th IEEE **INFOCOM**, San Francisco, CA, April 1-3, 2003

Yu Liu, Jack Lau, "**An Analysis of TCP over Different Routing Protocols in Ad hoc Networks,**" accepted for presentation in The Mobile Ad Hoc Network 2002 on March 2002.

Joseph Macker and Scott Corson, "**IETF Mobile Ad Hoc Networks (MANET) Charter**",http://www.ietf.org/html.charters/manet-charter.html.

David A. Maltz, Josh Broch, Jorjeta Jetcheva and David B. Johnson," **The Effects of On-Demand Behavior in Routing Protocols for Multihop Wireless Ad Hoc Networks**" .IEEE Journal on Selected Areas in Communications Special Issue on Mobile and Wireless Networks, pages 1439-1453, August 1999.

M. K. Marina and S. R. Das, "**Performance of Route Caching Strategies in Dynamic Source Routing**", Proceedings of International Workshop on Wireless Networks and Mobile Computing (WNMC) in conjunction with IEEE International Conference on Distributed Computing Systems (ICDCS), April 2001.

Padmini Misra, "**Routing protocols for ad hoc mobile wireless networks**", Available at http://www.cis.ohio-state.edu/~jain/cis788-9/adhoc_routing/index.html. 2/7/2000 10:38:34 AM

N. Panchal and N. B. Abu-Ghazaleh, "**Active Route Cache Optimization for Ad Hoc Routing Protocols**", Infocom 2002.

V. Park, and S. Corson, "**Temporally-Ordered Routing Algorithm (TORA) Version 1 Functional Specification**", Internet draft. 1997. Downloaded from http://www.ics.uci.edu/~atm/adhoc/paper-collection/papers.html

Seungjin Park and Brian VanVoorst ,"**Anticipated Route Maintenance (ARM) in Location-Aided Mobile Ad Hoc Networks**", SAW02-4, IEEE GlobeCom 2001.

Charles Perkins," **Ad Hoc On Demand Distance Vector (AODV) Routing**", IETF Mobile Ad Hoc Networking Working Group, INTERNET DRAFT, Sun Microsystems, 20 November 1997.

Charles E. Perkins, Elizabeth M. Belding-Royer," **Ad-hoc On-Demand Distance Vector Routing"**. WMCSA 1999: 90-100.

Charles E. Perkins, Elizabeth M. Royer**, "The Ad Hoc On-Demand Distance-Vector Protocol"**, in Ad Hoc Networking, edited by Charles E. Perkins, Chapter 6, pp. 173-220, Addison-Wesley, 2001.

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

Charles E. Perkins, "**Ad Hoc Networking An Introduction** ", in Ad Hoc Networking, edited by Charles E. Perkins, Chapter 1, pp. 1-28, Addison-Wesley, 2001.

Peter Pham**." Performance Analysis of Reactive Shortest Single-path and Multi-path Routing Mechanism With Load Balance".** INFOCOM 2003

Liang Qin**,** Thomas Kunz, **"PRO-ACTIVE ROUTE MAINTENANCE IN DSR"**, ACM Press, 2002.

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

Martha Streenstrup, Bolt Berank & Newman, "**Cluster-Based Networks**" , in Ad Hoc Networking, edited by Charles E. Perkins, Chapter 4, pp. 75-138, Addison-Wesley, 2001.

Andrew S. Tanenbaum. (2003)." **Computer networks".**(4th ed).Upper Saddle River, New Jersey 07458: Prentice Hall PTR

Venugopalan Ramasubramanian, Zygmunt J. Haas and Emin Gun Sirer. **"SHARP: A Hybrid Adaptive Routing Protocol for Mobile Ad Hoc Networks"** In Proceedings of the ACM Symposium on Mobile Ad Hoc Networking and Computing (Mobihoc), Annapolis, Maryland, June 2003.

S. Wang, A. Helmy, "**Effects of Small Transfers and Traffic Patterns on Performance and Cache Efficacy of Ad Hoc Routing"**, ACM MOBICOM Conference, San Diego, CA, September 2003.

Erik Weiss, Markus Frewel, Guido Hiertz, Bangnan Xu. (2004). **"Improving Ad Hoc Routing for Future Wireless Multihop Networks"**, Proccedings of the 5th European Wireless Conference: Mobile and Wireless Systems beyond 3G, pp. 358-363.

Shih-Lin Wu, Sze-Yao Ni, Yu-Chee Tseng, and Jang-Ping Sheu, **"Route Maintenance in a Wireless Mobile Ad Hoc Network"**, telecommunication Systems, vol. 18, issue 1/3, September 2001, pp. 61-84. (SCI)

Liangzhong Yin and Guohong Cao, **"Supporting Cooperative Caching in Ad Hoc Networks"**, IEEE INFOCOM, 2004.

Xiang Zeng, Rajive Bagrodia, Mario Gerla; **"GloMoSim: a Library for Parallel Simulation of Large-scale Wireless Networks"**, Proceedings of the 12th Workshop on Parallel and Distributed Simulations -- PADS '98, May 26-29, 1998 in Banff, Alberta, Canada

Hongbo Zhou, Matt W. Mutka, and Lionel M. Ni, **"IP Address Handoff in the MANET"**, to appear in Proceedings of the 23rd Conference of IEEE Communication Society (INFOCOM 2004), Hong Kong, China, March 2004.

OPNET simulator homepage. Retrieved July 8, 2004, from **http://www.opnet.com**.

APPENDIX

1. Using the simulation package GloMoSim

The following configuration was used for performing all the simulations described in this Thesis:

Computer: Pentium 4, Dell Optiplex Gx240, 128MB RAM

Operating System: Microsoft windows XP Professional

GloMoSim version: GloMoSim 2.02

2. How to install GloMoSim on windows XP

- Download the GloMoSim tar file
- Unzip the tar file
- Extract the folder into any path.
- Create a path called c:\glomosim
- Place everything in the unzipped GloMoSim folder into the newly created path
- Create a path called c:\parsec
- Open the unzipped parsec folder, choice the suitable platform folder; place everything in that folder in the newly created parsec folder.
- Create an environment variable called PCC_DIRECTORY with the value of C:\parsec (the parsec path)
- Place the "parsec path" in your path
- a C compiler is also a requirement including it's associated environment variables

3. GloMoSim simulation model

The simulation environment needs to be set-up carefully to produce meaningful results. The simulation set-up is called 'simulation scenario' which is the input to the simulation model. In GloMoSim this is usually provided by a file called "config.in". In general, a simulation scenario consists of three different main components (or dimensions).

1)Routed Topology

- Defines the number of nodes and their connectivity
- Defines the links connecting the nodes, their bandwidth, delay, queuing discipline and other characteristics
- Defines the routing protocols running in the nodes in terms of unicast and multicast routing

2)Connections, Traffic and Agents/end-host protocols

- Agents are end-host protocols, such as TCP, ftp, Telnet, UDP.. etc.
- Traffic source can be: (uses a traffic model)
    - constant bit rate (CBR) at rate R
    - with distribution (Poisson or Pareto)
    - trace driven
- Connection definition includes spatial distribution of connections (TCP sender/receiver, or group members)

3)External events and Failures

- External events include the temporal distribution of connections:
    - start and end of TCP connections
    - join/leave patterns for multicast group members

- Failures include link failures, packet loss patterns and congestion patterns and may follow a failure model or distribution
- Events may also include movement patterns of mobile hosts

This information is provided by "config.in" with the use of a number of parameters which are parsed and assigned to variables in the simulation model. The general model consists of a number of models. The communication model, the mobility model, the propagation model, the radio model…etc. These models are clearly shown in Figure 22. We have previously mentioned the specifications of the models of our interest. The following define our choice for the rest of the models:

Mac layer: The IEEE 802.11 distributed coordination function (DCF) MAC is used. It uses the RTS/CTS/DATA/ACK pattern for all unicast packets and simply sends out DATA for all broadcast packets.

Radio Propagation Model: The Radio Propagation Model uses Friss-space attenuation (1/r2) assuming each node is at a line-of-site (LOS) from the node it is transmitting to (i.e., no barriers between them).

Antenna: An omni-directional antenna with unity gain is used by mobile nodes.

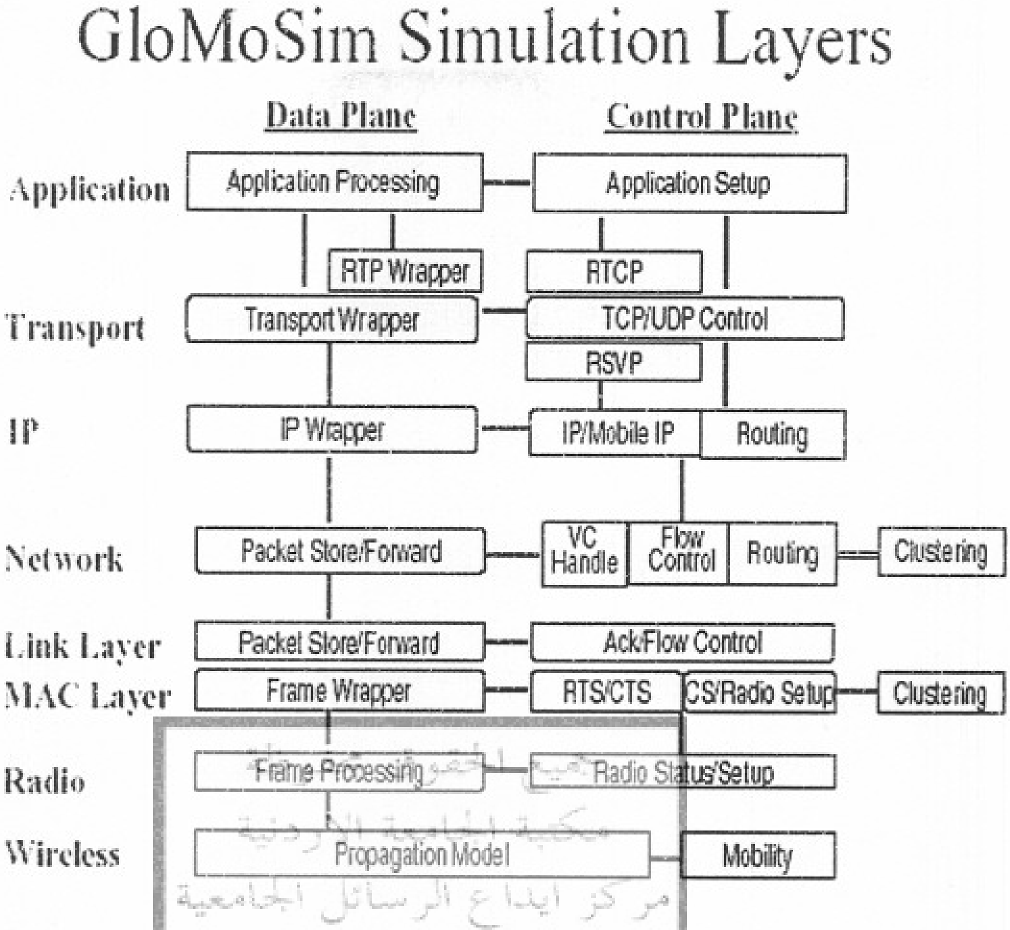


Figure 22: the GloMoSim Simulation Model

3. CBR model generation

In order to generate the CBR communication pattern we created our own generation visual basic program. When creating this program we took into consideration a number of issues:

- Random node number selection for severs and clients; in GloMoSim nodes are given
- A node cannot be the server and the client of the same connection
- For our own interest, the node could be the source node of only one route to a specific destination at a specific time.

The program would produce a ".in" file with the following format:

CBR [src] [dest] [items to send] [item size] [interval] [start time] [end time]

[src] is the client node number.

[dest] is the server node.

[items to send] is how many application layer items to send.

[item size] is size of each application layer item.

[interval] is the interdeparture time between the application layer items.

[start time] is when to start CBR during the simulation.

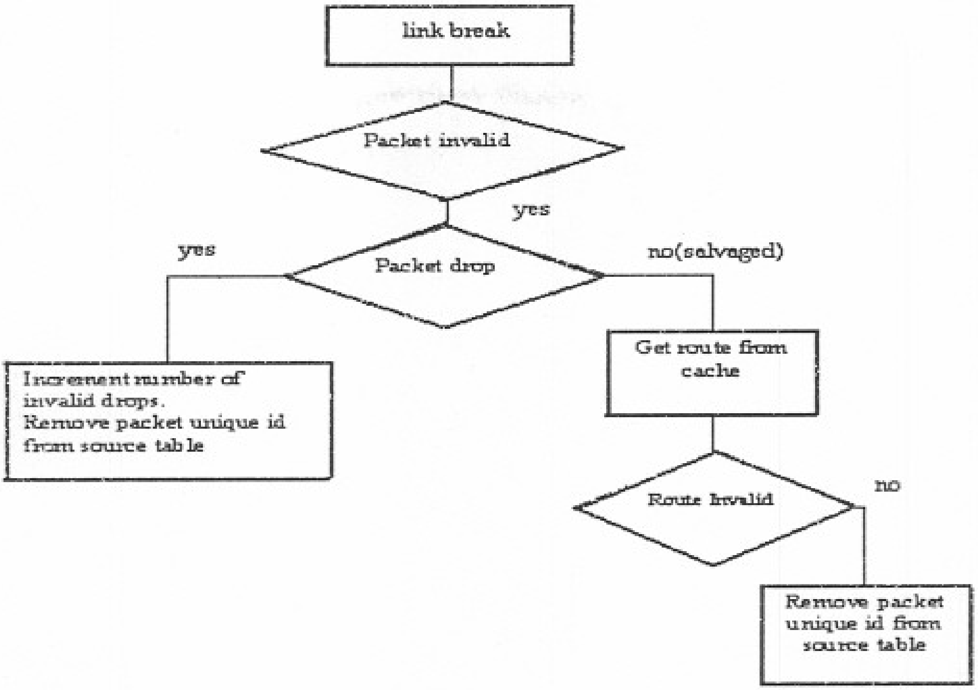
[end time] is when to terminate CBR during the simulation

this formant and other info can be found in the app.conf file which is located in: c:\glomoism \bin\

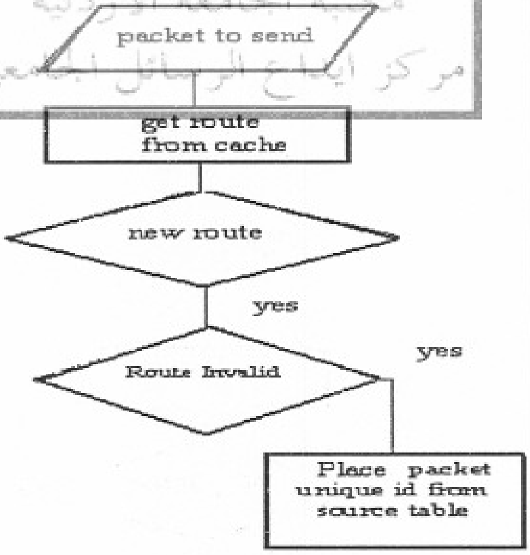
4. Retrieving results in GloMoSim for both AODV and DSR

- In order to retrieve info not already available in GloMoSim new stats need to be added in the protocol header file. The stats are represented by a structure of different variables with different values. The stats of a protocol are part of the routing information that belong to a specific node.
- In the ".pc" file for a specific protocol, there is a function called at the end of the simulation. This function is called the finalize function, this function prints out the state information into the trace file.

At the end of the simulation the trace file "glomo.stat" is produced. This file will contain information about the layers you specify in your configuration file. information about the stats maintained by each layer are presented as specified in the finalize function of each layer. Figure 23 shows the retrieval algorithm for the invalid drop packet ratio. Figure 24 shows retrieval algorithm for invalid delay.



a) Intermediate node handling



b) Source Node handling

Figure23: Invalid Drop Packet Ratio Retrieval algorithm .

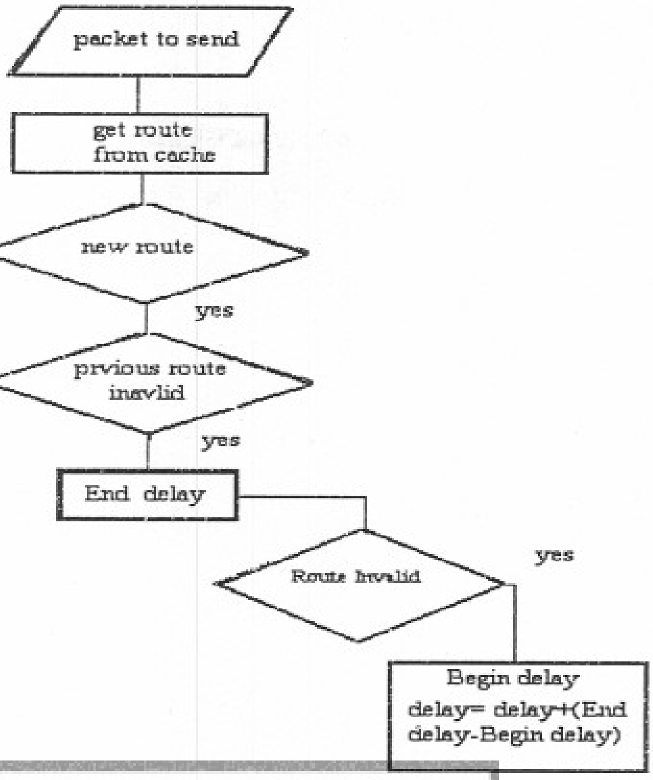


Figure 24: Invalid Delay Retrieval algorithm at Source node.

تحليل صيانة المسلك في بروتوكولات المسلك التفاعلية في شبكة المتنقل العشوائي

اعداد : إناس خالد الطراونة

المشرف: د. وسام المبيضين

المشرف المشارك: د. عماد قدورة

ABSTRACT (In Arabic)

شبكات التنقل العشوائي من أحدث الشبكات اللاسلكية ، تتمتع هذه الشبكات بالقدرة على التشكل المباشر دون دعامة ثابتة، يمكن الاستفادة من هذه الشبكات في المجالات العسكرية و التعليمية و التجارية و عدة مجالات أخرى ، و توفر تقنيات التخبئة مسالك إلى مختلف الغايات في شبكة المتنقل العشوائي في التو و اللحظة. هذا يساعد على إنقاص الزمن اللازم لإتمام عملية النقل و يخفف من الحمل الزائد على الشبكة . قد تصبح المسالك في المخابئ غير صحيحة نتيجة للتنقل المتكرر في شبكات المتنقل العشوائي . في حال استخدام المسالك غير الصحيحة في المخابئ يقوم بروتوكول المسلك التفاعلي على صيانة المسلك. خلال عملية صيانة المسلك يتم فقد بعض رزم المعلومات، مما يقلل من كفاءة البروتوكول في توصيل المعلومات. كما أن الزمن اللازم لصيانة المسلك يؤخر وصول المعلومات إلى الغاية المرجوة .

تعتمد DSRعلى عملية صيانة المسلك لزيادة نسبة صحة المسالك المخبئة ، و تعتمد AODVعلى عملية اكتشاف المسلك لذات السبب لهذا AODV تؤدي وظائفها بشكل أفضل في DSR في حال استخدام النقليات قصيرة الأمد . لتخفيض تأثير عملية صيانة المسلك على أداء DSR ؛ تقدمنا بحلين مقترحين ، الأول يتلخص بزيادة صحة المسالك المتعددة في DSR باستخدام WEN ، و الأخر هو تفضيل إحدى جزئي تقنية التخبئة الخاصة في AODV (والتي تتضمن المسلك الواحد للغاية الواحدة ، تفضيل المسلك الأحدث) على DSR بإزالة المسالك المتعددة للغاية الواحدة . كما و نقوم بدراسة عملية ملائمة عملية صيانة المسلك كأساس التقنية المخبئة WEN .

تشير النتائج الى ان تقنيات التخبئة المغتمدة على عملية صيلنة المسلك ، كالبروتكول DSR، منلسبة للشبكات التي تقوم بعمليات النقل قصيرة الأمد. اما تقنيات التخبئة المغتمدة على عملية اكتشاف المسلك ، كالبروتكول AODV، منلسبة للشبكات التي تقوم بعمليات النقل قصيرة الأمد عند تطبيق احد اجزاء تقنية التخبئة ل AODV علىDSR ،وهي المسلك الواحد للغاية الواحدة،زادت كفاءة التوصيل للبروتوكول DSR وتقلص الزمن المهدر في

عمليات صيانة المسالك غير الصحيحة لعمليات النقل قصيرة الأمد.هذا التحسن لم يكن بصورة كاملة بسبب عدم تطبيق الجزء الثاني لتقنية التخبئة الخاصة بAODV على DSR، وهي تفضيل المسلك الأحدث.