A ROBUST ROUTING PROTOCOL FOR WIRELESS MOBILE AD HOC NETWORKS

WANG QIANG

NATIONAL UNIVERSITY OF SINGAPORE

2002

A ROBUST ROUTING PROTOCOL FOR WIRELESS MOBILE AD HOC NETWORKS

WANG QIANG
(B.Eng (Hons.), NUS)

A THESIS SUBMITTED
FOR THE DEGREE OF MASTER OF ENGINEERING
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE

ABSTRACT

In Mobile Ad Hoc Networks (MANETs), routing protocols are challenged with establishing and maintaining multi-hop routes in the presence of mobility, bandwidth limitation and power constraints. In this thesis, we first study the various routing strategies for MANETs. On-demand and table-driven schemes are analysed and compared with each other. Our study shows that on-demand protocols in general achieve better and more stable performance especially at high mobility due to their efficient utilization of control overhead.

We next introduce a new scheme AODV-RR (Ad Hoc On-demand Distance Vector Protocol with Redundant Routes) with improved robustness by incorporating route redundancy into AODV. The presence of redundant routes allows provision of immediate alternative routes to salvage time-critical traffic flows upon link failures, hence increasing throughput and minimizing delay. Our evaluation proves that route redundancy indeed helps boost overall routing robustness as well as efficiency.

Lastly we experiment further improving performance by making Route Expiry Timeout (RET) adaptive to mobility. Adaptive RET proactively prevents aging of routes especially at higher mobility and hence helps reduce the chances of using potentially stale routes. Our investigation suggests that by selecting optimal RET values depending on mobility, substantial performance gain can be achieved at affordable increase in control overhead.

ACKNOWLEDGEMENT

The project work described in this thesis was carried out between July 2000 and July 2002 as a partial fulfilment of the requirements for the degree of Master of Engineering at the National University of Singapore.

I would like to express my sincere gratitude and appreciation to Professor Lawrence Wong Wai Choong for his valuable advice, guidance and supervision that helped make this project into reality. His many comments and ideas have always been the source of innovation and inspiration along the progress of this project.

I would also like to thank Mr. Raghuraman Ramiah from the Mobile Multimedia Research Laboratory and Mr. Henry Tan from the Communications Laboratory for their helpful facility arrangement and technical assistance during the course the project. My gratitude also goes out to the secretary Ms. Dorothy Goh for her administrative support.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABR	Associativity-Based Routing
AODV	Ad Hoc On-Demand Distance Vector Routing
AODV-BR	AODV with Backup Routing
AODV-LL	AODV with Link Layer detection of link failures
AODV-RR	AODV with Redundant Routes
ARP	Address Resolution Protocol
ATM	Asynchronous Transfer Mode
CBR	Constant Bit Rate
CBRP	Cluster-Based Routing Protocol
CEDAR	Core-Extraction Distributed Ad Hoc Routing
CGSR	Cluster Gateway Switch Routing
CMU	Carnegie Mellon University
DARPA	Defence Advanced Research Projects Agency
DCF	Distributed Coordination Function
DSDV	Destination-Sequenced Distance-Vector Routing
DSDV-SQ	DSDV with Sequence Number for update trigger
DSR	Dynamic Source Routing
DSSS	Direct-Sequence Spread Spectrum
FSR	Fisheye State Routing
IMEP	Internet MANET Encapsulation Protocol
MAC	Medium Access Control
MANET	Mobile Ad Hoc Networks
ns-2	Network Simulator - 2
OLSR	Optimised Link State Routing
PDR	Packet Deliver Ratio
QoS	Quality of Service

RERR	Route ERRor control packet
RET	Route Expiry Timeout
RREP	Route REPly control packet
RREQ	Route REQuest control packet
SSA	Signal Stability based Adaptive Routing
TCP	Transport Control Protocol
TORA	Temporally Ordered Routing Algorithm
TTL	Time To Live
UC	University of California
WRP	Wireless Routing Protocol
ZRP	Zone Routing Protocol

CHAPTER ONE
INTRODUCTION

This introductory chapter first describes the project background highlighting the definition of Mobile Ad hoc Networks (MANETs) and routing in such networks. The project objective is declared next, followed by a brief summary on the project achievements and related works. The organization of the thesis is outlined last.

1.1 Background

Mobile computing has gained tremendous popularity in recent years. Advancement in technology has been able to support computing devices that are much more capable, portable while less expensive. This makes it possible to extend access to information and personal communications beyond the traditional wired domain. While constrained by dynamic topology and unpredictable links, mobile networks clearly offer much greater flexibility in user access and network configuration as compared to fixed wired networks. This project concerns a relatively new type of mobile networks, the Mobile Ad Hoc Networks (MANETs).

1.1.1 Mobile Ad Hoc Networks (MANETs)

A Mobile Ad hoc Network (MANET) is a collection of wireless mobile nodes dynamically forming a temporary network without the use of any existing network infrastructure or centralized administration [1]. In contrast to conventional cellular or wireless ATM networks, mobile nodes in MANET communicate peer-to-peer directly through the dynamic network without relying on any wired base-stations. Each MANET mobile node operates not only as a host, but also as a router [1] that dynamically discovers and maintains routes to other nodes in the network whenever necessary. The idea of ad hoc networking is sometimes called infrastructureless networking [2].

The major advantage of MANETs is their quick deployment. Requiring no fixed infrastructure, MANETs are considerably easy and inexpensive to setup and configure. This makes MANETs particularly suitable for applications in temporary, private or emergency situations. Examples of such applications include home network, indoor e-conferencing, automotive interaction, personnel coordination during natural disasters recovery, military maneuvers in enemy territory, and so on.

The main concern about MANETs remains on their reliability and robustness in performance. When mobile nodes move arbitrarily, the network topology changes frequently and unpredictably [3]. Moreover there are constraints on the link bandwidth and battery power. All these factors bring challenges to the design of MANET operating protocols. Lower levels in the protocol stack such as the physical layer, multiple access control (MAC) and routing layers deserve exclusive attentions to cater for the unique characteristics of MANETs.

1.1.2 Routing in Mobile Ad Hoc Networks

The unique characteristics of MANETs dictate that their routing schemes vastly differ from existing shortest-path algorithms designed for static networks. Firstly, given the higher rate of topology change in MANETs, the routing scheme must react and converge fast enough to keep routes up-to-date. On the other hand, the routing scheme must not be over-sensitive to topology changes to avoid causing oscillating update events hence degraded stability. In addition, limited battery power supplies in mobile nodes require that less computation and routing overhead is desired. All these combined call for a robust, stable and efficient routing scheme suitable for the MANET environment.

Since the advent of Defense Advanced Research Projects Agency (DARPA) packet radio networks in the early 1970s [4], numerous MANET routing protocols have been developed and proposed. To name a few, these include Destination-Sequenced Distance-Vector Routing (DSDV) [5], Cluster Gateway Switch Routing (CGSR) [6], Wireless Routing Protocol (WRP) [7], Ad Hoc On-Demand Distance Vector Routing (AODV) [8], Dynamic Source Routing (DSR) [9], Temporally Ordered Routing Algorithm (TORA) [10], Associativity-Based Routing (ABR) [11], Signal Stability based Adaptive Routing (SSA) [12], Fisheye State Routing (FSR) [13], Core-Extraction Distributed Ad Hoc Routing (CEDAR) [14], Zone Routing protocol (ZRP) [15], and so on. These protocols adopted different approaches to deal with the typical limitations of MANETs and exhibit quite distinct behaviors.

In general, these routing protocols can be classified into two categories: Table-driven (Proactive) or Source-initiated (Reactive, Demand-driven) [16]. Table-driven routing protocols attempt to maintain consistent, up-to-date routing information from each node to every other node in the network. Each node is required to maintain routing information in local tables and respond to topology changes by propagating updates throughout the network in order to keep its routing information up-to-date. Table-driven schemes enable routes to be immediately ready or quickly established when needed. However, such proactive schemes in general produce excessive network overhead through regular update broadcasts and sustain potentially stale routes when topology changes are frequent. Examples of table-driven schemes include DSDV, CGSR, and WRP. Unlike table-driven schemes, source-initiated on-demand routing protocols do not consistently maintain routing information, but create routes only when desired by the source node. This approach significantly reduces the network overhead resulted from frequent update broadcasts, but introduces additional latency during route discovery and maintenance procedures. Examples of source-initiated schemes include AODV, DSR, TORA, ABR and so on. In summary, the comparison between table-driven protocols and source-initiated protocols exhibits a trade-off between robustness and efficiency.

1.1.3 Routing Metrics and Multimedia Traffic Requirement

Performance metrics that are most commonly used in evaluating MANET routing protocols are the Packet Delivery Ratio (PDR) and Routing Overhead. Packet Delivery Ratio is the fraction of data packets that are successfully delivered from source node to destination node over the total number of data packets generated by the source node. The higher the ratio, the more robust is the protocol. Routing Overhead is the amount of any overhead packets transmitted other than the data packets. The less amount of overhead packet spent per data packet, the more efficient is the protocol. These two parameters have been taken predominantly as means to indicate the capability of MANET routing protocols so far.

As many mobile applications involve human-to-human communications, there is strong need to support multimedia traffic such as voice and images over the wireless network. In such scenarios, real-time support with quality of service (QoS) constraints is critical. Specifically, low latency is in favor of multimedia traffic. Hence when assessing the real-time capability of MANET routing protocols, latency behavior such as the average end-to-end delay becomes an important metric to consider.

1.2 Objectives

This project aims to explore possible approaches to improve the robustness of exiting MANET routing protocols. A primary study is first conducted on early MANET routing protocols to evaluate their performance and review their relative strengths and weaknesses. Mechanisms to improve the robustness of such protocols are investigated next. Specifically, ideas of route redundancy and adaptive route expiry timeout are experimented and their performance evaluated. While these enhancements can apply to other protocols, in general, we select AODV as the benchmark.

1.3 Achievements

The following briefly summarizes the project achievements.

- Review and comparison of some early MANET routing protocols;
- Exploiting route redundancy to improve routing robustness; and
- Exploiting adaptive route expiry timeout to mobility to improve robustness.

A review study is first conducted on some early MANET routing protocols including DSDV, DSR, AODV and TORA. Their performance are evaluated and compared through simulations using Network Simulator-2 (*ns*-2) [17] following similar environment set-up as in [1]. The results obtained are consistent with those given in [1], which provides insight into how these protocols behave differently. AODV is selected as the basis for later enhancement.

The idea of route redundancy is then explored. A complete and systematic algorithm is designed to establish and maintain a mesh of redundant routes dynamically. With the above enhancement, a new scheme AODV with Redundant Routes (AODV-RR) is developed and its performance evaluated. As is evident from the results that AODV-RR achieves higher PDR, smaller end-to-end delay while using less routing overhead as compared to the original AODV, it is proven that the redundant route mechanism is capable of improving overall robustness of MANET routing protocols.

The project also studies the impact of making Route Expiry Timeout (RET) adaptive to mobility on routing performance. Instead of a fixed value of RET for all mobility cases, we experiment with smaller values of RET for higher mobility so as to improve the freshness of route information kept in the cache. Through simulations using *ns*-2, it is observed that adaptive RET mechanism further improves performance in PDR at the expense of larger routing overhead.

1.4 Related Works

Since the advent of DARPA packet radio networks in the early 1970s [4], numerous MANET routing protocols have been developed and proposed by ongoing research activities around the world. Some of these works are presented in [5-15]. Specifically, table-driven schemes [5-7] maintain constant network view by proactive and regular exchange of routing tables among all nodes. Demand-driven schemes [8-12] initiate route discoveries only when required by traffic demand at source nodes. There are also hybrid schemes such as ZRP [15] that applies proactive and reactive approaches to nodes in different areas (within and outside the zone).

Classic reviews on early MANET routing protocols include [1, 16, 3]. Royer [16] gave comprehensive introductions on the theoretical background and working mechanism behind a few MANET protocols, classifying them into either table-driven or source-initiated. Broch, et al [1] evaluated the performance of four early routing protocols through simulations and laid out the results in details for comparison. Lee, et al [3] focused on the behaviours of multicast protocols.

The idea of redundant routes aims to improve robustness by providing backup routes when the primary route fails. The work by Lee and Gerla [18] was one of the first to explore this idea. It proposed to rely on last-minute effort to broadcast data packets to neighbours when encountering a link failure. In contrast, our design uses a systematic algorithm to establish a mesh of redundant routes and support switching of data flows at the times of link failures.

1.5 Organization of Thesis

The rest of the thesis is organized as follows. **Chapter 2** reviews four early MANET routing protocols: DSDV, DSR, AODV and TORA closely comparing their behaviours and performance through simulations using *ns*-2. This initial part of the project aims to provide insight into the distinct characteristics of the few protocols as well as to ensure proper software settings to prepare for simulations later.

Chapter 3 presents a systematic design that exploits redundant routes mechanism to improve the overall robustness of MANET routing protocols. The chapter explains in details the AODV-RR implementation based on AODV and compares their performance.

Chapter 4 deals with an investigation into the impact of adaptive route expiry timeout (RET) with respect to mobility. We experiment with different values of RET for each mobility case until the optimal is reached, based on which a hypothetic formula relating RET to mobility is then proposed. The performance of a system with such adaptive RET formula is evaluated.

Lastly as a concluding note, **Chapter 5** summarizes the project's achievements and findings. The chapter ends with some suggestions for future work.

CHAPTER TWO

REVIEW OF EARLY ROUTING PROTOCOLS

This chapter reviews four early MANET routing protocols: DSDV, DSR, AODV and TORA. The chapter first briefly discusses different classifications of MANET routing protocols. As examples, the methodologies behind each of the above four protocols are then described. With a bit of introduction on the software simulator used for this project, the chapter proceeds to present the performance results of the four protocols for comparison.

2.1 Introduction

MANET routing protocols can be classified according to many different criteria. Firstly, depending on when route-discoveries are initiated, MANET protocols can be either Proactive (Table-driven) or Reactive (On-demand). Next, depending on where routing information is maintained, there are protocols with either Source Routing or Distributed Routing strategies. In addition, where the address space is concerned, MANET protocols can have either a Flat structure or a Hierarchical structure. Whereas when route selection criteria are considered, MANET protocols can be based on either Link State or Distance Vector. Each of these strategies has its relative advantage and disadvantages. **Table 2.1** gives a summary on these classifications with some protocols named as examples.

Table 2.1 Classifications of MANET Routing Protocols

Criteria	Classification	Characteristics	Advantage/Disadvantage	Examples
When to initiate route discovery	Proactive (Table-driven)	Regular exchange of routing information	Routes ready for use immediately / Large routing overhead	DSDV[5] CGSR[6] WRP[7]
	Reactive (On-Demand)	Initiate route request only when needed by traffic	Smaller routing overhead / Delay in route-discovery	AODV[8] DSR[9]
Where to maintain routing information	Source Routing	Routes maintained by source node	Complete source control / Large storage requirement & poor scalability	DSR[9]
	Distributed Routing	Routes maintained by source and intermediate nodes	Less burden on source & better scalability / More complex route computation	AODV[8] TORA[10]
Structure of address space	Flat structure	All nodes function as routers for one another	Easy route computation & maintenance / Poor scalability	DSDV[5] AODV[8]
	Hierarchical structure	Nodes grouped in clusters and routing handled by cluster heads	Good scalability / Complex route computation and maintenance	CGSR[6] CBRP[19]
Route selection criteria	Distance Vector	Select routes with shortest path	Easy route computation / Slow convergence or sub-optimal	DSDV[5] AODV[8]
	Link State	Select routes with best link quality	Optimal routes / Complex route computation	FSR[13] OLSR[20]

One of the classic reviews on early MANET routing protocols is given in [1]. It was the first to provide a realistic and quantitative analysis comparing the performance of a variety of MANET routing protocols [1]. With extensive use of the Network Simulator *ns*-2 [17], the paper presented detailed simulation results showing the relative performance of four proposed protocols: DSDV, DSR, AODV and TORA.

Due to the contributions from continuous research efforts, both the Network Simulator *ns*-2 and the individual protocol implementation codes have evolved substantially since the paper was published. This part of the project aims to re-evaluate the performance of the above four early MANET routing protocols through simulations using *ns*-2 with similar environment settings. The performance results provide much insight into the unique protocol behaviours.

2.2 Overview of Early Protocols

As representatives of different classes (see **Table 2.1**), the four early routing protocols of interest: DSDV, DSR, AODV, TORA were built on different mechanisms and exhibit unique characteristics. Despite the many differences, all protocols dealt with two issues: the initial route discovery and subsequent route maintenance.

2.2.1 Destination-Sequenced Distance Vector Routing (DSDV)

DSDV [5] is a distributed (hop-by-hop) table-driven distance-vector routing protocol requiring each node to periodically broadcast routing updates. The key advantage of DSDV over traditional distance vector protocols is that it guarantees loop-freedom [1].

Each DSDV node maintains a routing table listing all reachable destinations with the “next hop” address as well as the number of hops to each destination. Each routing table entry is tagged with a sequence number originated and monotonically increased by the destination node. The sequence numbers enable the mobile nodes to distinguish stale routes from new ones, thereby avoiding the formation of routing loops [16]. To maintain the consistency of route tables in a dynamically varying topology, routing table updates are transmitted throughout the network periodically, and immediately when significant new information is available. When comparing two routes to the same destination, the route with a higher sequence number is always more favourable; or in the event that two routes have the same sequence number, the route with shorter metric is considered more favourable. The above propagation of route updates is triggered periodically and whenever new network conditions are detected, regardless of network traffic demand. When such routing updates converge in time, all nodes should keep a consistent view of the whole network and maintain a shortest route to each reachable destination in its routing table. Hence routes are immediately ready for use when demanded by any network traffic. Such an approach of proactive exchange and computation of routing information is known as proactive routing.

When a node decides that its route to a destination is broken, it updates the metric to that destination to infinity, increments the sequence number for that route by one and advertises this new route entry to the rest of the network through a broadcast. This will cause all upstream nodes of the broken route to incorporate the infinity-metric into their own routing tables until they hear newer route updates with a higher sequence number from that destination later.

The major advantage of DSDV as a proactive algorithm is that routes are available immediately whenever needed. In addition, the exchange of routing updates is triggered proactively regardless of traffic demand; hence the amount of routing overhead incurred is theoretically constant for all mobility and traffic load cases.

However, the periodic network-wide broadcasts of routing table updates usually cost excessive routing overhead, which are sometimes unnecessary especially when mobility or traffic load is low. Such massive routing overhead not only consumes precious wireless bandwidth and mobile node battery power, but also prevents the distributed network view from converging quickly when mobility is high and change of network topology is frequent.

2.2.2 Dynamic Source Routing (DSR)

DSR [9] is an on-demand routing protocol based on the concept of source routing. Source nodes maintain the complete route information, and route each packet by inserting in its header the complete ordered list of nodes through which the packet must pass. The key advantage of source routing is that intermediate nodes do not need to maintain routing information for the packets they forward.

When a mobile source node has a packet to send to some destination, it first consults its route cache to determine whether an unexpired route to that destination is available. If yes, it appends the complete list of nodes in that route to the packet header and forwards the packet to the next hop. On the other hand if no such route is available in the cache, the source node initiates a route discovery process by broadcasting a Route Request (RREQ) packet containing both the destination address and the source address. As the RREQ packet propagates around in the network, all intermediate nodes which do not have a valid route to the requested destination in their caches will add their own address to the route record of the RREQ packet and forward the RREQ packet to all outgoing links. When the RREQ packet reaches either the destination, or an intermediate node that contains an unexpired route to the requested destination, a Route Reply (RREP) packet is generated and sent back to the source node. The RREP packet contains the complete list of nodes composing the discovered route, constructed from the route record in the RREQ packet and appended with the node list from the unexpired route to the requested destination in the cache of the intermediate node, if any. The RREP is forwarded back to the source by reversing the discovered route. Upon receiving the RREP, the source node updates its route cache, inserts the node list of the newly discovered route to the header of the waiting data packet for the requested destination, and forwards the packet to the next hop along the newly discovered route.

Route maintenance involves the use of Route Error (RERR) packets that are generated at a node when the data link layer encounters a fatal transmission problem. When notified by the RRER packet, the source node removes the hop in error from its route cache and truncates all routes containing the hop at that point. When requested by subsequent data packets, localized route discoveries are initiated to re-construct the complete route to that destination after the topology change.

As a typical on-demand source routing protocol, DSR eliminates the need for periodic routing table updates, and hence, has the advantage of significantly reduced routing overhead that is traffic and mobility oriented. Since the entire route is maintained in the source node and kept in the data packet header, less storage space and routing delay is required at intermediate nodes.

The main disadvantage of DSR being demand-driven in nature is that routes take time to be discovered while emerging data packets are delayed waiting in the source buffer. Moreover, the finite packet header size limits the maximum length of routes to carry and hence the diameter of the network. As a result, DSR suffers from poor scalability.

2.2.3 Ad Hoc On-Demand Distance Vector Routing (AODV)

AODV [8] is essentially a combination of DSR and DSDV. It borrows the basic on-demand mechanism of Route Discovery and Route Maintenance from DSR, plus the use of hop-by-hop routing, sequence numbers and optional periodic beacons from DSDV [1]. AODV improves on top of DSDV by creating routes on demand, hence minimising the number of required broadcasts. On the other hand, it improves over DSR by distributing route management among different nodes in a hop-by-hop manner. AODV is able to support both unicast and multicast communications. This project limits its scope to the unicast application of AODV.

Similar to DSDV, each AODV node uses a routing table to store reachable destinations with the next-hop address, number of hops to that destination as well as the destination sequence number for each. In addition, for each destination the node maintains a precursor list, which contains previous-hop addresses that have used this route recently, for route maintenance purpose.

When a source node needs a route to a destination node, it initiates the route discovery process by broadcasting a Route Request (RREQ) message to its neighbours. The RREQ message is flooded through the network in a controlled manner by use of a Broadcast ID mechanism. Each RREQ message originated by the source node and all of its subsequent copies made by forwarding nodes carry a same unique Broadcast ID. The Broadcast ID will ensure only the first copy of RREQ received is accepted at each node while all subsequent arrivals of RREQ copies are discarded. Each node forwarding the RREQ message creates a reverse route in its routing table from itself back to the source node. When the RREQ reaches either the destination node or an intermediate node with an unexpired route to the requested destination, a Route Reply (RREP) message is generated containing the number of hops and last-known sequence number for the requested destination. The RREP message is then unicast back to the source node along the reverse route. Each node that participates in forwarding this RREP message back to the source creates a forward route in its routing table from itself to the requested destination.

In order to maintain routes, AODV detects link failures through either periodic exchange of HELLO message among neighbours, or underlying physical layer or link layer methods. When a link goes down, the node brings down the route in its routing table by incorporating an infinity metric for that route, and informs all upstream nodes that have recently forwarded on this broken route via an Unsolicited Route Reply (UREP) message. The UREP message contains an infinity metric for that unreachable destination and is locally broadcast among neighbours. Upon receiving the UREP message, a node brings down all routes to that unreachable destination in its routing table, and informs all upstream nodes in the precursor lists of these broken routes by forwarding the UREP message. On demand of subsequent arrival of data packets, route discoveries are initiated locally to repair the broken route.

AODV is an improvement over DSDV because it no longer requires periodic network-wide broadcasts for route updates and therefore the route overhead is drastically reduced. By participating in the distributed hop-by-hop routing, the intermediate nodes can react faster to topology changes, hence potentially making the established routes more adaptive and optimal as compared to source routing. In addition, it enjoys better scalability than DSR.

The demand-driven nature of AODV implies that delays in route discoveries and repairs are inevitable. There always exists a tradeoff in favour of reduced delay at the possible expense of increased number of route finding broadcasts.

2.2.4 Temporally-Ordered Routing Algorithm (TORA)

TORA [10] is a highly adaptive distributed routing algorithm based on the concept of link reversal. It is designed to discover routes on demand and provide multiple routes for a source/destination pair. The key design concept of TORA is the localization of control messages to a very small set of nodes near the occurrence of a topology change [16]. Route optimality (shortest-path routing) is considered of secondary importance, and longer routes are often used to avoid the overhead of discovering newer routes [1].

In TORA, each node has a "height" metric with respect to the destination that is computed by the routing protocol. When a source node needs a route to a particular destination, it broadcasts a QUERY packet containing the requested destination address through the network. The final recipient of the QUERY packet (either the destination or an intermediate node with a route to the destination) will broadcast an UPDATE packet listing its height relative to the destination. As this UPDATE packet propagates through the network, all nodes receiving this packet sets its height to a value greater than the height of the neighbour from which the UPDATE is received. This has the effect of creating a series of directed links rooted at the destination that grows up to the source node. Data packets are then forwarded down the directed links like water flows through pipes from upstream to downstream.

When a node discovers that a downstream route to a destination is no longer available, it adjusts its height to a local maximum relative to all its neighbours and transmits an UPDATE packet, which has the effect of reversing one or more directed upstream links to reflect the change of topology.

TORA is layered on top of the Internet MANET Encapsulation Protocol (IMEP) [21], which is required to provide reliable, in-order delivery of all routing control messages, aggregation and encapsulation of control messages to save overhead, as well as link state sensing.

TORA's innovative link reversal concept enables multiple routes for a source/destination pair to be created efficiently. The localized algorithmic reaction to topology changes plus demand-driven routing finding broadcasts significantly reduces the routing overhead.

However, the proper functioning of TORA requires highly reliable and in-order delivery of control messages, which is difficult to achieve in scenarios where there is traffic congestion in the network. In addition, TORA has the potential for oscillations to occur [16] though short-lived. These degrade TORA in its robustness.

2.2.5 Summary of Protocols

Key characteristics and properties of the above four protocols are briefly summarized in **Table 2.2**.

Table 2.2 Summary of DSDV, DSR, AODV and TORA

Protocols	**DSDV**	**DSR**	**AODV**	**TORA**
Route Finding	Proactive	On-Demand	On-Demand	On-Demand
Route Selection	Shortest Path	Shortest Path	Freshest and Shortest Path	Shortest Path
Distributed	Yes	No	Yes	Yes
Loop Free	Yes	Yes	Yes	Yes
Periodic Messages	Routing Tables	None	(Optional) HELLO Messages	IMEP messages
Key Concepts	Bellmen-Ford algorithm; Periodic route updates	On-demand; Source routing; Entire route	On-demand; Hop-by-hop; Sequence numbers	Link reversal; Directed links; Localization
Strengths	Routes ready immediately; Optimal routes	Small routing overhead	Small routing overhead; Versatile	Small routing overhead
Weakness	Large routing overhead	Delay in route establishment; Poor scalability	Delay in route establishment	Delay in route finding; Less robustness

2.3 Simulation Setup

The Network Simulator (*ns*-2) [17] is a discrete event simulator developed as part of the VINT project [22] jointly by UC Berkeley, LBL, USC/ISI and Xerox PARC. Originally *ns*-2 was built to support simulations of TCP and other protocols in conventional networks, but it lacked support for physical aspects or MAC protocols in multi-hop wireless network environments. The Monarch Project (Originally at CMU and recently moved to Rice University) [23] made extensive modifications to *ns*-2 to provide new elements at the physical, data link, MAC, and routing layers of the simulation environment. Using these elements, it is possible to construct detailed and accurate simulations of wireless subnets, LANs, or multi-hop ad hoc networks [22]. Since then, *ns*-2 has become a popular software platform used by many researchers for their MANET simulations. *ns*-2 has always included substantial contributions from other researchers, including wireless code from the UCB Daedelus, CMU Monarch projects and Sun Microsystems [17]. These enhancements in turn greatly assist and push forward on-going effort of research and development in the MANET area. This section describes the *ns*-2 elements for wireless mobile environments together with our project methodologies.

2.3.1 Environment

The wireless physical model of *ns*-2 combines both a free-space propagation model and a 2-ray ground reflection model. When a transmitter is within the reference distance of the receiver, Friss free-space propagation model is used where the signal attenuates as $1/r^2$. Outside the reference distance, an approximation to 2-ray ground reflection model is used where signal attenuates as $1/r^4$. An omni-directional antenna with unity gain positioned 1.5m above ground is used in each mobile node. The wireless shared media interface for each mobile node is implemented to approximate the Lucent WaveLan direct-sequence spread spectrum (DSSS) radio interface [24] operating at 914MHz frequency with 2Mbps in bandwidth.

The link layer of *ns*-2 for wireless environments implements the complete IEEE802.11 standard [25] Medium Access Control (MAC) protocol Distributed Coordination Function (DCF) to accurately model the contention for wireless medium. In addition, an implementation of Address Resolution Protocol (ARP) similar to the BSD Unix implementation [26] is included to resolve IP addresses used by the routing protocol to underlying link layer addresses.

The routing protocol simulated in this project is one of the four early MANET routing protocols: DSDV, DSR, AODV and TORA (see **Chapter 2**); as well as the new protocol proposed by this project: AODV with Route Redundancy (AODV-RR) (see **Chapter 3**) with its further enhancement (see **Chapter 4**).

2.3.2 Methodology

Consistent with settings in [1], all our protocol evaluations in this project are based on simulations of a MANET of 50 wireless mobile nodes moving about and communicating with each other in a 1500m × 300m rectangular flat area for 900 seconds of simulated time.

Mobile nodes in the simulations move according to the "random waypoint" model [1]. In this model, each node begins the simulation by remaining stationary for pause time seconds. The node then selected a random destination in the 1500m × 300m space and moves to that destination at a speed distributed uniformly between 0 and some maximum speed. Upon reaching the destination, the node pauses again for pause time seconds, selects another random destination, proceeds there as previously described, repeating this behaviour for the duration of the simulation run. Hence, two key parameters in this model characterizing the movement scenarios are the pause time and maximum speed. We run our simulations with movement patterns generated for 7 different pause times: 0, 30, 60, 120, 300, 600 and 900 seconds. A pause time of 0 seconds corresponds to continuous motion, and pause time of 900 seconds corresponds to no motion. Because the performance of protocols is very sensitive to movement pattern, we randomly generate 10 scenario files for each of the 7 pause times. All four protocols in this part of the project are run on the same 70 movement pattern files. With varying pause time as means to control average mobility, we base our simulations primarily on a fixed maximum speed of 20 meters per second (average speed of 10 meters per second).

The communication model in our simulations uses constant bit rate (CBR) traffic sources. We experiment with different traffic load with 10, 20 and 30 CBR connections randomly started in between 0 ~ 180 second, all sending 64-byte packets at a rate of 4 packets per seconds for the remainder of the simulation run.

2.3.3 Routing Protocol Decisions

To create consistency with [1], we make similar implementation decisions for the individual routing protocols. For DSDV, we simulate the DSDV-SQ (sequence number) [1] version whereby the receipt of a new sequence number for a destination causes a triggered update. On the other hand for AODV, we use the AODV-LL (link layer detection) [1] version whereby link failures are detected by solely relying on link layer methods rather than the periodic HELLO message mechanism.

2.3.4 Summary

Table 2.3 summaries the simulation settings for this part of the project.

Table 2.3 Summary of Simulation Settings (Part I)

Parameter	**Setting**
Number of Mobile Nodes	50
Movement Area	1500 m × 300 m Flat Rectangular
Total Simulated Time	900 s
Movement Model	Random Way-point
Mobile Node Pause Time	0, 30, 60, 120, 300, 600, 900 s
Maximum Mobile Speed	20 m/s
Traffic Source	Constant Bit Rate (CBR)
Number of Connections	10, 20, 30
Packet Size	64 bytes
Packet Sending Rate	4 packets/s
Connection Start Time	Uniformly Random between 0 ~ 180 s
Medium Access Control	IEEE 802.11 Standard MAC with DCF
Address Resolution Protocol	Approximation of BSD Unix Implementation
Shared Media Interface	Approximation of Lucent WaveLan DSSS Radio
Mobile Node Antenna	Unity-gain, Omni-directional, 1.5m above Ground
Propagation Model	Free-space (near) + Two-ray Ground Reflection (far)
Routing Protocol	DSDV-SQ, DSR, AODV-LL, TORA
Simulator Tool	Network Simulator – 2 (*ns*-2) version allinone-2.1b7
Platform	Redhat Linux 7.0 [27]

2.4 Simulation Results

We test and cross-compare the performance of the above four early MANET routing protocols at different mobility and traffic load levels, in terms of the Packet Delivery Ratio (PDR), Routing Overhead and Average End-to-end Delay.

2.4.1 Packet Delivery Ratio

Figures 2.1 ~ 2.4 give the PDR performance of DSDV-SQ, DSR, AODV-LL and TORA at different traffic load levels with respect to pause time. The smaller the pause time, the higher the average overall mobility.

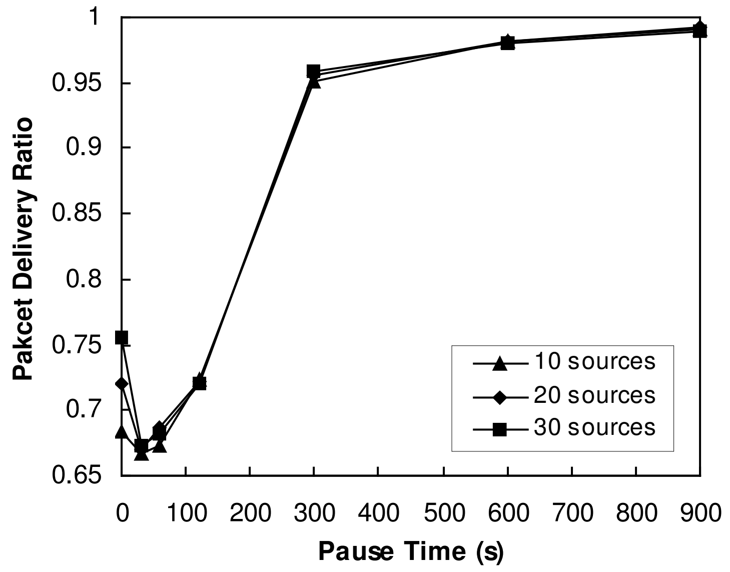


Figure 2.1 DSDV-SQ Packet Delivery Ratio Performance

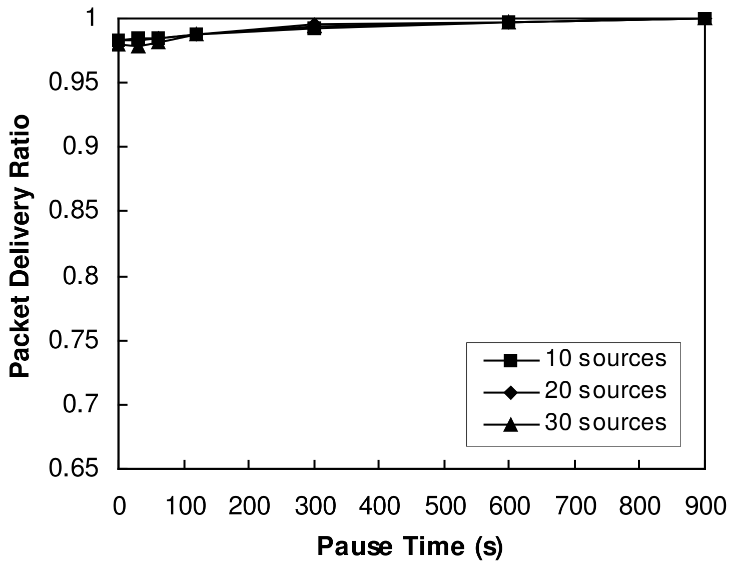


Figure 2.2 DSR Packet Delivery Ratio Performance

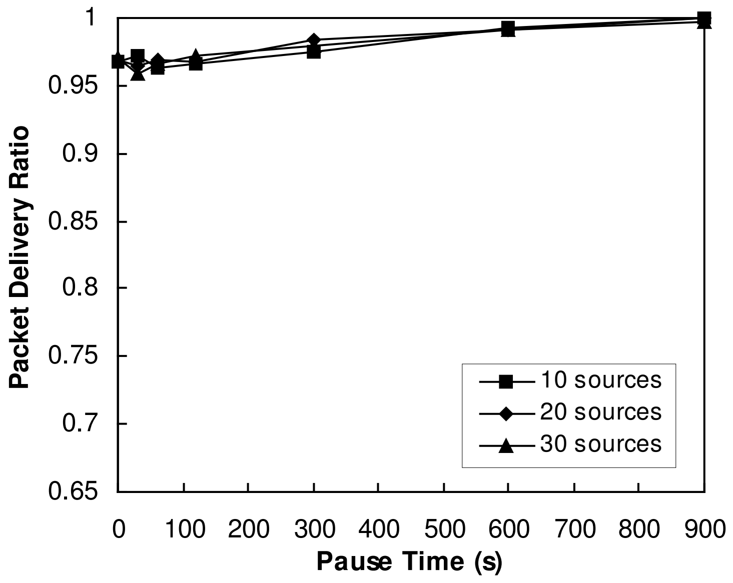


Figure 2.3 AODV-LL Packet Delivery Ratio Performance

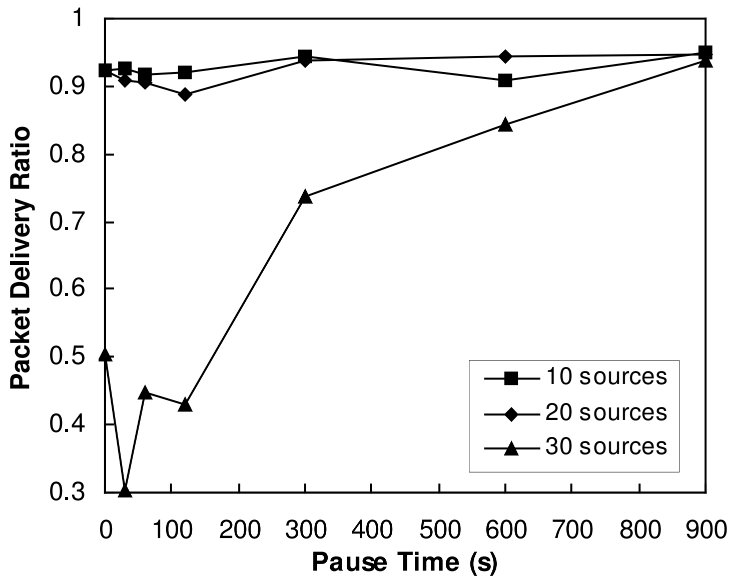


Figure 2.4 TORA Packet Delivery Ratio Performance

In general, all protocols perform better at lower mobility (larger pause times) and their performance degrades as mobility increases (smaller pause times). Moreover, DSDV-SQ, DSR and AODV-LL exhibit very small differences among their PDR curves with different number of sources. This means, the PDR performance of these routing protocols are rather insensitive to the variation of the number of traffic sources within the given range. When total network traffic is light, spatial diversity of the mobile nodes could largely diminish the impact of increase in data packets volume due to more traffic sources. In DSR and AODV that maintains PDR over 95% in all conditions, there is little room left for major further improvement. Essentially, the stability of DSDV-SQ, DSR and AODV-LL ensures that increasing the traffic sources to 30 does not cause any serious degradation in their PDR performance.

Specifically, we observe that DSDV-SQ fails to stabilize at pause times below 300s, dropping to a 70% PDR (see **Figure 2.1**). At high mobility, the periodic network-wide broadcasts of route updates are no longer able to keep up with the fast changing topology or maintain a consistent network view in all mobile nodes. In addition, the inefficient event-triggered updates cause excessive channel usage. As a result, more routing information becomes out-of-date, which causes more failures in packet deliveries hence more packets are dropped. On the other hand, both AODV-LL and DSR perform particularly well with both protocols delivering between 95% and 100% of packets in all cases (see **Figures 2.2 and 2.3**). Their efficient on-demand approach to discover and maintain routes does not exhaust system resources in the scope of the whole network when topology changes are frequent, making the two protocols well adapted to higher mobility. TORA does well with 10 or 20 traffic connections, delivering above 88% of packets even at high mobility (see **Figure 2.4**). The majority of packet drops are due to the creation of short-lived routing loops that are a nature part of its link-reversal process [1]. With 30 connections, TORA's average PDR drops significantly to about 45% at high mobility. In most of these scenarios TORA essentially undergoes congestive collapse, which causes TORA to erroneously perceive link breakages to neighbouring nodes and hence drops most of the data packets.

A comparison of the PDR performance of the four protocols in the case of 20 traffic connections is given in **Figure 2.5**. As is evident from the graphs, AODV-LL and DSR perform particularly well delivering over 95% of data packets in all cases, whereas DSDV-SQ and TORA exhibit some instability in their PDR performance at higher mobility.

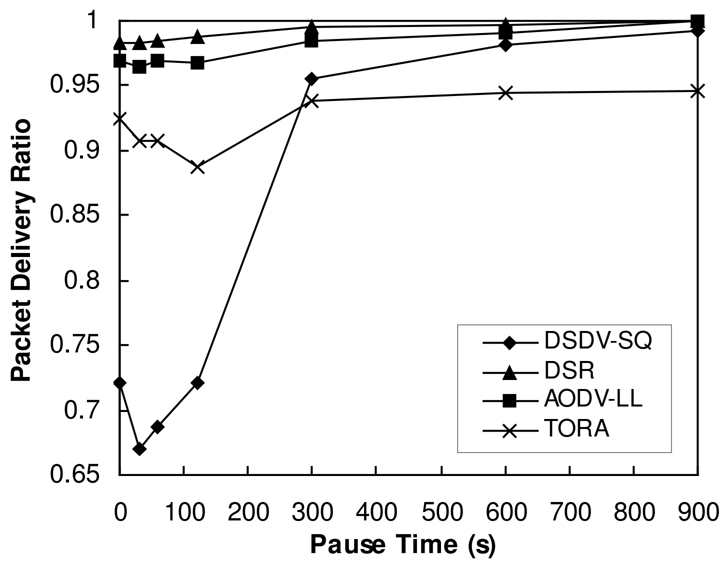


Figure 2.5 Comparison of PDR Performance (20 connections)

2.4.2 Control Overhead

Figures 2.6 ~ 2.9 give the Control Overhead performance of DSDV-SQ, DSR, AODV-LL and TORA at different traffic load levels with respect to pause time.

As one of the most important performance metrics for MANETs, the Control Overhead is defined as the total number of control packets transmitted within the network in order to support certain amount of data traffic. This includes the control packets transmitted during both unicasts and broadcasts as per required by the routing protocol. On the other hand, receptions of control packets are not counted, because one packet transmitted typically results in multiple receptions depending on the population of mobile nodes nearby. During the "Expanding Ring search" route request broadcast in AODV for example, each control packet transmitted is counted into the control overhead, regardless of how many nodes receives this packet. The multiple receiving nodes may or not may not forward this packet to their neighbours, and only when they do so, the packets they transmit on are counted into the total control overhead. Therefore, broadcast has a multiplicative effect on the total control overhead. The larger is the span of such broadcasts, the more mobile nodes will participate in forwarding the control packets to neighbours, and the more control overhead will be incurred.

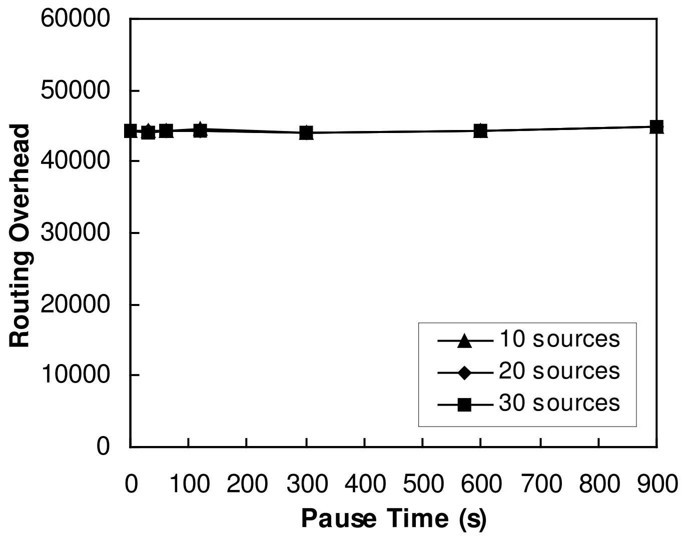


Figure 2.6 DSDV-SQ Routing Overhead Performance

DSDV-SQ has approximately constant routing overhead, regardless of movement speed or offered traffic load (see **Figure 2.6**). In DSDV, each destination node broadcasts a periodic update with a new sequence number every 15 seconds [1]. With 50 unsynchronised mobile nodes in the simulation, at least one node broadcasts a periodic update during each second. Considering the fact that the receipt of a new sequence number for a node is important enough to trigger an immediate advertisement through network-wide flooding, it is reasonable to see that each node effectively transmits triggered updates at the maximum permitted rate of one per second, although the base periodic rate is once every 15 seconds. Therefore, for a 50-node network in 900-second simulations, a constant amount of overhead of about 45,000 packets is obtained for DSDV-SQ, regardless of mobility and traffic load.

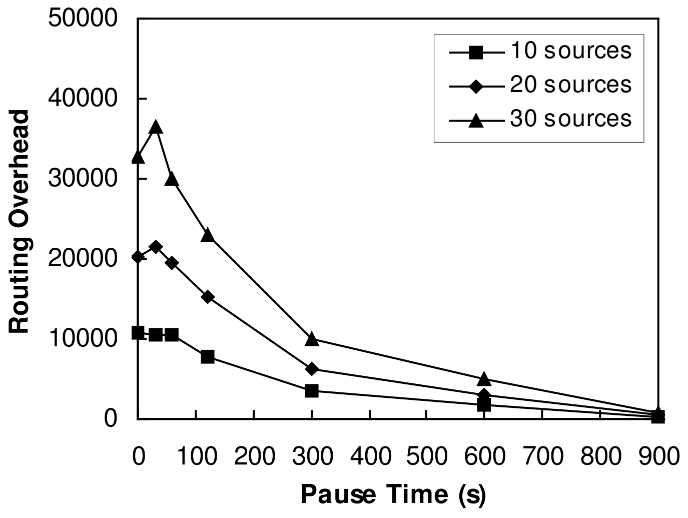


Figure 2.7 DSR Routing Overhead Performance

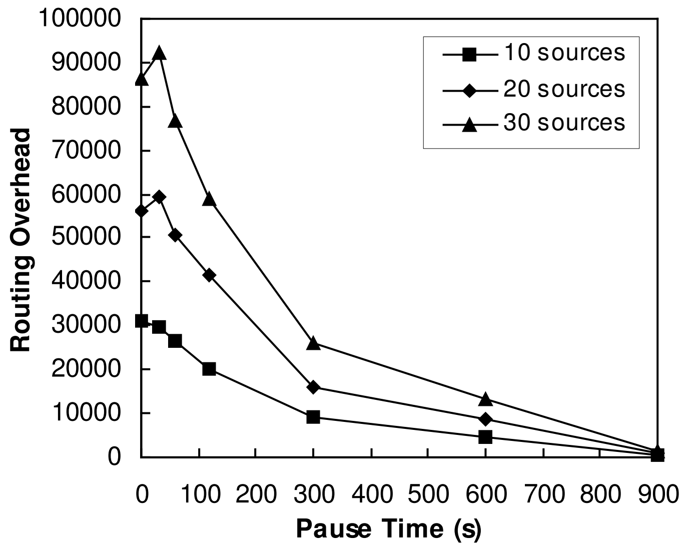


Figure 2.8 AODV-LL Routing Overhead Performance

Applying similar on-demand mechanisms, AODV-LL and DSR show almost identically shaped curves in their routing overhead performance (see **Figures 2.7** and

Figure 2.8). As the majority of control overhead is dedicated to maintenance and repair of broken routes, fast changing topology causes more route breaks and hence incurs larger routing overhead. Compared to AODV-LL, DSR requires less absolute overhead in terms of the number of control packets, largely because DSR stores the entire route information within one control packet, whereas AODV-LL relies on multiple control packets for its distributed hop-by-hop routing. Moreover, DSR typically makes use of promiscuous mode to obtain routing information from forwarded data packets as well as non-propagating RREQs to establish routes quickly from neighbours; whereas AODV's RREQs mostly flood the whole network.

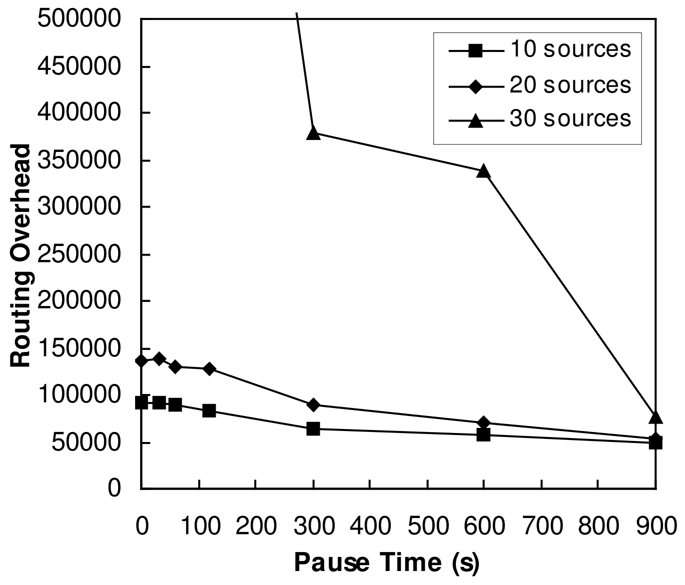


Figure 2.9 TORA Routing Overhead Performance

TORA's overhead consists of the constant mobility-independent overhead from IMEP's periodic beaconing for neighbour discovery, plus a variable mobility-dependent overhead from TORA's route creation and maintenance. In the case of 30 connections, TORA undergoes congestive collapse and fails to converge (see **Figure 2.9**), whereby the number of TORA/IMEP control packets causes numerous MAC-layer collisions, which in turn causes more data and control packets to be lost, and consequently, more and more erroneous perception of link breakage and hence severe local congestion.

Figure 2.10 gives a comparison of the Control Overhead performance of the four protocols in the case of 20 traffic connections. While the proactive scheme DSDV-LL has a constant overhead profile capped by its maximum periodic update transmission rate, on-demand schemes DSR, AODV-LL and TORA exhibits adaptive overhead depending on mobility and traffic load. The results prove that on-demand schemes, especially DSR and AODV-LL, are very cost-effective in all cases compared to DSDV-LL.

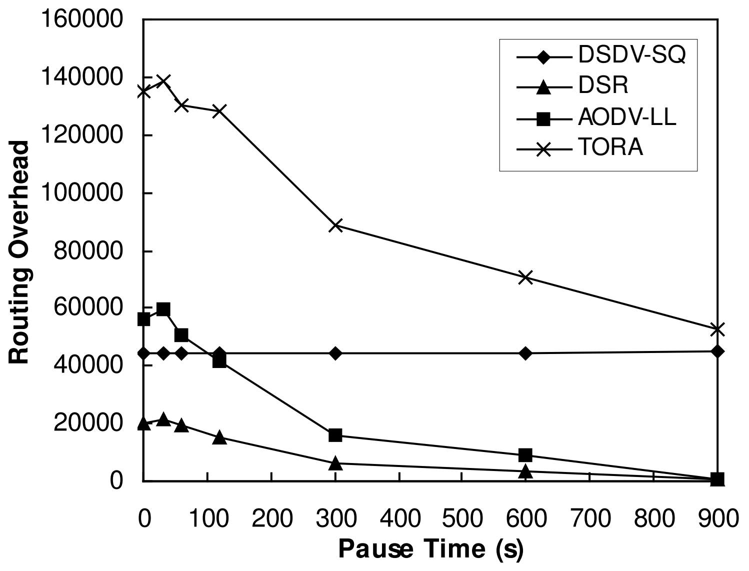


Figure 2.10 Comparison of Routing Overhead Performance (20 connections)

2.4.3 Average End-to-end Delay

The average end-to-end delay measures the average time it takes for a data package to travel from the source node to the destination node. The delay performance of the above four protocols are presented and compared in **Figures 2.11~2.14**.

The graphs show two common traits among the four delay characteristics. Firstly, all protocols produce larger end-to-end delay for data packets when mobility increases. This is expected because at higher mobility, routing information expires faster, whether such information is refreshed through periodic updates as in a proactive network or reactively on-demand. As a result, when mobility increases, nodes start to lose the precise or converging view of the network topology, and more packets are forced to travel along the less optimal routes, or spend more time waiting in cache for routes to be repaired.

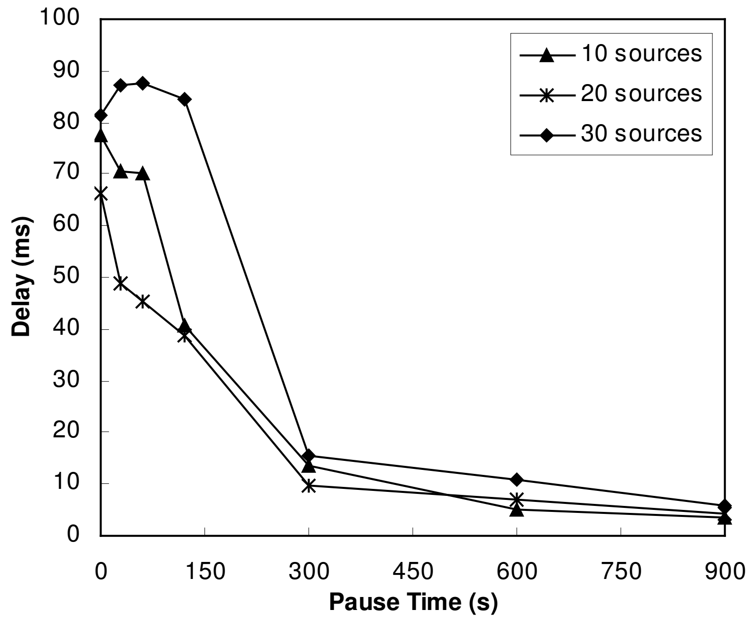


Figure 2.11 DSDV-SQ Average End-to-end Delay Performance

Another common trait is also observed. In all four protocols, the lowest average end-to-end delays occur in the medium traffic load case of 20 sources, rather than the lighter or heavier traffic cases of 10 or 30 sources. This implies that average delay does not increase linearly with growing volume of data packets all the time. Rather, the average delay sees some initial decrease while the data packet volume grows to a certain threshold, beyond which the delay starts to increase together with data packet volume.

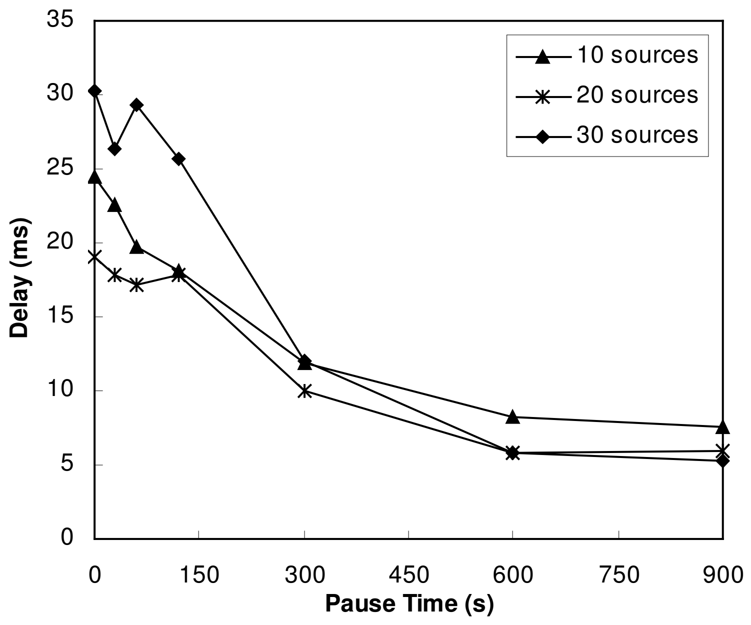


Figure 2.12 DSR Average End-to-end Delay Performance

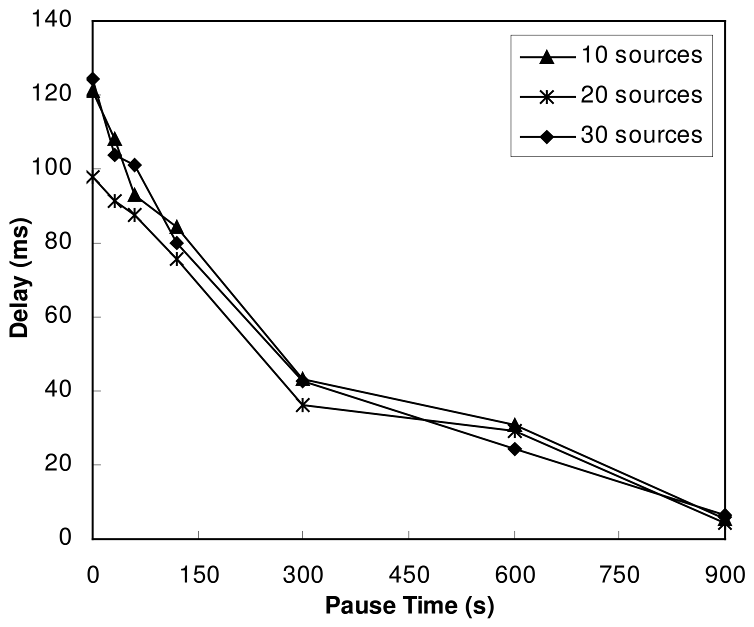


Figure 2.13 AODV-LL Average End-to-end Delay Performance

The above observation reveals that data packet volume is an important factor that contributes to the effectiveness of MANETs. In proactive networks, some level of data traffic helps discover broken links that periodical updates are unable to track, especially when mobility is high. In reactive networks, the initial volume of data packet flows creates the useful demand that triggers route discovery activities by all involving nodes. In general under light load conditions, increase in the number of traffic sources calls for more frequent network updates with wider coverage, and therefore creates more optimum routes and shorter average delay. Such effect will continue until the traffic volume grows to a level that leads to network congestion and starts to negatively impact the delay performance.

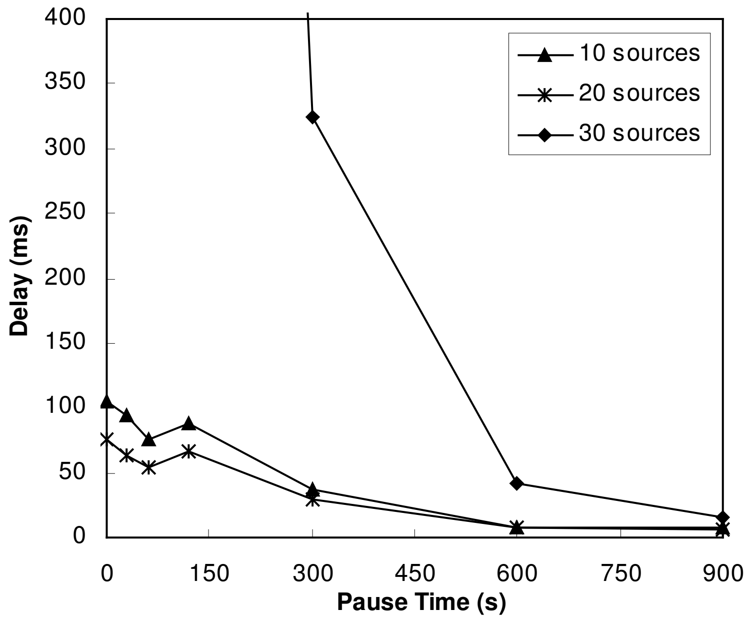


Figure 2.14 TORA Average End-to-end Delay Performance

However, the above four protocols have vastly different levels of sensitivity towards traffic volume. For example, the delay performance of AODV-LL is almost unaffected by change in the number of traffic sources. This is largely related to the highly distributed and localized link update and repair mechanisms used in AODV-LL. As a result, route information gets refreshed very quickly and efficiently without affecting many other nodes in the network. This advantage of AODV-LL not only reduces the impact of increasing traffic volume under light load conditions, but also raises the threshold when traffic load starts to congest the network under heavy load conditions. In contrast, DSDV-SQ encourages massive broadcast of route updates throughout the whole network, DSR relies on lengthy control packets due to centralized routing, and TORA uses a complex algorithm that becomes unstable at higher traffic load. These factors make DSDV-SQ, DSR and TORA more sensitive to changes in traffic volume.

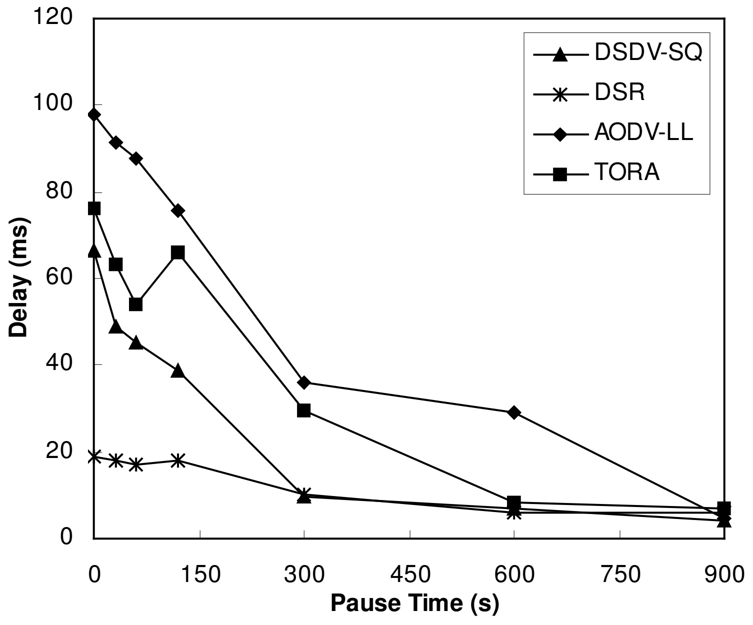


Figure 2.15 Comparison of End-to-end Delay Performance (20 connections)

We can see from **Figure 2.15** that source routing protocols such as DSR exhibit the smallest average end-to-end delay as compared the other protocols. In source routing schemes, the control packet carries the entire route information, rather than only the next-hop information in a distributed routing scheme. This enables data packets to feed through all intermediate nodes along the route with minimal transit delay. In distributed schemes like DSDV-SQ, AODV-LL and TORA, packets stop at each intermediate node to check for the next step to go, which introduces more delay.

2.4.4 Average Delay Jitter

The average delay jitter measures the extent to which the end-to-end delay varies across different data packets. Smaller delay jitter is desirable for delay-sensitive traffic such as multimedia stream data, since it requires less buffering and sequencing of incoming data packets at the destination nodes. **Figures 2.16~2.19** show the delay jitters of the four protocols, calculated as the standard deviation of delays.

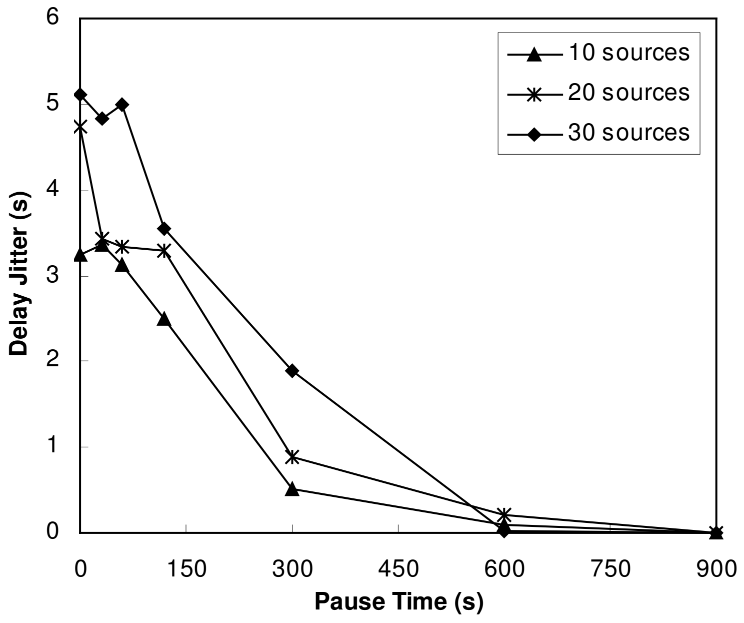


Figure 2.16 DSDV-SQ Average Delay Jitter Performance

The delay jitter curves of these four protocols somewhat resemble their end-to-end delay curves in the previous section, except the linear relation between increasing delay jitter and growing traffic volume. Despite the lowest end-to-end delay, DSR sees a big variation in delay when traffic load changes. AODV-LL on the other hand, maintains very close and steady delay jitters in almost all cases.

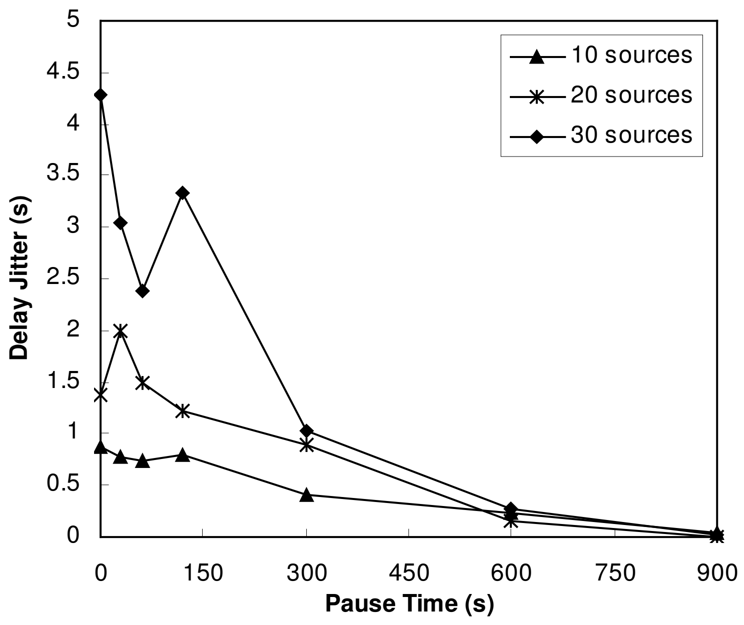


Figure 2.17 DSR Average Delay Jitter Performance

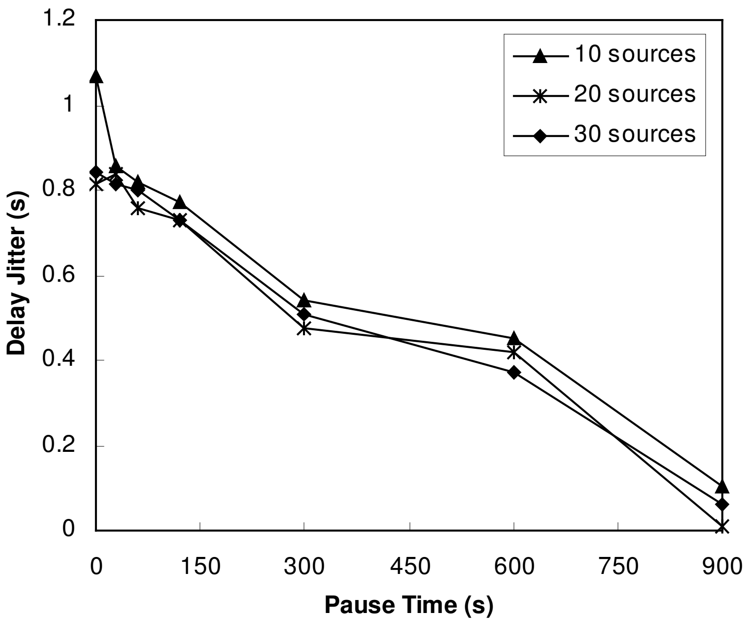


Figure 2.18 AODV-LL Average Delay Jitter Performance

From the graphs we see that in all protocols, while the end-to-end delays are in the millisecond range, the delay jitters easily go up to a few seconds. This means the bulk of the delay jitter comes from the extra time spent on data packet waiting at intermediate nodes, rather than the extra propagation delays due to longer routes. For this reason, delay jitter is mostly related to the how fast the routing mechanism finds a valid route for a waiting data packet, rather than route optimality in terms of how far the data packet needs to travel.

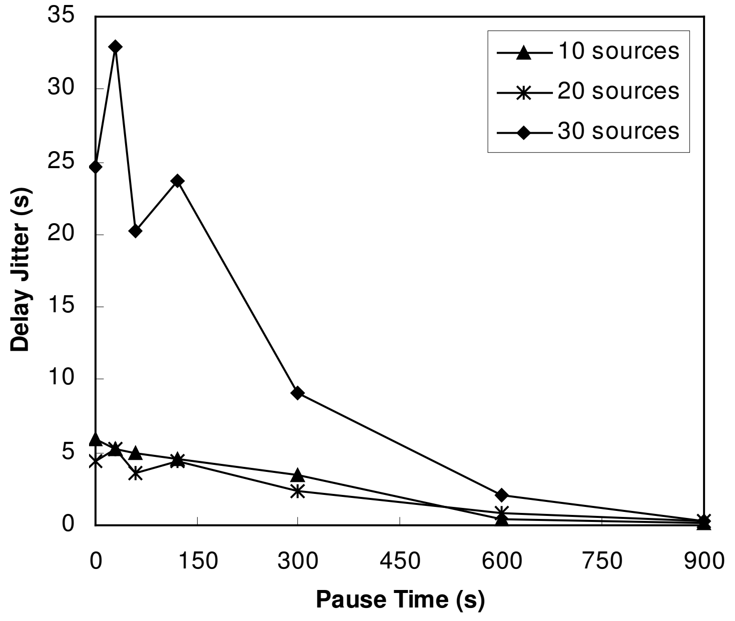


Figure 2.19 TORA Average Delay Jitter Performance

The small delay jitter in AODV-LL again benefits from its efficient distributed route repair mechanism. Because route repair is localized to a small region in AODV-LL, in the event of link failure, smaller extra delay is required for a data packet waiting till the new path is up, and such extra delay is less sensitive to the length of the route or where the broken link locates along the route. In source routing schemes such as DSR, once a link breaks, the entire downstream of the route needs to be re-established. When the broken link is far from the destination, such route repair could take a long time that adds large variation to the overall average delay. In DSDV-SQ and TORA, the massive route update broadcasts and complex routing algorithm limit their ability to keep up with higher mobility. This explains the sudden increase of delay jitter at smaller pause times shown in **Figure 2.20**.

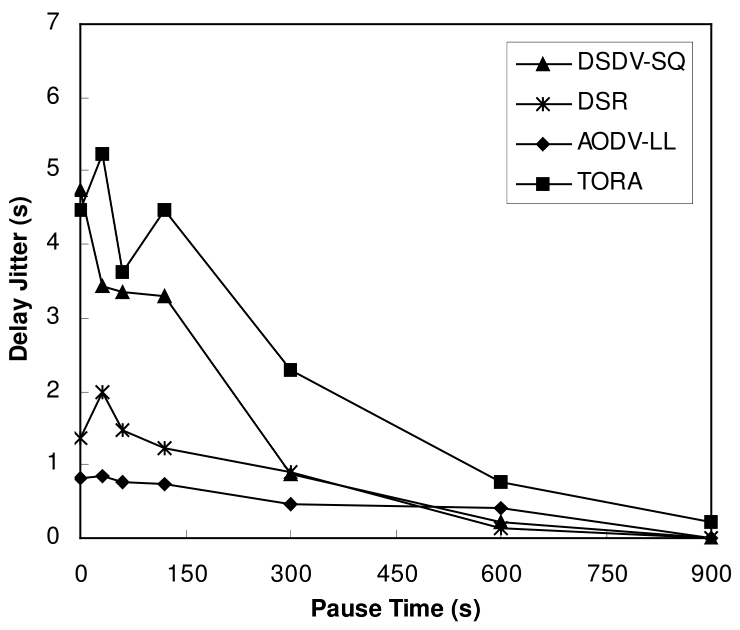


Figure 2.20 Comparison of Average Delay Jitter Performance (20 Connections)

2.4.5 Remarks and Limitations

With similar simulation settings as in the classic review MobileCom'98 paper [1], we obtain similar results as those presented in [1] provided some differences due to the *ns*-2 and protocol codes evolution. In particular, the AODV-LL in our simulations incurs much less routing overhead than the version of AODV-LL used in [1]. This is because the model implementation codes of AODV-LL have been improved much over the version used in [1], reflecting the evolution of the AODV specifications over a few Internet Drafts since the publishing of [1]. The use of better pacing in route discoveries and the introduction of expanding ring search algorithm in the current implementation significantly reduce the routing overhead of AODV-LL.

The control overhead measured in the number of bytes is not accounted in this project, because it is highly dependent on software implementation for the given simulation platform. As a distributed effort from many researchers, the protocol implementation codes in *ns*-2 at the current stage do not adhere strictly to the constantly evolving protocol specifications such as the control packet structures and sizes. Taking the consistent approach as majority of prior works, this project examines only the control overhead in the number of packets, which is a clear indication of how often mobile nodes need to exchange information in order for the routing protocol to operate properly. In **Chapter 3** and **4**, we derive a few variants of the AODV protocol with some enhanced features, without altering the structures and sizes of the AODV control packets. Therefore when comparing their control overheads, counting the number of control bytes would not give significantly different results.

2.4.6 Accuracy of Simulation Estimates

As described in Methodology (**Section 2.3.2**), the simulation for each combination of mobile speed and traffic load is run over 10 different randomly generated node movement scenario files. Each run covers a long period of 900 simulated seconds. The end result for each combination is an average over the 10 outputs from the 10 runs. The purpose of all these is to minimize errors due to inconsistencies in certain scenario files generated. From the experience in [1], results from such simulation settings are internally consistent [1].

While mathematical verification or physical prototyping incurs tremendous complication and constraints, simulation provides a relatively simple and fast characterization process. Indeed, simulations are estimates, and the results depend heavily on the scenario and environment settings. What we are trying to achieve here, is to establish a common simulation environment for all the different routing algorithms, to evaluate and compare their characteristics on a fair basis. The results of such experiment may not be exactly realistic, but they do provide important insights into the similarities and differences of the various protocols operating in a nearly practical environment.

2.5 Conclusion

As the outcome of on-going research effort, many routing protocols for MANETs have been proposed in recent years. In this chapter, we have reviewed different classifications of MANET routing protocols analysing their key characteristics and relative strengths and weaknesses. We then focus our studies on four early MANET routing protocols: DSDV, DSR, AODV and TORA outlining their distinct approaches in details and evaluating their performance through simulations using *ns*-2. The comparison among the four protocols show that on-demand algorithms like DSR and AODV-LL are very efficient and well adapted to different mobility and traffic load cases, while DSDV has its limits on refreshing routing information according to fast changing topology, and TORA needs to improves on its overhead congestion resolution.

CHAPTER THREE
EXPLOITING ROUTE REDUNDANCY

This chapter explores the idea of route redundancy to improve the overall robustness of MANET routing protocols. Briefly addressing the background and motivation for this part of the project, the chapter first gives a detailed review of the structure and operations of AODV as the base protocol for our enhancement. The systematic design of AODV-RR by incorporating redundant routes mechanism into AODV is then described fully. Lastly, the performance of the new protocol AODV-RR is evaluated and compared to that of AODV.

3.1 Background and Motivation

Many of MANET applications, such as e-conferencing and disaster recovery teamwork, involve not only exchange of data and information, but also real-time interaction among different users. In such scenarios, the need to support time-critical multimedia traffic over the dynamic wireless topology brings challenges to the robustness of MANET routing protocols. While conventional protocols focused mainly on performance metrics like packet delivery ratio, routing overhead, and path optimality in terms of the number of hops, parameters in the delay aspect such as route establishment latency and route re-discovery latency tend to be neglected. In the requirement of real-time traffic, delay performance becomes important in measuring the robustness of MANET routing protocols.

Protocols built on different mechanisms behave differently in the delay aspect. Proactive schemes like DSDV constantly maintain routing information regardless of traffic demand. Routes are immediately ready for use at any time when needed. Hence the delay in route discovery is theoretically zero. However, when mobility increases, the periodic network-wide updates in such proactive schemes fail to keep the routing information up-to-date and hence severely degrade data throughput. Therefore, proactive schemes have minimum route discovery latency but are, in general, too inefficient to support intensive real-time traffic. On the other hand, demand-driven schemes like DSR and AODV are more cost-effective and ensure good data throughput in all mobility cases, despite the fact that some delays in route discovery and repair are inevitable in these schemes. This part of the project seeks to explore mechanisms to reduce route discovery or repair latencies in demand-driven protocols in order to achieve improved robustness at minimum cost. AODV is chosen as the base protocol for enhancement.

In the unicast operation of AODV, a packet is either dropped or buffered waiting for repair if it encounters a link break midway along an established route according to routing tables. For time-critical multimedia traffic, such delay incurred due to retransmission or waiting is least desirable. By incorporating some degree of route redundancy and maintaining multiple routes for a single source/destination pair, the interrupted traffic flow can be switched from the broken primary route to an alternative backup route at the point of link failure, thus not only improving the overall data throughput, but also more importantly eliminating possible delay in waiting or retransmission. This is the basic rational behind the redundant routes mechanism.

The work on backup routing in MANETs presented in [18] is one of the first to explore the use of backup routes to salvage data packets at the point of link breakage.

The proposed scheme AODV-BR [18] establishes fish-spine shaped mesh of 1-hop alternative routes by promiscuously overhearing packets transmitted by neighbouring nodes. When a link failure is detected, the data packet is broadcast to all neighbours in the hope that some of these neighbours may carry an alternative 1-hop path to the destination bypassing the broken link. Rather, the AODV-BR proposed by [18] is a simplistic and non-systematic scheme. Strictly, the fish-spine structure of the primary and 1-hop alternative routes is an incomplete mesh, providing only one route for any source/destination pair. The route switching is done by last-minute broadcast of the data packet to all neighbours without precise knowledge of the backup route being selected. In contrast, our design is a systematic algorithm for developing multiple routes between a given source/destination pair, as well as switching data flow from the broken primary route to an properly maintained alternative. We implement this new design based on AODV, and name the new scheme as AODV with Redundant Routes (AODV-RR).

3.2 AODV in Details

In this section we review the structure and operation of AODV in more details. We examine the *ns*-2 simulation codes of AODV for *ns*-2 version 2.1b7 [17], which basically conforms to the AODV Internet Draft version 7 [29]. We first look at the structure of the AODV routing table as well as its control packets. With an overview of AODV operation by illustration, we then decompose the AODV operation into several event-triggered actions in handling various control packets. These are the features we will base our modifications on later.

3.2.1 Routing Table and Control Packets

Each AODV routing entry consists of the fields listed in **Table 3.1**.

Table 3.1 AODV Routing Table Entry Fields

Field	**Remarks**
Destination Address	Address of destination of this route
Destination Sequence Number	Sequence number for the destination
Next Hop Address	Address of next-hop along this route
Hop Count	Number of hops to the destination
Lifetime	Expiration time of this route
Flags	Routing flags
Precursor List	List of previous-hops who have recently used this route, for route maintenance
RREQ Timeout, RREQ Count, Last RREQ TTL, Route Discovery Latency History	For Expanding Ring Search in route discovery; RREQ propagation radius and timeout are increased gradually in subsequent attempts to save routing overhead

As a distributed scheme, AODV maintains only the next-hop address and the hop count of each route to a destination, instead of the entire route as done in source routing schemes such as DSR. This approach reduces storage overhead and localizes protocol reaction to topology change. In addition, AODV uses destination sequence number to achieve loop-freedom. The lifetime field allows automatic deletion of stale routes without additional service. The precursor list is built so that upstream nodes can be notified instantly once a link break is detected. The RREQ timeout and propagation radius (TTL) in subsequent route request attempts are increased gradually according to past RREQ propagation radius (TTL), applying the expanding ring search algorithm to save overhead.

Table 3.2 shows the AODV control packet structures. AODV basically uses three types of control packets: RREQ, RREP and RRER messages.

Table 3.2 AODV Control Packets

Control Packet	Fields	Remarks
RREQ	Type	Packet Type = RREQ
	Hop Count	Number of hops to the source node issuing RREQ
	Broadcast ID	RREQ broadcast ID
	Destination Address	Address of requested destination
	Destination Sequence Number	Last known sequence number for the requested destination
	Source Address	Address of source node issuing RREQ
	Source Sequence Number	Sequence number for source node
	Timestamp	Time at which RREQ is sent
RREP	Type	Packet Type = RREP
	Hop Count	Number of hops to the requested destination
	Destination Address	Address of requested destination
	Destination Sequence Number	Sequence number for the requested destination
	Source Address	Address of source node which issued RREQ
	Lifetime	Lifetime of RREP being valid
	Timestamp	Time at which the corresponding RREQ is sent
RERR	Type	Packet Type = RERR
	Unreachable Destination Address	Address of the unreachable destination due to the detected link failure
	Unreachable Destination Sequence Number	Last known sequence number for the unreachable destination due to the detected link failure

Each of the three types of control packets contains a field indicating the type of the control packet. The RREQ message is used for route discovery. The source node issuing the RREQ fills in the address and the last-known sequence number for the requested destination so that the destination nodes upon receiving the RREQ can recognize and acknowledge. The source address and hop count are intended for all nodes receiving the RREQ to establish reverse routes from themselves to the source node. The broadcast ID is important for ensuring loop-freedom in the reverse routes by making each node accept only one copy of the RREQ message and discard all duplicates of the same RREQ message. The timestamp field in RREQ records the time at which the RREQ is sent, meant for computing route discovery latency later.

On the other hand, the RREP message is generated by the destination in reply to the request upon receipt of the RREQ from the source. The RREP contains the number of hops to the destination as well as the address and sequence number of the destination, allowing all nodes receiving the RREP to establish forward routes to this destination. The lifetime field in RREP indicates how long the forward routes should be valid. The timestamp in RREP is actually inherited from the RREQ being corresponded to, intended for the source node to compute route discovery latency later. As RREP is unicast back to the source along the reverse route, no broadcast ID for RREP is needed.

Lastly the RERR message (equivalent to UREP message mentioned in AODV introduction in **Chapter 1**) is used to notify neighbouring nodes of a detected link failure during route maintenance. The RERR contains the address and last-known sequence number of the unreachable destination due to the link failure. By sending RERR to upstream nodes in the broken route's precursor list, the topology change information is propagated back to the affected source nodes, which forces updates of corresponding route information.

3.2.2 An Overview By Illustration

The AODV operation consists of two phases: route discovery and route maintenance, which are illustrated by **Figure 3.1** and **Figure 3.2**, respectively. In these diagrams, a mobile node is denoted by a square box; the source and destination nodes are marked by **S** and **D** respectively.

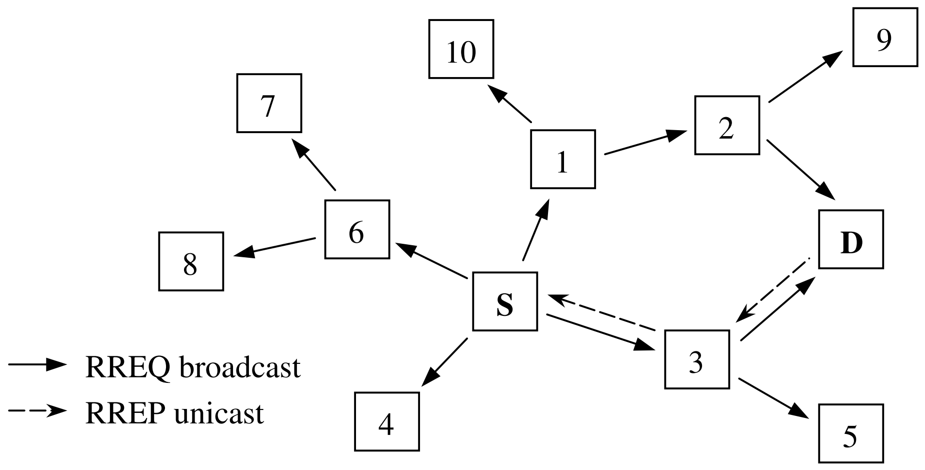


Figure 3.1 Illustration of AODV Route Discovery

The AODV route discovery initiates with the source node broadcasting RREQ to all surrounding neighbours, and completes with the destination unicasting RREP back to the source in reply to the request received. In our illustration, all nodes receiving the RREQ message, in other words each of nodes 1~10 and the destination **D** create a reverse route in their routing tables to the source **S**, using the previous-hop of the RREQ message as the next-hop of the reverse route. Depending on the distance between **S** and **D**, the source **S** may try several attempts re-broadcasting RREQ, with gradually increasing RREQ propagation radius (TTL) following the expanding ring search algorithm. The RREQ broadcast ID ensures each node accepts and forwards only one and the first copy of RREQ, hence preventing duplicated broadcasts and most importantly formation of loops in reverse routes. Therefore, as a result of RREQ broadcast, all neighbouring nodes create unique and shortest reverse routes to the source. Upon receipt of the RREQ, the destination **D** generates a RREP message and unicasts it back to the source **S** along the newly created reverse route. All nodes receiving the RREP, in other words, node 3 and the source **S** establish a forward route in their routing tables to the destination **D**, using the previous-hop of the RREP message as the next-hop of the forward route.

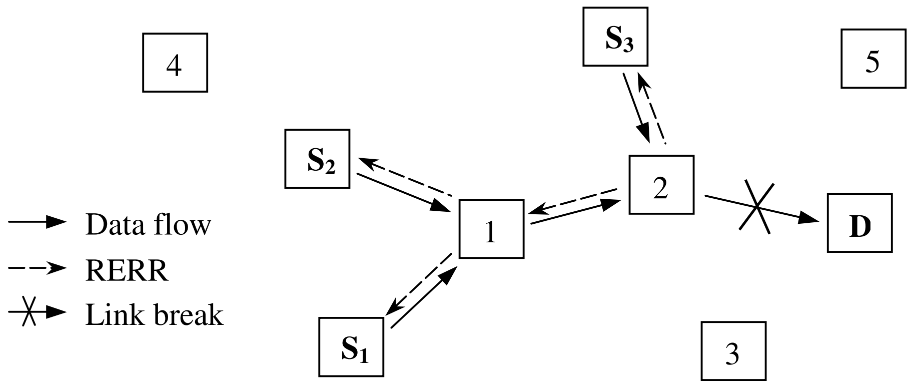


Figure 3.2 Illustration of AODV Route Maintenance

The AODV route maintenance makes use of the RERR message together with the route precursor list feature. Consider the case in our illustration as shown in **Figure 3.2** where three sources **S_1 ~ S_3** have established forward routes to a destination **D** through node 1 and 2. As node 1 and 2 forward data packets from **S_1 ~ S_3** to **D**, they update the precursor list of the route to **D** in their routing tables by inserting the previous-hops of the data packets. When node 2 detects its link to **D** has broken, it generates a RERR message containing **D** as the unreachable destination and forwards it to all neighbours in the precursor list of its broken route to **D**. All nodes receiving the RERR will bring down any of their routes to **D** whose next-hops are the same as the previous-hops of the RERR received, and continue to forward the RERR to all nodes in the precursor lists of these routes they have brought down. This forwarding of RERR continues until the precursor list of the broken route is empty. In this way, upstream nodes of a broken route are quickly informed of the downstream breakage and are forced to update their routing information.

In the next few sections we examine the AODV operation in terms of separate actions handling different control packets and events.

3.2.3 Upon Receipt of RREQ

The flowchart in **Figure 3.3** outlines the handling of an incoming RREQ.

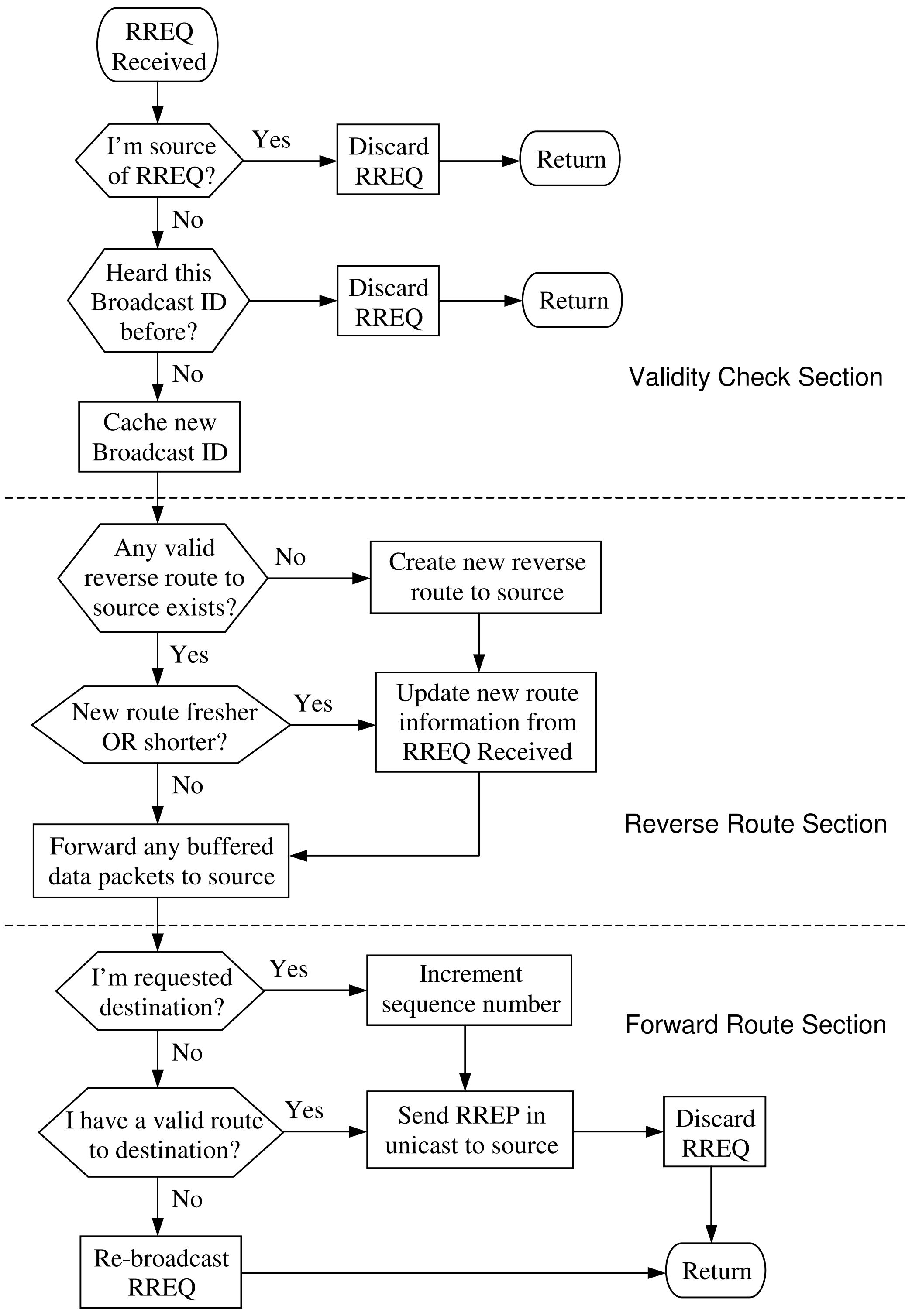

RREQ Received
I'm source of RREQ?
Yes
Discard RREQ
Return
No
Heard this Broadcast ID before?
Discard RREQ
Return
No
Validity Check Section
Cache new Broadcast ID
Any valid reverse route to source exists?
No
Create new reverse route to source
Yes
New route fresher OR shorter?
Yes
Update new route information from RREQ Received
No
Forward any buffered data packets to source
Reverse Route Section
I'm requested destination?
Yes
Increment sequence number
Forward Route Section
No
I have a valid route to destination?
Yes
Send RREP in unicast to source
Discard RREQ
No
Re-broadcast RREQ
Return

Figure 3.3 AODV RREQ Handling Routine

A node only accepts a RREQ message if it is neither the source that issued this RREQ nor it has heard this Broadcast ID before. The node next creates or updates a reverse route to the source in its routing table with the information carried in the RREQ, either if there is no existing such reverse route, or if the existing reverse route has an older sequence number or same sequence number but larger routing metric than those carried by the incoming RREQ. Lastly the node replies to the route request if it is either the destination or has a valid route to the destination. In case it is the destination, it sends a RREP back to the source with a newly incremented sequence number for itself; otherwise if it is an intermediate node, it fills the RREP with the sequence number for that destination recorded in its routing table. In both cases, the node discards the RREQ received. Other nodes (that cannot reply to the request) simply proceed to re-broadcast the RREQ to their neighbours.

3.2.4 Upon Receipt of RREP

Figure 3.4 shows the handling of an incoming RREP message. As RREP is unicast from the requested destination back to source, each node receiving the RREP creates or updates a forward route from itself to the destination in its routing table with the information carried in the RREP, either if there is no existing such forward route, or if the existing forward route has an older sequence number or same sequence number but larger routing metric than those carried by the incoming RREP. If the source node receives the RREP, it simply discards the RREP. Otherwise, if an intermediate node receives the RREP, it continues to forward RREP back to the source through a reverse route established previously. In both cases, any data packets queuing for that destination are sent on the newly created forward route.

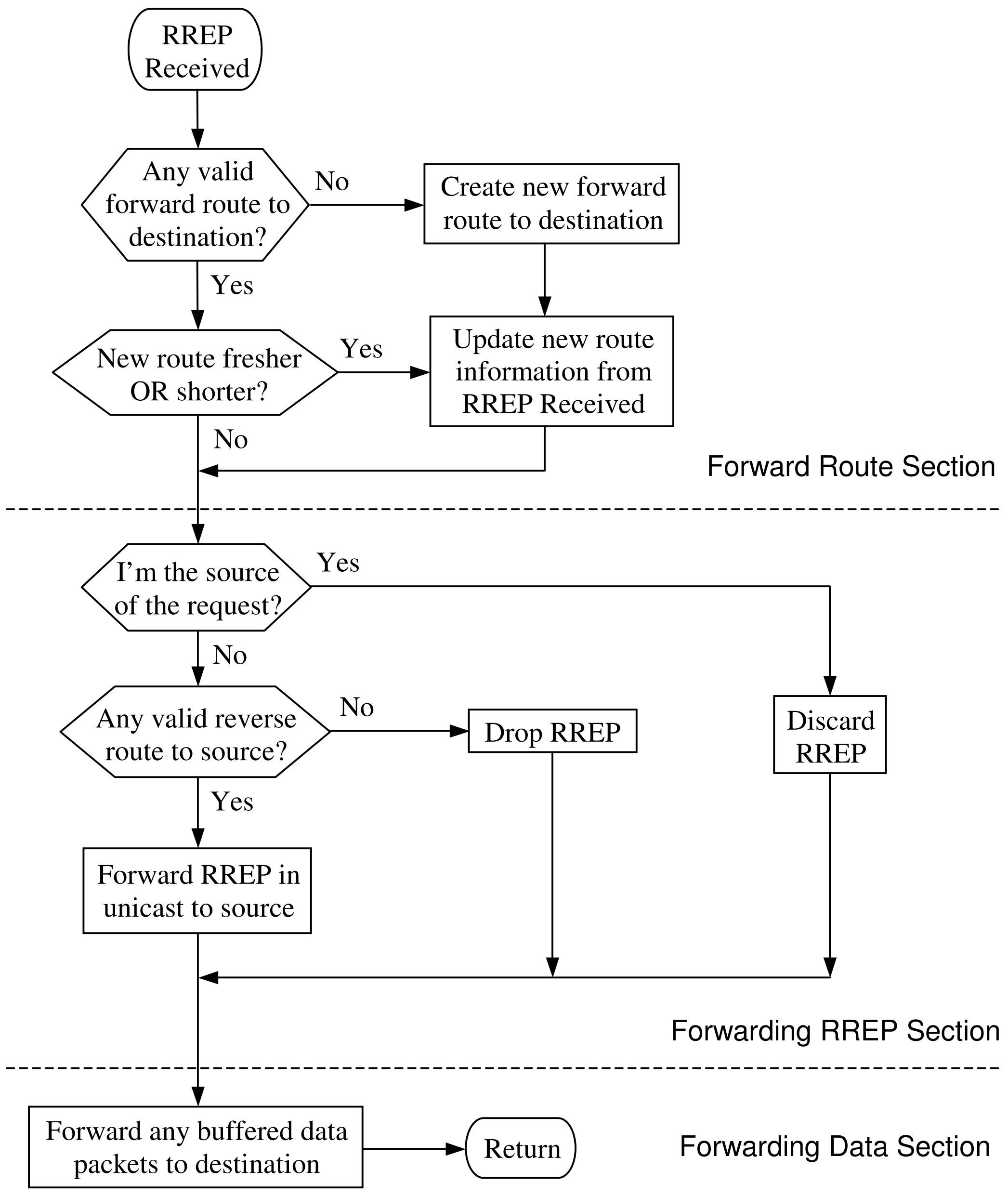
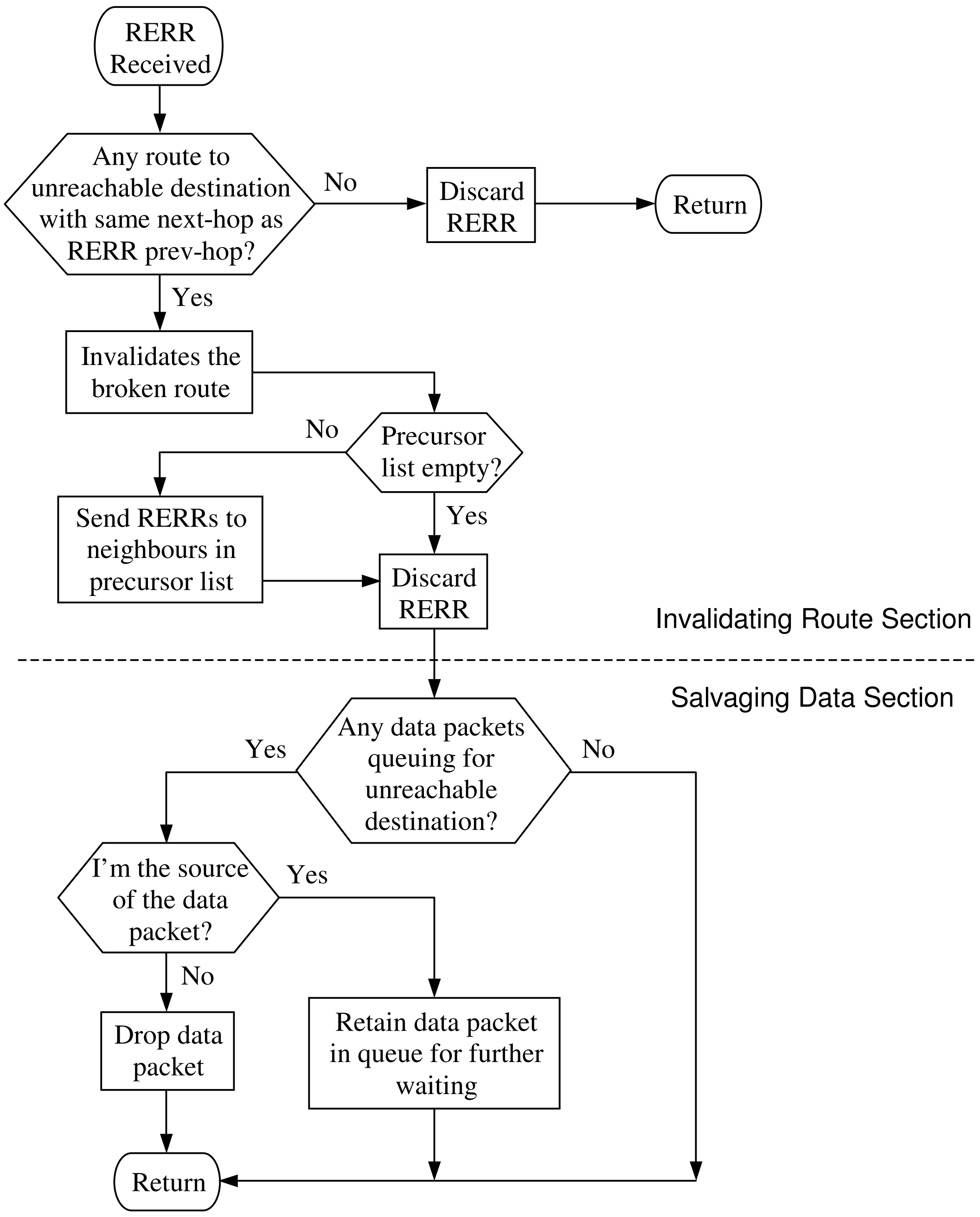

RREP Received
Any valid forward route to destination?
No
Create new forward route to destination
Yes
New route fresher OR shorter?
Yes
Update new route information from RREP Received
No
Forward Route Section
I'm the source of the request?
Yes
No
Any valid reverse route to source?
No
Drop RREP
Discard RREP
Yes
Forward RREP in unicast to source
Forwarding RREP Section
Forward any buffered data packets to destination
Return
Forwarding Data Section

Figure 3.4 AODV RREP Handling Routine

3.2.5 Upon Receipt of RERR

Figure 3.5 shows the handling of an incoming RERR message. Upon receiving a RERR message, a node locates and invalidates the route to the unreachable destination in its routing table whose next-hop is the neighbour from which the RERR comes in, and sends more copies of RERR messages to the neighbours in the precursor list, if any, to notify them about the route breakage. If there are data packets queuing for the unreachable destination, only those data packets for which this node is the source are salvaged and retained in the queue for further waiting. Other data packets forwarded at this intermediate node are dropped.



RERR Received
Any route to unreachable destination with same next-hop as RERR prev-hop?
No
Discard RERR
Return
Yes
Invalidates the broken route
No
Precursor list empty?
Yes
Send RERRs to neighbours in precursor list
Discard RERR
Invalidating Route Section
Salvaging Data Section
Yes
Any data packets queuing for unreachable destination?
No
I'm the source of the data packet?
Yes
No
Drop data packet
Retain data packet in queue for further waiting
Return

Figure 3.5 AODV RERR Handling Routine

3.2.6 Route Resolution

Figure 3.6 shows how data packets are serviced. The node first checks its routing table for any valid route to the destination of the data packet. If such a route exists, the data packet is forwarded. Otherwise, the node queues the data packet if it is the source of this data packet; or, as an intermediate node for the data packet, this node drops the packet and informs upstream nodes about the unreachable destination.

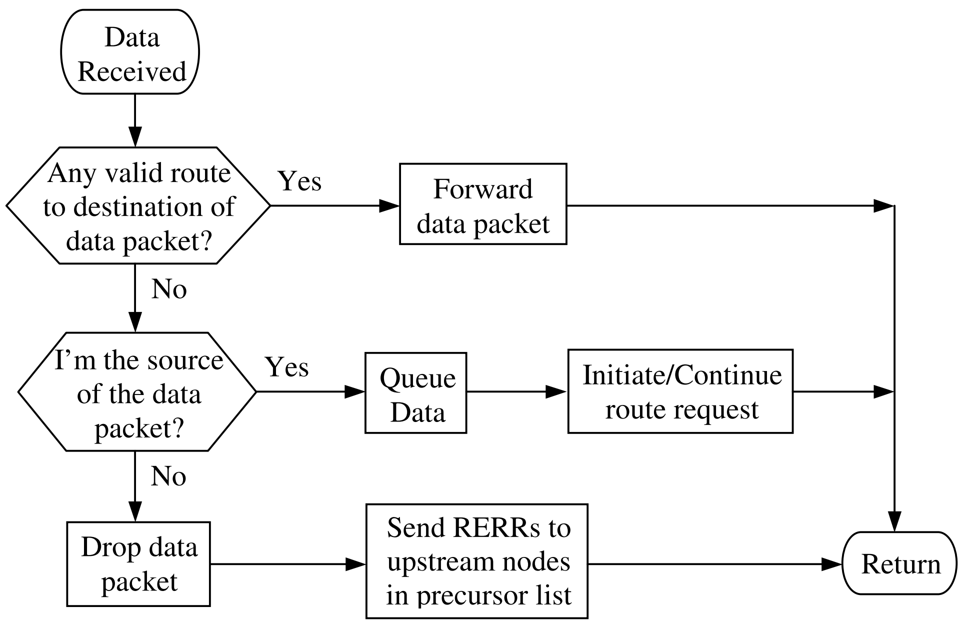

Data Received
Any valid route to destination of data packet?
Yes
Forward data packet
No
I'm the source of the data packet?
Yes
Queue Data
Initiate/Continue route request
No
Drop data packet
Send RERRs to upstream nodes in precursor list
Return

Figure 3.6 AODV Route Resolution For Data Packets

3.2.7 Route Breakage Control

Figure 3.7 shows the action taken when a route breakage is detected. In AODV-LL, a link failure is detected when the MAC layer fails to deliver a packet to a next-hop neighbour. In such an event, the routing protocol quickly increments the sequence number for the unreachable destination, and proceeds to invalidate the route and salvage data packets in a similar fashion as handling RERR messages.

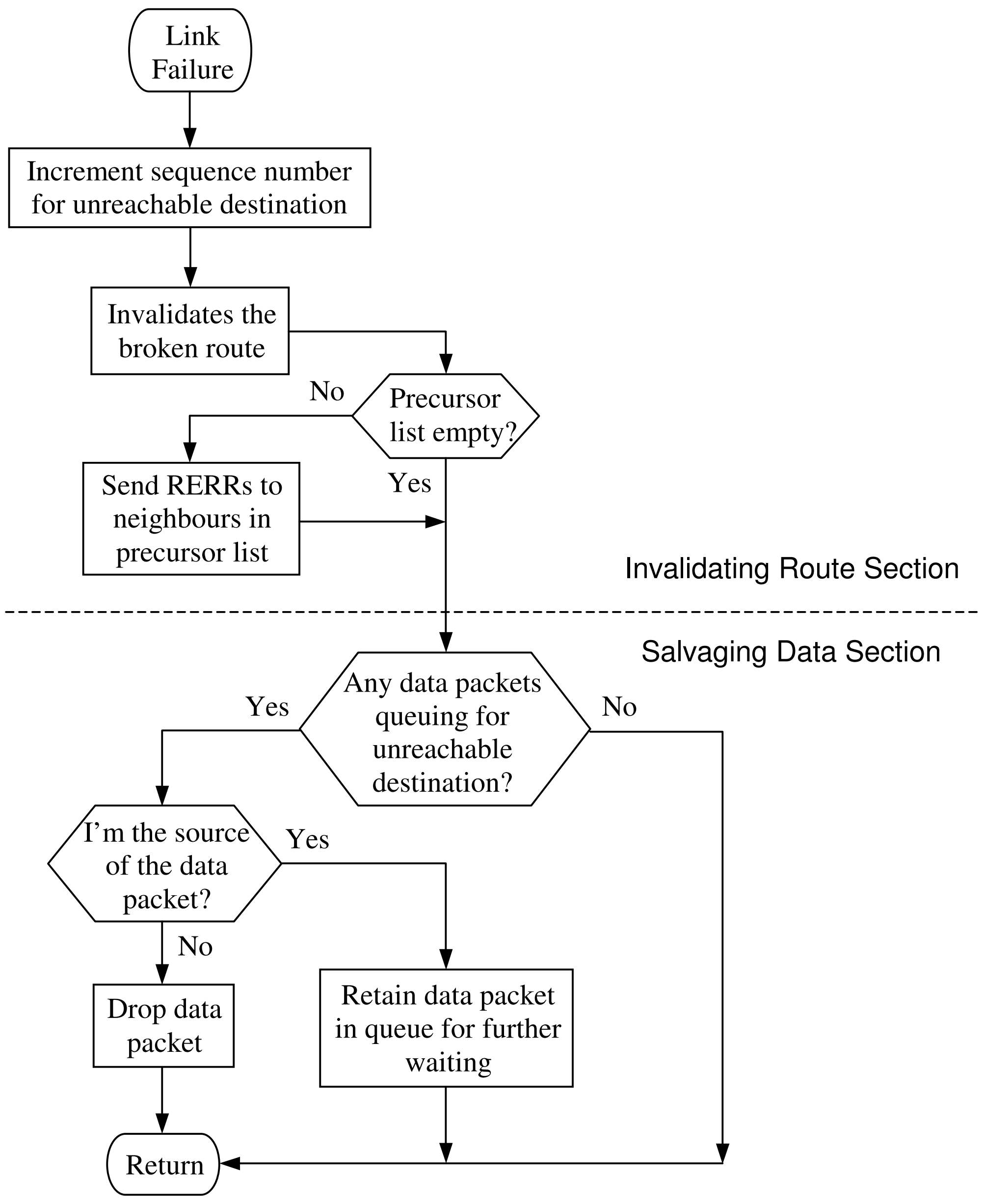

Link Failure
Increment sequence number for unreachable destination
Invalidates the broken route
Precursor list empty?
No
Yes
Send RERRs to neighbours in precursor list
Invalidating Route Section
Salvaging Data Section
Any data packets queuing for unreachable destination?
Yes
No
I'm the source of the data packet?
Yes
No
Drop data packet
Retain data packet in queue for further waiting
Return

Figure 3.7 AODV Route Breakage Control

The AODV proposal [29] defines the local repair mechanism, which allows intermediate nodes to initiate route request to repair a broken route locally, if the breakage point is nearer to the destination end. According to the AODV simulation codes, however, the local repair mechanism does not show significant performance improvement so far. Therefore the local repair mechanism has been disabled in the current AODV simulation codes, and similarly in our later enhanced version of AODV-RR simulation codes.

3.3 Route Redundancy Strategies

The idea of route redundancy seeks to supply backup routes for switching the traffic flow with minimum latency when the primary route fails. For this mechanism to work, multiple routes must be established for a given source/destination pair. We consider several possible methods for building redundant routes into AODV.

3.3.1 Exploiting RREQ Broadcast

We first think of creating multiple reverse routes from each intermediate node to the source node, as a means to create multiple forward routes later. This can be achieved by allowing each node to accept more than one copy of RREQ messages during the RREQ broadcast. Subsequently the destination can send multiple RREP messages back to the source acknowledging the multiple reverse routes created, hence establishing multiple forward routes. This approach is illustrated in **Figure 3.8**.

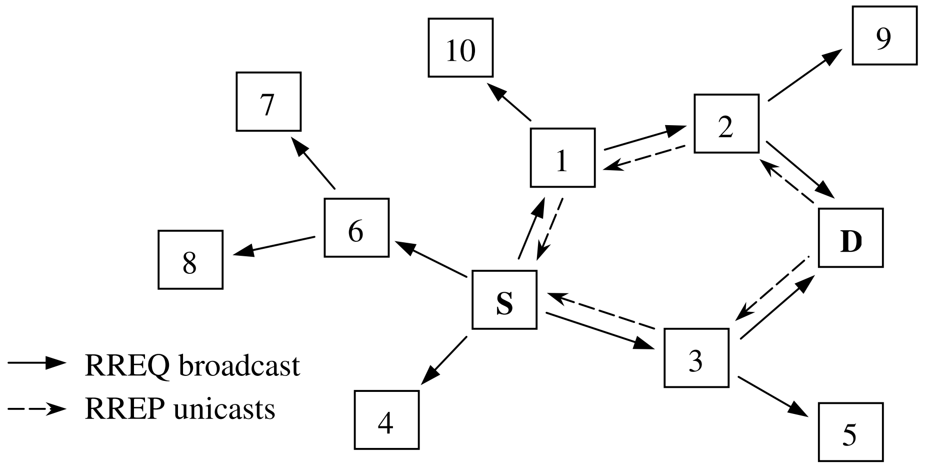


Figure 3.8 Multiple Reverse Routes From RREQ Broadcast

In the above illustration, by accepting more copies of RREQ messages from both node 3 and node 2, **D** creates two reverse routes back to **S** through node 3 and node 2 respectively. With **D** sending multiple RREP messages to acknowledge each of these reverse routes, **S** eventually establishes multiple forward routes, from **S** to **D** through node 3 and node 1 correspondingly.

After some preliminary experiments, we realize a few difficulties with this approach. Firstly, multiple RREP messages introduce a substantial amount of overhead considering the scope of RREQ broadcast. Secondly, as the multiple RREQs come in at different time instants, handling of different sequence numbers of the corresponding multiple RREPs requires complex algorithms. Lastly but definitely most importantly, accepting multiple RREQ messages may create loops in reverse routes, as illustrated in **Figure 3.9**.

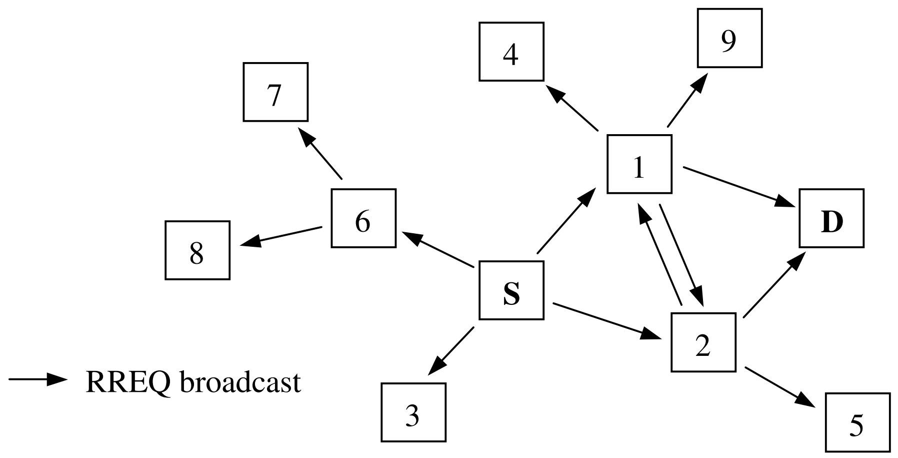


Figure 3.9 Loop in Reverse Route from RREQ Broadcast

In the above scenario, by accepting the RREQ forwarded from each other, both node 1 and node 2 create a second reverse route from themselves to **S** through each other. If the links from nodes 1 and 2 to **S** break, node 1 and node 2 will forward any packet to **S** to each other back and forth endlessly, as wrongly perceived alternatives to reach **S**. Subsequently, the loop in reverse routes directly causes a loop in the forward routes after the RREP unicasts, which will result in endless data packet exchanges between nodes 1 and 2 if the links from nodes 1 and 2 to **D** break.

3.3.2 Exploiting RREP Broadcast

Since accepting multiple copies of any control packets in broadcast inevitably causes routing loops, we abandon the previous approach and consider making use of RREP broadcast as a means to build redundant routes. Since existing controlled RREQ broadcast effectively establishes a unique shortest reverse route from each node to the source, a broadcast of RREP in a same controlled manner should similarly establish a unique shortest forward route from each node to the destination. We can imagine the result of such RREQ broadcast and RREP broadcast as the growth of two spanning trees of routes rooted at the source and the destination respectively, as illustrated in **Figure 3.10**. The branches of these two trees intersect at multiple points, creating multiple alternative paths connecting nodes at the various intersection points to the root at the destination.

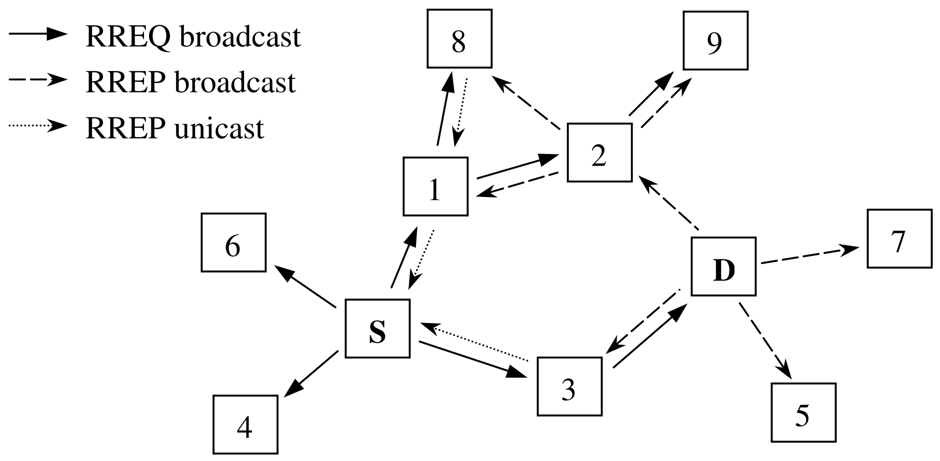


Figure 3.10 Multiple Forward Routes from RREP Broadcast

In the above illustration, the RREQ broadcast from **S** creates a tree of unique shortest reverse routes from nodes 1, 2, 3, 4, 6, 8, 9 to **S**; whereas the RREP broadcast from **D** creates a tree of unique shortest forward routes from nodes 3, 5, 7, 2, 1, 8, 9 to **D**. The branches of these two trees intersect at various nodes 3, 1, 2, 8, and 9. By unicasting RREP messages back to **S** from the different intersection points at nodes 3, 2 and 8, upstream nodes such as nodes 1 and **S** eventually establish multiple forward routes to **D**, through nodes 3, 2 and 8 respectively. The shortest forward route **S**→3→**D** becomes the primary route, and the alternative paths **S**→1→2→**D** and **S**→1→8→2→**D** serve as backup routes once the primary route fails.

It does not matter if some of these alternative routes have common links, such as link 2→**D** shared by alternate routes **S**→1→2→**D** and **S**→1→8→2→**D** in the above example. Given the distributive nature of AODV-RR, each node only sees the next-hop, and it will create multiple route entries only when it sees multiple different next-hops to the same destination. So in our example, node 2 only sees one route to destination **D** through the next-hop **D**, whereas node 1 sees two next-hops: 2 and 8 that both lead to **D.** In other words, node 2 has only one choice when routing packets to **D**, while node 1 has two choices. Essentially, common links are where multiple routes merge and defeat the benefit of route redundancy. If a common link breaks, the upstream node has no choice but to trigger the route repair.

Despite the expected increase in the routing overhead due to RREP broadcast, the above method a simple and reliable way to build route redundancy, not only from the source to the destination, but also from all nodes under the overlapping coverage of the two spanning trees to the destination. This approach requires minimum modification to the existing AODV route discovery procedure and protocol structure. The controlled RREP broadcast is simple to implement. The major remaining design issue is how such multiple routes are managed in the routing table. Next we present our new scheme AODV-RR with systematic algorithms for creating and maintaining redundant routes in detail.

3.4 AODV with Redundant Routes (AODV-RR)

We adopt the above mentioned method of creating redundant routes from RREP broadcast, and develop systematic algorithms for storing, selecting and managing these redundant routes in the routing table. In the next few sections, we first examine the routing table structure and control packet formats of the new scheme AODV-RR, next overview its operation by illustration, and last describe its behaviour in terms of handling of various control packets and events, highlighting the differences and modification made based on the original AODV (see **Section 3.2**).

3.4.1 Routing Table and Control Packets

The original AODV allows a unique freshest and shortest route between each source/destination pair. In contrast, with route redundancy incorporated in the new scheme, nodes establish and maintain more than one route to a given destination. These multiple route entries to the same destination are distinguished by their different next-hops. They share a same destination sequence number as they point to the same destination that issued the RREP messages. **Figure 3.11** and **Figure 3.12** compare the route table structures of AODV and AODV-RR by examples.

AODV Routing Table at node S:					
Route 1:	To Dest **D$_1$**	Nexthop = A	Hops = 2	Seqno = 100	…
Route 2:	To Dest **D$_2$**	Nexthop = B	Hops = 3	Seqno = 200	…
Route 3:	To Dest **D$_3$**	Nexthop = C	Hops = 1	Seqno = 300	…
…	…	…	…	…	

Figure 3.11 AODV Routing Table Structure

AODV-RR Routing Table at node S:						
Route 1-1:	To Dest $\mathbf{D_1}$	Nexthop = A	Seqno = 100	Hops = 2	Cat = Pri	…
Route 1-2:	To Dest $\mathbf{D_1}$	Nexthop = B	Seqno = 100	Hops = 3	Cat = Alt	…
Route 1-3:	To Dest $\mathbf{D_1}$	Nexthop = C	Seqno = 100	Hops = 3	Cat = Alt	…
Route 2-1:	To Dest $\mathbf{D_2}$	Nexthop = E	Seqno = 200	Hops = 2	Cat = Pri	…
Route 2-2:	To Dest $\mathbf{D_2}$	Nexthop = A	Seqno = 200	Hops = 2	Cat = Alt	…
Route 3-1:	To Dest $\mathbf{D_3}$	Nexthop = F	Seqno = 300	Hops = 1	Cat = Pri	…
Route 3-2:	To Dest $\mathbf{D_3}$	Nexthop = G	Seqno = 300	Hops = 2	Cat = Alt	…
Route 3-3:	To Dest $\mathbf{D_3}$	Nexthop = B	Seqno = 300	Hops = 3	Cat = Alt	…
…	…	…	…	…	…	…

Figure 3.12 AODV-RR Routing Table Structure

For each destination, the shortest route created first is marked as primary route. Other subsequent routes to the same destination are marked as alternative routes, for use as backup routes once the primary route fails. Alternative routes carry the same destination sequence number as the primary route. If a route is created with a fresher destination sequence number, it replaces the old primary route for that destination, and deletes all corresponding old alternative routes in the routing table too. Since alternative routes are always created later than the primary route, they have larger or equal number of hops away from the destination compared to the primary route. Two or more alternative routes to a same destination may have equal hop counts. When selecting a backup route from these alternatives, the alternative route entry that is newer in time is preferred. For example (see **Figure 3.12**), in the event that primary route 1-1 breaks, the latest alternative route 1-3 is preferred over 1-2 despite the same hop counts. We define the collection of the primary and alternative routes to a same destination as a "route group" from the current node to that destination.

As listed in **Table 3.3,** the format of AODV-RR route entries is similar to that of AODV's (see **Table 3.1**) but with some additional fields and differences.

Table 3.3 AODV-RR Routing Table Entry Fields

Field	**Remarks**
Destination Address	Address of destination of this route
Destination Sequence Number	Sequence number for the destination
Next Hop Address	Address of next-hop along this route
Hop Count	Number of hops to the destination
Lifetime	Expiration time of this route
Flags	Routing flags
Route Category (Cat) Flag	Category of route: Primary or Alternate
Route Acknowledged Flag (Ack)	Indicates if this reverse route has been acknowledged with a unicast RREP
Precursor List	List of previous-hops who have recently used this route, for route maintenance
RREQ Timeout, RREQ Count, Last RREQ TTL, Route Discovery Latency History	For Expanding Ring Search in route discovery; RREQ propagation radius and timeout are increased gradually in subsequent attempts to save routing overhead

Leaving the original AODV routing flags field intact, two additional flags are added to each AODV-RR routing table entry: Route Category Flag and Route Acknowledged Flag. The Route Category Flag marks the category of this route, either primary or alternative. Primary routes are shortest and created first. Alternative routes are created subsequently with same sequence number for that destination. The Route Acknowledged Flag indicates whether a reverse route has been acknowledged by sending a RREP in unicast back to source. This is to prevent each reverse route from being acknowledged more than once with a RREP carrying the same destination sequence number, which otherwise causes wastage of system resources and control overhead. For all forward routes, the Route Acknowledged Flag is set.

In the context of redundant route, the hop count field of each route entry may not be the shortest distance from this node to the destination. Rather it is the actual distance to the destination along the primary or alternative route, which may not be optimal. Associating a precursor list to every route entry requires more storage overhead. In addition, the sharing or inheritance of precursor lists among routes within a route group becomes difficult to handle, since the primary and alternative routes have different service times and lifetimes. Therefore we abandon the use of precursor lists in AODV-RR. Instead of unicasting RERRs to all nodes in the precursor list during the route breakage control phase, we rely on 1-hop broadcasts of RERRs to propagate link breakage information to all affected nodes nearby. This method is much easier to implement and it greatly reduces the overhead from transmitting RERR messages.

Table 3.4 shows the AODV-RR control packet formats. Compared to AODV (see **Table 3.2**), AODV-RR introduces a new type of RREP message meant for broadcast. To differentiate the two types of RREP messages, we denote the RREP for broadcast as RREP-b, and the original RREP for unicast as RREP-u.

The RREP-b is the same as the RREP-u except for an extra RREP Broadcast ID field for controlling its broadcast. In contrast to the RREQ Broadcast ID field in the RREQ message, the RREP Broadcast ID is issued by the destination, whereas the former is issued by the source. The RREP-u carries the same destination sequence number as RREP-b, since they originate from the same destination.

The RERR message in AODV-RR is identical to that in AODV. However, it is to be noted that the RERR in AODV-RR is meant for 1-hop broadcast, whereas the RERR in the original AODV is for unicast. In AODV-RR, broadcast and forwarding of RERR messages is limited among those upstream nodes affected by the link breakage only. Hence no broadcast ID is necessary for controlling the RERR in AODV-RR. The TTL of RERR broadcast is always set to one.

Table 3.4 AODV-RR Control Packets

Control Packet	**Fields**	**Remarks**
RREQ	Type	Packet Type = RREQ
	Hop Count	Number of hops to the source node issuing RREQ
	Broadcast ID	RREQ broadcast ID
	Destination Address	Address of requested destination
	Destination Sequence Number	Last known sequence number for the requested destination
	Source Address	Address of source node issuing RREQ
	Source Sequence Number	Sequence number for source node
	Timestamp	Time at which RREQ is sent
RREP-b	Type	Packet Type = RREP
	Hop Count	Number of hops to the requested destination
	Broadcast ID	RREP broadcast ID
	Destination Address	Address of requested destination
	Destination Sequence Number	Sequence number for the requested destination
	Source Address	Address of source node which issued RREQ
	Lifetime	Lifetime of RREP being valid
	Timestamp	Time at which the corresponding RREQ is sent
RREP-u	Type	Packet Type = RREP
	Hop Count	Number of hops to the requested destination
	Destination Address	Address of requested destination
	Destination Sequence Number	Sequence number for the requested destination
	Source Address	Address of source node which issued RREQ
	Lifetime	Lifetime of RREP being valid
	Timestamp	Time at which the corresponding RREQ is sent
RERR	Type	Packet Type = RERR
	Unreachable Destination Address	Address of the unreachable destination due to the detected link failure
	Unreachable Destination Sequence Number	Last known sequence number for the unreachable destination due to the detected link failure

3.4.2 An Overview By Illustration

To incorporate route redundancy, the route discovery and maintenance of AODV-RR significantly differ from those in AODV. We overview these operations through illustrations in **Figure 3.13** and **Figure 3.14** respectively.

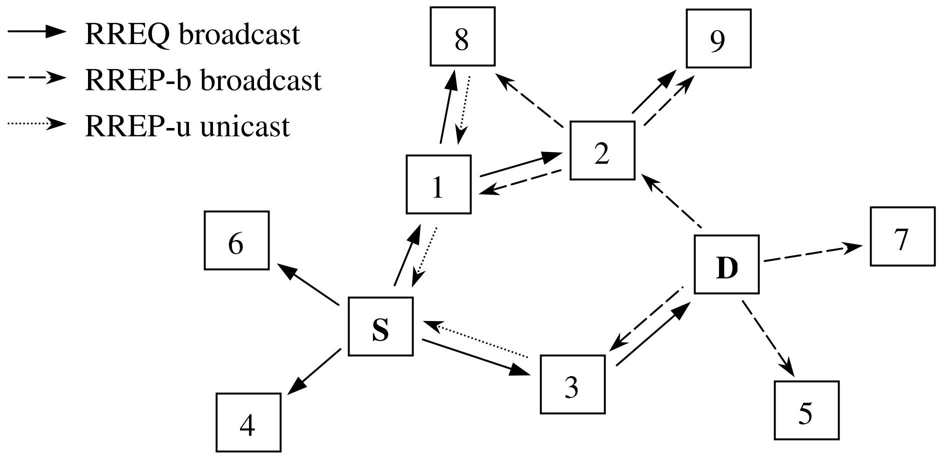


Figure 3.13 Illustration of AODV-RR Route Discovery

The source **S** initiates route discovery by broadcasting RREQ. As the RREQ floods the network, nearby nodes 1, 2, 3, 4, 6, 8, 9 create unique shortest reverse routes from themselves to **S**. The RREQ broadcast is controlled such that each node accepts only one and first copy of RREQ they hear. The first copy of RREQ received by the destination **D** (from node 3) triggers the broadcast of RREP-b immediately after **D** creates a reverse route to **S** in its routing table. With a TTL equal to the number of hops to **S** indicated in the RREQ (=2), the RREP-b broadcast is controlled in the same manner as the RREQ broadcast, causing nearby nodes 3, 5, 1, 2, 7, 8, 9 to establish unique shortest forward routes to **D**. The above two spanning trees from RREQ and RREP-b broadcasts interact with each other according to following basic rules:

- When a node accepts a RREQ, it always creates a shortest reverse route back to **S**. If the node is the destination or already has a valid forward route to **D** created from a previous RREP-b/u, the node unicasts an RREP-u back to **S** acknowledging the new reverse route, and stops broadcasting the RREQ further. Otherwise, the node proceeds to broadcast the RREQ to its neighbours.

- When a node accepts an RREP-b, it always creates a fresh shortest forward route to **D**. If the node already has a valid but unacknowledged reverse route to **S** created from a previous RREQ, it unicasts an RREP-u back to **S** along the unacknowledged reverse route. In any case, the node proceeds to broadcast the RREP-b to its neighbours unless the TTL of the RREP-b becomes zero.
- When a node accepts an RREP-u, it always creates a fresh shortest forward route to **D**. If the node already has a valid but unacknowledged reverse route to **S** created from a previous RREQ, it forwards the RREP-u back to **S** along the unacknowledged reverse route; otherwise it simply discards the RREP-u.

For example, node 2 first creates a reverse route to **S** through node 1 from the RREQ broadcast. Later upon receiving the RREP-b from **D**, node 2 creates a forward route to **D**, and unicasts an RREP-u back to **S** along the unacknowledged reverse route through node 1. In addition, node 2 further forwards the RREP-b in broadcast to its neighbours. As a result, node 8 receives the RREP-b and creates a forward route from itself to **D**. Since node 8 already has a reverse route created earlier, it also unicasts an RREP-u back to **S** along the unacknowledged reverse route. Correspondingly, node 1 will receive two consecutive copies of RREP-u messages from node 2 first and node 8 next respectively. Hence node 1 establishes two forward routes to **D**, one through node 2 as primary route, and the other through node 8 as alternative. After creating the primary forward route, node 1 forwards the RREP-u back to **S** acknowledging its reverse route. The second copy of RREP-u is discarded since node 1 has no unacknowledged reverse route to **S**. Therefore, the source **S** only sees the availability of a forward route to **D** through node 1, not knowing further redundant routes from node 1 downwards. In other words, in such a distributed protocol as AODV-RR, each node is aware of route redundancy at its immediate next-hop only. In the above example, prior to the knowledge of forward route to **D** through node 1, **S** also has received RREP-u from node 3. Consequently **S** establishes two forward routes to **D**, the primary through node 3, and one alternative through node 1.

There is one exception when forwarding RREP-u. If the node receives an RREP-b from the next-hop of its unacknowledged reverse route, the RREP-u is never forwarded along the reverse route. This prevents the formation of routing loops between this node and the next-hop of its reverse route. An example of this case is node 9 in our illustration; when it receives the RREP-b from node 2, it does not unicast an RREP-u back to **S**, because the next-hop of its reverse route to **S** is through node 2.

Figures 3.14(a~c) illustrate the route maintenance in AODV-RR. With multiple forward routes established, the primary route is always preferred over alternative routes. So in our example, initially the source **S** sends all its data packets to **D** along its primary route to **D** through node 3. We explain the protocol reactions in a few cases when different links break.

In general, with redundant routes established, the breakage of a single next-hop link does not render the destination unreachable, unless all routes in the route group are down. Therefore, the RERR propagation is triggered only after all routes in the route group to that destination have broken. If the next-hop link of the one route breaks, the node switches the traffic flow to the next optimal valid alternative route, and triggers a RERR broadcast to upstream nodes only if no more alternative routes are available.

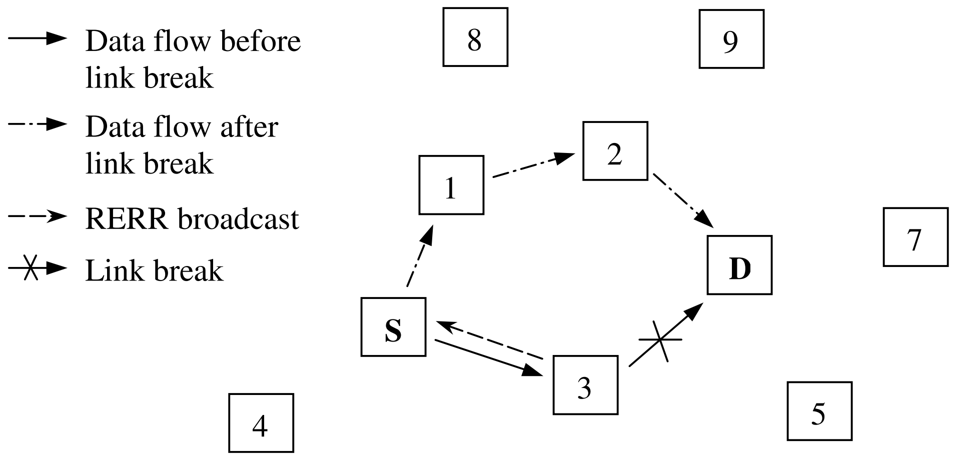


Figure 3.14(a) Illustration of AODV-RR Route Maintenance (Link 3→D Breaks)

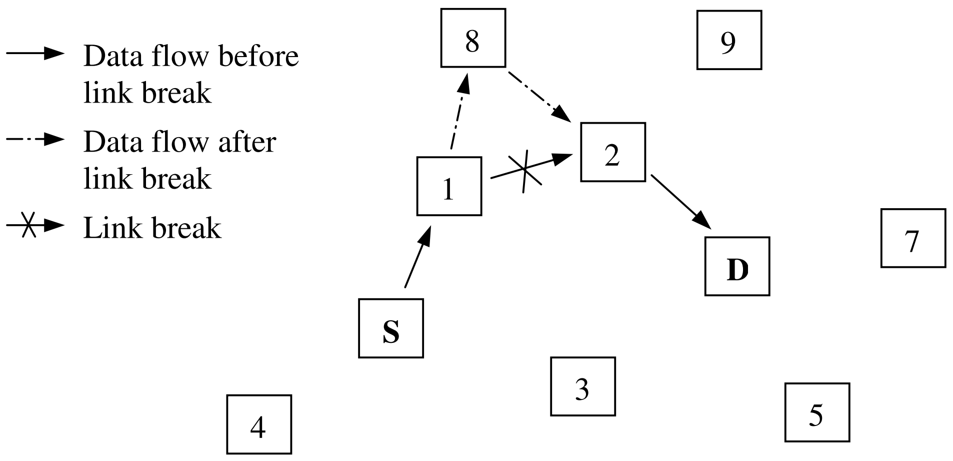


Figure 3.14(b) Illustration of AODV-RR Route Maintenance (Link 1→2 Breaks)

As shown in **Figure 3.14(a)**, when link 3→**D** breaks, since node 3 does not have any alternative route to **D**, it broadcasts RERR to its neighbours. Upon receiving the RERR from node 3, **S** invalidates its primary route to **D** through node 3, and switches its traffic flow for **D** onto the next optimal valid alternative route to **D** through node 1. **S** does not further broadcast the RERR because the link break at 3→**D** does not render **D** unreachable to **S**. All other neighbouring nodes receiving the RERR but with no valid routes to **D** through node 3 will not further broadcast the RERR. This applies to **D** itself too.

As shown in **Figure 3.14(b)**, when link 1→2 breaks, node1 still has valid alternative route to **D** through node 8. So node 1 silently invalidates the broken route, and switches its traffic flow for **D** over onto the next shortest alternative route to **D** through node 8, without further broadcasting the RERR.

As shown in **Figure 3.14(c)**, when link 2→**D** breaks, node 2 has no alternative choice but to broadcast RERR to notify its upstream nodes 1 and node 8 about the unreachable destination **D**. Node 8 in turn invalidates its only route to **D** through node 2, and forwards the RERR to node 1. After receiving RERRs from both node 2 and node 8, node 1 invalidates all its routes to **D**, and thus further broadcasts the RERR to the source **S**. With this last alternative route to **D** through node 1 broken, **S** will be forced to initiate new route request for **D** if it needs to send more data to the destination **D**.

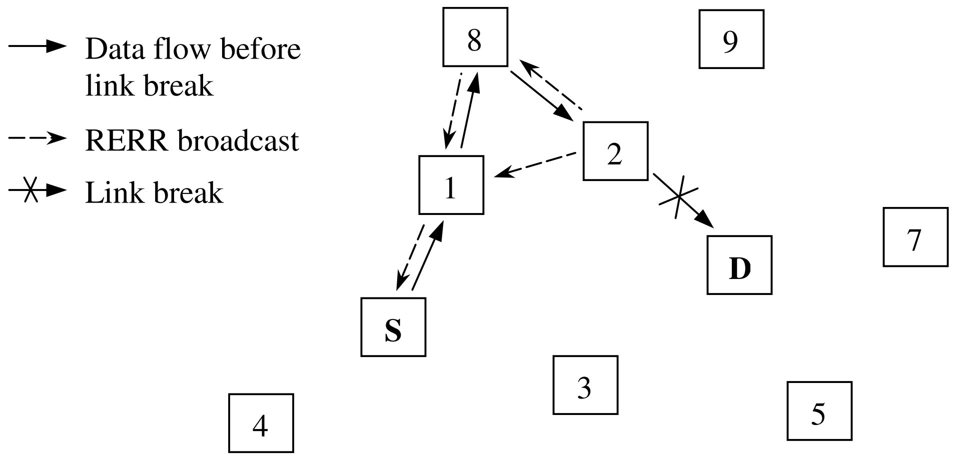


Figure 3.14(c) Illustration of AODV-RR Route Maintenance (Link 2→D Breaks)

In summary, the introduction of route redundancy provides immediate backup routes to salvage data flows at the point of link failure. In a distributed scheme like

AODV-RR, backup routes are built and selected automatically at every intermediate node without the supervision of the source. These key characteristics help greatly reduce the need for packets waiting in buffer and the likelihood of packet being dropped, hence resulting in smaller end-to-end delay and larger data delivery ratio.

3.4.3 On Receipt of RREQ

The flowchart in **Figure 3.15** outlines the handling of an incoming RREQ in AODV-RR. Using the same format of RREQ message and the same RREQ broadcast mechanism as in AODV, the processing of an incoming RREQ in AODV-RR is basically similar. However, modifications are made for handling different categories of routes (primary/alternative) and their status (Acked/UnAcked), as well as the different kinds of RREP response being triggered, in the context of redundant routes.

Only non-source nodes accept RREQ and each node accepts RREQ only once. It is to be noted that, unlike the use of multiple acknowledgements in creating multiple forward routes, the creation of reverse routes solely relies on the controlled broadcast of RREQ. Therefore each node creates only one primary shortest reverse route to the source. The node marks the newly created reverse route as unacknowledged to prepare for incoming RREPs in the future. Before proceeding, the node must delete any stale routes with older sequence numbers to the source from its routing table; because although no redundancy is built for reverse routes, there may exist redundant forward routes from this node to the source that become out-of-date.

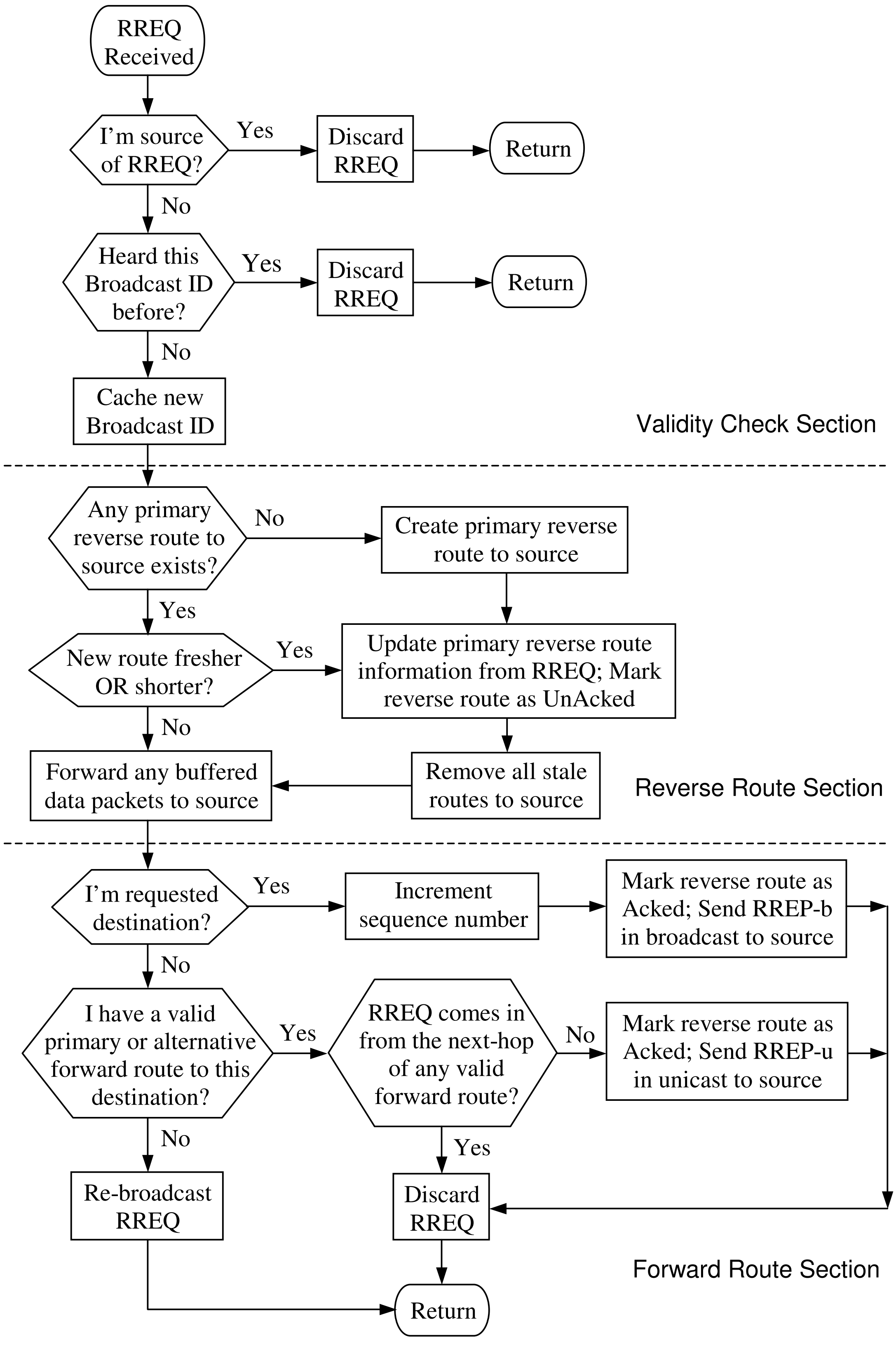

RREQ Received
I'm source of RREQ?
Yes
Discard RREQ
Return
No
Heard this Broadcast ID before?
Yes
Discard RREQ
Return
No
Cache new Broadcast ID
Validity Check Section
Any primary reverse route to source exists?
No
Create primary reverse route to source
Yes
New route fresher OR shorter?
Yes
Update primary reverse route information from RREQ; Mark reverse route as UnAcked
No
Forward any buffered data packets to source
Remove all stale routes to source
Reverse Route Section
I'm requested destination?
Yes
Increment sequence number
Mark reverse route as Acked; Send RREP-b in broadcast to source
No
I have a valid primary or alternative forward route to this destination?
Yes
RREQ comes in from the next-hop of any valid forward route?
No
Mark reverse route as Acked; Send RREP-u in unicast to source
No
Yes
Re-broadcast RREQ
Discard RREQ
Return
Forward Route Section

Figure 3.15 AODV-RR RREQ Handling Routine

If this node is the requested destination, the reception of the RREQ triggers an RREP-b broadcast for building route redundancy. Otherwise, if this node has a valid primary or reverse forward route to the requested destination, it must ensure that the RREQ did not come in from the next-hop of any existing valid forward routes to the requested destination before it unicasts a RREP-u to acknowledge the reverse route, to prevent potential routing loops. In both cases, the previously created reverse route is marked as acknowledged. In addition, the RREQ broadcast stops here. A node further broadcasts the RREQ only if it is unable to acknowledge the route request.

3.4.4 On Receipt of RREP

In order to build route redundancy, AODV-RR uses two subtypes of RREP messages: RREP-b for broadcast and RREP-u for unicast. Despite their differences, RREP-b and RREP-u carry essentially the same information regarding the discovered destination (see **Table 3.4**). Therefore AODV-RR merges the processing of both incoming RREP-b and RREP-u into one single routine, as is outlined by the flowchart in **Figure 3.16**.

Since the incoming RREP may be either a RREP-u from unicast or a RREP-b from broadcast, the node must check in the case of an incoming RREP-b and ensure that it is not the destination that issued the RREP-b and that only the first copy of RREP-b is accepted. Depending on the incoming RREP and existing routing information, the node creates an alternative forward route to the requested destination using the RREP received only if all of the following four conditions are met:

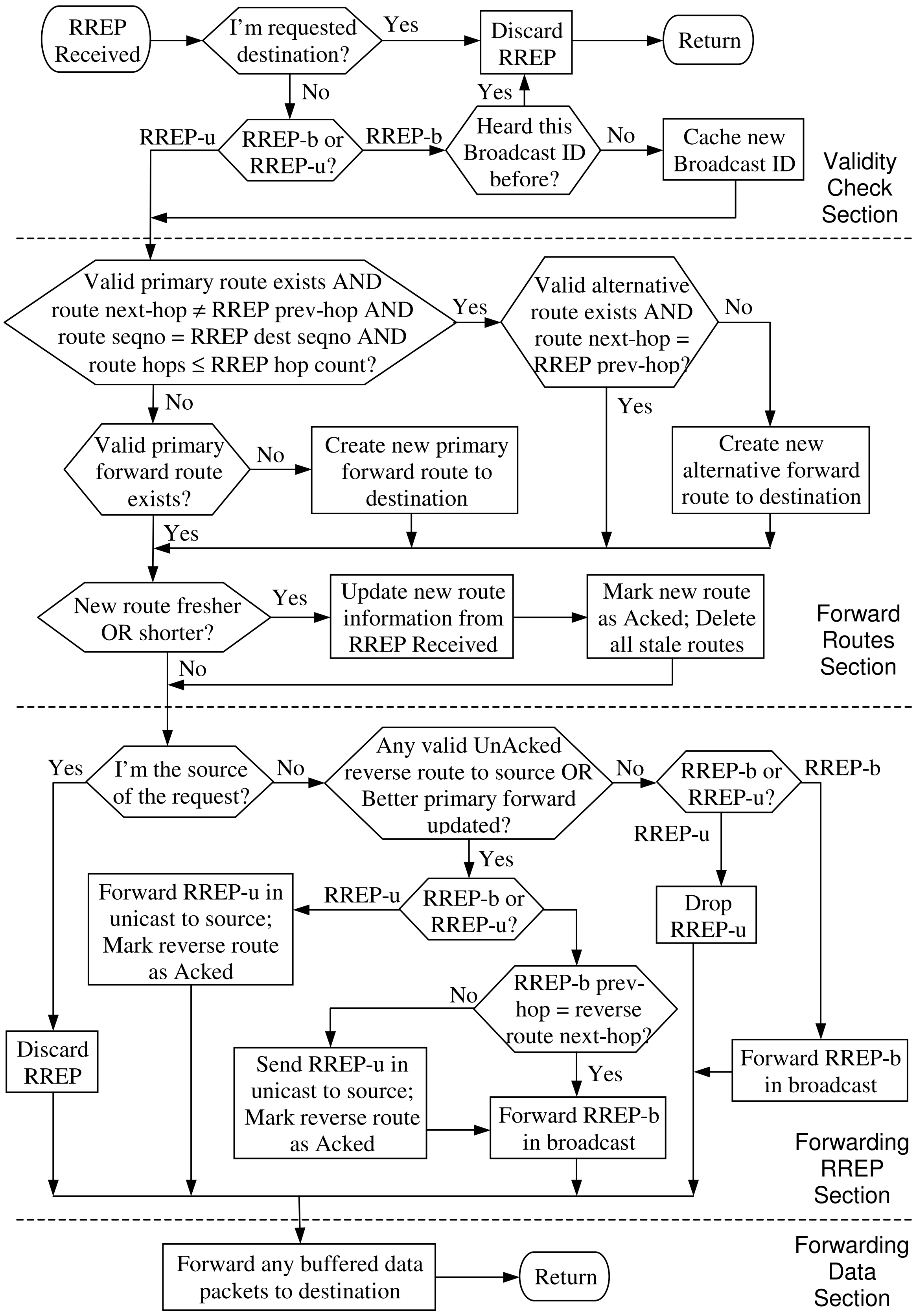

RREP Received
I'm requested destination?
Yes
Discard RREP
Return
No
Yes
RREP-u
RREP-b or RREP-u?
RREP-b
Heard this Broadcast ID before?
No
Cache new Broadcast ID
Validity Check Section
Valid primary route exists AND route next-hop ≠ RREP prev-hop AND route seqno = RREP dest seqno AND route hops ≤ RREP hop count?
Yes
Valid alternative route exists AND route next-hop = RREP prev-hop?
No
No
Yes
Valid primary forward route exists?
No
Create new primary forward route to destination
Create new alternative forward route to destination
Yes
New route fresher OR shorter?
Yes
Update new route information from RREP Received
Mark new route as Acked; Delete all stale routes
No
Forward Routes Section
Yes
I'm the source of the request?
No
Any valid UnAcked reverse route to source OR Better primary forward updated?
No
RREP-b or RREP-u?
RREP-b
RREP-u
Yes
Forward RREP-u in unicast to source; Mark reverse route as Acked
RREP-u
RREP-b or RREP-u?
Drop RREP-u
No
RREP-b prev-hop = reverse route next-hop?
Discard RREP
Send RREP-u in unicast to source; Mark reverse route as Acked
Yes
Forward RREP-b in broadcast
Forward RREP-b in broadcast
Forwarding RREP Section
Forward any buffered data packets to destination
Return
Forwarding Data Section

Figure 3.16 AODV-RR RREP Handling Routine

- There exists a valid forward route to the requested destination; AND
- This primary route has different next-hop as the RREP previous-hop; AND
- This primary route has same destination sequence number as the RREP; AND
- This primary route has smaller or equal hop count than the RREP.

These conditions ensure that all alternative routes are distinguished by different next-hops but carry the same destination sequence number and are equal or longer than the primary route. Once the node decides which category of routes (primary or alternative) to create, the update of routing information for that particular route is based on the same fresher and shorter route preference rule in AODV. This newly created forward route is then marked as acknowledged. The node must remove all stale existing routes in its table to ensure consistency among all its multiple forward routes to the same destination.

Only non-source nodes need to further forward the RREP. An incoming RREP-b message is always forwarded for further broadcast as long as it has nonzero TTL. In addition, the RREP-b triggers the unicast of an RREP-u message back to the source under either of the following two conditions:

- There exists a valid unacknowledged reverse route; OR
- A fresher or better primary forward route has been created previously; the reason being, the creation of a new primary forward route typically implies a change in topology downstream. Hence it is necessary to notify upstream nodes through RREP-u regardless of whether or not the reverse route is acknowledged.

Apart from the above two conditions, there is one more exception when deciding whether the RREP-b should trigger an RREP-u unicast. If the RREP-b comes in from the next-hop of this reverse route, the node must not acknowledge the reverse route with an RREP-u, in order to prevent potential routing loops. On the other hand, an incoming RREP-u message is always forwarded back to the source along a valid reverse route if either of the above two same conditions is met, in other words, either if there exists an unacknowledged reverse route to the source, or if a new primary forward route has been created previously. If neither of these two conditions holds, the RREP-u is dropped.

3.4.5 On Receipt of RERR

Figure 3.17 shows the handling of an incoming RERR message. In AODV-RR, a destination becomes unreachable only if all its primary and alternative routes in the route group are broken. Therefore the reception of a RERR from a neighbour merely invalidates the route to the destination through this neighbour as next-hop. It does not trigger more RERRs being sent to upstream neighbours, unless there is no other valid alternative route available to the destination. Hence upon receiving a RERR message, the node locates and brings down the route to the RERR- listed destination through the neighbour from which the RERR comes in. It then searches for any valid shortest newest alternative route to the listed destination. If such alternative is found, the node redirects any queuing data packets onto this new alternative route. Otherwise, if no alternative choice is available, the node proceeds as normal to inform upstream nodes with a new RERR broadcast and salvage any buffered data.

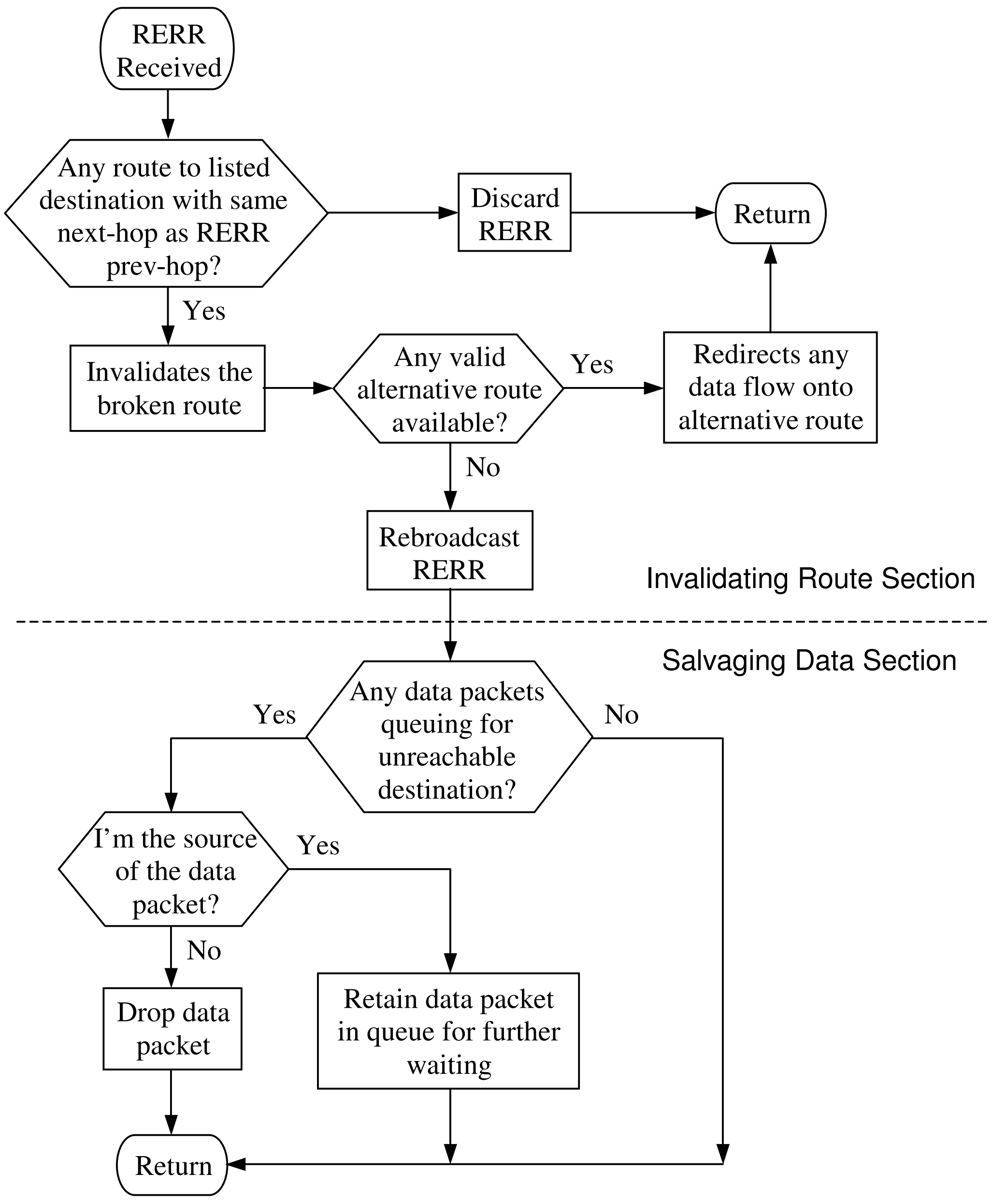

RERR Received
Any route to listed destination with same next-hop as RERR prev-hop?
Discard RERR
Return
Yes
Invalidates the broken route
Any valid alternative route available?
Yes
Redirects any data flow onto alternative route
No
Rebroadcast RERR
Invalidating Route Section
Salvaging Data Section
Any data packets queuing for unreachable destination?
Yes
No
I'm the source of the data packet?
Yes
No
Drop data packet
Retain data packet in queue for further waiting
Return

Figure 3.17 AODV-RR RERR Handling Routine

3.4.6 Route Resolution

Figure 3.18 shows how data packets are serviced in AODV-RR. With route redundancy, more routes are available for selection; hence a smaller chance of data packets drops. The node usually starts with the primary shortest route. If the primary route fails, the node automatically tries the next shortest newest alternative available. Such attempts are repeated until all routes in the table are broken.

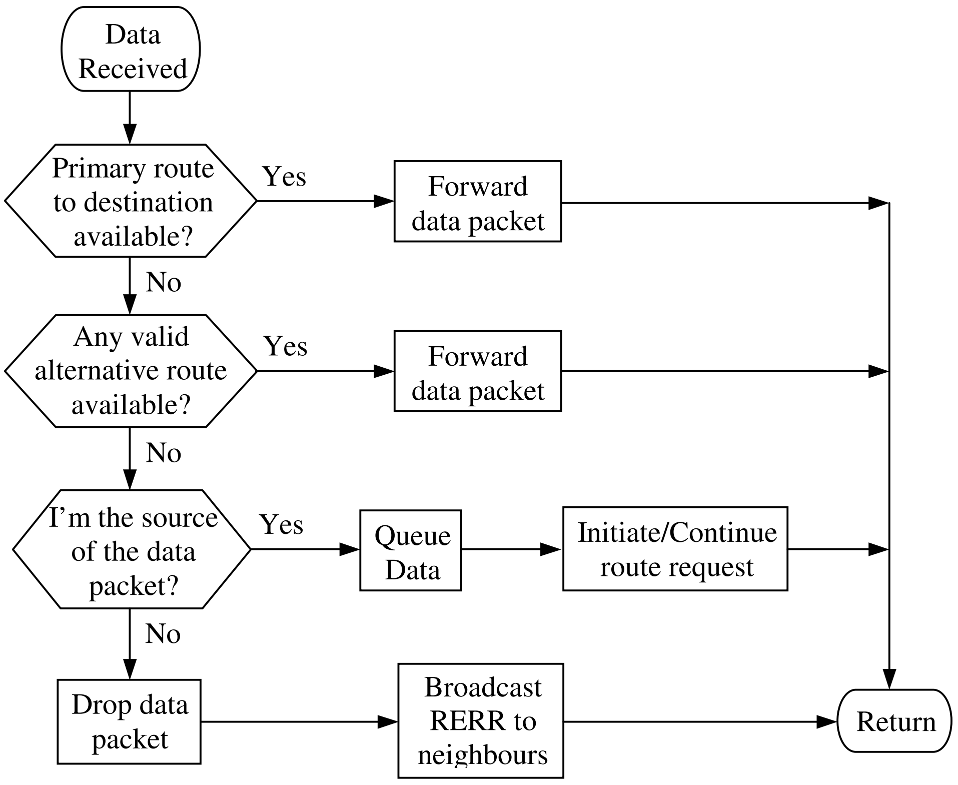

Data Received
Primary route to destination available?
Yes
Forward data packet
No
Any valid alternative route available?
Yes
Forward data packet
No
I'm the source of the data packet?
Yes
Queue Data
Initiate/Continue route request
No
Drop data packet
Broadcast RERR to neighbours
Return

Figure 3.18 AODV-RR Route Resolution For Data Packets

3.4.7 Link Breakage Control

Figure 3.19 shows the protocol reaction to a route breakage. When AODV-RR detects a link failure from MAC layer feedback, it brings down not only the route under the unsuccessful attempt, but also all existing routes through this broken link using the unreachable neighbour as their next-hops. To determine whether the link failure has caused the destination of the unsuccessful route unreachable, the node searches for any valid shortest newest alternative route to that destination. If such alternative route is found, the data flow is switched. Otherwise, if no alternative choice is available, the node notifies upstream nodes about the connection breakage with the destination and salvage data packets, if any.

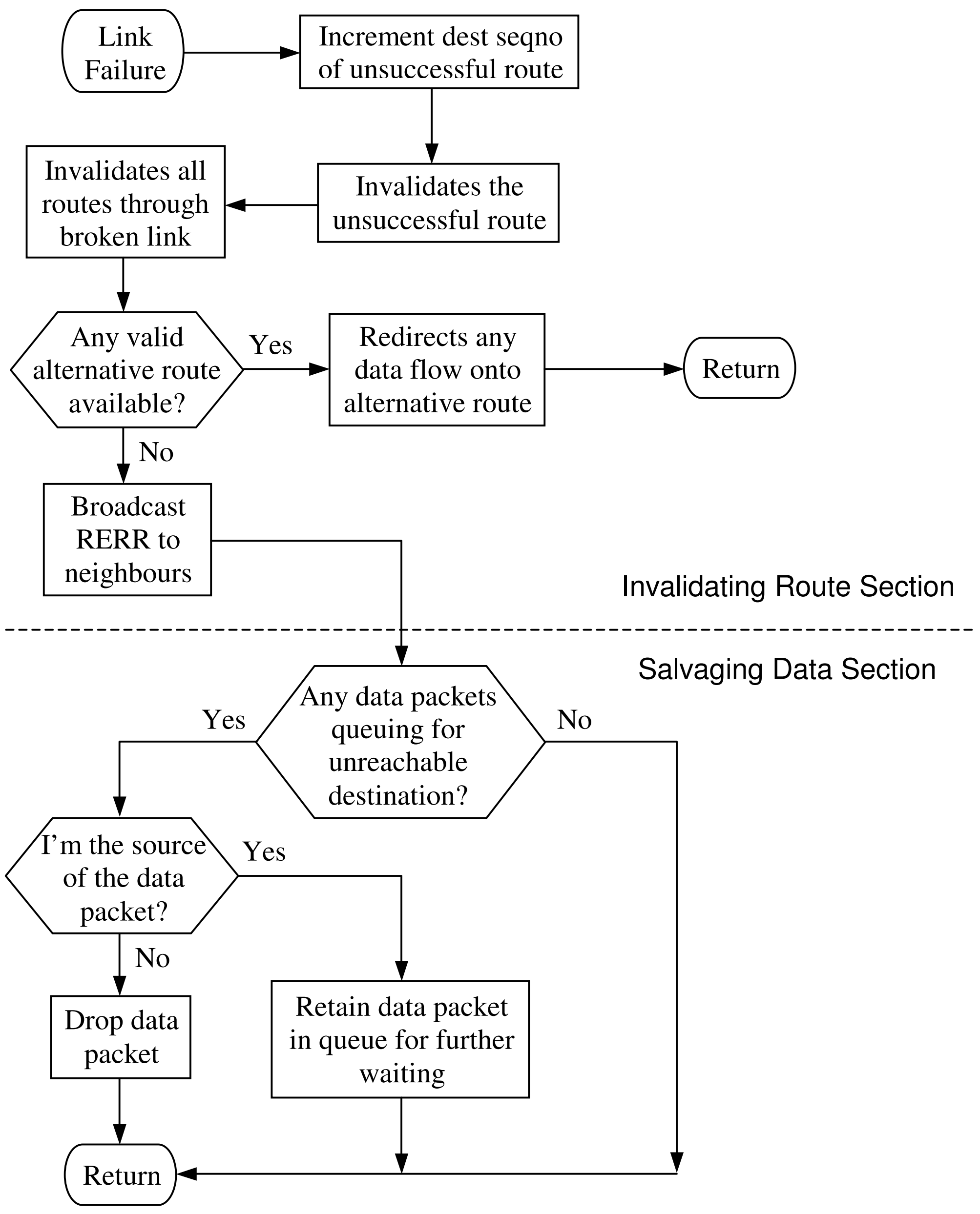

Link Failure
Increment dest seqno of unsuccessful route
Invalidates the unsuccessful route
Invalidates all routes through broken link
Any valid alternative route available?
Yes
Redirects any data flow onto alternative route
Return
No
Broadcast RERR to neighbours
Invalidating Route Section
Salvaging Data Section
Any data packets queuing for unreachable destination?
Yes
No
I'm the source of the data packet?
Yes
No
Drop data packet
Retain data packet in queue for further waiting
Return

Figure 3.19 AODV-RR Route Breakage Control

3.5 Simulation Results

To measure the impact of the route redundancy mechanism, we evaluate the performance of AODV-RR through simulations in *ns*-2 and compare with that of AODV. **Table 3.5** lists the simulation settings, which are similar to those used in part one of the project. We vary the traffic load level with two different values of packet size: 64/512 bytes, but fixed the number of connections at 20. We examine the protocol performance in terms of not only packet delivery ratio and routing overhead, but also the end-to-end delay, delay jitter and route optimality.

Table 3.5 Summary of Simulation Settings (Part II)

Parameter	**Setting**
Number of Mobile Nodes	50
Movement Area	1500 m × 300 m Flat Rectangular
Total Simulated Time	900 s
Movement Model	Random Way-point
Mobile Node Pause Time	0, 30, 60, 120, 300, 600, 900 s
Maximum Mobile Speed	20 m/s
Traffic Source	Constant Bit Rate (CBR)
Number of Connections	20
Packet Size	64, 512 bytes
Packet Sending Rate	4 packets/s
Connection Start Time	Uniformly Random between 0 ~ 180 s
Medium Access Control	IEEE 802.11 Standard MAC with DCF
Address Resolution Protocol	Approximation of BSD Unix Implementation
Shared Media Interface	Approximation of Lucent WaveLan DSSS Radio
Mobile Node Antenna	Unity-gain, Omni-directional, 1.5m above Ground
Propagation Model	Free-space (near) + Two-ray Ground Reflection (far)
Routing Protocol	AODV-LL, AODV-RR
Simulator Tool	Network Simulator – 2 (*ns*-2) version allinone-2.1b7
Platform	Redhat Linux 7.0 [27]

3.5.1 Packet Delivery Ratio

Figure 3.20 and **Figure 3.21** give the Packet Delivery Ratio (PDR) performance of AODV-RR and AODV with respect to mobility under different traffic load levels (64-byte and 512-byte respectively). As expected, in general more packets are successfully delivered at lower mobility (larger pause time).

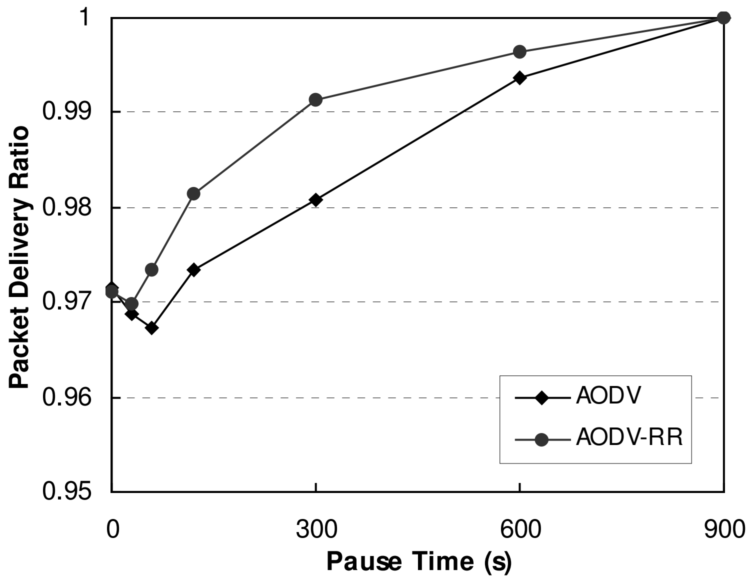


Figure 3.20 PDR Performance of AODV-RR and AODV (64-byte packets)

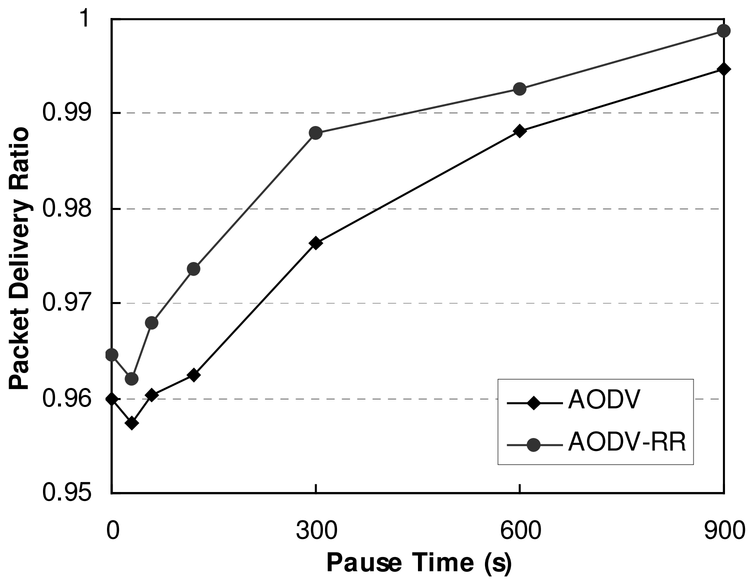


Figure 3.21 PDR Performance of AODV-RR and AODV (512-byte packets)

It is evident from the above results that AODV-RR performs better in PDR than AODV, and the difference is more significant with larger traffic load. With route redundancy incorporated, AODV-RR can supply backup routes to salvage data packets immediately at the point of link failures, hence greatly reducing the chances of data packets being dropped due to unavailability of routes. With larger data packet size, the transmission time for each packet increases. Topology changes are more likely to occur within the transmission time of a packet, increasing the chances of delivery failure. In this case, the presence of redundant routes helps to a larger extent to ensure proper delivery of data packets, hence achieving larger performance gain when the data packet size is larger.

In addition, we also observe that for both traffic load levels, the performance gain of AODV-RR compared to AODV is most significant in the medium mobility range while less when mobility is high or low. This is because at high mobility, the breakage of one route probably accompanies the breakages of many other alternative routes in the vicinity too. The slow on-demand link failure detection causes nodes to blindly try alternative routes that are actually down, which degrades the benefit from route redundancy at high mobility. On the other hand, when all nodes move slower, less topology changes occur and packets are dropped less frequently. Therefore redundant routes seldom become necessary in this case, causing smaller performance gain too.

3.5.2 Routing Overhead

Figure 3.22 and **Figure 3.23** compare the routing overhead performance of AODV-RR and AODV with respect to mobility. As expected from the on-demand nature of both protocols, in general the routing overhead of both AODV-RR and AODV increases positively with mobility (decreasing pause time).

It is interesting to observe that AODV-RR actually incurs less routing overhead than AODV does in all mobility and traffic load cases, despite the fact that AODV-RR relies on RREP broadcast to establish route redundancy.

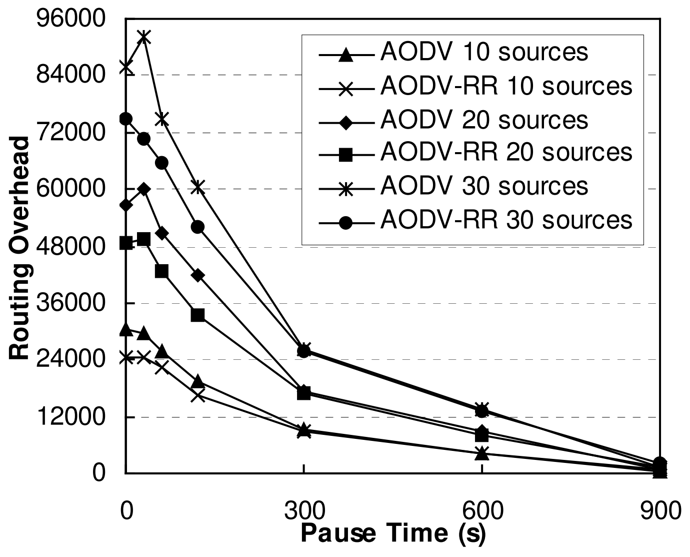


Figure 3.22 Overhead Performance of AODV-RR and AODV (64-byte packets, 10/20/30 sources)

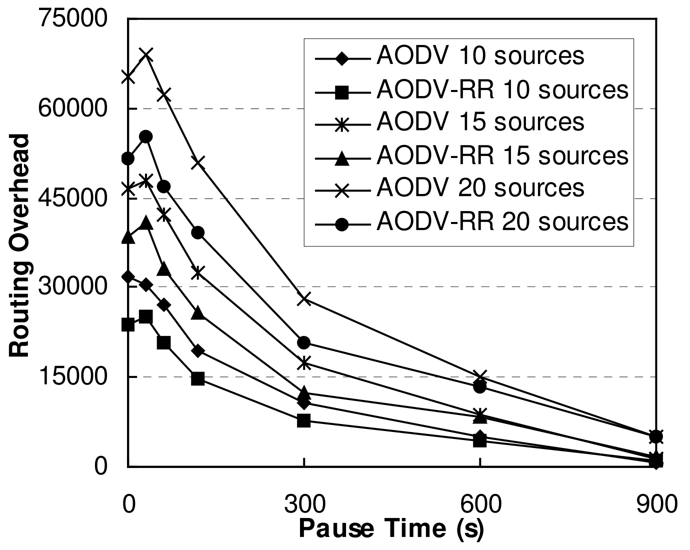


Figure 3.23 Overhead Performance of AODV-RR and AODV (512-byte packets, 10/15/20 sources)

In contrast to our expectation, AODV-RR achieves better PDR performance yet at the cost of smaller routing overhead than AODV in all cases. Despite the use of RREP-b broadcast and multiple RREP-u unicasts, AODV-RR works more efficiently in delivering more data packets than AODV. To investigate the reason behind this result, we analyse the composition of each protocol's routing overhead and compare the usage of different control packets, as shown in **Figures 3.24 ~ 3.27**. In the case of AODV-RR, we group RREP-b and RREP-u together and treat them both as belonging to the RREP category of control packets.

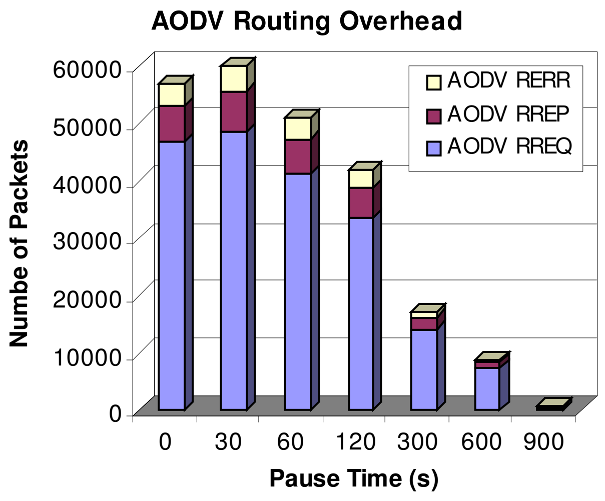


Figure 3.24 Routing Overhead Composition of AODV (64-byte packets, 20 sources)

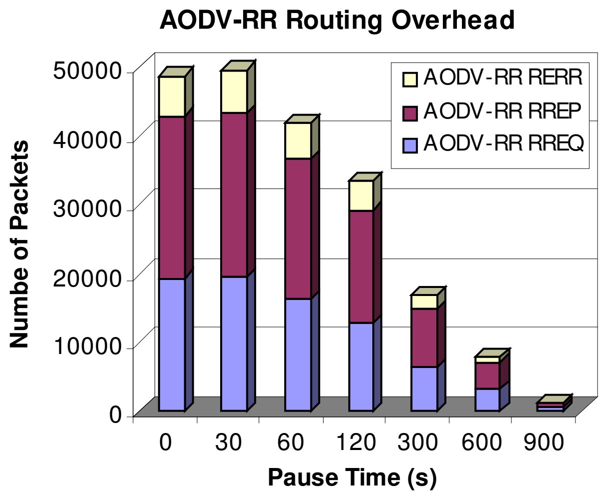


Figure 3.25 Routing Overhead Composition of AODV-RR (64-byte packets, 20 sources)

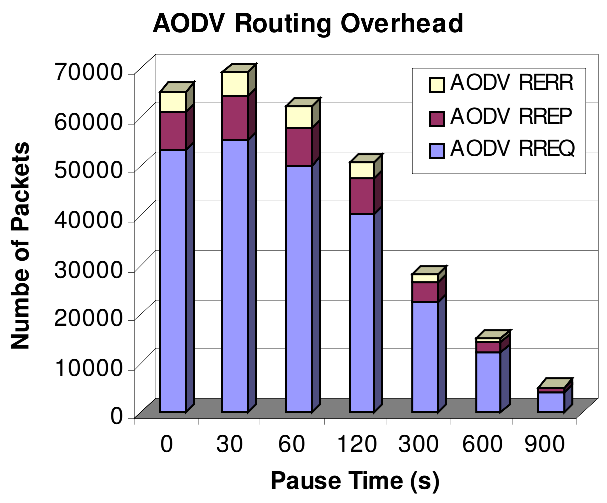


Figure 3.26 Routing Overhead Composition of AODV (512-byte packets, 20 sources)

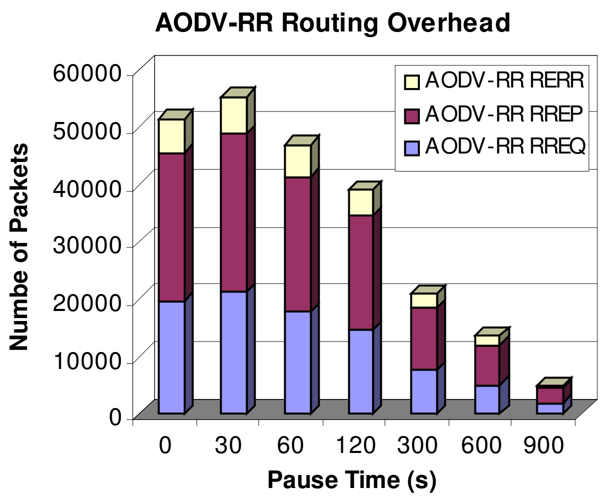


Figure 3.27 Routing Overhead Composition of AODV-RR (512-byte packets, 20 sources)

We can see that indeed AODV-RR uses more RREP messages in order to support its RREP-b broadcast and multiple RREP-u unicasts. Meanwhile, however, the number of RREQ messages in AODV-RR is much smaller than that in AODV, with a margin more than enough to offset the increase in the number of RREP messages in AODV-RR, resulting in a smaller gross amount of overhead than AODV.

The implication here is that, while requiring more RREP overhead in building route redundancy, AODV-RR effectively to a large extent reduces the frequent need for nodes to request for routes through broadcast of RREQ messages. In AODV, RREQ messages dominate and flood the network frequently, causing all nodes to create shortest reverse routes to the source. On the other hand, in AODV-RR, RREQ and RREP messages take almost equal share in the control overhead, allowing much better balance between the creation of reverse routes from surrounding nodes to the source, and forward routes from surrounding nodes to the destination.

In addition, we also observe that AODV-RR incurs slightly more RERR messages than AODV does. This is the result of two forces. Firstly, the availability of alternative routes in AODV-RR reduces the frequency of sending RERR messages. Moreover, replacing RERR unicasts to precursor list members in AODV with 1-hop RERR broadcast in AODV-RR further reduces the number of RERRs sent per node. Secondly, however, with richness in route information, one link failure typically affects many nearby nodes and forces them to do route breakage control. While AODV sends RERRs to upstream neighbours who have recently used this broken route, AODV-RR informs all upstream neighbours who have this broken link as next-hop in their routing tables. This actually increases the scope of RERR propagation. Hence the result on RERR reflects the combined effect of the above two factors.

3.5.3 End-to-End Delay

Figure 3.28 and **Figure 3.29** reveal the average end-to-end delay experienced by data packets in AODV and AODV-RR respectively. End-to-end delay is the time duration from the instant when a data packet is generated and passed down to the route layer at the source to the instant when this packet arrives at the routing layer at the destination. Delay is largely determined by route length along which packets propagate, plus queuing delay at the source and intermediate nodes.

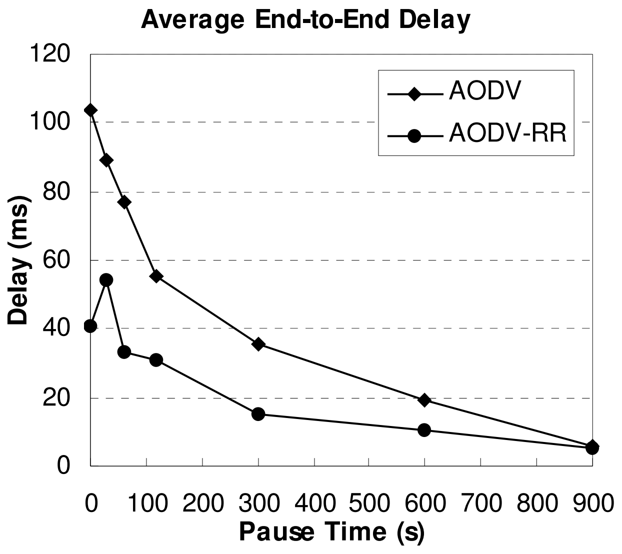


Figure 3.28 Delay Performance of AODV and AODV-RR (64-byte packets)

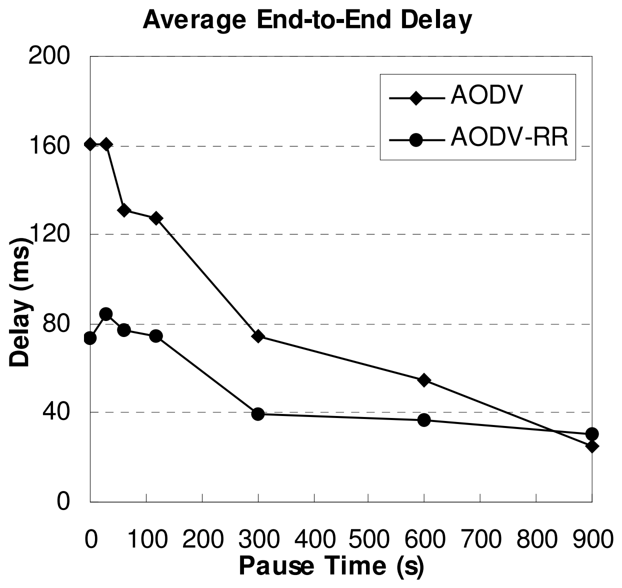


Figure 3.29 Delay Performance of AODV and AODV-RR (512-byte packets)

In general, data packets on the average experience larger delay at higher mobility. When nodes move faster, route information becomes out-of-date easily due to frequent topology changes. More delay is incurred for source nodes to request for fresh routes and for data packets to wait or try longer sub-optimal routes.

It is clear that AODV-RR produces much smaller end-to-end delay than AODV, and the improvement becomes more significant at higher mobility and larger traffic load. By supplying immediate backup routes to carry over data flow at the point of link failure, AODV-RR minimizes delay incurred in data packets waiting for the route to be recovered. While the unique forward route in AODV may break easily, the subsequent multiple RREP acknowledgements allows AODV-RR to adapt to changing topology state over a longer time frame. In terms of hop counts, some of these alternative routes are equally optimal as the primary route. Therefore, the propagation delay along such alternative routes is nearly equal to that along the broken primary route. In summary, the use of redundant routes in AODV-RR minimizes queuing delay without significantly enlarging propagation delay.

3.5.4 Delay Jitter

Other than the metric End-to-end delay, for multimedia traffic such as video streaming and internet radio, the variance of delay or jitter is also a large concern. Delay jitter measures the extent to which the end-to-end delay varies across different data packets. We compute the delay jitter as the standard deviation of the end-to-end delay across all data packets, and compare the results for AODV and AODV-RR in **Figures 3.30~3.31**.

In both protocols, smaller delay jitter is incurred when mobility is lower. When there are less frequent topology changes, route information stays valid for longer, and less often there are needs for packets to stop and wait at intermediate nodes for route repairs that cause extra delays.

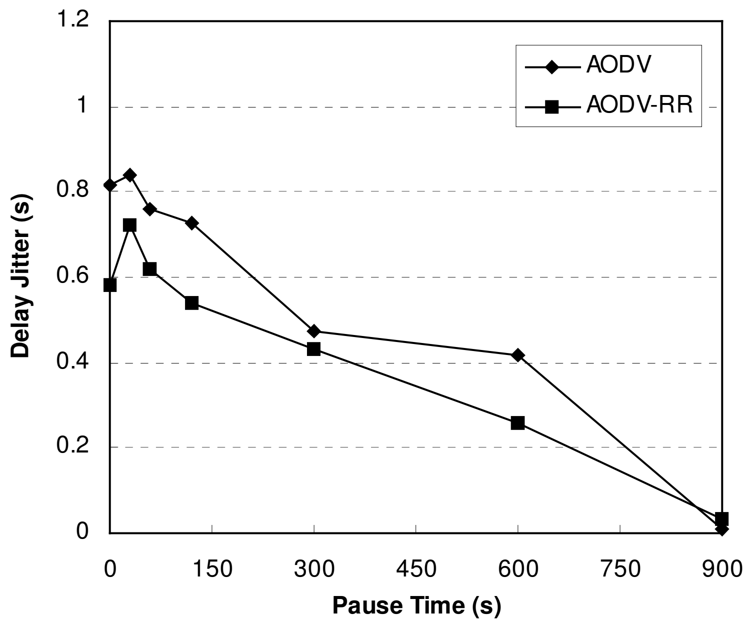


Figure 3.30 Delay Jitter Performance Comparison (64-byte packets, 20 connections)

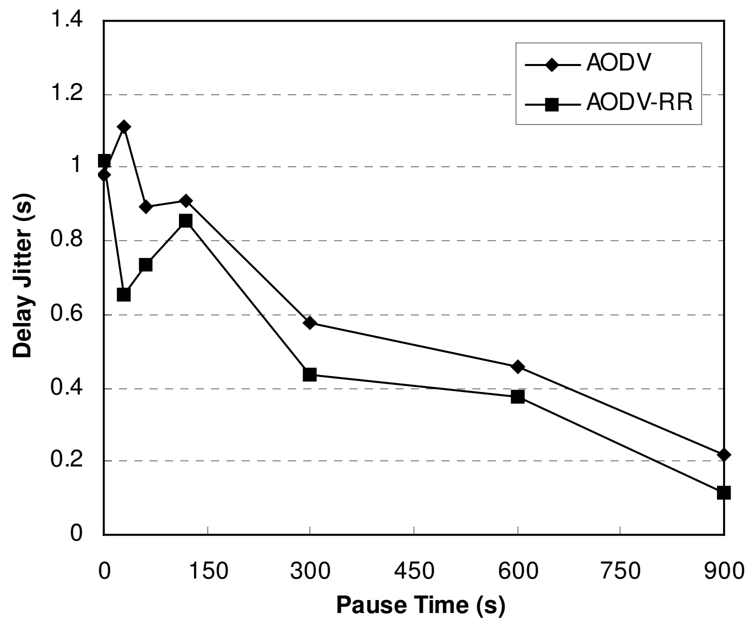


Figure 3.31 Delay Jitter Performance Comparison (512-byte packets, 20 connections)

The graphs show that AODV-RR has some slight improvement over AODV in terms of delay jitter. As we have analysed in **Section 2.4.4**, the route repair mechanism in AODV is highly distributive and localized among a small group of nodes near the broken link. This results in the very efficient route repair mechanism in AODV. AODV-RR inherits the same distributiveness and localization in its route repair mechanism from that of AODV. And on top of that, AODV-RR exploits route redundancy to further speed up the provision of alternate routes when links break. When in AODV a packet needs to stop at an intermediate node and wait for a new route to be created, it may simply be switched to an alternate route in AODV-RR at no extra delay. Essentially, the presence of route redundancy in AODV-RR reduces the chances for nodes to create new routes, hence lowering not only the average end-to-end delay, but also the jitter of such delay, as compared to AODV.

We can also observe that at higher mobility with pause times below 120s, both AODV and AODV-RR see some fluctuations in their delay jitter curves, and AODV-RR seems to have bigger fluctuations than AODV. This implies that the delay jitter of AODV-RR is more sensitive to increase in mobility as compared to that of AODV. As we know, at high mobility, route information gets obsolete faster. When mobility grows high enough to make multiple routes in the group fail, AODV-RR still needs to patiently verify every redundant route one by one until there is no choice left. The more redundant routes AODV-RR keeps, the more extra delay is potentially wasted. And such potential is largely dependant on mobility that causes the aging of routes. Therefore, high mobility has a greater negative impact on delay jitter performance of AODV-RR than it does on that of AODV.

3.5.5 Route Optimality

We also examine the optimality of routes created by both protocols. **Figure 3.32** and **Figure 3.33** report the distribution of extra hops in actual routes compared to shortest optimal routes. An extra hop of zero indicates that the actual route used is equally optimal as the shortest route. Smaller extra hop means more optimality in actual routes.

The results show no significant difference in route optimality between AODV and AODV-RR. Most of the shortest alternative routes used in AODV-RR are equally optimal in terms of hop count as the primary route, which minimizes queuing delay while causing no significant increase in propagation delay.

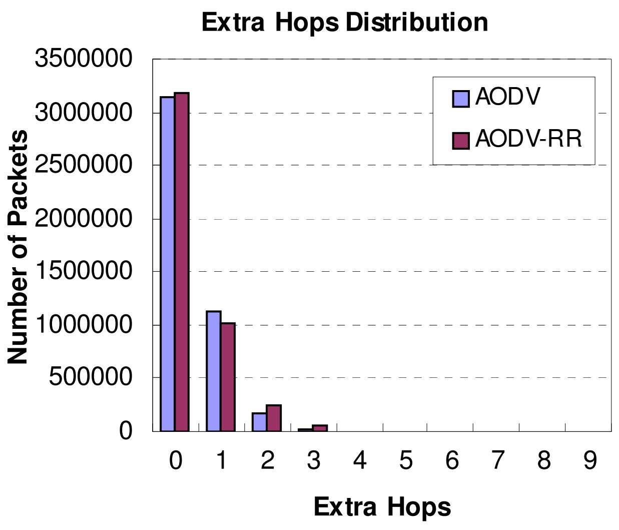


Figure 3.32 Route Optimality of AODV and AODV-RR (64-byte packets)

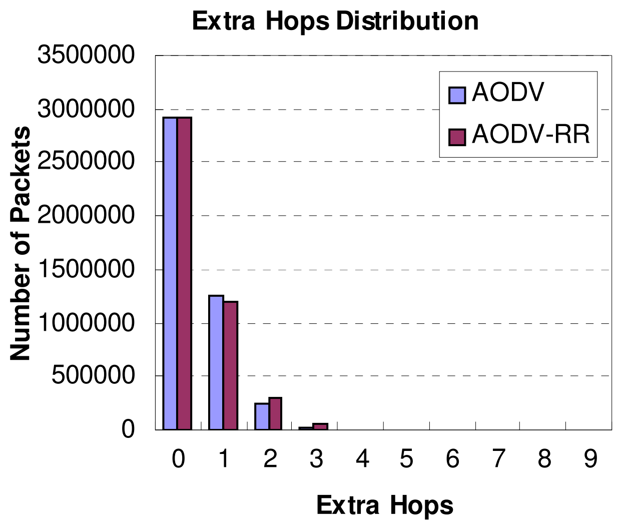


Figure 3.33 Route Optimality of AODV and AODV-RR (512-byte packets)

3.5.6 Storage and Computational Complexity

Essentially, AODV-RR improves robustness by exploring redundant routes knowledge, additional to the unique shortest path information in AODV. Through the same AODV route request process, AODV-RR nodes collect and maintain richer routes information, which requires more storage and processing capacity as compared to AODV nodes.

The increased storage requirement for AODV-RR nodes is obvious. For each given destination, the AODV-RR routing table in the source node keeps not only the single optimal shortest route as in AODV, but also as many the next shortest alternate routes as the route table can hold. In other words, the extent of such redundant route information stored by AODV-RR is constrained by either a prescribed capacity limit, or by the actual availability of all possible alternate routes, whichever is smaller. The extent of redundant route information directly determines how much more storage is needed in AODV-RR nodes. Therefore we deduce the following equation relating storage space consumptions in AODV-RR and AODV.

Storage in AODV-RR = N × Storage in AODV ; where, **<3.1>**

N = Smaller of {Prescribed maximum number of routes kept per destination, Average number of neighbours around each node in the network}.

To compare the computational complexity of AODV and AODV-RR, we break down the routing algorithms into three phases: route discovery, route maintenance and route resolution. Firstly, the route discovery phases in both schemes start with very similar route request processes that exploit the same controlled RREQ broadcasts from the source node to its neighbours (See **Figure 3.1** and **Figure 3.13**). The numbers of nodes receiving and forwarding the RREQs are the same in both schemes. During the subsequent route reply process, however, AODV only allows those nodes along the unique valid forward route to unicast an RREP back to the source; whereas AODV-RR allows all nodes potentially linking the destination to the source to return an RREP either through controlled broadcast or unicasts. The number of nodes sending or forwarding RREP messages in AODV-RR is equal to the number of nodes within the overlapped span region of both the RREQ broadcast tree from the source, and the RREP broadcast tree from the destination. If we suppose nodes are evenly spaced, and the destination is **d hops** away from the source, then only **d** nodes will forward the RREP during AODV route reply process, as compared to $\mathbf{(d+1)^2-1}$ nodes in AODV-RR case. **Figure 3.34** and **Figure 3.35** illustrate the above estimation. The average value of **d** equals to half of the network diameter.

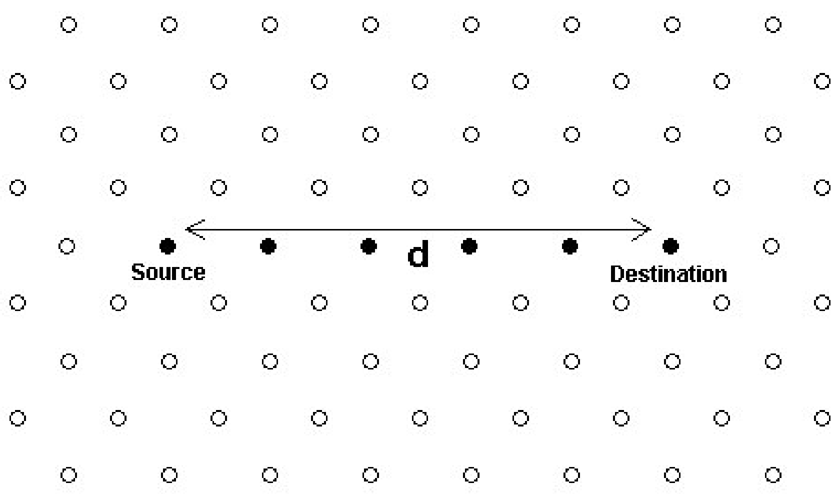


Figure 3.34 Single RREP Return Path in AODV

Secondly, during the route maintenance phase in both AODV and AODV-RR, the node at the point of failure link simply sends RERR to notify all upstream nodes. The difference is, in AODV, the number of affected upstream nodes is much smaller, on the average of **d/2**, assuming the point of link failure distributes uniformly along the unique forward route (See **Figure 3.34**). In AODV-RR, however, due to route redundancy, the number of upstream nodes that need to be notified and engaged in receiving and forwarding the RERRs can average to **$[(d+1)^2-1]/2$**, assuming a uniform distribution of the point of link failure within the route mesh shown in **Figure 3.35**.

Lastly, during route resolution, AODV simply picks the unique forward route when one is needed. If this route fails, AODV simply proceeds to route creation or maintenance. In AODV-RR, nodes will continue to try the next optimum alternate route until there is no more valid choice. The more alternate routes AODV-RR stores, the more routes it may end up trying before considering the destination to be totally out of reach. Therefore, route redundancy actually requires AODV-RR spending more effort in validating these multiple routes for each destination. If we assume fully random movement of mobile nodes and equal chance for each route to fail, then the average number of routes that AODV-RR needs to test before it hits a valid route roughly equals **N/2**, where **N** is again the number route entries stored for each destination, or the maximum route cache size for each destination, whichever is smaller. In the case of AODV, **N** always equals to 1.

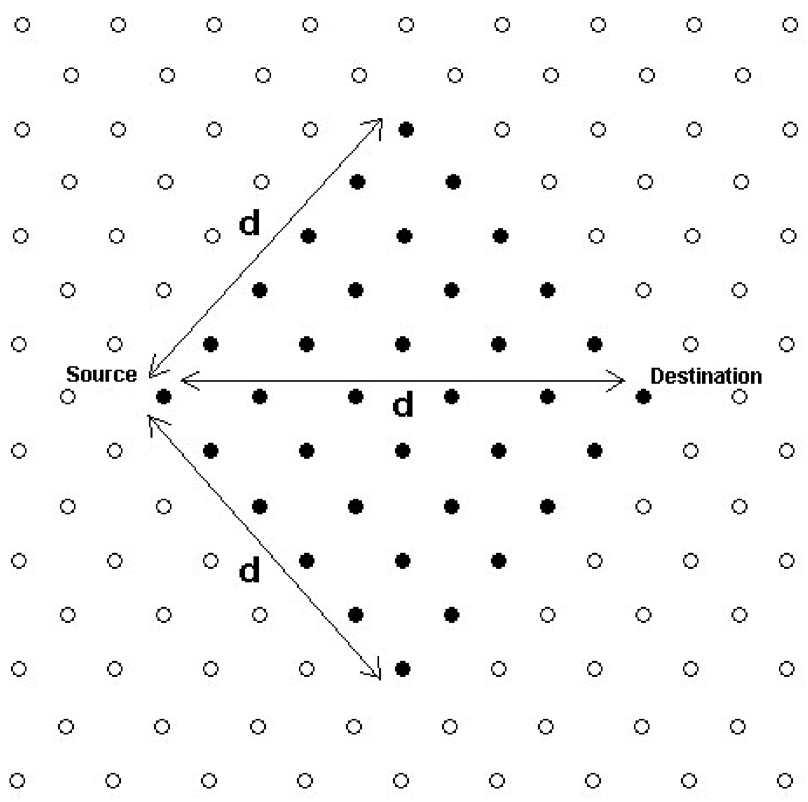


Figure 3.35 Multiple RREP Return Paths in AODV-RR

Summarizing the above analysis, if we look across the network from the macroscopic view, the computational overhead in AODV-RR will be **O[(d+1)^2-1]**, as compared to the **O[d]** in AODV, where **O[]** denotes the order of, and **d** averages to be half of the network diameter **D**. On the other hand, if we examine every node from the microscopic view, then the computation overhead in AODV-RR will be **N** times of that in AODV, where **N** is the smaller of the redundant routes available and allowed for each destination as defined in **Equation 3.1**.

3.5.7 Summary

In summary, the introduction of route redundancy achieves better performance in terms of higher PDR and more importantly smaller end-to-end delay, while costing less total control overhead with no significant loss in route optimality. The penalty for these advantages is less stable delay jitter when adapting to high mobility, and increased complexity in routing algorithm and storage requirement for mobile nodes.

3.6 Conclusion

In this chapter, we have explored the idea of route redundancy to improve the robustness of MANET routing protocols, especially in the delay aspect for time-critical traffic. The rational behind this approach is to build multiple routes for each source/destination pair so that immediate backup routes can be supplied at the point of link failure to salvage time-critical traffic at minimum delay. We experiment with different approaches of building redundant routes, and develop a new scheme AODV-RR with systematic algorithms to establish, select and maintain such redundant routes. We evaluate the performance of the new scheme in comparison with AODV. It is evident from the results that, with route redundancy incorporated, AODV-RR requires less control overhead to successfully deliver more data packets within shorter end-to-end delays. Albeit slightly less stable delay jitter at high mobility, the reduced end-to-end delay in AODV-RR is, still an important advantage for time-critical traffic such as voice and images over MANET. Overall, route redundancy is able to improve the efficiency and robustness of MANET routing protocols. The penalty for such improvements is increased processing complexity and storage requirement.

CHAPTER FOUR
ADAPTIVE ROUTE TIMEOUT

This chapter investigates the impact on performance of making the expiry timeouts of routes adaptive to mobility. The background, motivation and objective for this part of project are first described. Some experiments are next carried out on the how adaptive route expiry timeout can influence routing performance. After recognising such effects, the chapter proceeds with more trials to find the optimal value of route expiry timeout for each mobility case. Lastly, the performance of such adaptive route timeouts is evaluated and its advantages concluded.

4.1 Background and Motivation

In original AODV, a newly created route is given a fixed expiry timeout. If this route has not been used to forward any packet over this timeout interval, the route information expires and is removed from the routing table. Otherwise, every time a packet gets forwarded along this route, the route is given another timeout interval over which it remains active from the current time instant onward. The purpose of introducing expiry timeouts for routes is to allow automatic deletion of stale routing information without requiring special management.

Despite different values assignable to them, these expiry timeouts remain fixed during the entire protocol operation (simulation), regardless of different mobility or traffic conditions. Route information expires naturally after the same timeout interval no matter how fast nodes move or how frequently topology changes. At high mobility, however, route information becomes out-of-date easily, and nodes need to refresh their routing table more often to keep consistency with the network topology. In this case, using a same route expiry timeout as in the low mobility case will cause nodes to falsely trust stale routing information until damage is actually incurred upon failing to deliver data packets on these routes later.

In our opinion, Route Expiry Timeout (RET) should relate to mobility. When nodes move faster, the route should expire faster too. A smaller expiry timeout at high mobility will force the stale routing information to be deleted more quickly so that nodes can discover fresh routes on a more proactive basis, instead of waiting for delivery failures passively. Based on this reason, we expect the adaptation of route expiry timeout to mobility would result in more up-to-date routing information in mobile nodes, hence better routing performance, but with potentially larger control overhead from the proactive route discoveries at high mobility.

This part of the project advances in following three stages. First, the effect of adaptive Route Expiry Timeout (RET) on routing performance is tested. With such impact confirmed, we next seek to find the optimal RET value for different mobility cases on an experimental basis. Lastly, with RET adapted to these optimal values, the performance of the routing protocol is evaluated and compared to that of the original with fixed RET, with the characteristics of this idea concluded.

We base our experiments on both the original AODV and AODV-RR; so that any accumulative benefits from adaptive RET on top of that from route redundancy can be observed. **Table 4.1** lists common simulation settings used in this part of our project; while different maximum mobile speed and RET values are tested in each of the three stages.

Table 4.1 Summary of Simulation Settings (Part III)

Parameter	**Setting**
Number of Mobile Nodes	50
Movement Area	1500 m × 300 m Flat Rectangular
Total Simulated Time	900 s
Movement Model	Random Way-point
Mobile Node Pause Time	0, 30, 60, 120, 300, 600, 900 s
Maximum Mobile Speed	1, 5, 20 m/s and varies
Traffic Source	Constant Bit Rate (CBR)
Number of Connections	20
Packet Size	64, 512 bytes
Packet Sending Rate	4 packets/s
Connection Start Time	Uniformly Random between 0 ~ 180 s
Medium Access Control	IEEE 802.11 Standard MAC with DCF
Address Resolution Protocol	Approximation of BSD Unix Implementation
Shared Media Interface	Approximation of Lucent WaveLan DSSS Radio
Mobile Node Antenna	Unity-gain, Omni-directional, 1.5m above Ground
Propagation Model	Free-space (near) + Two-ray Ground Reflection (far)
Routing Protocol	AODV, AODV-RR with and without Adaptive RET
Simulator Tool	Network Simulator – 2 (*ns*-2) version allinone-2.1b7
Platform	Redhat Linux 7.0 [27]

4.2 Effect of Adaptive RET

To investigate the impact of adaptive RET, we first compare the performance of different RET values applied on the same protocol, both AODV and AODV-RR. We also conduct statistical studies to gain further understanding on how different RETs affect the behaviours of the routing protocol.

4.2.1 Performance Results

We experiment with different values of RET on both AODV and AODV-RR. The results are given in **Figures 4.1 ~ 4.6**.

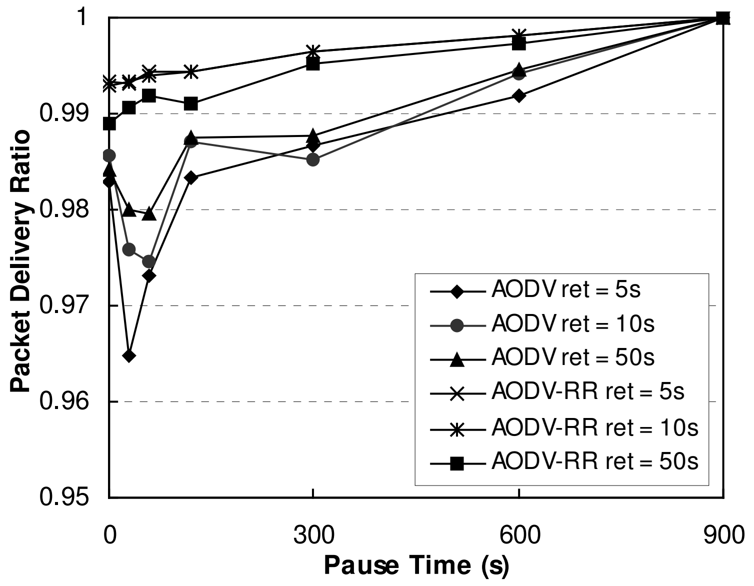


Figure 4.1 PDR Performance with Different RETs 64 byte/pkt, 4 pkt/s, 20 conn, max speed 5 m/s (ret = Route Expiry Timeout)

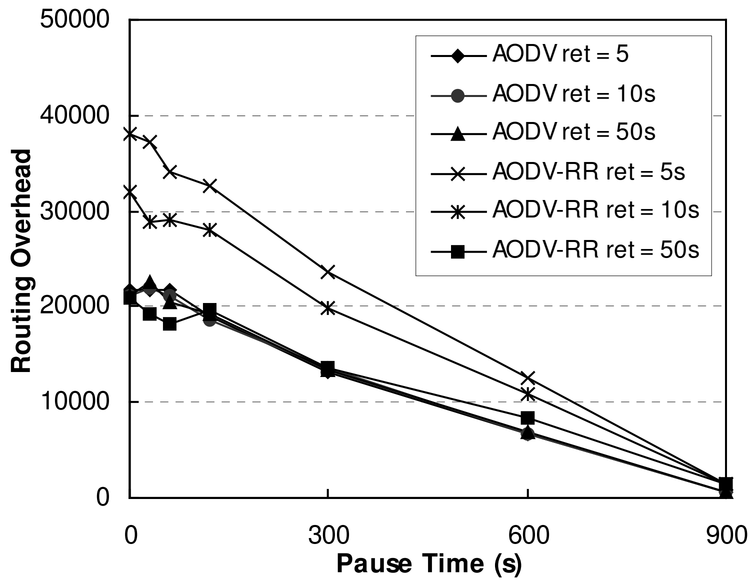


Figure 4.2 Overhead Performance with Different RETs 64 byte/pkt, 4 pkt/s, 20 conn, max speed 5 m/s (ret = Route Expiry Timeout)

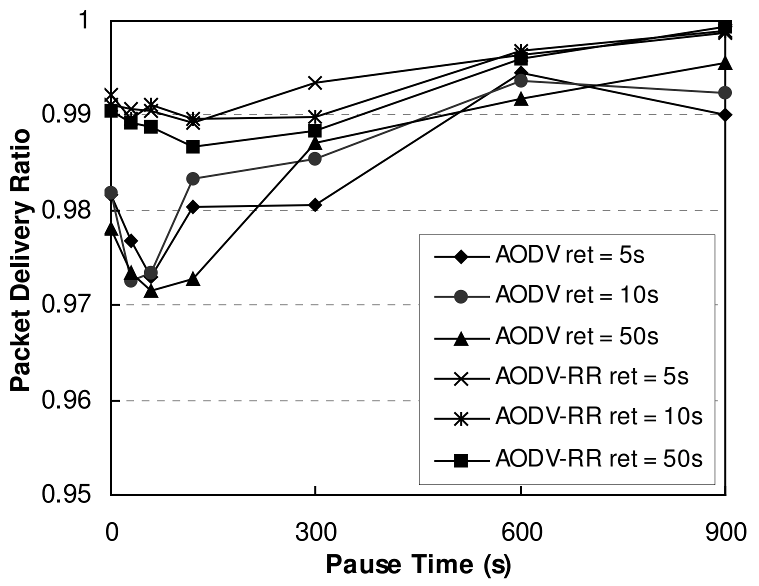


**Figure 4.3 PDR Performance with Different RETs
512 byte/pkt, 4 pkt/s, 20 conn, max speed 5 m/s
(ret = Route Expiry Timeout)**

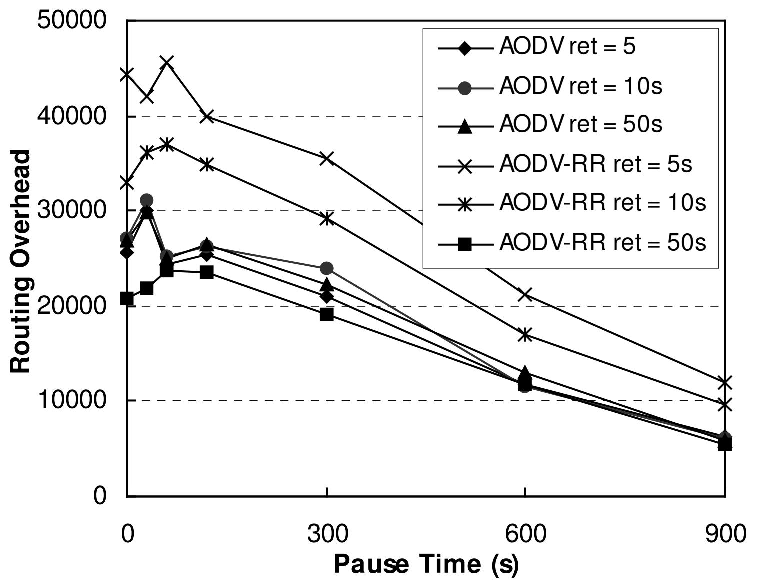


**Figure 4.4 Overhead Performance with Different RETs
512 byte/pkt, 4 pkt/s, 20 conn, max speed 5 m/s
(ret = Route Expiry Timeout)**

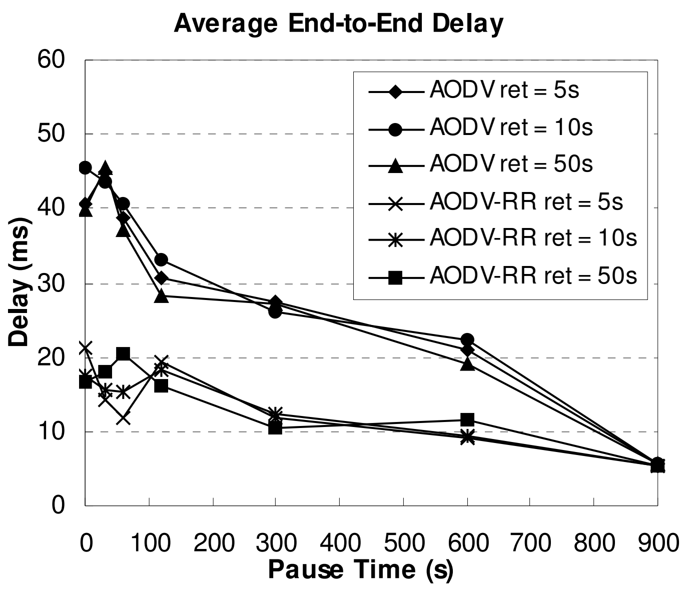


Figure 4.5 Delay Performance with Different RETs
64 byte/pkt, 4 pkt/s, 20 conn, max speed 5 m/s
(ret = Route Expiry Timeout)

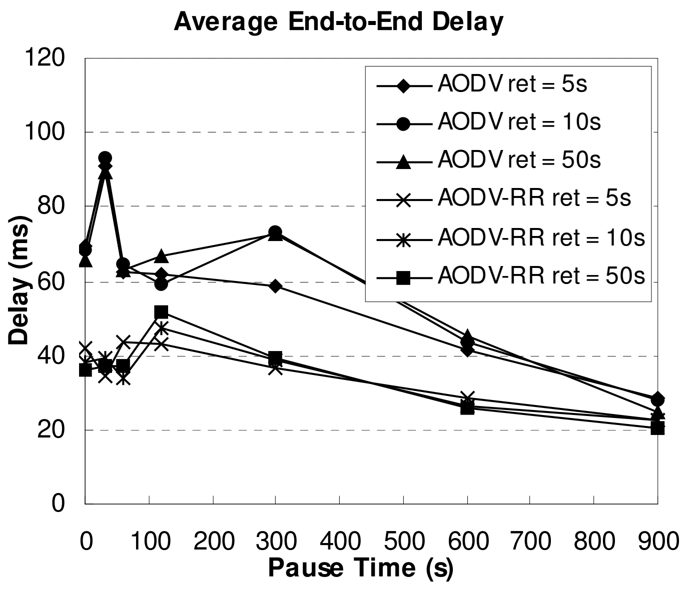


Figure 4.6 Delay Performance with Different RETs
512 byte/pkt, 4 pkt/s, 20 conn, max speed 5 m/s
(ret = Route Expiry Timeout)

We reduce the maximum mobile speed to 5 meters per second to set a smaller dynamic range for RET values. From the PDR curves in **Figure 4.1** and **Figure 4.3** we see that firstly, with the help from route redundancy, AODV-RR performs better than AODV in PDR at all values of RET. Secondly, changes in RET values cause larger fluctuations in the PDR curves for AODV then in those for AODV-RR. This implies that to some extent, the multiple backup routes built in AODV-RR do help compensate the impact of aging route information, which makes AODV-RR much less sensitive towards increasing RET than AODV is. In fact, at such a small mobile speed of 5 m/s, increasing RET from 5s to 10s has so little effect on AODV-RR that its PDR curves for these two cases are extremely close. It is reasonable to expect that, if we set a large maximum mobile speed, we would observe some more obvious differences among PDR curves for different RET cases.

However, the above improvement in PDR comes at a heavy penalty of routing overhead, as shown in **Figure 4.2** and **Figure 4.4**. Firstly, we observe that given the same RET value, AODV-RR actually incurs more routing overhead than AODV. This is because, at such low mobility with the small maximum mobile speed, the number of route requests issued by nodes become much less frequent, hence the savings from RREQ messages in AODV-RR have been reduced drastically and the increase in RREP overhead is no longer compensated. Secondly, smaller RET values cause significantly more routing overhead in AODV-RR, while having little impact on overhead in AODV. This is because in AODV-RR, more frequent expiration of routing information directly produces more overhead from widespread RERR broadcast in order to propagate the link error information; whereas in AODV with its local repair mechanism disabled, invalidating a route proactively or on-demand makes not much difference in overhead cost.

As far as average end-to-end delay is concerned, AODV-RR still outperforms AODV in general; however, no evident difference has been observed in **Figure 4.5** and **Figure 4.6** among cases with difference RET values. Adaptive RET aims to down-limit the degree of freshness of routing information; whereas the average delay only encompasses packets delivered successfully. Since stale routes typically cause data packets to be dropped, the average delay is not able to reflect the age of the route related to RET.

4.2.2 Statistical Results

In order to gain further insight into how RET values influence routing behaviour, we gather statistics about link breaks and route composition, and see how these measures vary with different RET values, as given in **Figures 4.7 ~ 4.9**.

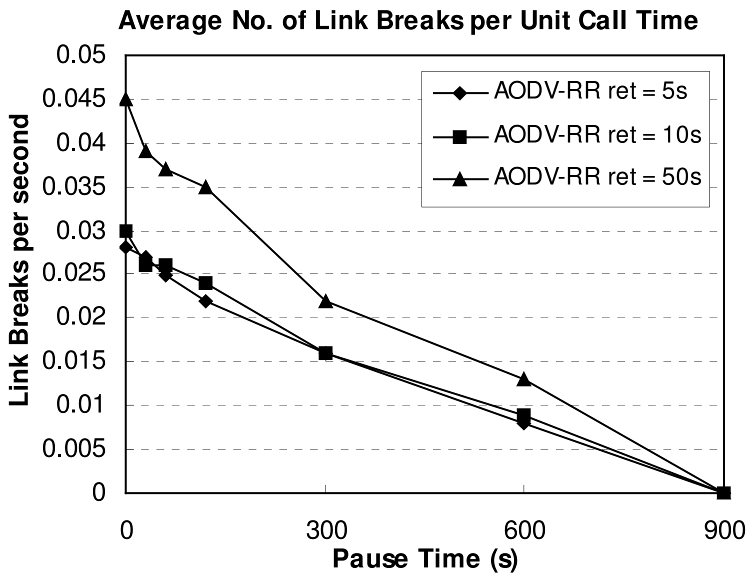


Figure 4.7 Average Number of Link Breaks per Unit Call Time AODV-RR, 64 byte/pkt, 4 pkt/s, 20 conn, max speed 5 m/s (ret = Route Expiry Timeout)

The average number of link breaks per unit call time indicates how often a data packet encounters delivery failure due to link breakage at the source or intermediate nodes, whether or not the data packet gets delivered to the destination eventually. With larger values of RET, routing information expires slowly and nodes are less frequently forced to update routes. Consequently, these potentially stale routes may not truly reflect the current topology condition and therefore should increase the chances of link failure encountered. The result in **Figure 4.7** agrees with our expectation, especially with the large difference between the 50-second RET case and the other two smaller RET cases. The proximity of the results for 5s and 10s RET cases implies that, at such as a small mobile speed of 5m/s, increasing RET from 5s to 10s causes no significant degradation in the freshness of the route cache information given the slow topology changes.

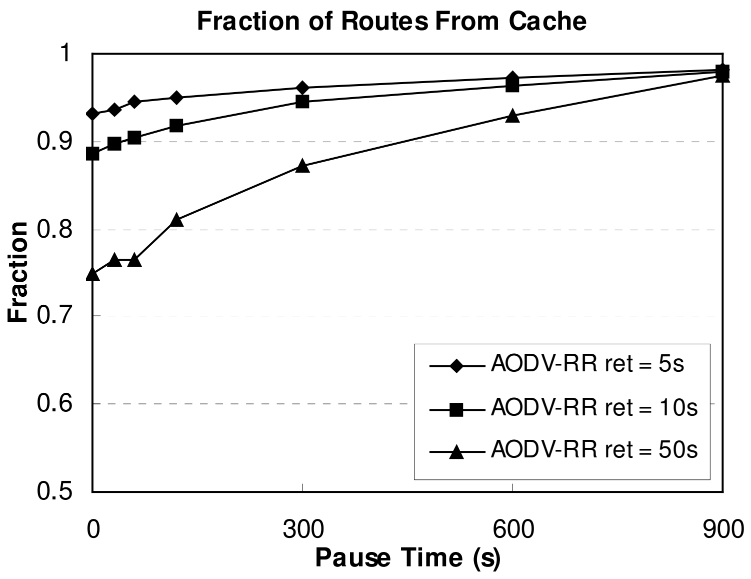


Figure 4.8 Fraction of Routes Found from Table (First Attempt) AODV-RR, 64 byte/pkt, 4 pkt/s, 20 conn, max speed 5 m/s (ret = Route Expiry Timeout)

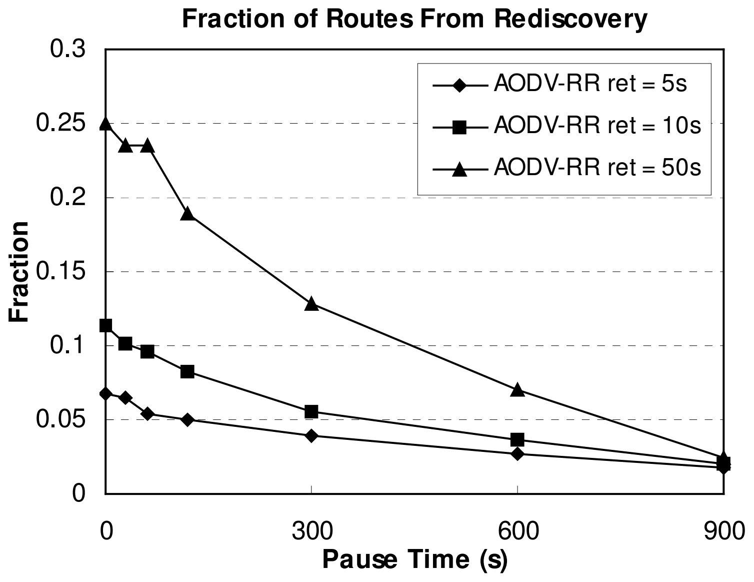


Figure 4.9 Fraction of Routes Found from Subsequent Attempts AODV-RR, 64 byte/pkt, 4 pkt/s, 20 conn, max speed 5 m/s (ret = Route Expiry Timeout)

In AODV-RR, the route selected for use could be from either the shortest primary route initially, or if the primary route fails, any shortest alternative route subsequently. We collect the fraction of successful routes found from the first primary route in the table and the complementary fraction of successful routes found from subsequent attempts in **Figure 4.8** and **Figure 4.9** above. As is evident from the figures, smaller values of RET result in more successful routes to be selected from the initial primary route, and correspondingly fewer routes from subsequent retries. This is because, smaller RETs force routing information to be refreshed more frequently, following topology changes more closely; hence larger chances for the primary route information to be correct, and correspondingly less need to rely on alternative choices after the primary route failure.

4.2.3 Conclusion

From the previous observations we can conclude that RET values indeed affect the performance and behaviour of routing protocols. It sets the lower limit on the freshness and hence the credibility of routing information. Smaller RETs reduce the potential of link failure and hence improve PDR performance, however, at the penalty of increased routing overhead.

4.3 Finding Optimal RET

Recognising the impact of RET on routing performance, we next seek to find the optimal RET value for a certain given mobility case on an experimental basis. By observing the results of such experiments, we expect to deduce a formula or rule of thumb relating the optimal RET value to mobility.

As concluded earlier, smaller RET improves PDR performance but incurs more control overhead. The optimal RET value should correspond to a good compromise between the two. Our methodology is to evaluate the routing performance with a large number of different RET values applied to different mobility cases and then locate the most cost-effective point in the curves. The results are given in **Figures 4.10 ~ 4.12**.

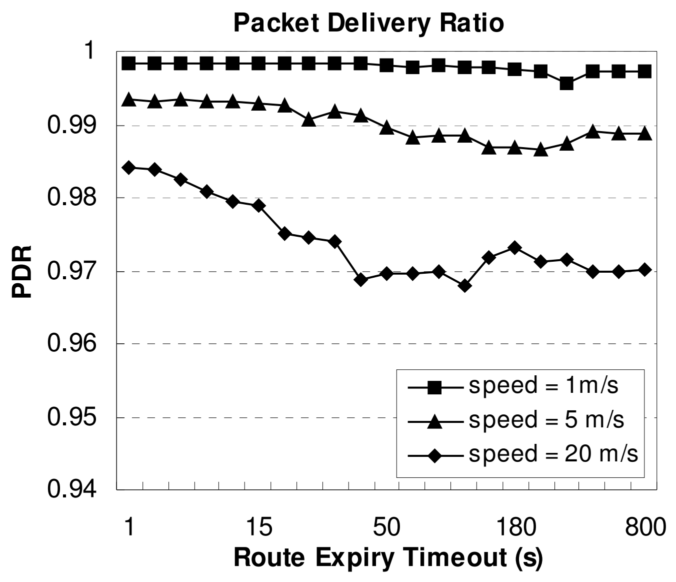


Figure 4.10 PDR Performance with Various RETs AODV-RR, 64 byte/pkt, 4 pkt/s, 20 conn, pause time 0s

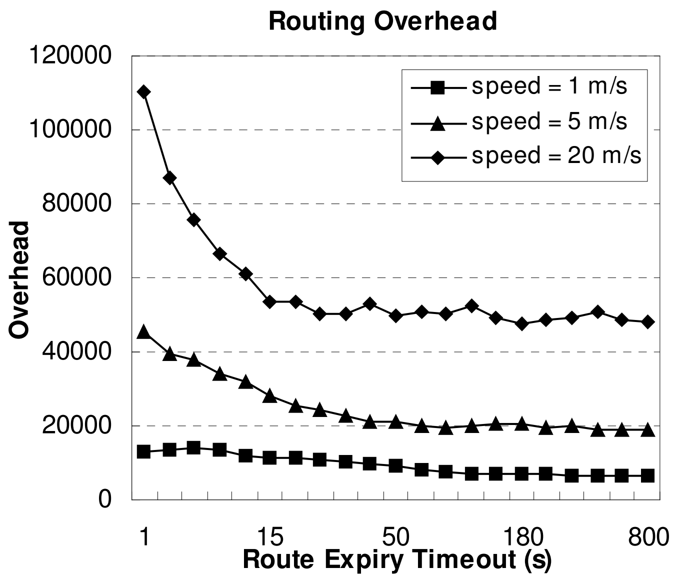


Figure 4.11 Overhead Performance with Various RETs AODV-RR, 64 byte/pkt, 4 pkt/s, 20 conn, pause time 0s

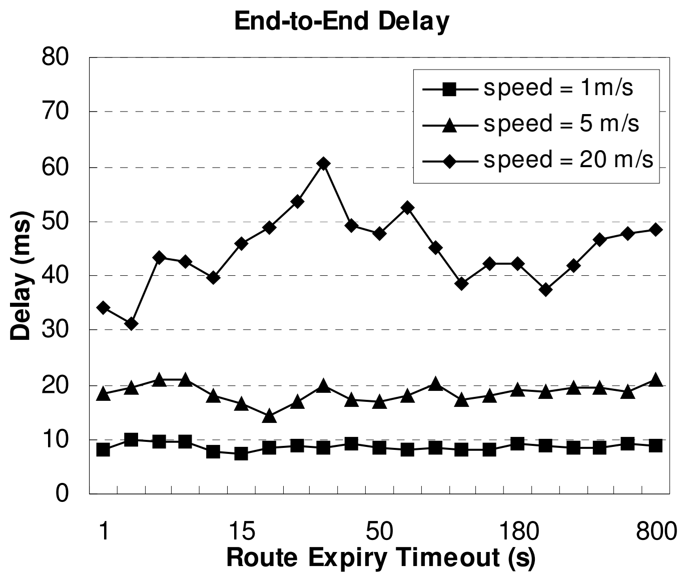


Figure 4.12 Delay Performance with Various RETs AODV-RR, 64 byte/pkt, 4 pkt/s, 20 conn, pause time 0s

In the random waypoint movement model [1], overall mobility depends on two parameters: the maximum mobile speed and the pause time. In this section we vary mobility by applying different maximum mobile speed (1, 5, 20 m/s) while fixing the pause time at 0 second (nodes continuous moving). Numerous sample RET values are tried ranging from 1 to 800 seconds.

From the above figures, varying RET has most significant impact on both PDR and overhead performance when mobility is high (20 m/s case). This implies that the benefit from enforcing route freshness depends on mobility. When mobility is low, topology changes slowly, routes do not break as often, hence forcing routes to expire faster brings less improvement. When nodes move faster, the chances of encountering a link break during the propagation interval of a packet greatly increases, hence it becomes more beneficial to ensure route freshness at high mobility. On the other hand, varying RET seems to have no significant or stable effect on end-to-end delay aspect of the performance.

The optimal RET point marks the maximum cost-effectiveness, hence it should be taken where the improvement in PDR starts to diminish while the incremental cost in control overhead starts to become larger. From the figures, it seems that the suitable optimal RET values are about the following:

- Maximum speed = 1 m/s → RET = 200 s
- Maximum speed = 5 m/s → RET = 50 s
- Maximum speed = 20 m/s → RET = 15 s

Hence we can deduce the following formula as a rule of thumb to estimate the suitable RET value for a given maximum mobile speed:

Suitable RET (s) = 200 / Maximum Mobile Speed (m/s) (4.1)

This **Equation (4.1)** applies for AODV-RR running with 64-byte packets, 4 packets per second, 20 connections and nodes continuously moving.

4.4 Adaptive RET Performance

Finally we test the performance of a system with adaptive RET for a range of mobile speeds, using **Equation 4.1** above to adjust the RET value for each mobility. We compare the performance of three systems: original AODV (with fixed RET), AODV-RR with fixed RET, and AODV-RR with Adaptive RET.

Once again, we vary the mobility by using different maximum mobile speeds and fixing the pause time at 0 second. The range of maximum mobile speed and corresponding suitable RET values according to **Equation 4.1** are given in **Table 4.2**.

Table 4.2 Maximum Mobile Speed and Suitable RET

Maximum Speed (m/s)	Suitable RET (s)
1	200
5	40
8	25
10	20
15	13
18	11
20	10

Figures 4.13 ~ 4.15 show the comparative performance of the three systems. For AODV-RR with adaptive RET, a different RET value computed using **Equation 4.1** and given in **Table 4.2** above is applied at each mobility. For the other two systems, the original fixed RET value is applied for all mobility cases.

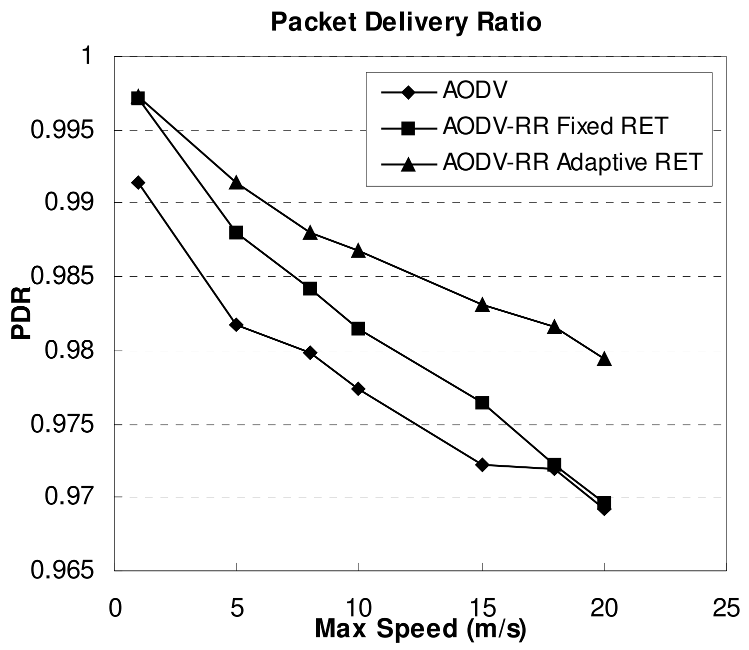


Figure 4.13 PDR Performance of Adaptive RET 64 byte/pkt, 4 pkt/s, 20 conn, pause time 0s

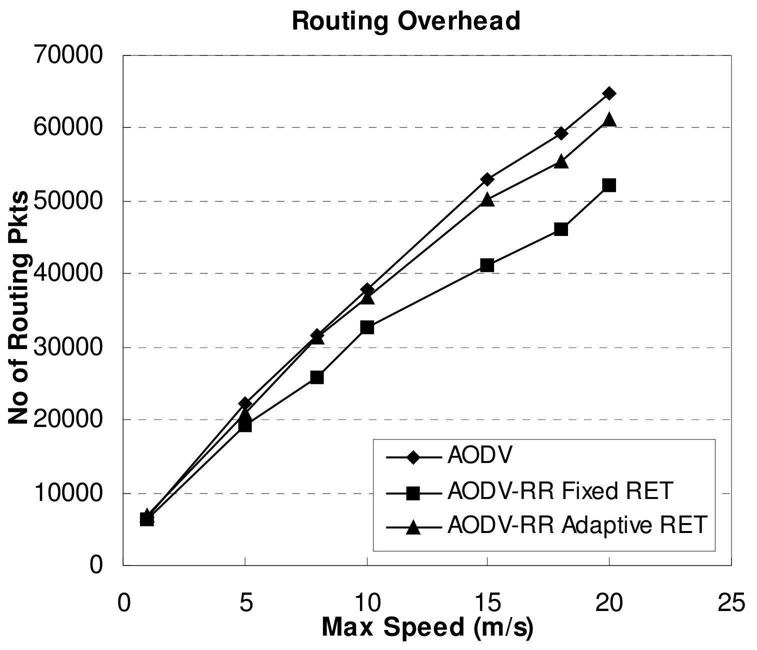


Figure 4.14 Overhead Performance of Adaptive RET
64 byte/pkt, 4 pkt/s, 20 conn, pause time 0s

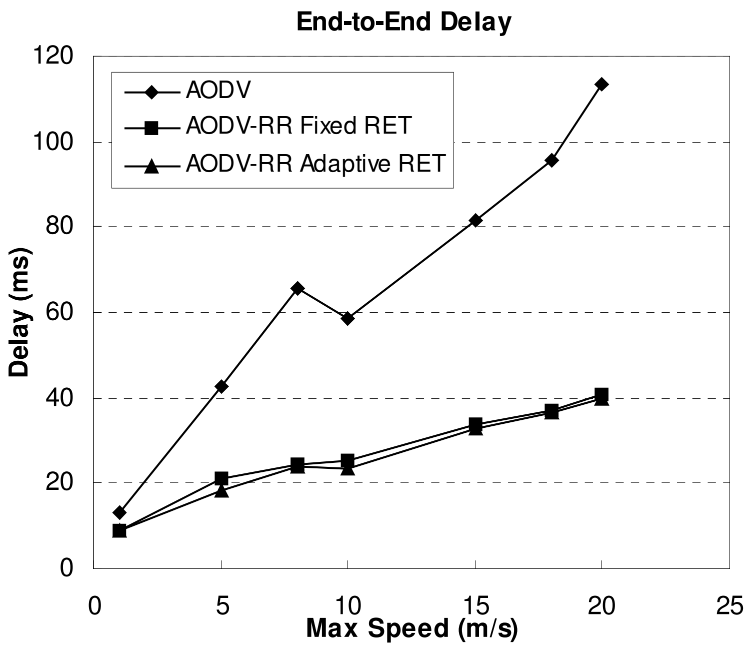


Figure 4.15 Delay Performance of Adaptive RET
64 byte/pkt, 4 pkt/s, 20 conn, pause time 0s

It is evident from the figures that AODV-RR with adaptive RET achieves much better PDR than AODV-RR with fixed RET, further improving from the original AODV. Moreover, the performance gain becomes more significant at higher mobility (larger maximum speed). Depending on mobility, adaptive RET forces routing information to be refreshed at an appropriate frequency thus preventing the use of potentially stale routes, and hence, reducing the chances of data delivery failures. In fact, from **Figure 4.13** we observe two types of convergence among the PDR curves. One is between the Fixed RET and Adaptive RET versions of AODV-RR when mobility decreases together with the maximum speed. This is expected because at higher mobility, routes expire faster, making the role of adaptive RET more crucial in maintaining route quality, hence the PDR performance gain due to adaptive RET goes larger. The other convergence happens between AODV-RR with Fixed RET and AODV when maximum speed increases, which is also within our understandings. AODV-RR with Fixed RET relies merely on route redundancy to improve robustness on top of AODV. As we have observed in **Section 3.5.1**, at higher mobility, the PDR performance gain from route redundancy is constrained by the degradation of cost-effectiveness due to the massive route verification process in AODV-RR.

However, in order to support more frequent route update activities driven by adaptive RET, additional routing overhead is needed. Therefore, the improvement in PDR comes at the penalty of larger routing overhead in AODV-RR with adaptive RET as compared to AODV-RR with fixed RET. Nonetheless by selecting the most cost-effective RET value, the increase in overhead can be kept within affordable range. For example, using our formula to estimate the optimal RET for each given maximum speed, the AODV-RR with adaptive RET still incurs less routing overhead compared the original AODV (see **Figure 4.14**).

On the other hand, once again adaptive RET has little impact on end-to-end delay. No evident difference in delay performance is observed between AODV-RR with adaptive RET and that with fixed RET. This again is due to the fact that average delay is unable to reflect the age of routes since it only encompasses successfully delivered data packets. Therefore the proximity in **Figure 4.15** between the delay curves for AODV-RR with and without Adaptive RET is expected and consistent with our observations from **Figure 4.5** and **4.6**.

4.5 Conclusion

From these experiments, we can conclude that adaptive RET indeed improves PDR performance by removing stale routes proactively. However, such improvement requires additional routing overhead to support more frequent route refreshing.

CHAPTER FIVE

CONCLUSION

MANETs have seen growing popularity and potentials recently. The self-organizing ability and easy deployment of MANET provide much greater flexibility in their applications. However fast changing topology, limited bandwidth and battery power in MANET environment bring big challenges to the reliability and robustness of its routing protocol without relying on pre-existing backbone infrastructure.

5.1 Contributions

We first study the different classifications of MANET routing protocols and their associated characteristics. We conduct a performance evaluation of various early routing protocols of different styles to compare their relative strengths and weaknesses. Our analysis agree that on-demand protocols achieves good performance and are well suited to MANET due to their efficient utilization of control packets.

We next explore the idea of route redundancy as a means to improve the robustness of MANET routing protocols. We develop a new scheme AODV with Redundant Routes (AODV-RR) that builds multiple routes for each source/destination pair and supplies immediate backup route to salvage traffic flows at the point of link failures. Our evaluation reveals that the new scheme with route redundancy not only achieves higher Packet Delivery Ratio (PDR) and substantially reduces average end-to-end delay, but also costs less total control overhead. Our studies prove that providing alternate and multiple routes indeed increases robustness of routing protocol and is especially beneficial for time-critical traffic.

We lastly experiment further improving routing performance by adapting Route Expiry Timeout (RET) to node mobility. We use smaller RET values at higher mobility to proactively prevent aging of routes, and hence, to reduce the chances of potential delivery failures. Our investigation suggests that applying adaptive RET values related to mobility is capable of largely boosting Packet Delivery Ratio (PDR) performance with an affordable increase in routing overhead.

5.2 Future Work

The approaches in our studies aim to improve the robustness of MANET routing protocols, with emphasis on delay performance to provide better support for time-critical traffic. With the benefits and advantages of these approaches observed, their application in a more realistic system carrying different types of traffic has not been tested fully. Given traffic with different priorities and characteristics including normal data, voice, video, and web interaction traffic, some mechanisms can be introduced to enable traffic discrimination and provide the most cost-effective service accordingly, for example by varying the degree of route redundancy or adjusting the proper range of RET values with different traffic priorities taken into consideration. In addition, the implementation of route redundancy mechanism in our project operates closely with the distributed routing in AODV. It will be interesting to investigate the performance of redundant routes in conjunction with other MANET routing strategies.

REFERENCES

[1] J. Broch, D. A. Maltz, D. B. Johnson, Y-C Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols", In the Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'98), October 25-30, 1998, Dallas, Texas, USA.

[2] National Science Foundation, "Research Priorities in Wireless and Mobile Communications and Networking" Report of a workshop held March 24-26, 1997, Airlie House, Virginia. Available at http://www.cise.nsf.gov/anir/ww.html.

[3] S-J Lee, W. Su, J. Hsu, M. Gerla, and R. Bagrodia, "A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols", In the Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000.

[4] J. Jubin and J. Tornow, "The DARPA Packet Radio Network Protocols", In the Proceedings of IEEE, Volume 75, No. 1, 1987, pp. 21-32.

[5] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", In the Proceedings of ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications, pp. 234-244, August 1994.

[6] C-C Chiang, H-K Wu, W. Liu and M. Gerla, "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel", In the Proceedings of IEEE SICON'97, April 1997, pp. 197-211.

[7] S. Murthy and J. J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks", ACM Mobile Networks and Applications Journal, Special Issue on Routing in Mobile Communication Networks, 1996.

[8] C. E. Perkins and E. M. Royer, "Ad-hoc On-Demand Distance Vector Routing", In the Proceedings of 2nd IEEE Workshop on Mobile Computer Systems and Applications, February 1999, pp. 90-100.

[9] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks", Mobile Computing, edited by Tomasz Imielinski and Hank Korth, chapter 5, pp. 153-181, Kluwer Academic Publishers, 1996.

[10] V. D. Park and M. S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks", In the Proceedings of INFOCOM'97, pp. 1405-1413, April 1997.

[11] C-K Toh, "A Novel Distributed Routing Protocol To Support Ad-Hoc Mobile Computing", In the Proceedings of 1996 IEEE 15th Annual International Phoenix Conference on Computing and Communications, March 1996, pp. 480-86.

[12] R. Dube, C. D. Rais, K-Y Wang and S. K. Tripathi, "Signal Stability-Based Adaptive Routing (SSA) for Ad Hoc Mobile Networks", IEEE Personal Communications, February 1997, pp. 36-45.

[13] A. Iwata, C-C Chiang, GY Pei, M. Gerla and T-W Chen, "Scalable Routing Strategies for Ad Hoc Wireless Networks", IEEE Journal on Selected Areas in Communications, Volume 17, No. 8, August 1999, pp. 1369-79.

[14] R. Sivakumar, P. Sinha and V. Bharghavan, "CEDAR: A Core-Extraction Distributed Ad Hoc Routing Algorithm", IEEE Journal on Selected Areas in Communications, Volume 17, No. 8, August 1999, pp. 1454-65.

[15] M. R. Pearlman and Z. J. Haas, "Determining the Optimal Configuration for the Zone Routing Protocol", IEEE Journal on Selected Areas in Communications, Volume 17, No. 8, August 1999, pp. 1395-1414.

[16] E. M. Royer, University of California, Santa Barbara, and C-K Toh, Georgia Institute of Technology, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks", IEEE Personal Communications, April 1999.

[17] Internet Homepage for The Network Simulator – ns-2, available at http://www.isi.edu/nsnam/ns/.

[18] S-J Lee and M. Gerla, "AODV-BR, Backup Routing in Ad Hoc Networks", In the Proceedings of IEEE Wireless Communications and Networking Conference (WCNC), Chicago, September 2000, pp. 1311 – 1316.

[19] ML Jiang, JY Li and Y.C. Tay, "Cluster Based Routing Protocol(CBRP)", Internet Daft, draft-ietf-manet-cbrp-spec-01.txt, July 1999.

[20] T. Clausen, P. Jacquet, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum and L. Viennot, "Optimized Link State Routing Protocol", Internet Draft, draft-ietf-manet-olsr-06.txt, September 2001.

[21] M. S. Corson and V. D. Park, "An Internet MANET Encapsulation Protocol (IMEP) Specification", Internet Draft, draft-ieft-manet-imep-spec-00.txt, November 1997.

[22] K. Fall and K. Varadhan, "The *ns* Manual (formerly *ns* Notes and Documentation)", The VINT Project – A Collaboration between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC, October 14, 2000, available at http://www.isi.edu/nsnam/ns/ns-documentation.html.

[23] Internet Homepage for The Monarch Project, available at http://www.monarch.cs.cmu.edu/ and http://www.monarch.cs.rice.edu/.

[24] B. Tuch, "Development of WaveLan, an ISM band wireless LAN", AT&T Technical Journal, 72(4):27-33, July/August 1993.

[25] IEEE Computer Society LAN MAN Standards Committee, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", IEEE Std 802.11-1997, IEEE, New York, 1997.

[26] G. R. Wright and W. R. Stevens, "TCP/IP Illustrated, Volume 2: The Implementation", Addison-Wesley, Reading, Massachusetts, 1995.

[27] Redhat Linux Homepage, available at http://www.redhat.com/.

[29] Ad Hoc On-Demand Distance Vector Routing (AODV) Homepage, available at http://moment.cs.ucsb.edu/AODV/aodv.html.