PROGRESSIVE ROUTE DISCOVERY PROTOCOLS FOR WIRELESS MESH NETWORKS

by

XUHUI HU

A dissertation submitted to the Graduate Faculty in Engineering in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York

2008

UMI Number: 3296960

UMI Microform 3296960

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

This manuscript has been read and accepted for the Graduate Faculty in Engineering in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

January 28, 2008
Date

Myung J. Lee
Professor **Myung J. Lee**
Chair of Examining Committee

January 28, 2008
Date

Mumtaz Kassir
Dean **Mumtaz Kassir**
Executive Officer

Professor **Tarek N. Saadawi**
The City University of New York

Professor **Yi Sun**
The City University of New York

Professor **Kaliappa Ravindran**
The City University of New York

Professor **Jizhong Xiao**
The City University of New York
Supervision Committee

THE CITY UNIVERSITY OF NEW YORK

PROGRESSIVE ROUTE DISCOVERY PROTOCOLS FOR WIRELESS MESH NETWORKS

by

XUHUI HU

Advisor: Professor Myung J. Lee

ABSTRACT

Wireless mesh routing is responsible to establish low-cost, high-quality routes in a dynamic environment. At the same time, it needs to consider the efficient usage of network resources. How to find the optimal route(s) with the minimum overhead is a challenging research topic.

There are several types of routing approaches. Proactive approaches are not suitable for dynamically changed networks since valuable network resources are wasted to keep nodes updated with unused links and routes. In flooding-based reactive approaches, as all network nodes are required to participate in the relays of route request packets, substantial control overhead is still inevitable.

In this thesis, we propose a set of progressive route discovery protocols (PRD), which obtains high-quality routes with little overhead in different network situations. The basic idea of PRD is to concentrate route discovery efforts in the regions that most likely contain the optimal route(s). More specifically, PRD divides the route discovery processes into several stages and progressively finds cost-efficient routes. In the

network scan, namely, the first stage, the whole network is roughly explored and a preliminary route is set up between each source-destination pair with very little overhead. The network scan stage may not locate the best route, however, it helps locate route discovery regions, thus significantly decreasing route discovery overhead at the subsequent stages. At the optimal route discovery stage, a route discovery region is defined in the neighborhood of each preliminary route; and the source device concentrates route discovery effort on discovering the cost-efficient route towards its desired destination only within this region.

The progressive route discovery protocols can be easily extended to cover multipath routing and route update cases. During multi-path route discovery stage, a one-hop insulating region is built around a single-path route, and a new route is then established outside the region. Since these two paths are isolated by the region, they can effectively avoid inter-path interference, and therefore can, effectively support simultaneous data transmission for throughput aggregation.

Both performance analysis and simulation results show that the progressive route discovery protocols are a set of cost-efficient routing protocols beneficial to various wireless mesh networks.

To my husband, my son, and my parents

三人行，必有我师焉，

择其善者而从之，其不善者而改之

— 孔子

Among those around me,

there must be someone whom I can learn from.

I will follow their virtues,

but avoid their shortcomings.

— Confucius

ACKNOWLEDGEMENTS

My first and most earnest acknowledgment must go to my advisor: Dr. Myung J. Lee. Throughout my doctoral study, he provided constant support and encouragement, as well as conscientious advices and guidance. His wide knowledge and his logical way of thinking have been of great value for me. None of my work would have been possible without him.

I want to show my great appreciation to Dr. Tarek N. Saadawi, Dr. Yi Sun, Dr. Kaliappa Ravindran, and Dr. Jizhong Xiao, the members of my supervisory committee, for taking time to review this dissertation and make valuable comments on my work. I am also grateful to Dean Mumtaz Kassir for providing me administrative advice throughout my graduate study.

I am fortunate to have the opportunity to work with a group of energetic people in Samsung joint research Lab. I have enjoyed every moment that we have worked together. All former and present members of Samsung joint Lab have taught me many things about life. I want especially to thank Dr. Yong Liu, Chunhui Zhu, Dr. Jianliang Zheng, Junjun Li, Hsin-Hui Juan, Baozhi Chen, June-Seung Yoon for their friendships and their collective encouragement to finish this dissertation. I also want to thank my colleagues I have worked with in the Center for Information Networking and Telecommunications (CINT). They are Peixiang Gong, Dr. Zhengliang Yi, Dr. Guanhua Ye, Dr. Osama Hussein, Dr. Ahmed Abd El Al, Alberto Aponte and our secretary Ms. Victoria Okai.

This Ph.D research was sponsored by U. S. Army Research Laboratory and the Samsung Advanced Institute of Technology. I want to thank them for financially supporting me.

I also owe a huge debt of gratitude to my parents, my mother-in-law and my husband for their continuous love and encouragement. This dissertation is dedicated to them. Most of all, I would like to thank my husband Yong for his patience and sacrifice throughout my study. It is impossible to have my research career without his love and support.

Last but not least thanks to my son Yuheng Liu for smiling every day during my thesis-writing period. It surely cheered me up.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ACK	Acknowledgement
AODV	Ad Hoc On Demand Distance Vector protocol
AODVjr	AODV junior
AODV-DM	AODV Decoupled Multipath routing protocol
AODVM	AODV-based node-disjoint Multipath routing protocol
CBR	Constant Bit Rate
cRREP	correction RREP
CTS	Clear To Send
DCF	Distributed Coordination Function
DIFS	DCF Inter-Frame Spacing period
DLOC	Destination LOCating packet
DSDV	Destination Sequenced Distance Vector
DSR	Dynamic Source Routing
E2E	End to End
EDSR	Extended Dynamic Source Routing
FSR	Fisheye State Routing
GPS	Global Positioning System
GPSR	Greedy Perimeter Stateless Routing
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
LAR	Location-Aided Routing
LS	Link State routing

MAC	Medium Access Control
MBCR	Minimum Battery Cost Routing
MCST	Minimum Cost Spanning Tree
MPR	MultiPoint Relaying
MTU	Maximum Transmission Unit
PCF	Point Coordination Function
PID	Path ID
PROC	Progressive Route Calculation algorithm
pRREP	primary Route REply Packet
PSN	Path Sequence Number
OLSR	Optimized Link State Routing
QL	Query Localization
QoS	Quality of Service
Q-OLSR	the QoS version of OLSR
RASB	Route Assembling packet
RCOL	Route COLlection packet
RDEF	Region DEFinition packet
RDP	Route Discovery Packet
REGR	REGion based Routing
RF	Radio Frequency
RERR	Route ERRor packet
RPRT	Region PRoTection packet
RREJ	Route reply REJection packet

RREP	Route REPly packet
RREQ	Route REQuest packet
RRP	Route Reply Packet
RTS	Request To Send
RU-SRC	Route-Update SouRCe
RU-DEST	Route-Update DESTination
SACKs	Selective Acknowledgement packet
SCTP	Stream Control Transmission Protocol
SIFS	Short Inter-Frame Spacing
SMR	Split Multipath Routing algorithm
sRREP	secondary Route REPly packet
TBR	Tree-Based Routing
TCP	Transmission Control Protocol
TS	Topology Server
TSNs	Transmission Sequence Number
TTL	Time-To-Live
TREP	Topology REPort
UDP	User Datagram Protocol
UNNQ	UNsettled Node Queue
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network
ZRP	Zone Routing Protocol

1. INTRODUCTION

1.1 Background

In areas where there is no communication infrastructure, or the infrastructure is expensive or inconvenient to be deployed, wireless users may still be able to communicate with each other via the form of wireless mesh networks. A wireless mesh network is an autonomous system that is composed of a group of intelligent wireless nodes. In this system, a node dynamically determines a path from itself hop by hop through the network to other nodes, without depending on any existing network infrastructure or centralized administration (like base stations and access points). Besides, it may automatically fix the break of paths caused by mobility of nodes, changes of RF propagation, etc. Since a mesh network can be deployed rapidly and flexibly, it is attractive to numerous potential applications, ranging from multi-hop

wireless broadband Internet access to intelligent transportation system to voice and video communications for disaster areas.

Although mesh networks provide advantages in deployment, cost, and distributed intelligence, they incur new challenging issues. First is the issue of resource constraints such as battery capacity, processing capability, and lack of memory space. Second, due to the dynamic nature of wireless environment, the bandwidth is severely limited. Last, the topology of wireless network experiences frequent changes which is caused by the node mobility and error prone transmissions. All these constraints join to contribute significant challenges for wireless mesh networks, especially the operation of routing protocols. How to make the best use of limited network resources for routing packets is the key issue in wireless mesh networks.

1.2 Brief Review of Existing Routing Protocols

There have been numbers of routing protocols for wireless mesh networks. The following subsections classify and analyze some of the existing approaches.

1.2.1 Proactive Routing Protocols

In proactive routing algorithms like DSDV [1], each node calculates and maintains one or more routing tables for all other nodes in the network. These tables are refreshed periodically and/or updated whenever network topology changes [2-5]. Proactive approaches are beneficial for traffic models in which a large number of nodes communicate each other; and source-destination pairs change with time. Since all nodes have a consistent and up-to-date view of all other nodes, proactive approaches

bring low route discovery latency. However, as network-wide routing information is maintained disregarding the practical traffic conditions, valuable network resources are wasted to refresh and keep unused routes. Some proactive routing protocols such as OLSR [6] try to minimize the overhead of flooding packets by reducing redundant retransmissions. However, the efficient retransmission mechanism used in OLSR is not sufficient to derive the optimal routes when route metrics other than hop count are considered. Another approach to avoid the excessive update traffic is to employ tree structures [7] for multihop data transmission. Unfortunately, a tree route is most likely non-optimal.

1.2.2 Reactive Routing Protocols

To keep nodes from being updated with unused links and routes, network-wide flooding based reactive routing protocols like AODV [8] permit nodes to form and store only the routing entries of active routes [9-13]. Reactive routing protocols are favored in dynamic networks since route discovery overhead is efficiently controlled. To search the active routes, the flooding based reactive protocols probe all directions of the network with the same effort. However, only the area between the source-destination pair deserves thorough exploration. Nodes located outside this area waste their power to relay the route request packets (RREQs). These unnecessary RREQ relays not only burden the whole network but also interfere with normal data traffic. The problem is even more serious when multiple paths are required because traditional flooding based multi-path routing approaches consume much more control overhead.

Some efficient broadcast schemes [14-17] can be utilized to suppress redundant RREQ relays, but they often suppress the discovery of optimal routes, too.

1.2.3 Location-aware Routing Protocols

To further reduce overhead of route discovery, some location-based approaches such as Location-Aided Routing (LAR) [18] and Greedy Perimeter Stateless Routing (GPSR) [19] [76] suggest using location techniques (for example, global position system (GPS)) to make routing decisions. LAR protocol limits the search for a new route to a smaller "request zone" of the network by using location information, and thus results in a significant reduction in the number of routing messages. GPSR makes greedy forwarding decisions using only information about a router's immediate neighbors. LAR and GPSR assume that the wireless nodes are location-aware. But in some situations, equipping all nodes with GPS or other location techniques is not economic and accurate. Other techniques are required for efficient route discovery and maintenance.

1.2.4 Local Repair based Routing Protocols

Node mobility, malfunction and power-depletion may lead to breakage or energy exhaustion of existing routes. Good route maintenance and update schemes, therefore, become extremely important.

Among the existing route repair routing protocols, AODV local repair tries to fix a broken link locally by sending RREQ from the upstream node of the broken link with a small TTL. However, the updated route may become longer than the optimal path and

with the continuous movement of the destination, the route is extended longer and longer. In [20], a Query Localization (QL) scheme is proposed to utilize the directional information implied in the old route for efficient route repair. QL puts a counter in each RREQ to limit its travel steps away from the old broken route. Therefore, only a few neighbors of the old route members participate in the RREQ relays. QL is very efficient in the repair of sporadic node or link failures; however, as the updated route always shares the main body with the old route, it does not fit in the cases requiring route-wide renewal.

1.2.5 Multipath Routing Protocols for Throughput Aggregation

Sometimes, a single route cannot provide sufficient capacity to support traffic intensive applications. Multipath mechanism can be used for throughput aggregation. Most traditional multipath routing methods tend to establish link-disjoint or node-disjoint paths with minimal path costs [21-34]. As paths created by these methods are typically close to each other, they may cause serious inter-path interference when serving for concurrent data transmissions. To eliminate the inter-path interference, M. R. Perlman et al. proposed to use multiple channels to construct contention-free paths [21]. In [22], D.Saha et al. suggested using directional antennas to reduce radio interferences between different paths. Although these approaches can decouple adjacent paths, they require extra resources like directional antenna or multi-channel supported equipments, which may not be practical in low-cost wireless mesh networks.

1.3 Motivation and Contributions of the Thesis

Although there have been a lot of routing protocols for wireless mesh networks, they need further improvements to perform more efficiently. In this thesis, we propose a set of progressive route discovery protocols, which divide the route discovery processes into network scan stage, the optimal route discovery stage, as well as route update and proliferation stage. By finding the optimal or near optimal route(s) gradually, the progressive approach greatly reduces the spread of the route discovery packets without losing route optimality.

The design and analysis of the progressive route discovery protocols focus on the following issues:

A. ***Avoid the waste of network resources***

The conventional routing algorithms are inclined to combine the destination exploration and the optimal route discovery. As a result, every node in the network is involved in searching the desired destinations. In fact, with the help of some sketchy network scan schemes or efficient broadcast schemes, the destination discovery itself does not cause much control overhead since only a small portion of nodes are required to relay the destination search packets. Our progressive route discovery protocols try to save control overhead by separating destination exploration process from the optimal route discovery process.

The sketchy network scan or efficient broadcast scheme might not be able to detect the optimal route; nevertheless, they can roughly delimit a region between each source and the desired destination, in which the optimal route or near-optimal route is

very likely contained. The optimal route discovery process can then be activated within this region to dig out the best route.

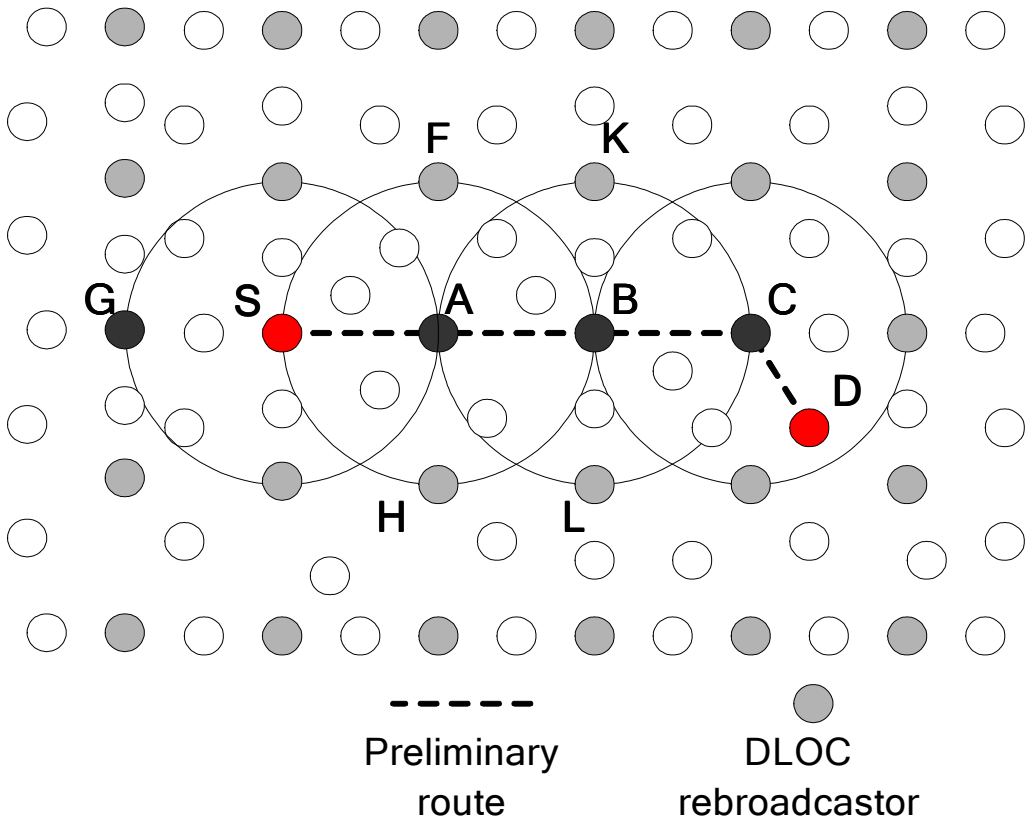


Figure 1-1. Example of preliminary route discovery

Figure 1.1 shows a simple example of preliminary route discovery process using a distance-based broadcast scheme [15]. An active source S initiates on-demand route discovery by first broadcasting a destination-locating message (DLOC). Ideally, only the border nodes of the message senders need to rebroadcast the message, which engenders very little control overhead. It is clear that in a dense network, the efficient broadcast of destination-locating messages engenders much less control overhead than the full flooding does.

The nodes relaying the destination-locating message have to create a backward routing entry for the source. So that when the DLOC arrives at the destination, a preliminary route is immediately ready in the backward direction (from destination to source).

*B. **Obtain the optimal or near-optimal route***

Although the preliminary route discovered during the first stage engenders very little overhead, it is usually not an optimal route.

In order to locate the optimal route, we propose to create a route refinement region in the neighborhood of the preliminary route, and then make the source broadcast a route discovery message only within this region. The width of the region can be easily adjusted based on network density and network resource usage. Figure 1-2 shows a simple example of the optimal route discovery process using RREQ broadcast approach. Since the route refinement region restricts route discovery activities to the nodes that most likely constitute the optimal or near-optimal route, it makes the optimal route discovery process quite efficient.

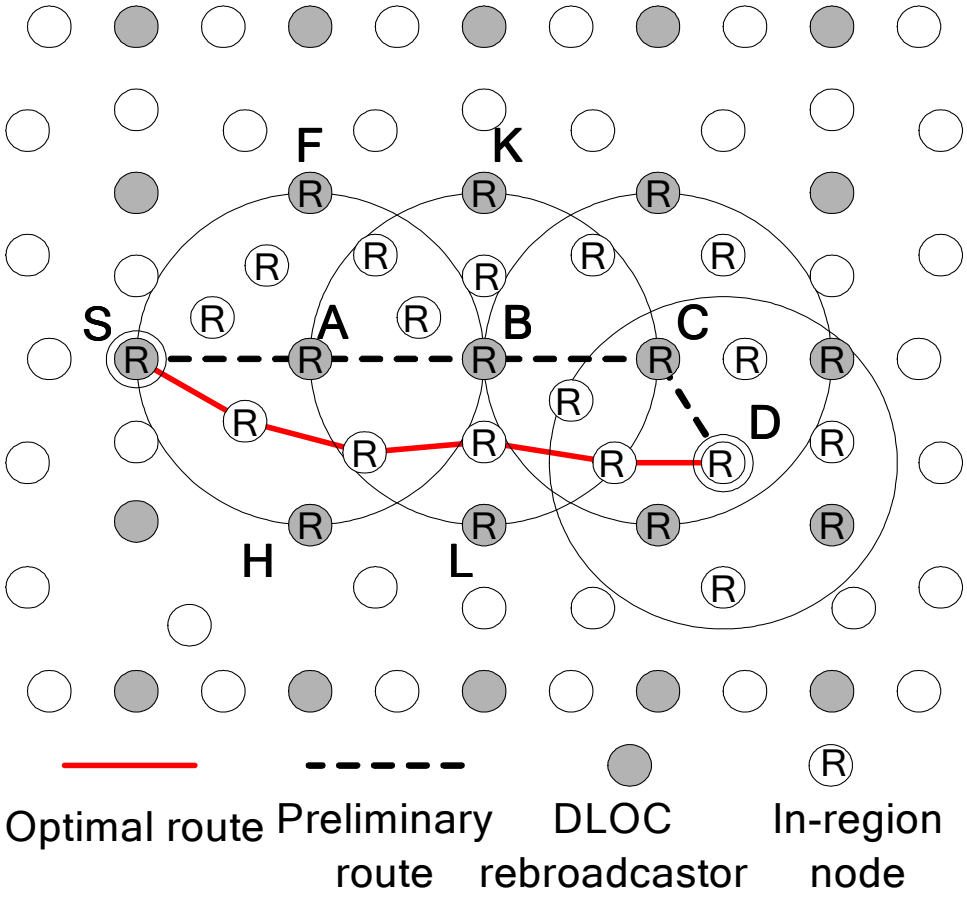


Figure 1-2. Example of the optimal route discovery

If every node is aware of its local topology information, we can further improve the route refinement process by making the members of the preliminary route propagate their local knowledge along the preliminary route, and jointly calculate the optimal route based on the region topology information.

*C. **Enhance or update an old route***

When a new route is demanded to repair, refresh, or enhance an old route, the source need not initiate a totally new route discovery process. It can simply follow the

progressive route discovery principle, and establish the new route in the nearby region of the old route.

For example, by building a one-hop insulating region in the neighborhood of a single-path route, and discovering a new route outside the insulating region (but inside the two-hop region of the single-path route), we can obtain two decoupled routes with very little route discovery overhead. The decoupled multipath can effectively avoid inter-path interference, and nicely serve in simultaneous data transmission. Figure 1-3 shows an example of decoupled multipath route proliferation process. Besides, by reducing the size of the insulating region to include only the single path members, we can easily obtain node-disjoint multipath, which can serve for fault tolerance and load balancing purposes.

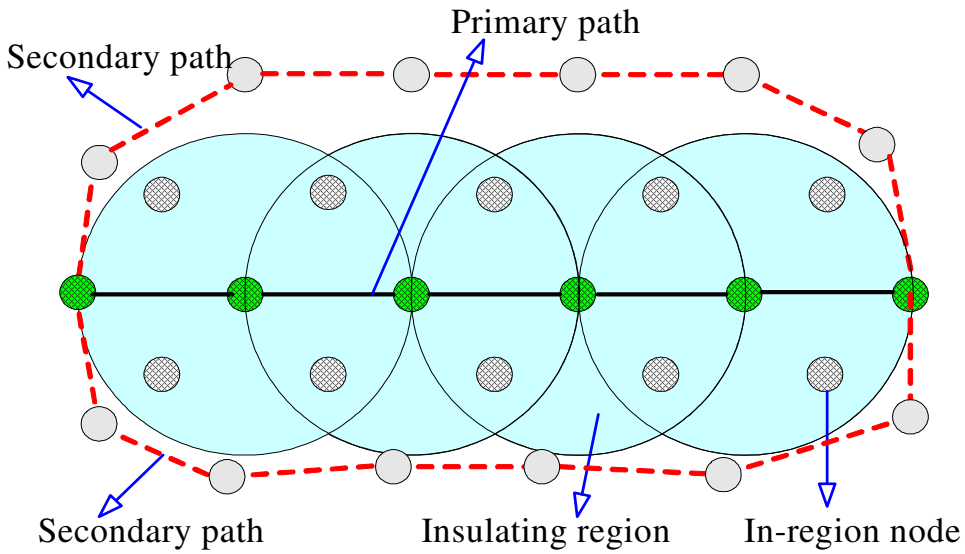


Figure 1-3. Example of route proliferation

1.4 Outlines of the Thesis

This thesis is organized as follows.

Chapter 1 is the introduction of the thesis. It sets forth the background, motivation and contributions of this research work.

Chapter 2 presents a progressive route discovery protocol to find a single optimal route. This chapter first describes two proactive schemes and one reactive scheme, which are used to roughly scan the whole network and efficiently determine preliminary routes between sources and their desired destinations; and then it introduces two route refinement approaches to elaborately search the optimal or near-optimal route(s) based on the preliminary routes. The efficiency of these schemes is evaluated.

Chapter 3 continues the progressive routing discovery process by addressing a route update protocol. Similar to route discovery process, the source node in the route update process tries to find a new optimal route towards its destination at a local area. However, it skips the destination discovery stage since the old route is a good "preliminary route". The route update approach is presented and analyzed in this chapter.

Chapter 4 extends the progressive routing protocol from single route discovery stage to multipath route discovery stage. In this chapter, a multipath routing structure for concurrent data transmission is described in detail. The cross-layer optimization of the multipath structure is also addressed.

Chapter 5 concludes the thesis and discusses the future research directions.

2 PROGRESSIVE ROUTE DISCOVERY — ROUTE CREATION PROTOCOL

2.1 Introduction

One of the crucial but challenging issues in multihop wireless networks is how to route packets efficiently in resource-limited wireless environments.

In traditional proactive routing protocols like DSDV [1], each node collects and updates routing information for every other node in the network, periodically and/or whenever network topology changes. As a result, all nodes have a consistent and up-to-date view of the network. Proactive routing protocols are suitable for static wireless networks with a large amount of short traffic sessions. Another main advantage of this category of protocols is their extremely low latency – once an active source needs to

contact a distant destination, an optimal route is immediately ready However, since network-wide routing information is recorded and updated by each node regardless of traffic requirement, valuable network resources will be wasted if these protocols are applied to dynamic environments, or networks where only a few network nodes transmit data with each other. On the contrary, network-wide flooding based reactive routing protocols like AODV [8,9] and DSR [12] permit nodes to form and store only the routing entries of active routes. To search an active route, the network-wide flooding based reactive protocols probe all directions of the network with the same effort. However, only the area between the source-destination pair deserves thorough exploration. Nodes located outside this area waste their power to relay the route request packets (RREQs). These unnecessary RREQ relays not only burden the whole network but also interfere with normal data traffic. The problem is even more serious when multiple paths are required because traditional flooding based multi-path routing approaches consume much more control overhead. It is obvious that the network-wide flooding based routing protocols is beneficial to networks with very few traffic sessions; nevertheless, substantial control overhead will be unnecessarily consumed if a large number of routes are established with these protocols. It is, therefore, a great challenge to design a routing approach that is cost-efficient in a wide range of wireless networks.

Some proposed proactive routing algorithms such as OLSR [6] try to reduce the overhead by removing unnecessary topology update efforts, but they are not sufficient to derive the optimal routes when route metrics other than hop count are considered. Other efficient broadcast schemes [14]-[18] help to reduce reactive route request

packets (RREQ) by allowing only a small portion of nodes relaying RREQs. However, as many nodes lying between the source and destination are excluded from the route search, the optimal routes are very likely ignored.

In this chapter, we propose a progressive route discovery protocol, which gradually finds the optimal or near optimal routes in order to concentrates route discovery efforts in a local region that most likely contains the optimal or near-optimal route. Our progressive route discovery protocol consists of two stages: network scan stage and optimal route discovery stage.

At network scan stage, efficient routing approaches are employed so that nodes can set up routing tables for other nodes with only a little control overhead. Based on the routing tables, the shortest routes are built and are to be functioned as “preliminary routes” during the optimal route discovery. Both proactive and reactive routing approaches are introduced in this chapter to sketchily scan the network and find the preliminary routes. Although the preliminary routes found by these approaches might not be the best route as viewed from link quality and energy efficiency, they are good direction indicators for further route refinement.

At the optimal route discovery stage, an active source concentrates route discovery efforts only within small regions around the preliminary routes. Two optimal route establishment schemes are addressed to locate the optimal routes: broadcast scheme and unicast scheme. The former simply asks the active source to broadcast a route discovery packet within the neighborhood region of each preliminary route to find the optimal route. The latter unicasts a topology report (TREP) packet along each

preliminary route and makes the members of the preliminary route jointly calculate the optimal route according to the region topology information.

The rest of the chapter is organized as follows. Chapter 2.2 introduced related works. Chapter 2.3 describes the detailed operations of the progressive routing discovery protocol. Chapter 2.4 discusses simulation results. Finally, Chapter 2.5 concludes the chapter.

2.2 Related works

The existing wireless routing discovery protocols can be divided into two categories: proactive routing protocols and reactive routing protocols. Most proactive routing approaches evolve from distance-vector [1] or link-state schemes [3]. In these approaches, each node collects and updates routing information for every other node in the network, periodically and/or whenever network topology changes. Since all nodes have a consistent and up-to-date view of all other nodes, proactive approaches bring low latency. However, as network-wide routing information is maintained regardless of the traffic conditions, valuable network resources are wasted to keep nodes updated with unused links and routes. The problem becomes acute in highly dynamic networks, where frequent link changes cause significant topology update overhead. Some proposed proactive routing algorithms try to reduce the overhead by removing unnecessary topology update efforts. For example, OLSR [6] uses a Multipoint Relaying (MPR) technique [36] to reduce topology control overhead in large and dense networks. In addition, it decreases the size of the topology update packets by broadcasting only partial topology information. Although the MPR technique together

with partial topology information is enough for a node to calculate the shortest path to any other node, it is not sufficient to derive the optimal routes when route metrics other than hop count are considered. Another approach to avoid the excessive update traffic is to employ tree structures for multihop data transmission. For instance, in a tree-based routing (TBR) protocol introduced in [7], a spanning tree that radiates from the tree root and connects all network nodes is built, so that each node can reach any other node by using a tree route. The disadvantage of TBR is that, since packets in TBR are always forwarded along tree links, the routes, especially those between leaf nodes, are far from being optimal. Moreover, as more traffic goes through the top portion of the tree, serious load unbalance is expected in TBR structure. QOLSR [37] is a QoS version of OLSR that takes account of QoS requirements in the selection of reported links. However, as the heuristic for the selection of MPRs is to ensure QoS, more neighbors are selected as MPRs in QOLSR. Besides, whenever any link/node cost of QOLSR is changed, the change is broadcast to the whole network. Thereby, QOLSR creates much more topology control overhead than OLSR, especially when the network is highly dynamic.

In reactive routing approaches, a source initiates a topology exploration on demand. Most reactive routing approaches such as AODV [8] and DSR [12] employ full flooding schemes to discover optimal routes. In these approaches, an active source initiates a route discovery process by flooding or gradually broadcasting a route request packet (RREQ) to the network. All nodes, except the source and the destination, are required to rebroadcast the first received RREQs, and/or the RREQs propagated from better routes. Reactive routing approaches efficiently control route discovery overhead

by keeping nodes from being updated with unused links and routes. Hence, they are favored in dynamic networks.

In the reactive routing approaches based on full flooding, all directions of the network are probed with the same effort. However, only the region between each source-destination pair deserves thorough exploration. Nodes located outside this region waste their power to rebroadcast the RREQ's. These unnecessary rebroadcasts burden the whole network and interfere with normal data traffic.

If all nodes in the network are aware of their own positions, and each source also knows the position of its desired destination, route discoveries can be tightly controlled within a small rectangle "request zone" delimited by the locations of the source-destination pair [18]. Nevertheless, most position-aware networks are designed to equip with GPS receivers, whose expense, accuracy and dependence on outdoor environment make them unsuitable for general mesh networks. In addition, the inquiries and updates of destination positions lead to large control overhead especially in highly mobile networks.

A few efficient broadcast algorithms [14]-[17] have been proposed to limit packet relays to a small portion of nodes. For instance, Ni, *et al*, [15] investigated the broadcast storm problem in mesh networks and presented several schemes to reduce redundant broadcasts. Some of these schemes, such as the probabilistic scheme, the count-based scheme and the distance-based scheme, are very simple and do not rely on any neighbor information. However, in order to achieve high reachability, these schemes still incur considerable broadcast overhead. Cartigny and Simplot [16], and

Feng, *et al*, [17], proposed to give rebroadcast privilege to border nodes. In [16], the distance between the message sender and receiver is approximated by the numbers of their respective neighbors and their common neighbors. The border nodes of the sender have bigger probabilities to rebroadcast the received message. In [17], the distance between the message sender and receiver is calculated from the message signal strength, the border nodes, with shorter transmission defer time, rebroadcast the message earlier than other receivers. An angle-based scheme is then used to eliminate redundant retransmissions. Dominant sets based protocols [35] aim at high reliability and low rebroadcast traffic. Nevertheless, they require each node to have the knowledge of two-hop neighbors or the accurate location information of one-hop neighbors. Utilizing efficient broadcast schemes to spread RREQ packets can greatly reduce route discovery overhead. However, as a lot of nodes lying between sources and destinations are excluded from the route discovery process, the optimal routes are very likely ignored.

2.3 Progressive Route Discovery Protocol

Progressive route discovery protocol consists of two stages: network scan and the optimal route discovery. The following sections specify the operations.

2.3.1 Network Scan

The purpose of network scan is to sketchily explore a network and derive a preliminary route between a source-destination pair with small routing overhead. Three network scan approaches are presented to achieve the purpose. Among them, OLSR-

based routing and tree-based routing approaches are proactive network scan approaches and efficient broadcast approaches are reactive network scan approaches.

2.3.1.1 Proactive Network Scan — OLSR-based Routing

In the OLSR-based routing approach, OLSR [6] is utilized to find preliminary routes. Based on network topology information obtained from OLSR topology control packets, each node derives a shortest-path spanning tree, by which it can reach any other node. When an active source wants to build the optimal route to a desired destination, it may utilize the shortest route derived by OLSR as a preliminary route, and initiate the optimal route discovery process based on the preliminary route. For example, in Figure 2-1, node S derives a shortest-path spanning tree based on the topology control information it obtains. The shortest route between node S and its desired destination D is employed as the preliminary route between the source-destination pair.

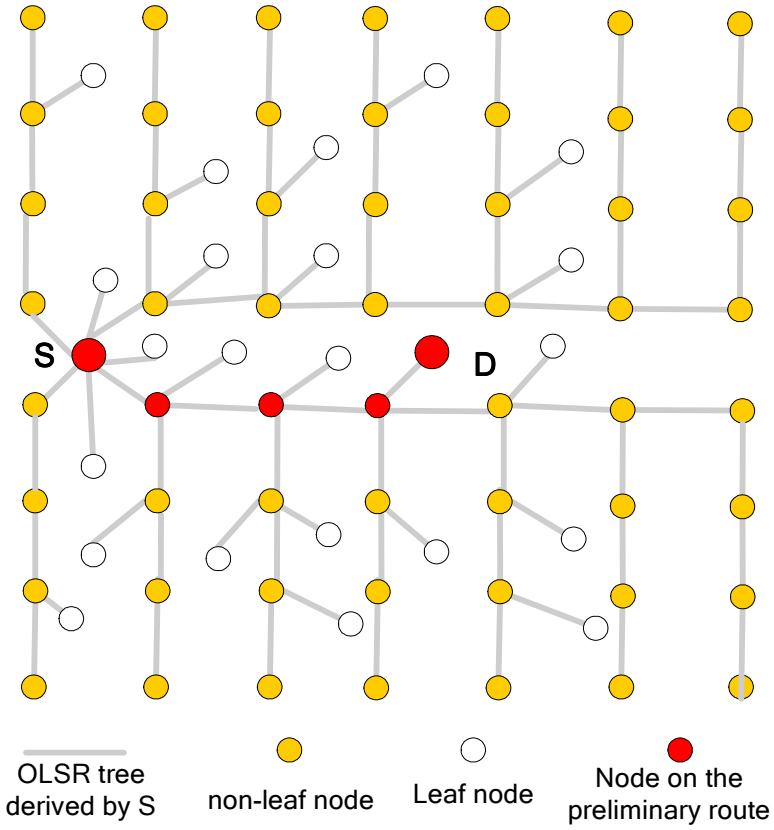


Figure 2-1. OLSR-based preliminary route scan

2.3.1.2 Proactive Network Scan — Tree-based Routing

Preliminary routes can also be established by using a centralized approach. In this approach, all nodes in a network are self-organized as a spanning tree [7]. This spanning tree is rooted at a network portal and extended to cover all other nodes, as shown in Figure 2-2. Each node on the spanning tree maintains routing entries or calculates routes [38] for the tree root as well as all descendants of the tree. Therefore, each node can reach any other node by using a tree route. Based on the tree structure, a topology server (TS) is then established to maintain the network skeleton. More specifically, TS first broadcasts its address to all other nodes in order to announce its

existence. Each node, upon receiving the announcement, reports its OLSR-based link states (i.e. the link states between it and its MPR selectors) to TS along a tree route. TS, after collecting the OLSR-based link states from all nodes, is able to derive a shortest path between any node pair (by using a shortest-path selection algorithm). When an active source wants to build the optimal route to a desired destination, it sends a route query packet to TS along the tree route. Upon receiving the route query packet, TS derives the shortest path between the active source and its desired destination (TS may also calculate the shortest path between any node pair in advance). TS includes a list of nodes on the shortest path in a route report packet, and sends the packet to the source along a tree route. The source employs the shortest path as the preliminary route and initiates the optimal route calculation based on the preliminary route.

2.3.1.3 Reactive Network Scan — Efficient Broadcast based Routing

In reactive network scan, an active source initiates route construction by first broadcasting a Destination LOCating (DLOC) Packet. Every neighbor of the source, upon receiving the DLOC, rebroadcasts it with a probability p, or backoffs a short period t before rebroadcasting it. There are many ways to choose p and t. For example, in [16], the value of p is related to the number of common neighbors between the relay node and the source. A distance-based defer-time scheme is proposed in [17] to set the t inversely proportional to the signal strength of the received packet. The nodes rebroadcasting the DLOC earlier can suppress the redundant rebroadcasts from other nodes [15]. After a node rebroadcasts the DLOC, its neighbors follow the

same efficient broadcast rules to further relay this packet. We use the scheme of [17] in the following descriptions and simulations.

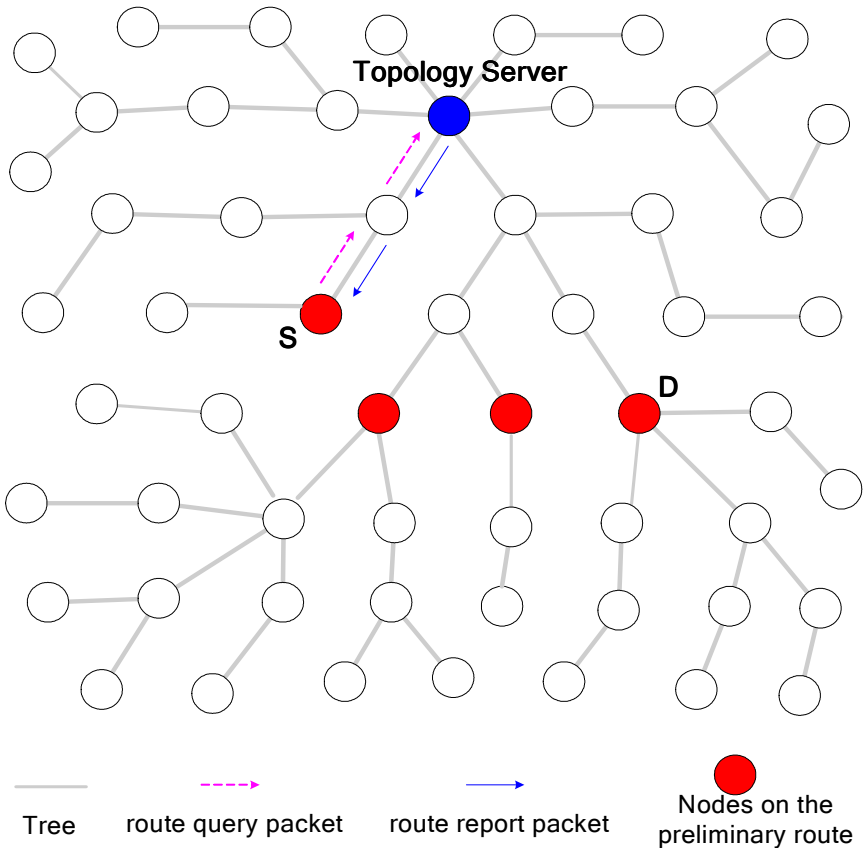


Figure 2-2 Tree-based preliminary route scan

We define nodes lying in the transmission boundary of the source as first-tier borderers, and the nodes lying in the transmission boundary of the first-tier borderers are called second-tier borderers, and so on. As shown in Figure 2-3, node S is source, then node A, F, G, H are first-tier borderers, and node B, K, L are second-tier borderers. According to the efficient broadcast rules in [17], borderers have the shortest backoff delays, so they always rebroadcast at the earliest time and suppress the

redundant rebroadcasts from other nodes. It is easy to see that, in a dense network, only a fraction of nodes really participate in the relays of the DLOC.

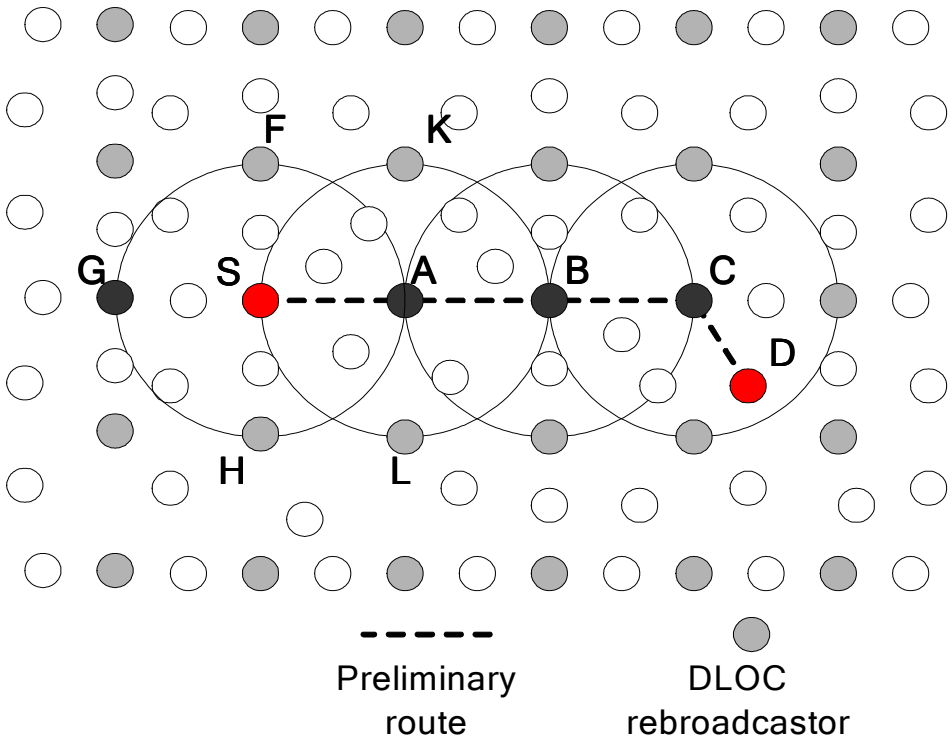


Figure 2-3 Efficient broadcasting of the DLOC

The nodes relaying the DLOC have to keep a record of the previous-hop[1] neighbors who send them the packet. So that when the DLOC arrives at the destination, a preliminary route is immediately ready in the backward direction (from destination to source). Multiple DLOCs may arrive at the destination from different routes. The destination selects the shortest route among them and names it as “preliminary route”.

[1] In this thesis, “previous-hop” means direction toward the source, “next-hop” means direction toward the destination

Like AODV, a ring-search technique can be used here to gradually enlarge the network exploration range until the destination is reached.

2.3.2 The Optimal Route Discovery

If hop count is considered as the criterion of route cost, the preliminary route is already a very good route, as shown in Figure 2-3. However, in wireless networks, the shortest route in many cases is no longer the optimal one. For example, although the preliminary route has small hop count, the distance between two neighboring nodes in the route tends to be large. Thereby, the link qualities along the route may become very poor and unstable. Also, some nodes in the preliminary route may have very low battery levels and cannot support data traffic for long time. In order to find the route with the best channels and/or power conditions, more nodes, especially the ones located between the source-destination pair, have to participate in the route discovery process.

Although the preliminary route may not be the best route as viewed from link quality and energy efficiency, it is a good direction indicator for further route discovery. Figure 2-4 shows that the vicinage of the preliminary route covers most nodes sitting between the source and the destination. Therefore, the optimal route or near-optimal routes very likely lie in this area. In the following section, two reactive approaches, Region-based Routing algorithm (REGR) and progressive route calculation algorithm (PROC) are presented separately to find the optimal route in the neighborhood of the preliminary route to limit the route exploration only inside this small area.

2.3.2.1 Region-based Routing Algorithm

The region-based routing algorithm, or REGR consists of region formation, and in-region route discovery. Figure 2-4 illustrates the operations of REGR.

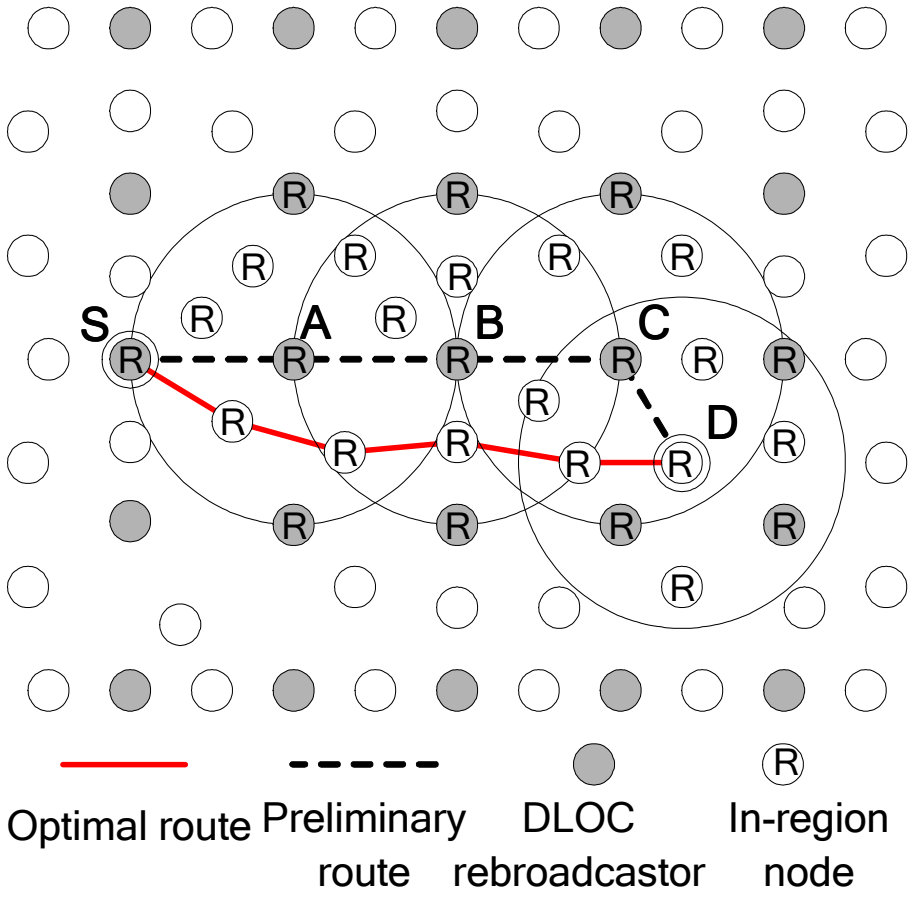


Figure 2-4 REGR protocol Illustration

A. Region Formation

After choosing a preliminary route, the destination broadcasts a Region DEFinition (RDEF) packet to its neighbors. The RDEF contains a "previous-hop-address" field to carry the address of the previous-hop neighbor in the preliminary

route. There is also a REGION_WIDTH value in the RDEF to name the size of the pre-routing region. When the RDEF starts from the destination, its initial Time-to-Live (TTL) equals to REGION_WIDTH.

All nodes receiving the RDEF mark themselves as “in-region” nodes. From the “previous-hop-address” field in the RDEF, the nodes learn whether they belong to the preliminary route or not. The nodes in the preliminary route update the “previous-hop-address” field of the RDEF, reset its TTL to REGION_WIDTH and rebroadcast it. Nodes not in the preliminary route firstly reduce the RDEF TTL by one and rebroadcast it when the deducted TTL is still larger than zero.

When the RDEF propagates to the source, a pre-routing region is also established in the neighborhood of the preliminary route. In the example of Figure 2-4, as REGION_WIDTH equals to one, the pre-routing region only includes the members of the preliminary route as well as their one-hop neighbors. The size of the pre-routing region can be enlarged by increasing REGION_WIDTH.

B. In-region Route Discovery

The destination broadcasts a RREQ packet a short period after it releases the RDEF. The delay between the transmissions of these two packets should be large enough to avoid packet collisions caused by hidden terminals. Only nodes with “in-region” mark participate in the rebroadcast of the RREQ. As shown in Figure 2-4.

In AODV, intermediate nodes relay only the first received RREQ’s. This may be good enough for hop-count based routing. However, if other route costs, such as link

quality and power conditions, are considered, the first received RREQ not necessarily comes from a good route. REGR makes intermediate nodes relay the duplicated RREQ's if they are transmitted from better routes. So the optimal route within the region can always be detected. At the same time, we do not increase much control overhead since the route discovery activities have already been limited to a very small portion of nodes.

Among the routes discovered by the RREQ's, the source chooses the best one and begins to forward data along the route. This completes the optimal route discovery process.

In the simulations, we use a Minimum Battery Cost Routing (MBCR) scheme [40] for in-region route discovery. Basically, each in-region node adds the reciprocal of its residual battery capacity as its node cost to the RREQ's, and the source selects the route with the minimum summation of node costs. Routing with battery level metric enables the efficient utilization of network energy resource, therefore prolonging network lifetime.

2.3.2.2 Progressive Route Calculation Algorithm

The progressive route calculation algorithm, or PROC consists of route calculation and route formation processes.

A. Route Calculation

Traditional link state routing protocols [3] assume that each node in a network maintains the weighted and connected graph information of the whole network; and thus, every node can derive a minimum cost spanning tree (MCST), by using Dijkstra's algorithm [39], to reach every other node in the network. In PROC, we assume that each node is aware of its complete local topology information. Hence, a single node can derive its local MCST. In order to build an MCST extended from a source to a desired destination, we employ a progressive Dijkstra's algorithm, which enables the nodes on a preliminary route to "relay" the MCST calculation hop-by-hop. The nodes on the preliminary route are called "anchors" hereafter. We assume that each node knows its own link state and the link state of those of one-hop neighbors as well. In a sparse network, each node can collect three-hop or even more topology information to increase the probability of hitting the optimal route.

Once the source selects the preliminary route, it initiates the route calculation process by applying Dijkstra's algorithm [39] to its one-hop and two-hop neighbors, which are called (1+2)-neighbors[2] for simplicity. The knowledge of its own link state and its neighbors' link states enables the source to derive an MCST covering its (1+2)-neighbors. The MCST specifies the "path-cost" and the "upstream[3] node" of each (1+2)-neighbor. Here, the "path-cost" means the cost from a node to the source and the "upstream node" indicates the node's next-hop node toward the source. Then, the

[2] For the sake of notation, we define a node that can be directly reached by node *k* as a *1-neighbor* of node *k*; and a node that cannot be directly reached by node *k*, but can be directly reached by any 1-neighbor of node *k* as a *2-neighbor* of node *k*. Any 1-neighbor or 2-neighbor of node *k* is called as a *(1+2)-neighbor* of node *k*.

[3] In this thesis, *downstream (upstream)* represents the direction towards the destination (source).

source delivers the path-costs of its (1+2)-neighbors to its downstream anchor via a Tree REPort (TREP) packet. When the downstream anchor receives the TREP, it also applies Dijkstra's algorithm to its (1+2)-neighbors, thus extending the MCST from the source's neighbors to its own neighbors. Similarly, it sends the path-costs of its (1+2)-neighbors to the next downstream anchor via a TREP. The MCST calculation and extension process is relayed hop-by-hop along the preliminary route until the MCST finally reaches the destination.

Figure 2-5 elaborates on the progressive Dijkstra's algorithm, and compares it with the original Dijkstra's algorithm. In the original Dijkstra's algorithm, the route deriver is assumed to know the complete information of the whole network topology. At the initialization stage, all nodes except the route starting-point are added to an unsettled node queue (UNNQ). Nodes are removed from the UNNQ one by one based on their path-costs. The newly removed node needs to update the path-costs of its neighbors that still stay in the UNNQ.

In the progressive Dijkstra's algorithm, the route derivers, i.e. the anchors, are aware of only local topology information completely. In order to derive the optimal route around the preliminary route, the anchors need to share their knowledge of local topologies and coordinate their route calculation efforts. As shown in Figure 2-5, the route calculation begins from the route starting-point (source) and continues on the anchors on the preliminary route. Each anchor adds its (1+2)-neighbors to the UNNQ, and tries to empty the UNNQ by finding the least path-costs for the UNNQ members.

Dijkstra's algorithm vs. Progressive Dijkstra's algorithm

(The grey boxes show the major differences between the two algorithms)

Objective: find an optimal path from vertex *S* to vertex *D* in an undirected graph $G = (V, E)$.

Definitions:

V: the set of all vertices in the graph *G*
E: the set of all edges in the graph *G*
$N_k,\ k \in V$: the set of 1-neighbors of vertex *k*
$c(k),\ k \in V$: the path-cost from vertex *k* to vertex *S*
$l(i, j),\ i, j \in V$: the link cost between vertex *i* and vertex *j*
$p(k),\ k \in V$: the next upstream router in the path from *k* to *S*
Q : the unsettled node queue (UNNQ)

P: the set of anchors on the preliminary route
$M_k,\ k \in V$: the set of (1+2)-neighbors of vertex *k,* plus vertex *k*
$\Lambda_{down}(k),\ k \in P$: the downstream anchor of vertex *k* on the preliminary route
deriver: the anchor that is currently executing Dijkstra'a algorithm

Dijsktra's Algorithm:

1) Initialization:

$c(S) = 0$;

$$c(k) = \begin{cases} l(S,k), & k \in N_S \\ \infty, & otherwise \end{cases}$$

$Q = V - \{S\}$;

$p(k) = unset,\ \forall\ k \in Q$

2) Find $j \in Q$, such that

$c(j) = \min c(k),\ k \in Q$

3) $Q = Q - \{j\}$

4) If $j \neq D$, go to 5). Otherwise END.

5) $\forall k \in N_j$, and $k \in Q$,

if $c(k) > c(j) + l(j,k)$

$c(k) = c(j) + l(j,k)$;

$p(k) = j$;

go back to 2).

Progressive Dijkstra's Algorithm:

1) Initialization:

$c(S) = 0$;

$$c(k) = \begin{cases} l(S,k), & k \in N_S \\ \infty, & otherwise \end{cases}$$

$deriver = S$

$Q = M_{deriver} - \{S\}$

Apply Dijkstra's algorithm to M_S

$p(k) = unset,\ \forall\ k \in Q$

2) Find $j \in Q$, such that

$c(j) = \min c(k),\ k \in Q$

3) $Q = Q - \{j\}$

The derived path-costs are forwarded to the downstream anchor to help it continue Dijkstra's calculation

4) If $j \neq D$ or $deriver \neq D$, goto 5); otherwise END

5) If $Q \neq \varnothing$, go to 6). Otherwise,

send all $c(k), k \in M_{deriver}$ to $\Lambda_{down}(deriver)$

$deriver = \Lambda_{down}(deriver)$;

$Q = M_{deriver}$, $p(k) = unset,\ \forall\ k \in Q$

6) $\forall k \in N_j$, and $k \in Q$,

if $c(k) > c(j) + l(j,k)$

$c(k) = c(j) + l(j,k)$;

$p(k) = j$;

go back to 2).

Figure 2-5 Comparison of the original Dijkstra's algorithm and its progressive version

The derived path-costs are then forwarded to the downstream anchor to help it continue the calculation.

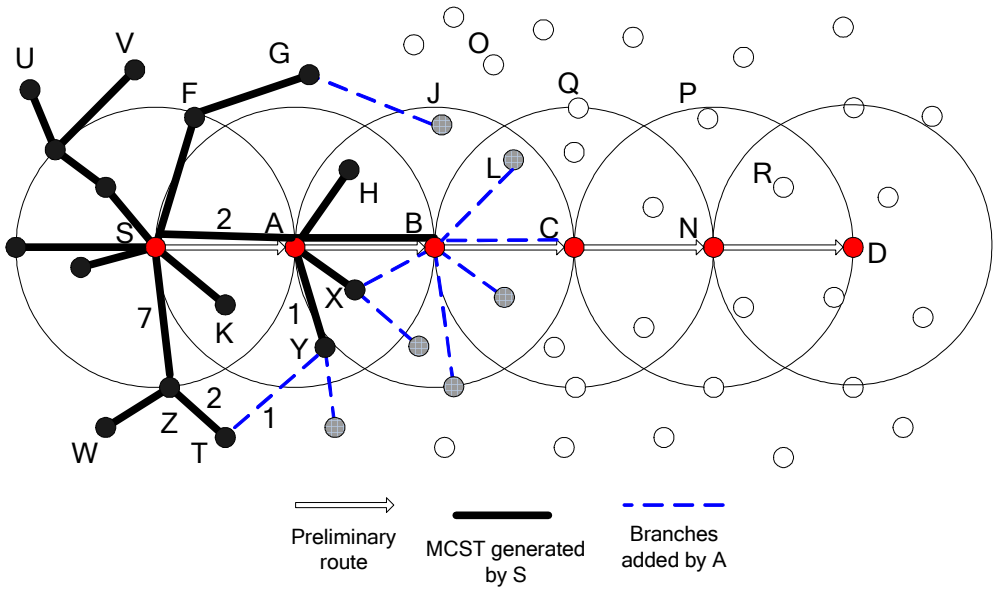


Figure 2-6 An example of the progressive Dijkstra's algorithm

Figure 2-6 illustrates the progressive Dijkstra's algorithm. Suppose **S-A-B-C-N-D** is the preliminary route between the source **S** and the destination **D**. **S** initiates the route calculation process by applying Dijkstra's algorithm to its (1+2)-neighbors. Dijkstra's algorithm enables **S** to build an MCST covering all its (1+2)-neighbors. In order to help node **A** continue the MCST calculation, **S** forwards the path-costs of the MCST members to **A** via a TREP. Some of **A**'s (1+2)-neighbors are already in the MCST, while the others are not (such as **J, L, C** and other grey nodes). **A** also runs Dijkstra's algorithm to add its (1+2)-neighbors to the MCST. As **A** is clearer about its neighbors than **S** is, **A** is able to correct some path-costs derived by **S**. For instance, the link cost

of **SZ** and **ZT** are 7 and 2 respectively. Due to limited topology information, **S** calculates the path-costs of **T** as 9 and the upstream node of **T** as **Z**. When **A** participates in the calculation, as it knows that the link cost of **YT** is 1, it finds a better path (**S-A-Y-T**) for **T**. So it corrects the path-cost of **T** as 4, and revises the upstream node of **T** from **Z** to **Y**. After **A** finishes the calculations, it delivers the path-costs of its (1+2)-neighbors to **B** via a TREP. Figure 2-7 shows the new branches added by different anchors. Some branches are added by one upstream anchor and later on corrected by another downstream anchor. Figure 2-8 removes those flawed branches, and shows the final MCST. Note that in this example, the preliminary route is not on the MCST.

B. Route Formation

After the MCST is extended from the source to the destination, the destination selects a branch of the MCST with the lowest path-cost as the desired route. Since the destination knows only a segment of the branch, which falls into its two-hop region, it seeks helps from other anchors to collect all pieces of the branch. To build the optimal route, the destination creates a Route Assembling (RASB) packet, and records in the RASB the branch segment it knows. For example, the destination D in Figure 2-8 selects a branch D-R, …, S as the optimal route and records D-R-P in a RASB.

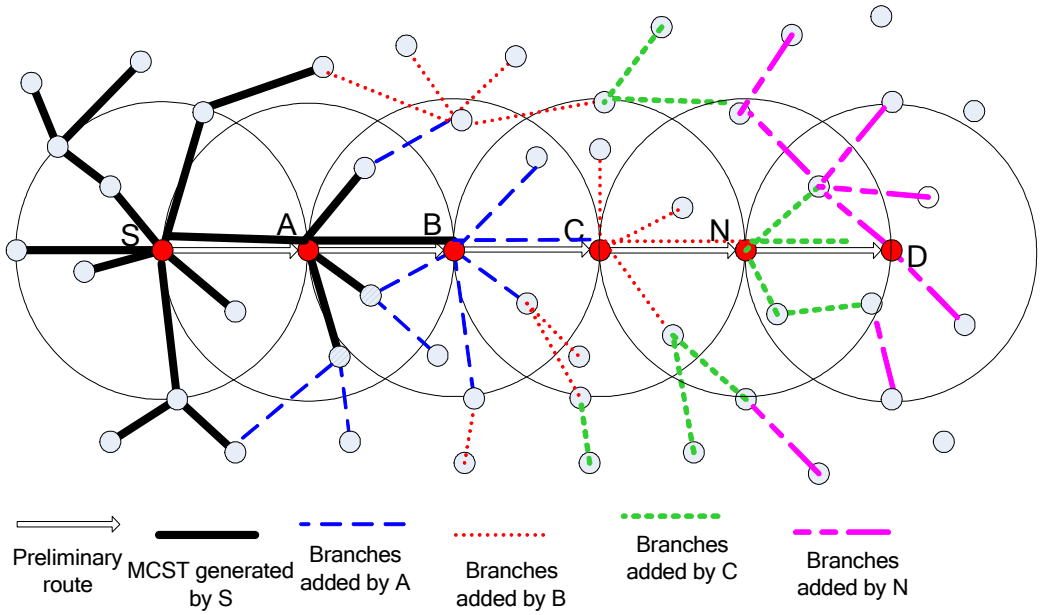


Figure 2-7 MCST establishment and extension

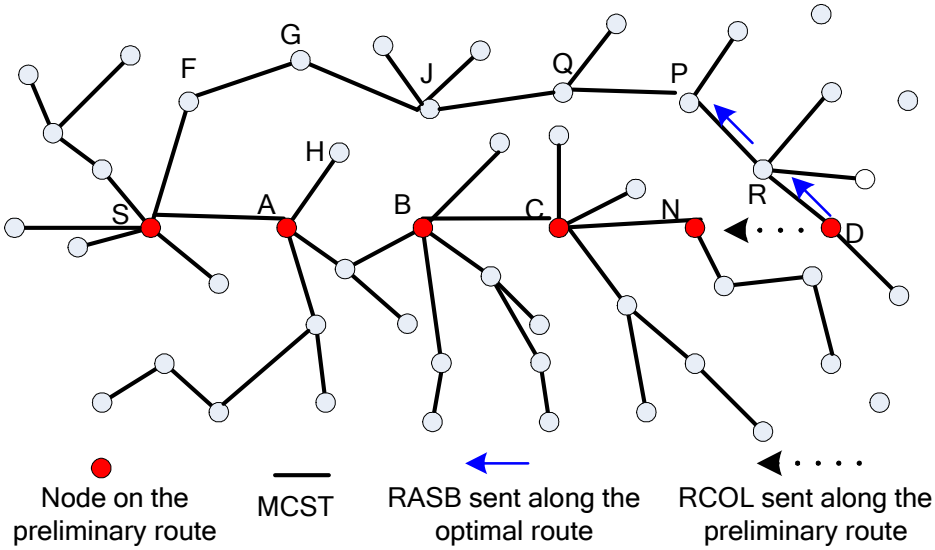


Figure 2-8 Route Formation: illustration 1

After the destination creates a RASB, it forwards the packet along the branch segment to the nodes on the optimal route. And the nodes on the optimal route create corresponding routing entries to the source and the destination according to the branch segment information recorded in the RASB. In Figure 2-8, D sends the RASB to R. R forwards the packet to P. Based on the route list in the RASB, D and R set the next-hop node to the source S as R and P respectively; while R and P set the next-hop node to D as D and R respectively. The destination then sends a Route COLlection (RCOL) packet to its upstream anchor, so that the upstream anchor can continue the route collection and assembling process. The RCOL contains the same route information as the RASB. In Figure 2-8, D sends a RCOL to its upstream anchor N.

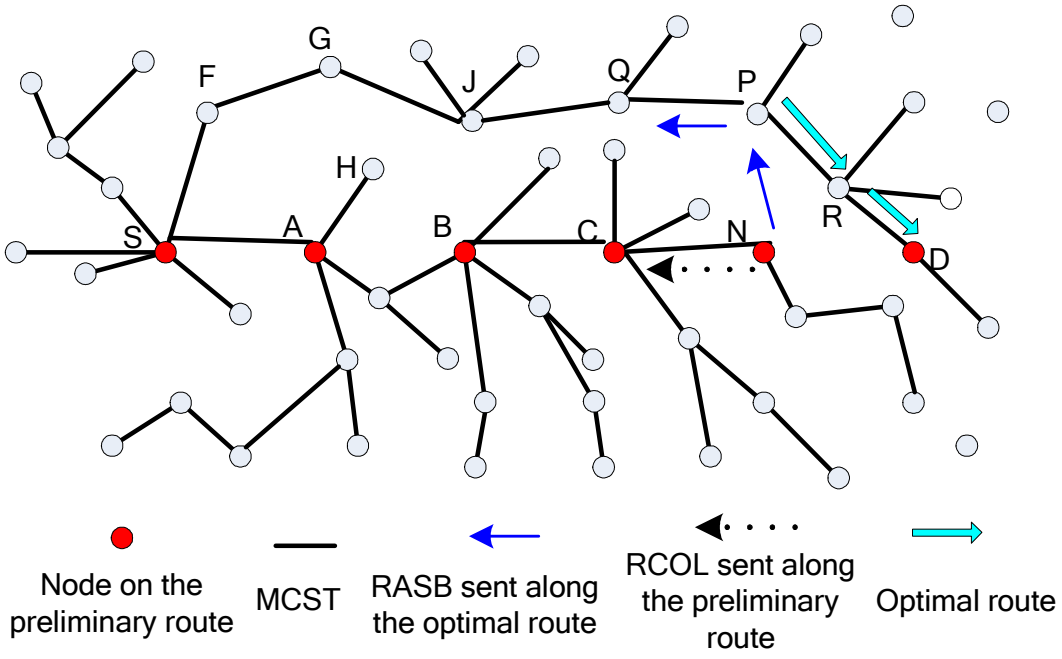


Figure 2-9 Route Formation: illustration 2

When the upstream anchor receives a RCOL from the destination, it figures out the branch segment falling into its own (1+2)-neighbors, appends the branch segment it

knows in the received RCOL, and copies the content of RCOL to a new RASB. It then sends the newly created RASB to the corresponding branch segment on the optimal route. Based on the route information in the RASB, the nodes on the segment of the optimal route create their routing entries. Note that the RASB is sent to the branch segment via a local route (each anchor knows a local route towards any of its (1+2)-neighbor). In Figure 2-9, the upstream anchor N updates the route list in the RCOL as D-R-P-Q, copies the route information to a new RASB, and sends the RASB to P, so that P can forward the packet along the branch segment to Q. Based on information in the RASB, P set the next-hop node to the source S as Q; while P and Q set the next-hop node to D as R and P respectively. Beside the RASB, the upstream anchor also sends the RCOL to its upstream anchor C. This route collection and assembling process continues until it stops at the source. Figure 2-10 shows the optimal route formed after route collection and assembling process.

C. Algorithm Improvement

In the progressive MCST calculation process, after an anchor applies Dijkstra's algorithm to its (1+2)-neighbors, it sends their path-costs to its downstream anchor. However, the path-costs of its 1-neighbors are actually useless to the downstream anchor since the downstream anchor knows the path-costs of all 2-neighbors that are extended from these 1-neighbors. The downstream anchor can continue extending the MCST solely based on the path-costs of the 2-neighbors (of the upstream anchor).

As shown in Figure 2-11, the local MCST built by node **S** covers its 1-neighnors as well as all 2-neigbors that extended from these 1-neighbors. When node **A** extends

the MCST to its neighborhood, it only attaches new branches to the 2-neighbors of node **S**. Therefore, node **A** only needs to know the path-costs of **S**'s 2-neighbors.

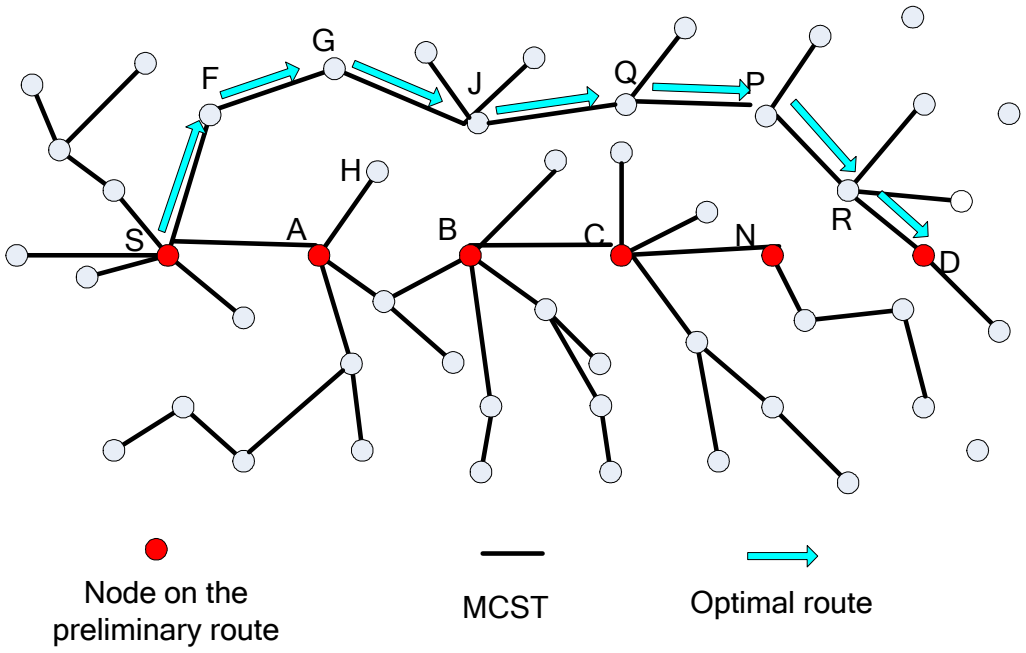


Figure 2-10 Route Formation: illustration 3

Based on this understanding, we can improve PROC by asking each anchor to include only the path-costs of 2-neighbors in the TREP. This improvement further reduces the route discovery overhead, while maintains the same route optimality as the original PROC.

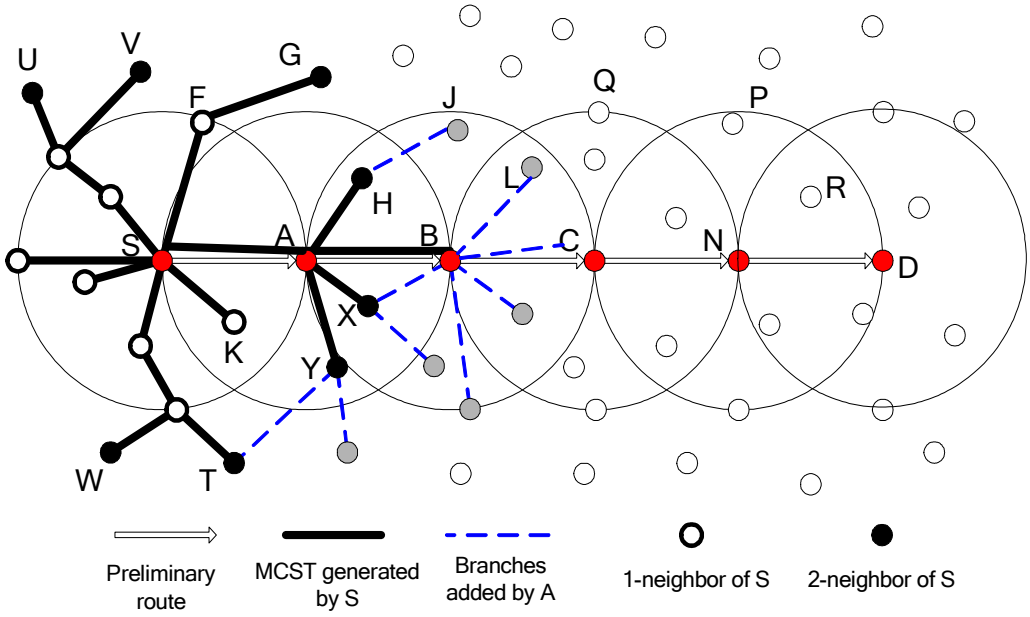


Figure 2-11 PROC Improvement

2.4 Performance Comparison

A. Performance Comparison of Network Scan

There is a tradeoff between the OLSR-based preliminary route calculation approach and the centralized preliminary route calculation approach. In the OLSR-based approach, each node broadcasts topology control packets to all other nodes in the whole network. While in the centralized tree-based approach, each node only unicasts topology control packets to TS. Obviously, the latter approach engenders less topology control overhead and causes less packet collisions than the former approach. On the other hand, the centralized approach has a single-point-of-failure problem, while the OLSR-based approach, as a totally distributed approach, does not have such a problem. The single-point-of-failure problem can be solved by deploying backup TSs, which is

out of the scope of this thesis. Another difference between these two approaches is that in the OLSR-based approach, each node maintains network-wide topology information, and derives the shortest route (preliminary route) between itself and all other nodes; while in the centralized approach, only TS takes the burden to maintain network topology information and derive the shortest routes (preliminary routes) between any source-destination pair.

B. Performance Comparison of Route Optimality

In the route calculation and route formation processes of optimal route calculation algorithm, all control packets, such as TREP, RASB, and RREP, are transmitted by unicasting, which makes the signaling very reliable. Also, an overhearing avoided scheme [41] is applied to the transmissions of long unicast packets. Specifically, a packet source and destination use RTS/CTS mechanism to reserve the wireless channel before the packet is transmitted. The neighbors of the packet source and destination, when receiving the RTS and/or CTS, turn off their transceivers until the packet transmission is completed. The unicast approach, as well as the overhearing avoided scheme, makes PROC much more power-efficient than REGR.

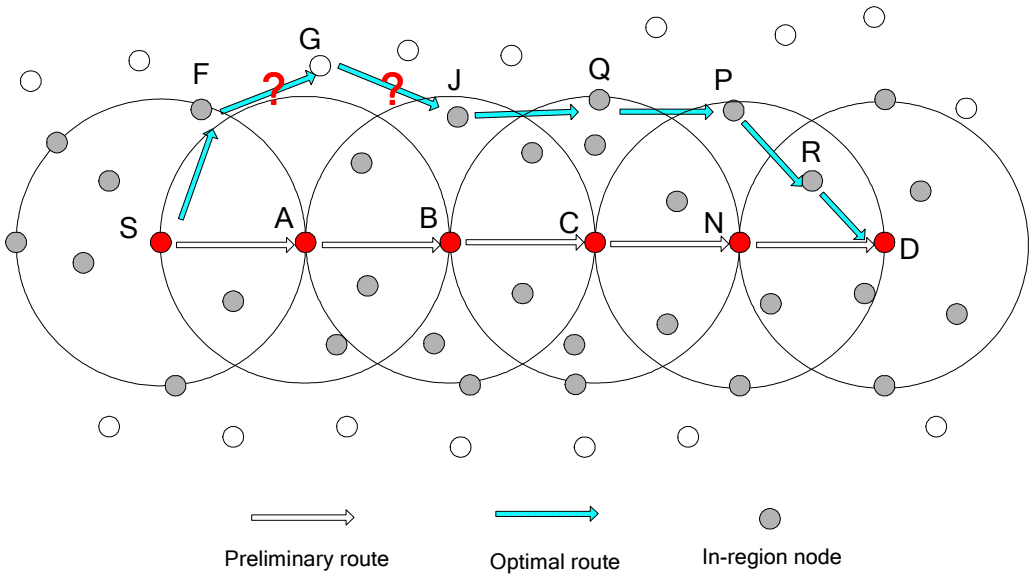


Figure 2-12 Minimum overhead cases: REGR vs. PROC

For REGR, the minimum overhead case is that the pre-routing region covers one-hop neighborhood of the preliminary route. For optimal route calculation algorithm, each anchor has to include at least (1+2)-neighbors in MCST calculations. This is the minimum requirement case, and also the minimum overhead case. Comparing these two minimum overhead cases, optimal route calculation algorithm has more route choices than REGR. This can be explained by the example of Figure 2-12. In this example, node G is not an “in-region” node; thereby REGR cannot detect any route that passes node G. In optimal route calculation algorithm, as node G is a 2-neighbor of anchor S, A, and B. All these anchors include node G in their MCST calculations. If the optimal route happens to be S-F-G-J-Q-P-R-D, as shown in Figure 2-12, PROC is able to detect the route. With more route choices and very reliable signaling mechanism, optimal route calculation algorithm achieves better route optimality than REGR in the

minimum overhead cases. As a tradeoff, each node in optimal route calculation algorithm has to have knowledge of (1+2)-neighbor nodes.

2.5 Simulation Results

We simulate progressive route discovery protocol in OPNET and compare its performances with both pure proactive routing protocols like OLSR, QOLSR and pure reactive routing protocols like the efficient-broadcast based routing protocol and the cost-based AODV. For PHY and MAC layers, we use IEEE 802.11 based WLAN model.

In our simulation, 100 wireless nodes are deployed in a 2000m × 2000m network. The transmission range of each node is 300m. The channel bandwidth is 1Mbps. The transmission, reception, and idling powers of each node are 0.66W, 0.395W and 0.035W respectively [42]. Each source sends out packets with a constant rate of 2 packets/second. The data packet size is 4096 bits. A link cost between any two neighbor nodes is randomly picked from [1, 2, 3, …, 7, ∞]. 1 represents a link cost with the highest quality and 7 represents a link cost with the lowest quality. ∞ means a link is broken. To study the influence of topology change on the performances of different routing approaches, we construct a network environment where link cost between any two neighboring nodes randomly changes with time. And to investigate the influence of the number of source-destination pairs, we design a traffic pattern in which a number of active source-destination pairs are randomly selected to transmit data. The starting time and the stopping time of these communication pairs are randomly determined as well.

Two performance metrics, i.e. route cost and route discovery overhead are chosen to address the performance issues discussed in the previous sections. The route cost is defined as the sum of the link costs of a certain route. The route discovery overhead calculates the total transmission, reception and idling powers that are consumed by nodes to establish new routes. REGR and PROC are evaluated under their minimum overhead cases. The influences of both topology change and the number of source-destination pairs on these performances are tested. Each test is repeated 5 times. The simulation time of each test is 650 seconds.

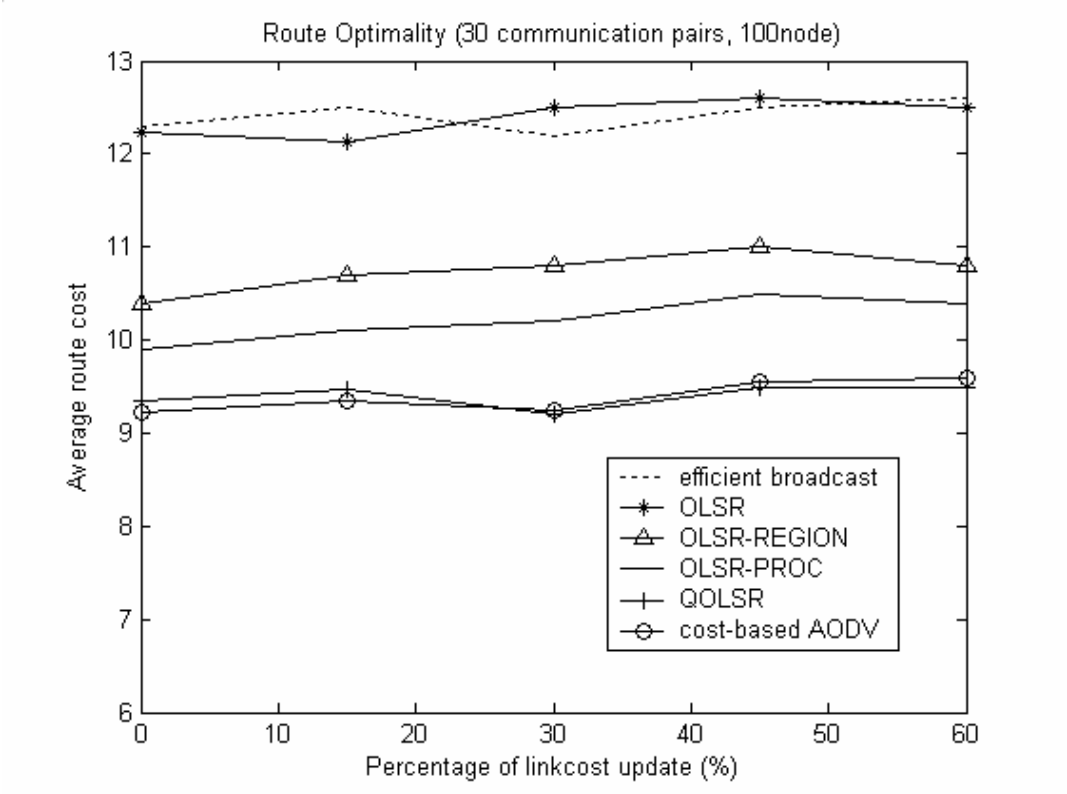
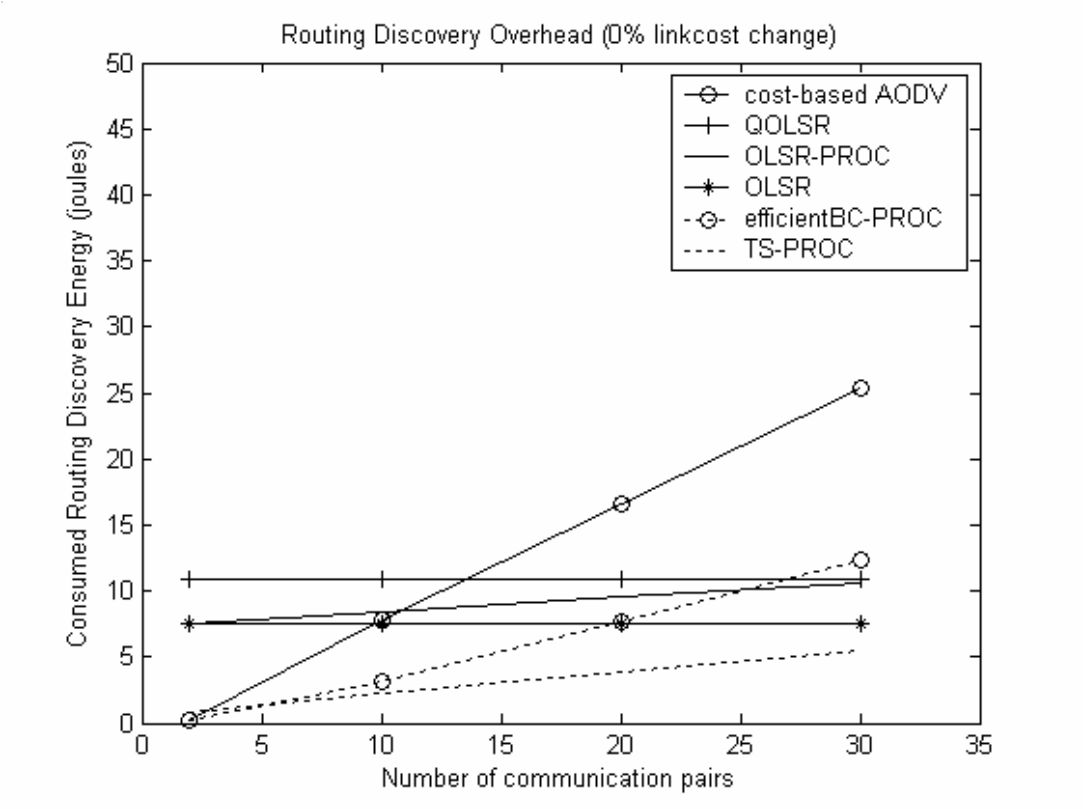


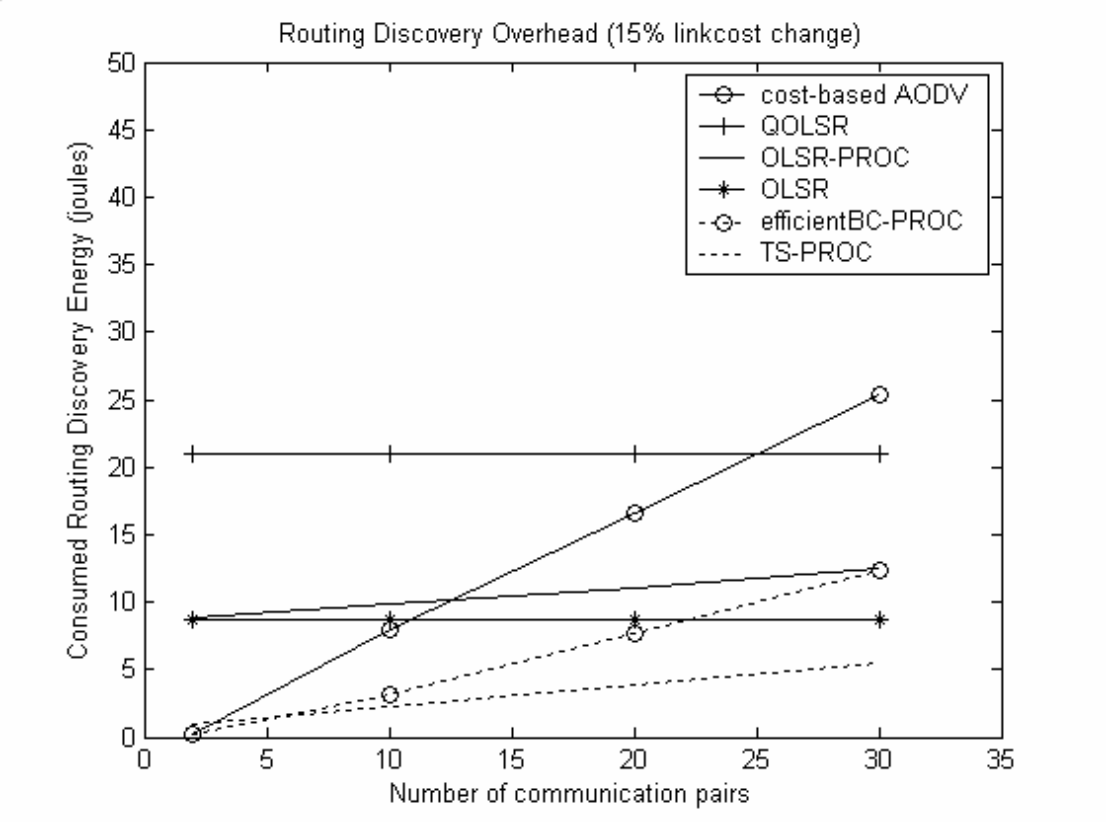
Figure 2-13. Route Optimality Comparison

Figure 2-13 compares route optimality of different routing protocols. The total number of active source-destination pairs is 30. The horizontal axis of the figure

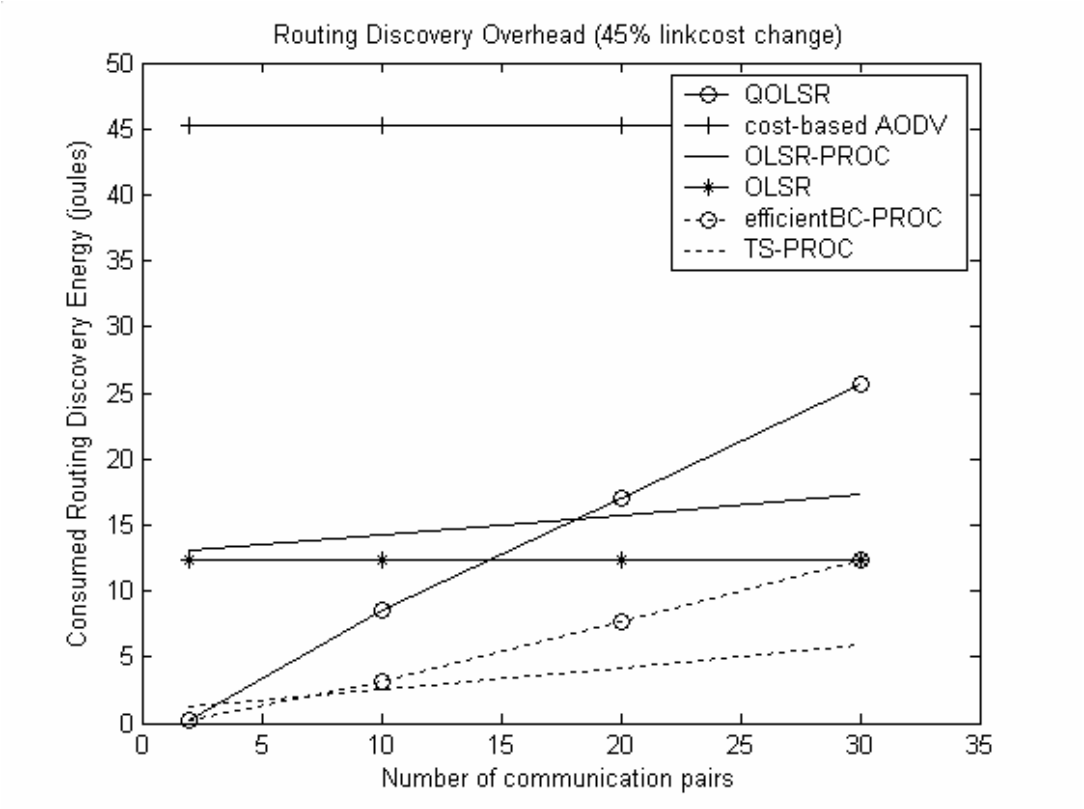
denotes the percentage of updated link cost in the whole network during simulation time. The vertical axis of it measures the average cost of the established routes. Figure 2-13 shows that due to the change of link cost, the average route cost is slightly fluctuating. Note, in particular, that the route cost of OLSR and the efficient broadcast based routing approaches are always much higher than the rest. The reason is that the heuristic of OLSR for the selection of MPRs is to ensure minimum topology control overhead instead of route cost. Hence, there is no guarantee that OLSR finds the optimal route in terms of QoS requirements. While in efficient broadcast based routing approaches, many nodes on the optimal routes are suppressed from relaying route request packets. On the other hand, the heuristic of QOLSR for the selection of MPRs guarantees that the routes with high QoS are always selected. And thus, the route cost of QOLSR is very low. The cost-based AODV finds routes with high quality as well. Since in this approach, each intermediate node in the network rebroadcasts redundant RREQ whenever the RREQ is from a route with less cost. OLSR-PROC can catch the local optimal route, which is also the global optimal route in many cases. And its route optimality is only a little behind QOLSR and the cost-based AODV scheme. The route built by OLSR-REGR, however, is not as good as OLSR-PROC because REGR has fewer chances to find the optimal route than PROC, as illustrated in Figure 2-12. Beside, the local optimal route of REGR is sometimes missed due to RREQ collisions within the small region. We do not show route costs of REGR and PROC whose preliminary routes are found with tree-based routing approach and efficient broadcast routing approaches since the results of them are very close to those of OLSR approaches.



(a) 0% link cost change



(b) 15% link cost change



(c) 45% link cost change

Figure 2-14. Route discovery cost Comparison

Figure 2-14(a-c) investigate influences of communication pairs and topology change on control overhead. The horizontal axes of these figures denote total number of source-destination pairs that transmit data during simulation time. The vertical axes of them represent the total power consumed by network nodes to set up routes, where the total power includes the total transmission power consumed by all nodes to send or relay the control packets and the total reception power consumed by all nodes to receive or overhear the control packets. Since the emphasis here is not power efficiency due to overhearing avoidance, which is discussed in chapter 2.4 (B), we do not apply overhearing avoidance scheme to PROC in these cases. And in order to focus on the

comparison between PRD and other one-step approaches, we do not show route discovery energy consumed by REGR in Figure 2-14, as the result of REGR is similar to that of PROC. We will discuss the energy consumption of REGR and PROC later on.

Figure 2-14(a) displays the effect of number of communication pairs on control overhead when the topology of the network keeps unchanged. It indicates that, first, the control overhead of QOLSR is larger than that of OLSR. It is because in QOLSR, to find routes with high quality, more information is included in topology control message and more nodes are selected to flood the message. Second, the slopes of control overhead for both OLSR and QOLSR do not go up with the number of communication pairs. As for the cost-based AODV, when the number of communication pairs is small, it shows predominance by generating much less control traffic than the proactive approaches; however, its control overhead increases quickly when more active source-destination pairs are involved in data transmission. It can be explained that the cost-based AODV uses network-wide reactive flooding and the overhead to flood the RREQ is proportional to the number of active communication pairs, yet the overhead of QOLSR and OLSR is proactively determined and it has nothing to do with the number of active communication pairs. Similar conclusions are made from Figure 2-14(b) in which the network topology changes comparatively infrequently (15% of link cost is updated) and from Figure 2-14(c) in which network topology changes comparatively frequently (45% of link cost is updated). Note that when topology changes, the difference between the overhead of QOLSR and OLSR increases. Figure 2-14 demonstrates proactive routing protocols are more beneficial for the networks where a

large amount of nodes communicate each other; while the reactive routing approaches like the cost-based AODV are more suitable for the scenarios in which the number of active source-destination pairs is small.

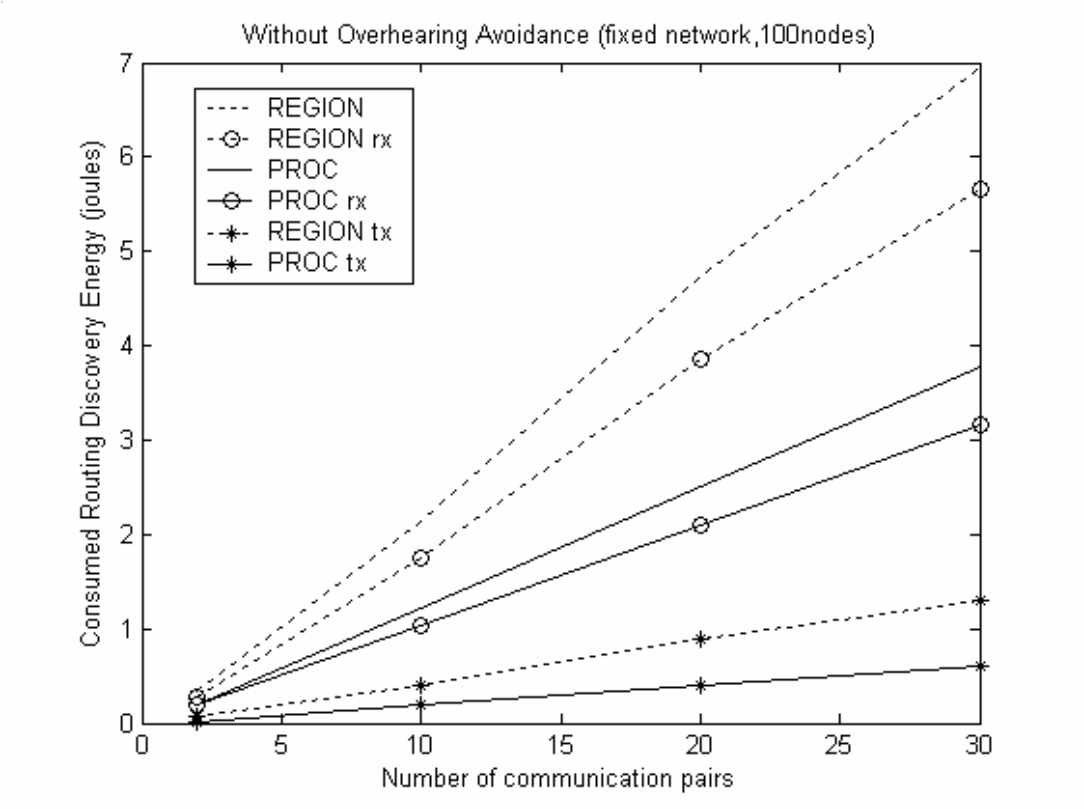
Now let us discuss OLSR-based PROC, a two-step progressive routing approach. Because the route calculation and formation process of OLSR-based PROC is based on the information obtained from OLSR, the total control overhead of the former is definitely more than that of the latter. Nevertheless, it takes only a little more overhead for OLSR-based PROC to build the optimal or near-optimal routes that are much superior to those set up by OLSR. From Figure 2-14 (a) to 2-14(c), we see that OLSR-based PROC integrates the advantages of both OLSR and efficient broadcast based reactive routing approaches. Compared with QOLSR, the control overhead of the OLSE-PROC is little when the number of communication pairs is small. And since the optimal route discovery stage of OLSR-based PROC is localized, the control overhead of it climbs up very slowly; therefore it is not as sensitive to the number of communication pair as the cost-based AODV. Figure 2-14 also shows that the control overhead of TS-based PROC is less than that of OLSR-based PROC. It can be explained that in the TS-based approach, each node sends topology control packets only to tree server; while in OLSR-based PROC, each node broadcasts topology control packets to all other nodes in the whole network. As the tradeoff, TS-based PROC may suffer from single-point-of-failure problem.

To study the effect of topology change on control overhead, let us compare the three figures in Figure 2-14. Our first observation is that the control overhead for both OLSR and QOLSR increases with dynamic networks, but the former goes up much slower than the later. The control overhead of OLSR and QOLSR consists of two parts: periodic topology updates, and dynamic topology updates due to the change of link state[1]. The update of link cost produces dynamic topology update packets, which is broadcast to the whole network and results in the increase of control overhead. In QOLSR, to maintain the best route information, whenever the cost of any link/node is changed (even if the link/node does not break), this change is broadcast to the whole network. While in OLSR, only link breaks or re-connection are considered as dynamic topology changes, which happen much less frequently than QOLSR. The difference between OLSR and QOLSR thereby increases. Our second observation is that, different from OLSR and QOSLR, the topology change has little impact on the control overhead of the cost-based AODV. The control overhead of the cost-based AODV consists of reactive route formation and reactive route repair overhead (As we noted earlier, periodic HELLO signaling is ignored here), and only the overhead due to route repair is relevant to topology change. Note that route repair results in localized flooding, which does not significantly increase routing overhead. Hence, topology change slightly influences the total control overhead of the cost-based AODV.

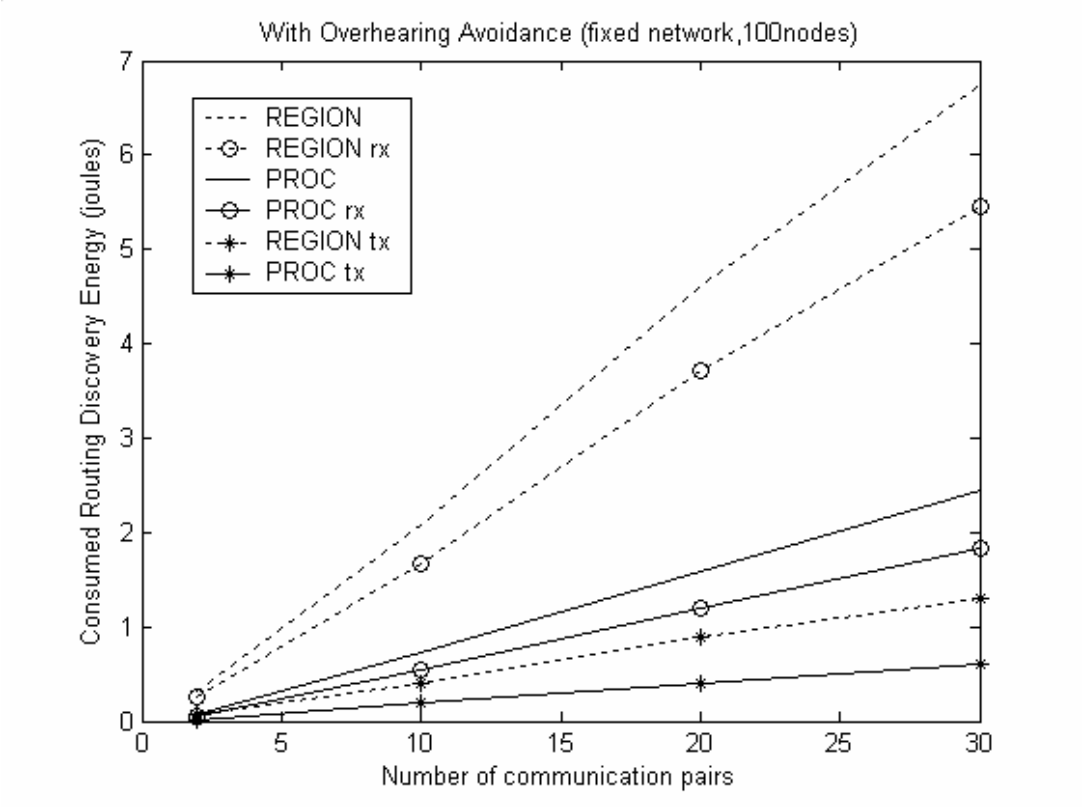
As for OLSR-based PROC, although the control overhead of it increases with topology change, similar to OLSR, the impact of topology change on it is much less than that on QOLSR. From analysis, we see that the control overhead of it consists of both network-wide scan and localized route refinement overhead, which calculates and

forms the optimal routes. The effect of topology change on OLSR-based PROC is limited because only when the link is broken or re-connected, topology control packets are flooded to re-construct preliminary routes, which is the same as in OLSR. On the other hand, the optimal route formation and repair process of the OLSR-based PROC approach is confined to a local region around a preliminary route, and thus the OLSR-based PROC is not as sensitive to topology change as the cost-based one-step proactive routing approaches like QOLSR. Consequently, the control overhead of it is a little more than that of OLSR and increases much slower than QOLSR when topology changes frequently. Similarly, TS-based PROC is affected by topology change less than QOLSR is.

In summary, the simulation results of Figure 2-14 demonstrate that one-step proactive routing approaches such as QOLSR are favored for the scenario of relatively fixed networks where a large number of nodes communicate each other and source-destination pairs change with time. In contrast, the one-step reactive routing approaches such as cost-based AODV work well in dynamically changing environment where only a few active communication pairs exist. Our progressive route discovery protocols, by gradually exploring the network topology, obtain the merits of both one-step routing approaches in most situations. Especially, it outperforms those one-step routing approaches when the topology changes frequently and substantial number of nodes communicate each other.


Without Overhearing Avoidance (fixed network,100nodes)
REGION
REGION rx
PROC
PROC rx
REGION tx
PROC tx
Consumed Routing Discovery Energy (joules)
Number of communication pairs
0
1
2
3
4
5
6
7
0
5
10
15
20
25
30

(a)



(b)

Figure 2-15. Influence of overhearing avoidance scheme on PROC and REGR

Figure 2-15 investigates the influence of overhearing avoidance scheme on REGR and PROC during route calculation and/or formation period. The horizontal axes of the figures denote the total number of source-destination pairs that transmit data during simulation time. The vertical axes represent power consumed by network nodes to calculate and/or establish new routes. The "tx cost" in the legend means the total transmission power consumed by all nodes to send or relay the control packets. The "rx cost" in the legend means the total reception power consumed by all nodes to receive or overhear the control packets. The "overall cost" is a sum of the transmission power and the reception power[1]. Figure 2-15(a) compares the power consumption of REGR

and PROC approaches when overhearing avoidance scheme is not applied. It is evident that the control overhead due to reactive routing is proportional to the number of active communication pairs. The figures manifest that overall reactive route discovery cost of the three protocols mostly results from packet-receiving and overhearing cost. It is because in REGR, each in-region RREQ broadcast is to be received and processed by all neighbors of the RREQ sender. In PROC, when an anchor unicasts a control packet to another anchor, all the neighbors of the packet sender and receivers unnecessarily waste their power to overhear the transmission activities.

Figure 2-15(b) compares the route discovery cost of two protocols when the overhearing avoidance scheme is adopted. We see that by making use of the overhearing avoidance scheme, the reception power of PROC falls off a lot, since only the desired receiver takes the burden to receive the unicast control packets. As a result, the overall power consumption significantly goes down. For example, with the overhearing avoidance scheme, the overall power of PROC decreases 30%. However, the overhearing avoidance scheme does not improve the REGR performance. It is because REGR employs broadcast scheme for route discovery; and overhearing avoidance scheme does not work for the RREQ broadcast scheme. The simulation results of Figure 2-15 support the analysis in chapter 2.4.

3 PROGRESSIVE ROUTE DISCOVERY — ROUTE UPDATE PROTOCOL

3.1 Introduction

In wireless mesh networks, the multihop routes between endpoints may suffer breakage or damage because of the movement, malfunction or power exhaustion of some nodes. Good route update schemes, therefore, become extremely important to keep a wireless mesh network running efficiently and with maximum possible longevity.

There have been several existing route repair and update protocols. In expanding ring search and local repair of Ad Hoc On Demand Distance Vector (AODV), the Time to Live (TTL) of the route request packet (RREQ) is set according to the length of the old route (if it is available). Therefore, the RREQ can be broadcast up to the neighborhood of the destination, but without going further. Essentially, AODV utilizes the length of the old route to estimate the distance of the destination, so that the RREQ broadcast range can be well confined. One problem of AODV approach is that the directional information provided by the old route is not used. RREQs are always

broadcast omni-directionally, but only RREQs propagate toward the destination are really needed. AODV local repair tries to minimize route recovery traffic, however, it sacrifices the optimization of new routes. This situation becomes worse when endpoints of one route keep moving, which makes the length of the route keep increasing. Non-optimal routes include more nodes in traffic than necessary, thus causing significant waste.

In [20], a Query Localization (QL) scheme is proposed to utilize the directional information implied in the old route for efficient route repair. QL puts a counter in each route query packet. The nodes along the old route rebroadcast each received query packet without increasing the counter value. All other nodes have to increase the counter by one before sending the packet out. When the counter in one query packet reaches a limit, this packet cannot be rebroadcast any more. If the counter limit is small, route query packets can only be relayed along the old route direction. Those broadcast to other directions quickly get their counters exceed the limit and stop propagation. In this way, an update route with most parts overlapping with the old route can be quickly found without much control overhead. QL is very efficient to locally repair sporadic node or link failures. However, it compromises route optimization. The route prolonging problem appearing in AODV local repair case also happens here. Another problem about QL is that it is very hard to use QL to discover a new route that is mostly disjoined with the old one. In sensor networks, when one route is used for a long time, most nodes along the route wear out; Then, a new fresh route should be found to replace the old one so that the whole network can maintain good energy-

balancing and survive longer. Apparently, QL is not good at this kind of route-wide update.

In this chapter, we propose a region-based route update protocol that reduces route rediscovery traffic and at the same time achieve good route optimization. The region-based route update protocol is similar to region-based route discovery protocol except that the former can skip the destination discovery stage since the old route is a good "preliminary route". More specifically, each node along the old route broadcasts a region packet and define the route update region; and then, a RREQ packet is released from the source or the upstream node of the broken link/node to discover new routes to the destination. Only the nodes within the route update region can rebroadcast the RREQ. In this way, route update activities are nicely confined to a narrow strip exactly covering the old route, which makes the route discovery overhead very small. Also, under the region-based route update protocol, either a new route mostly overlapping with the old route or a new route mostly separated from the old route can be found depending on the definition of the route cost function. This is different from QL. Given more freedom in route discovery and selection, our region-based route update protocol is able to set up new routes with better performance.

The rest of the chapter is organized as follows: in the next section, the basic operations of the region-based route update protocol is presented; after that, simulation results are shown to demonstrate the rationality and efficiency of region-based route update protocol; lastly, we conclude the chapter.

3.2 Region-based Route Update Protocol

We name the initiator of the route update process as route-update source (RU-SRC). The destination of the route update packets is called route-update destination (RU-DEST). We assume that the old route was the optimal route or at least a near-optimal route between the source and the destination. In the mobile endpoint case, the source or destination is assumed to leave the old route gradually. So the old route is always a proper direction indicator for route rebuilding.

The route update process can skip the destination discovery stage since the old route is a good "preliminary route". A pre-routing region can be directly established in the neighborhood of the old route to confine the route update activities to the local area.

A. Formation of Pre-routing Region

The RU-SRC initiates the route update process by broadcasting an RDEF packet. Since the RU-SRC may not be able to contact its next-hop neighbor in the old route, it sets the initial TTL of the RDEF equal to max [2, REGION_WIDTH].

Similar to the route creation process, nodes along the old route resets the RDEF TTL to REGION_WIDTH, so that the RDEF can be broadcast REGION_WIDTH hops away from the old route. All nodes visited by the RDEF are marked as "in-region" nodes.

In the route creation process, the preliminary route is freshly built by the DLOC, thereby the RDEF can propagate smoothly through the preliminary route to reach the

source. The route update process, however, uses the old problematic route to transmit the RDEF. The low power nodes and the broken points of the route may block the propagation of the RDEF, especially when REGION_WIDTH is small. A simple way to solve this problem is to make the RU-SRC increase REGION_WIDTH and try again. As a tradeoff, the route update delay and route discovery overhead are increased. A more efficient way is to enlarge the pre-routing region only at the problematic spots. In order to achieve this purpose, every capable node in the old route starts a "route-clear" timer after it broadcasts the RDEF out. If its next-hop neighbor is able to relay the RDEF, it will quickly receive an RDEF_ACK from this neighbor. On the contrary, if an old route member does not receive the RDEF_ACK before its "route-clear" timer expires, it believes the RDEF is blocked in the next-hop neighbor. So it increases the RDEF TTL and tries once more. When the RDEF bypasses the broken point and reaches the old route again, an RDEF_ACK is sent back to this node to end its "route-clear" timer. The RDEF, after passing the bad spot, has its TTL set back to REGION_WIDTH.

Figure 3-1 shows a mobile source example. When the source S moves to a new place and loses contact with its next-hop neighbor A, it broadcasts an RDEF with REGION_WIDTH = 1, TTL = 2. At the same time, S starts a "route-clear" timer to wait for the RDEF_ACK. F relays the RDEF to A, and A, as a old route node, sends an RDEF_ACK back to S to confirm its ability to rebroadcast the RDEF. B receives the RDEF relayed by A and rebroadcasts it with TTL = REGION_WIDTH = 1, but since its next-hop neighbor C is in low-power status, the RDEF propagation is blocked. When B's "route-clear" timer expires, it tries another RDEF with TTL = 2. This time,

G can relay the RDEF to E, thus the RDEF successfully bypasses the broken point. E sends an RDEF_ACK to B through the reverse RDEF path E-G-B. When E broadcasts the RDEF, it sets the TTL of this packet back to REGION_WIDTH. So, the pre-routing region is only enlarged in the neighborhood of node C.

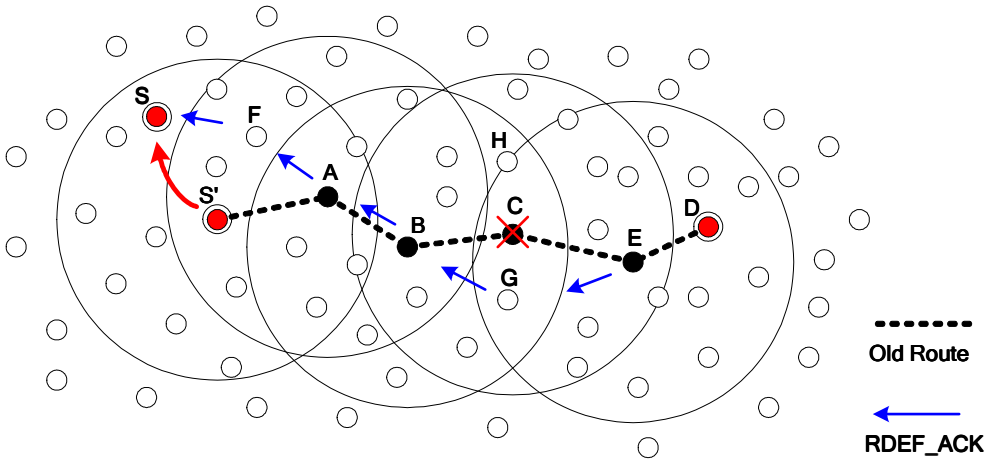


Figure 3-1 Region formation in route update process

When the RDEF arrives at the RU-DEST, the pre-routing region is also ready for new route discovery.

B. In-region Route Discovery

After the RU-SRC sends out an RDEF packet, it waits a while to ensure that its hidden terminals finish the relay of this packet, and then, it can safely release the RREQ to discover a new route.

When the RDEF is blocked by a broken point, the region construction also stops there. Since the follow-up RREQ can only be rebroadcast by “in-region” nodes, it is

detained at the “front-line” of the region. Later on, when a new RDEF with increased TTL is broadcast to expand the region, the detained RREQ’s are pushed forward to continue their propagations. In the example of Figure 3-1, The RDEF with TTL = 1 is blocked at node C, and the pre-routing region is extended to the neighborhood of B. If the RREQ’s arrive before B generates a new RDEF, B’s neighbors have to temporarily hold these RREQ’s. When B broadcasts a new RDEF with TTL = 2, its neighbors, such as G and H, are pushed to transmit the RREQ’s out after they finish the relay of the new RDEF.

Upon receiving the RREQ’s, the RU-DEST selects the best route and returns a route confirmation packet.

3.3 Performance Issues

Region-based route update achieves a good balance between route update overhead and route optimization. By exploring the network in a region around the existing route, region-based route update protocol gets rid of much unnecessary control traffic and at the same time ensures the good quality of the new discovered routes.

A. Routing Overhead

Progressive route discovery protocols achieve a good balance between route discovery overhead and route optimization. By exploring a wireless network in an asymmetric way, progressive route discovery gets rid of much unnecessary control traffic and at the same time guarantees high qualities of new discovered routes. In the example of Figure 2-4 in Chapter 2, if a brand new route is built between the source S

and the destination D, 40 out of 78 nodes participate in the route discovery process. If S simply refreshes an existing route, only 27 nodes are involved in the route update process. Clearly, when the network becomes larger and denser, and the network topology becomes more variable, REGR will be more efficient.

When route cost other than hop count is considered, all nodes participating in the route discovery process have to relay the duplicated RREQ's propagating from better routes. Progressive route update protocol, same as progressive new route discovery protocol, does not suffer much because only a small portion of nodes needs to relay routing information packets.

As an example, the following paragraph analyzes and compares the overhead consumption of REGR and AODV. In fact, the route discovery overhead of proactive routing protocols and PROC can be derived in the similar way.

For analytic tractability, a simple and idealized network scenario is used in our evaluation of communication overhead. We assume that wireless nodes are deployed in a circle field with radius l and node density d. All nodes have the same transmission range R. A mobile source moving with an average speed v sends P data packets to a fixed destination in T seconds. Since the average distance between the source and its downstream neighbor is $R/2$, the route broken rate, i.e. route update rate, is:

$$\frac{v}{R/2} = 2v/R$$

The total number of route updates during T seconds is:

$$m = 2vT / R .$$

The average route length is assumed to be proportional to the network radius. If hop-count is considered as route cost, the average route cost can be expressed as:

$$h = Kl / R \ \ (K < 1)$$

AODV

In the route creation stage, as the RREQ is flooded to the whole network, the route discovery overhead is: $\pi \cdot l^2 d$. Although the destination may send back multiple RREPs, we only count one RREP here. Since all nodes along the new discovered route participate in the relays of the RREP, the RREP transmission overhead is h . So, the total control overhead to create a new route is: $\pi \cdot l^2 d + h$.

When updating an old route with hop-count h , AODV limits the RREQ TTL to h , therefore, the one-time route update overhead is: $\pi (hR)^2 d + h$.

Since P data packets are transmitted from the source to the destination, the data transmission overhead is Ph .

Finally, we can obtain the total communication overhead as the summation of the route creation overhead, m route update overheads and the data transmission overhead:

$$O_{AODV} = (\pi \cdot l^2 d + h) + m(\pi h^2 R^2 d + h) + Ph \qquad (1)$$

REGR

If we use the distance-based broadcast scheme [15] and the counter based suppression scheme [17] in the destination discovery stage, and we assume the network is very dense, ideally, only nodes with direct distances nR, $n = 1,2,...,\lfloor l/R \rfloor$, from the source needs to rebroadcast the DLOC. All other nodes are suppressed to do so. As shown in Figure 3-2, if nodes on the dot-circles are dense enough, their broadcasts can prevent nodes lying between the circles from relaying the DLOC. Clearly, the control overhead generated in the destination discovery stage is:

$$(2\pi R + 2\pi * 2R + ... + 2\pi * bR)\sqrt{d} = \pi Rb(b+1)\sqrt{d}$$

here, $b = \lfloor l/R \rfloor$.

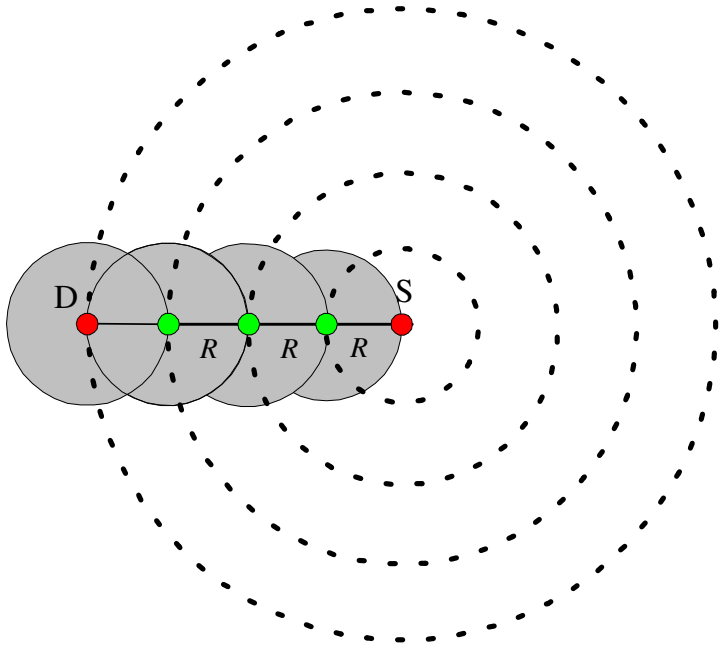


Figure 3-2 Calculation of REGR route creation overhead

In the region formation stage, when REG_WIDTH equals to one, the RDEF is only transmitted by the nodes along the preliminary route. So the control overhead in this stage is h.

If we assume that all nodes within the pre-routing region only rebroadcast the RREQ once, the RREQ transmission overhead can be calculated as:

$$(\pi R^2 h - 1.2284R^2(h-1))d$$

$\pi R^2 h - 1.2284R^2(h-1)$ is the area of the pre-routing region, as shown in Figure 3-2, and $1.2284R^2$ is the overlapping area between two circles. Finally, we have the expression of the REGR route creation overhead:

$$\begin{aligned} O_{REGR1} &= \pi Rb(b+1)\sqrt{d} + (\pi h - 1.2284(h-1))R^2 d + h \\ &= \pi Rb(b+1)\sqrt{d} + (1.9132h + 1.2284)R^2 d + h \end{aligned}$$

In the route update process, the mobile source first broadcasts a RDEF with TTL = 2; other nodes along the old route rebroadcast the RDEF with TTL = REG_WIDTH = 1. Therefore, the RDEF propagation produces an overhead of: $\pi R^2 d + h$. The transmissions of RDEF_ACKs add another h in the region formation overhead.

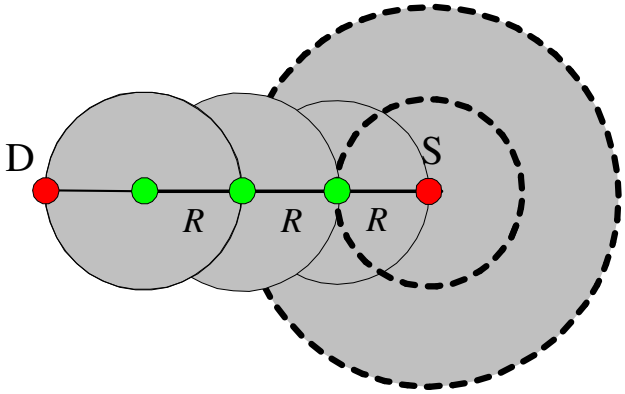


Figure 3-3 Calculation of REGR route update overhead

The RREQ transmission overhead in the route update process is a little larger than that in the route creation process because the pre-routing region includes two-hop neighbors of the source, as shown in Figure 3-3. We skip the detailed deduction and directly give the result of the RREQ transmission overhead:

$$(1.9132h + 8.5652)R^2 d$$

With the addition of the RREP overhead h, the one-time route update overhead becomes:

$$O_{REGR2} = \pi R^2 d + (1.9132h + 8.5652)R^2 d + 3h.$$

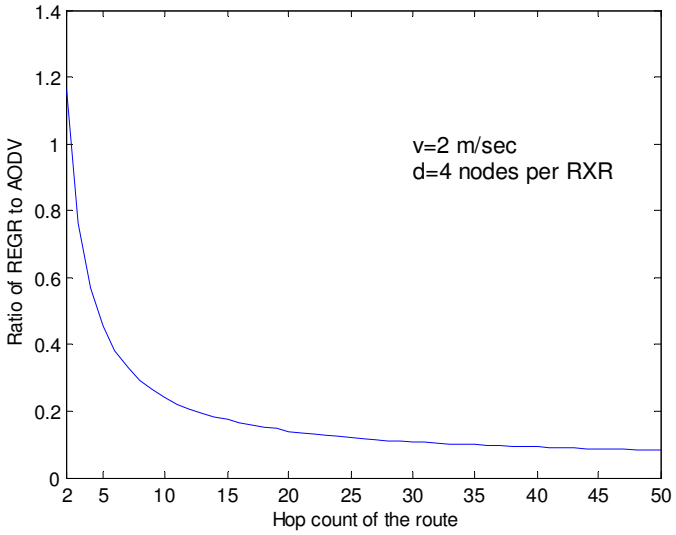
The overall communication overhead is:

$$O_{REGR} = O_{REGR1} + mO_{REGR2} + Ph \qquad (2)$$

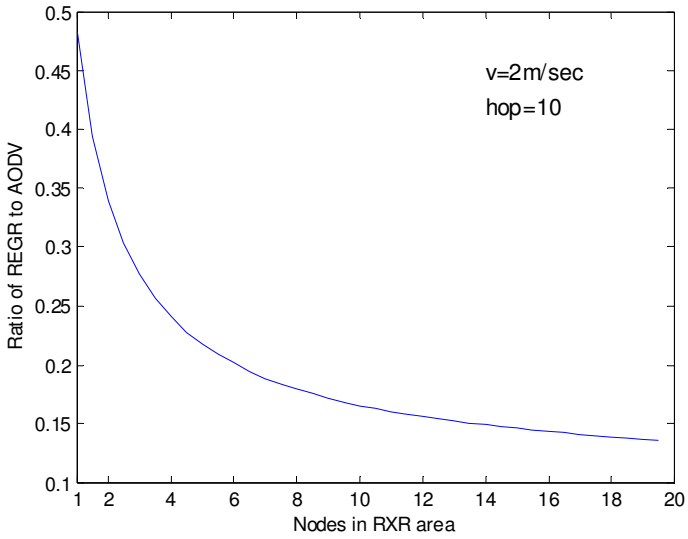
Overhead Ratio of AODV and REGR

We calculate the overhead ratio: O_{REGR} / O_{AODV} and observe its variations with the augments of route length, network density and source speed.

In the example of Figure 3-4, we set R = 30m, K=0.5, P = 600 packets and T = 300 seconds. The O_{REGR} / O_{AODV} ratio is shown to drop dramatically when the average route length increases. As the average route length is proportional to the network size, we can conclude that REGR is much more efficient than AODV in large-scale networks. Figure 3-4(b) also shows that REGR has great predominance in dense networks.



(a) Effect of Route length



(b) Effect of network density

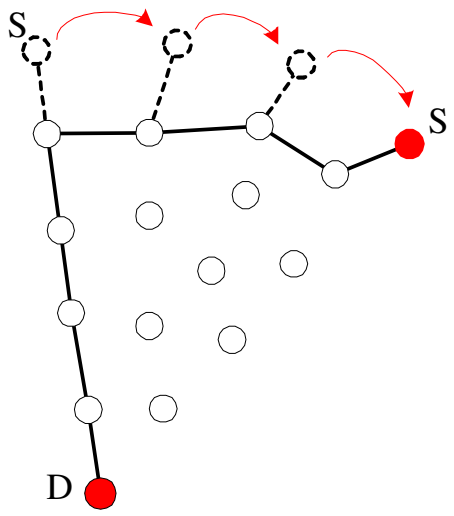
Figure 3-4 Ratio of REGR to AODV with various parameters

B. Route Optimization

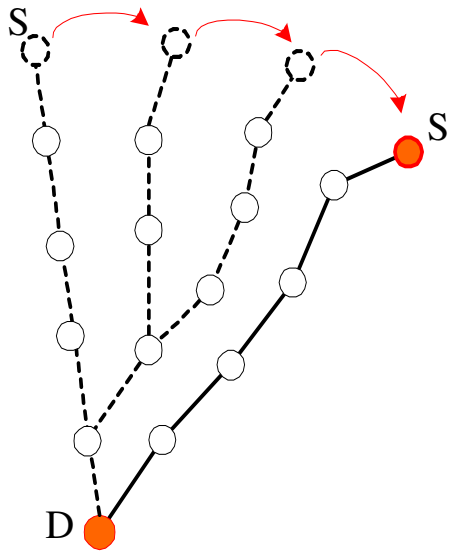
Since REGR carefully examines the region that most possibly contains the optimal route or near-optimal routes, good route quality can be guaranteed. Compared with the routing schemes based only on efficient broadcast algorithms, REGR can adapt to different route-cost requirements and always find desired routes.

In the route update case, although the local-repair type protocols like QL generate very little control overhead, the new discovered route depends too much on the old route. Route optimization is sacrificed. Figure 3-5 shows a mobile endpoint example, when the source S moves, QL only repairs the link between S and its next-hop neighbor, the other parts of the route is kept unchanged. So the route becomes longer and longer. REGR route update protocol conducts route-wide update in the

neighborhood of the old route. The new route is built around the old route, but it does not necessarily overlap with the old route. As shown in Figure 3-5 (b), REGR always builds a good route between the moving source and the destination. There is another example about the network power balancing: as QL keeps on using existing routes, nodes along old routes quickly exhaust their batteries, which may result in quick network partition and shortened network lifetime. REGR update protocol conducts route-wide update in the neighborhood of the old route. The new route is built around the old route, but it does not necessarily overlap with the old one. Therefore, the route freshness is assured and load balancing is achieved.



(a) QL



(b) REGR

Figure 3-5 Route update comparison in mobile endpoint case

3.4 Simulation Results

We simulate Region-based route update protocol using OPNET and compare its performances with those of AODV local repair [8] and QL [20]. For PHY and MAC layers, we use IEEE 802.11 WLAN model. Nodes are randomly placed in the network and have the same transmission range of 250m. The initial power of each node is 25 Joules. Node transmitting, receiving and idling powers are 0.66W, 0.395W and 0.035W, respectively. Each source sends out packets with a constant rate of 5 packets/second. The data packet size is 2048 bits. The RREQ and DLOC packets share the same size of 192 bits. The RREP and RDEF packets share the same size of 160 bits. The channel capacity is 1M bps.

In order to address the performance issues discussed in the previous section, we choose two performance metrics: routing overhead and the variance of residual power. The routing overhead accounts for all control packets that are consumed to repair broken routes in the entire simulation period. The variance of residual power is calculated among all nodes every 20s to measure the network-wide energy balancing. Since the reciprocal of residual battery capacity is used as node cost in Region-based route update protocol, we want to see its improvement of network energy utilization, which is a good index of route optimization.

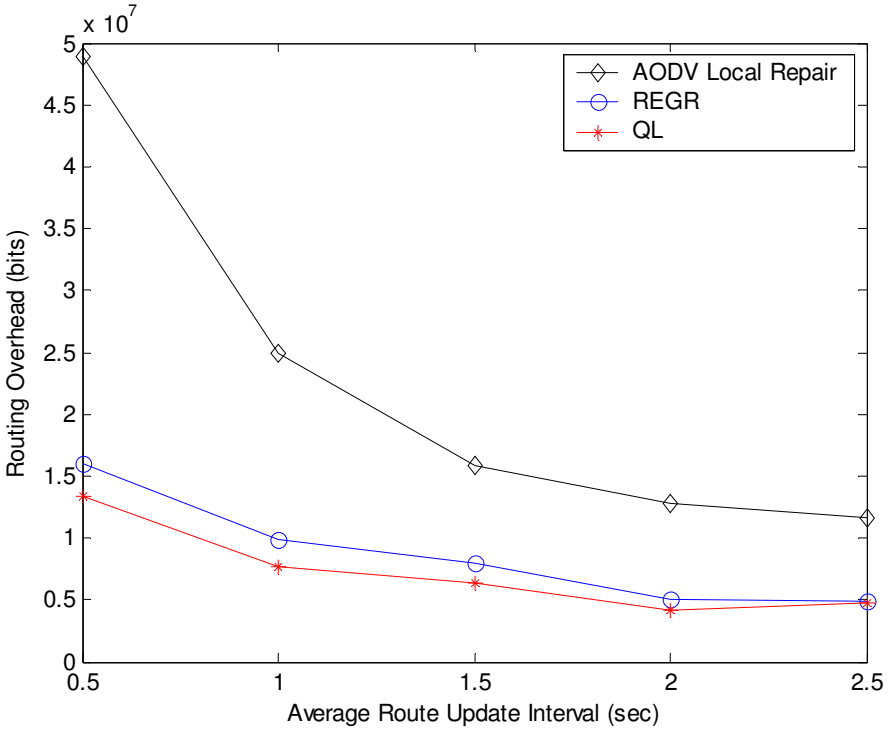


Figure 3-6 Impact of network dynamics in route update case

8 active source-destination pairs are randomly chosen from a network consisting of 300 nodes. The network size is 2000m × 2000m. In the network startup stage, a shortest route is established between each source-destination pair. At a series of

randomly selected time points, nodes randomly picked up from the 8 routes move out of the transmission range of their previous-hop or next-hop neighbors. The average interval between these time points, i.e. the average route update interval, is reduced from 2.5s to 0.5s. We compare the performance of the REGR route update protocol, AODV local repair and QL. Here, the reciprocal of residual battery capacity is considered as node cost in all approaches. With the increase of network mobility, thus the decrease of the route update interval, routing overhead is shown in Figure 3-6 to go up in all three approaches. However, REGR region based route update protocol suffers a much less overhead increase than AODV local repair because, in region based route update protocol, the route repair traffic is limited to a small region. QL has the merit to generate very small routing overhead since in most cases RREQ's are only transmitted BETWEEN the old route members and their neighbors, instead of propagating freely in the neighborhood of the old route. As the tradeoff, QL always burdens the existing routes and makes the nodes along the old routes quickly deplete their batteries. The results in Figure 3-7 are obtained by setting the average route update interval equal to 1s. QL is shown to have much larger power variance than the other two approaches. REGR region based route update protocol, with small routing overhead, achieves very good energy balancing. Its power variance values are very close to the results of AODV local repair.

3.5 Conclusion

In this chapter, we present REGR region-based route update protocol to provide energy-efficient and scalable routing in large-scale wireless mesh networks. Instead of

blindly spreading their route update queries to all directions, region-based route update protocol reduces communication overheads of route reconstruction by defining the packet propagation region along the old route, and limiting route rediscovery packets within only this region. Our simulation shows that region-based route update protocol reduces the traffic overheads substantially and still has much freedom to find optimal routes. Therefore, region-based route update protocol is an energy-efficient and scalable approach that is suitable for power-limited large scaled wireless mesh network environments.

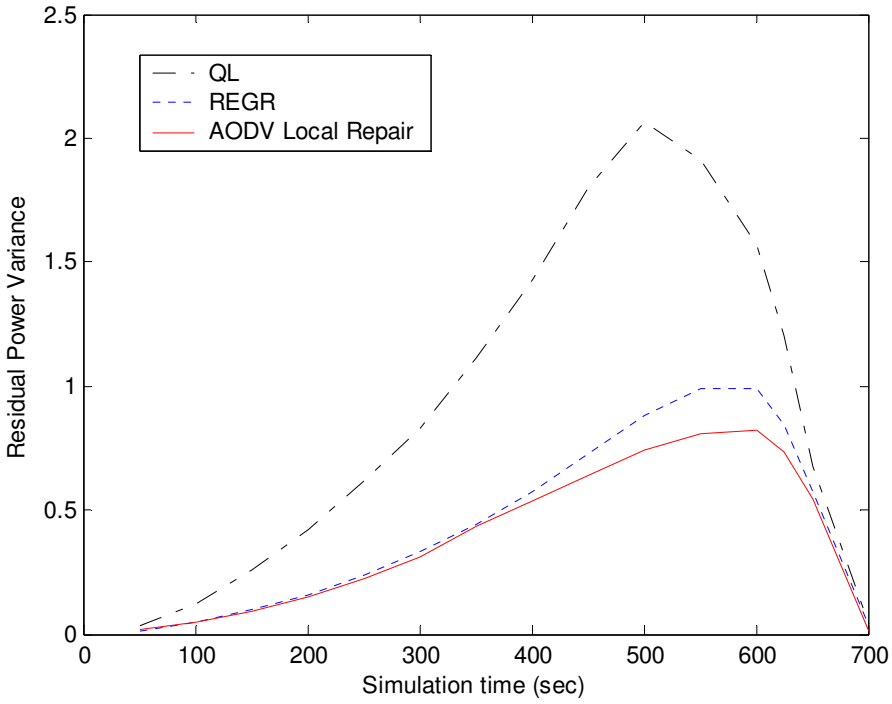


Figure 3-7 Network energy balancing in route update case

4 PROGRESSIVE ROUTE DISCOVERY — MULTIPATH ROUTING PROTOCOL

4.1 Introduction

Muitipath routing is one of the distinguished features of wireless mesh networks [23-34]. It can be used for various purposes. For example, a source device can establish a primary path as well as several backup paths toward a desired destination. Once the primary path breaks, the source can switch the on-going traffic to backup paths, instead of shutting down the end-to-end connection. In resource-constrained networks, a source-destination pair may evenly use the resources of multiple paths, instead of exhausting one path. Besides, when a single path cannot provide sufficient capacity to support traffic demands, a source may create additional paths and simultaneously deliver data through these paths. The focus of this chapter is to design a concurrent multipath structure for bandwidth aggregation and throughput enhancement.

Traditional multipath routing algorithms incline to build multiple link/node-disjoint paths with minimized path costs. Although these paths do not share common links or nodes, they are usually located around the shortest path, and stay very close to each other. When all these paths are utilized simultaneously to transmit data, they will heavily affect each other and cause route-coupling problem [21]. Ascribed to the broadcast nature of wireless medium and the hidden/exposed terminal problems [43,67-69], the route-coupling problem suppresses concurrent data transmissions through adjacent paths, and may make multipath routing perform worse than single path routing. In order to achieve throughput enhancement by using multiple paths, we need to seek ways to remove route coupling.

In this chapter, we introduce AODV-DM, an AODV Decoupled Multipath routing protocol, to efficiently establish multiple node-disjoint paths that are separated far enough to avoid inter-path interferences. In brief, a primary path is built like any other single-path routing protocol first; and then a region is defined around the primary path for insulating purpose. Finally, a secondary path is selected outside the region. Since the primary path and the secondary path are spaced out by the insulating region in view of radio coverage, they do not contend for the same piece of medium. Moreover, the AODV-DM algorithm is optimized to reduce the length of the secondary path. The two-path formation mechanism can be easily extended to establish three or more decoupled paths.

AODV-DM provides a good multipath structure at the network layer. However, as the transport layer controls the inputs to these paths, the network layer shall seek

support from the transport layer in order to work more efficiently. Unfortunately, the traditional TCP is designed for single-path routing and cannot adapt to the network layer multipath structure. We propose to employ a path-aware SCTP scheme [44][45], which possesses the abilities to identify congestions occurring in different paths, and independently control traffic rate of each path.

By integrating the multipath modules crossing the network and transport layers, we are able to derive an efficient multipath structure for concurrent data transfer in wireless mesh networks. Simulations verify the feasibility of our designs and show that our protocols can achieve much better performances than the single path and coupled multipath protocols.

The rest of this chapter is organized as follows. Chapter 4.2 gives a brief overview of related work. Chapter 4.3 and 4.4 describe the AODV-DM protocol as well as the path-aware SCTP scheme respectively. Chapter 4.5 presents our simulation results. Lastly, Chapter 4.6 concludes the multipath structure.

4.2 Related Works

Multipath routing in wireless mesh networks has been studied extensively in recent years [21,25-29]. Most multipath routing algorithms address the ways to construct link-disjoint or node-disjoint paths. While link-disjoint multipath is mainly utilized to tolerate link failures, node-disjointed multipath can not only protect transmissions from link and node failures, but also enable balanced usage of network resources. Therefore, node-disjoint multipath is more favored in error-prone and

resource-limited wireless networks. There have been quite a few papers regarding the creation of multiple node-disjoint paths in wireless mesh networks. For example, Lee and Gerla present a Split Multipath Routing algorithm (SMR) [25] to build maximally disjoint paths. EDSR, an Extended Dynamic Source Routing Scheme [26], discovers nod-disjoint paths by marking them with different colors. AODVM [27] is an AODV-based node-disjoint multipath routing protocol. We will discuss more about AODVM in chapter 4.3.

Although node-disjoint paths do not share any relay nodes, they may lie close to each other and generate serious inter-path interferences. This phenomenon is called route coupling. In order to resolve the route-coupling problem, M. R. Perlman et al. proposed to use multiple channels to construct contention-free paths [21]. In [28], D.Saha et al. suggested using directional antennas to reduce radio interferences between different paths. Although these approaches can decouple adjacent paths, they require extra resources, which may not be practical in low-cost wireless mesh networks.

A DSR based decoupled multipath scheme was proposed in [29]. In this scheme, each route reply (RREP) message carries not only the information of a newly discovered path, but also the neighborhood information of the path. With sufficient topology information supplied by RREPs, source node is able to pick out multiple decouples paths. One concern about this scheme is that the topology reporting through RREPs may cause large control overhead especially in large and dense networks.

Some recent researches indicated that traditional TCP does not work well over concurrent multipath routing [46-47]. One reason is that TCP cannot conduct congestion control in each individual path. Also, TCP ascribes disordered packet delivery to path congestion, however, for multipath routing, many out-of-sequence cases are caused by path diversity, which should not trigger the reduction of data transmission rate. A new transmission layer protocol, Stream Control Transmission Protocol (SCTP) [48], was proposed recently to enhance end-to-end transmission performances especially for multi-stream, multi-path applications. SCTP inherits the major features of TCP, such as connection setup, reliable transmission, and congestion control etc. In addition, it provides some new functions, like multi-streaming, multi-homing, and four-way handshake. The multi-homing scheme is particularly beneficial to the establishment and maintenance of multiple paths. Al et al. [44] and Ye et al. [45] improved the SCTP congestion control mechanism by making source-destination pairs respectively handle congestion control in different paths. This improvement shows significant advantage in concurrent multlpath data transport.

Some researchers declare that unless we establish a large number of paths, which is costly and infeasible, multi-path routing does not improve load balancing at all [81]. They reach this conclusion under the assumption that every node in the network incessantly sends messages to any other node via the shortest paths. If all nodes aggressively occupy network resources and keep the network busy, using additional paths would not benefit much. However, under normal operating conditions, much of network resources may become available in terms of time and space. By using multipath structure, a few overloaded source-destination pairs can utilize the

unoccupied network resources to enhance their end-to-end performances. Our work is aiming at this type of scenarios.

4.3 Decoupled Multipath Routing Protocol

The basic idea of our decoupled multipath routing protocol is to form two or more paths that are separated from each other by an insulating region. This decoupled multipath structure enables concurrent data transmissions through all established paths without generating inter-path interferences. It is designed, particularly for high-traffic source-destination pairs, to boost end-to-end throughput in large and dense wireless mesh networks.

We will mainly describe the formation of two decoupled paths, while the protocol can be easily extended to cover the cases of three and more paths. The whole routing process can be divided into three stages: in the first stage, a primary path is discovered by using the single path AODV protocol [8]; and then, an insulating region is formed around the primary path; and lastly, a secondary path outside the insulating region is selected and established.

An easy way to implement above process is to use two-round route discovery. The first round discovery tries to locate an optimal primary path. Then, an RREP is sent from the destination to the source to confirm the newly discovered path and define an insulating region around the path. All nodes in the primary path shall broadcast the RREP to their one-hop neighbors, and these neighbors mark themselves as “in-region” nodes. When the region is ready, the source initiates the second round discovery. Only

nodes without “in-region” marks can participate in the second round discovery. Eventually, an optimal path outside the insulating region is selected as the secondary path. Although this method is easy to implement, it requires two route discoveries, which cause large control overhead and engender significant interferences to normal data traffic.

We notice that AODVM [27] can establish node-disjoint paths with one-round route discovery. Similar mechanisms can be applied to build decoupled paths. Before going to the details of our protocol, we first give a brief introduction of AODVM.

In AODVM, a source that intends to create multiple disjoint paths first broadcasts a route request (RREQ) message. Intermediate nodes have to record all received RREQs in an RREQ table, but relay only one of them. No node is allowed to send back RREPs except the destination. The destination responds each received RREQ with an RREP. When an intermediate node receives an RREP, it picks up a neighbor node that has the shortest hop-count to the source, and forwards the RREP to that neighbor. To ensure that an intermediate node does not participate in more than one path, whenever a node overhears an RREP from one of its neighbors, it deletes the entry corresponding to that neighbor from its RREQ table. Therefore, the intermediate nodes that have relayed an RREP once will not be asked to forward other RREPs. By this means, path separation is guaranteed. AODVM fully utilizes the routing information revealed by one-round route discovery and establishes disjoined paths in a very efficient way.

In the following sections, we present the detailed process of the AODV-DM protocol.

4.3.1 Primary Path Discovery and Region Formation

When a source tries to reach a destination, but cannot find any available routing information, it initializes a route discovery process by flooding an RREQ to the entire network. When an intermediate node receives an RREQ, it records the information of RREQ sequence number, RREQ originator, hop-count to the source and immediate RREQ sender in its RREQ table. If the intermediate node later on receives duplicated RREQs from different senders, it shall also record these senders' information in its RREQ table. The intermediate node only rebroadcasts the first received RREQ. This process is the same as AODVM.

Multiple RREQs may arrive at the destination from different paths. Instead of responding each RREQ immediately with an RREP, the destination first responds the RREQ propogating from the shortest path, i.e. the primary path. The destination generates a primary route reply (pRREP) packet, and unicasts this packet to a neighbor that is closest to the source. The pRREP contains a "previous hop[4] address" field. When a relay node receives a unicast pRREP from the destination, it examines its RREQ table and picks out a neighbor closest to the source. It puts the address of this neighbor in the "previous hop address" field of the pRREP and broadcasts this packet out. When receiving a broadcast pRREP, a node marks itself as an "in-region" node. If the address of this node matches the "previous hop address" of the pRREP, the node shall create a routing entry for the destination. And then, it selects a neighbor that is closest to the source, and updates the "previous hop address" of the pRREP with this

[4] In this chapter, previous-hop means the direction toward source, and next-hop means the direction toward destination.

neighbor's address. After that, it broadcasts the pRREP to its neighbors. If the address of the node does not match with the "previous hop address" of the pRREP, the node shall discard the pRREP without further relaying it. Each pRREP sender, after sending out the pRREP, shall delete the entry corresponding to the "previous hop" neighbor from its RREQ table. Also, when a node receives a broadcast pRREP, it shall delete the entry corresponding to the pRREP sender from its RREQ table. Removing pRREP senders from their neighbors' RREQ tables can prevent these nodes from serving in more than one path.

When the RREP is relayed to the source, the primary path formation is completed. At the same time, an insulating region is established in the neighborhood of the primary path. To prevent future RREPs from entering the insulating region, each "in-region" node shall broadcast a region protection (RPRT) packet to its neighbors. Those neighbors outside the insulating region shall remove the "in-region" node from their RREQ table. In other words, they will not send RREPs to this node any more. Random backoff schemes can be applied to avoid concurrent RPRT broadcasts from neighboring nodes.

4.3.2 Secondary Path Discovery

After releasing the pRREP, the destination waits a short while to allow the pRREP to establish the insulating region, and then it begins to respond other RREQs. The response packet is called secondary route reply (sRREP) packet. The sRREP also contains the "previous hop address" field. Its propagation process is very similar to that of pRREP. The only difference is that (for two-path formation case), a node receiving a

broadcast sRREP does not mark itself as “in-region” node. Except the destination, all sRREP senders shall broadcast the sRREP to their neighbors in order to remove themselves from their neighbors’ RREQ tables. This can efficiently prevent loops in sRREP transmissions. When a node receives a sRREP, but has no available entry in its RREQ table to further forward the packet, it shall send a route reply rejection (RREJ) packet to the sRREP sender, which will try other entries in its RREQ table.

Figure 4-1 shows an example of AODV-DM. After the “in-region” nodes broadcast RPRTs, their “out-of-region” neighbors delete all entries leading into the region. Therefore, only nodes outside the region are able to serve as the sRREP relays. The secondary path established by this means is thus decoupled from the primary path.

If the neighbors of the “in-region” nodes happen to miss the RPRTs due to packet collisions and link failures, and do not delete the entries pointing into the insulating region. Some sRREPs may be erroneously sent to the “in-region” nodes. In this case, the “in-region” nodes shall send RREJs back to the sRREP senders and ask them to try other neighbors.

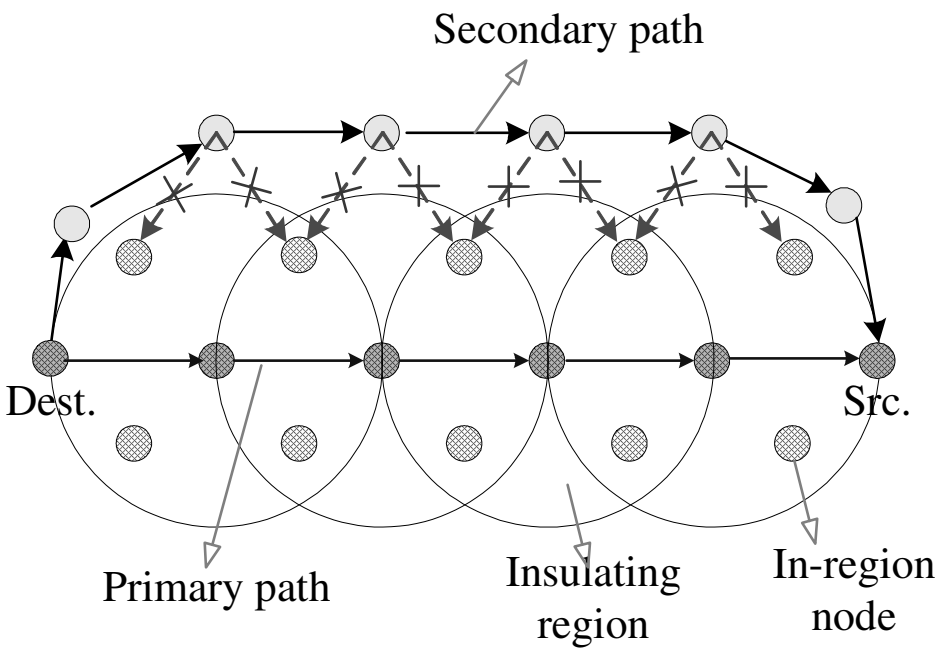


Figure 4-1. AODV-DM Illustration

4.3.3 Optimization of AODV Multipath Routing

AODV-DM and AODVM can avoid loop problems by removing RREP senders from their neighbors' RREQ tables, however, they may create "twisted" paths, as shown in the example of Figure 4-2.

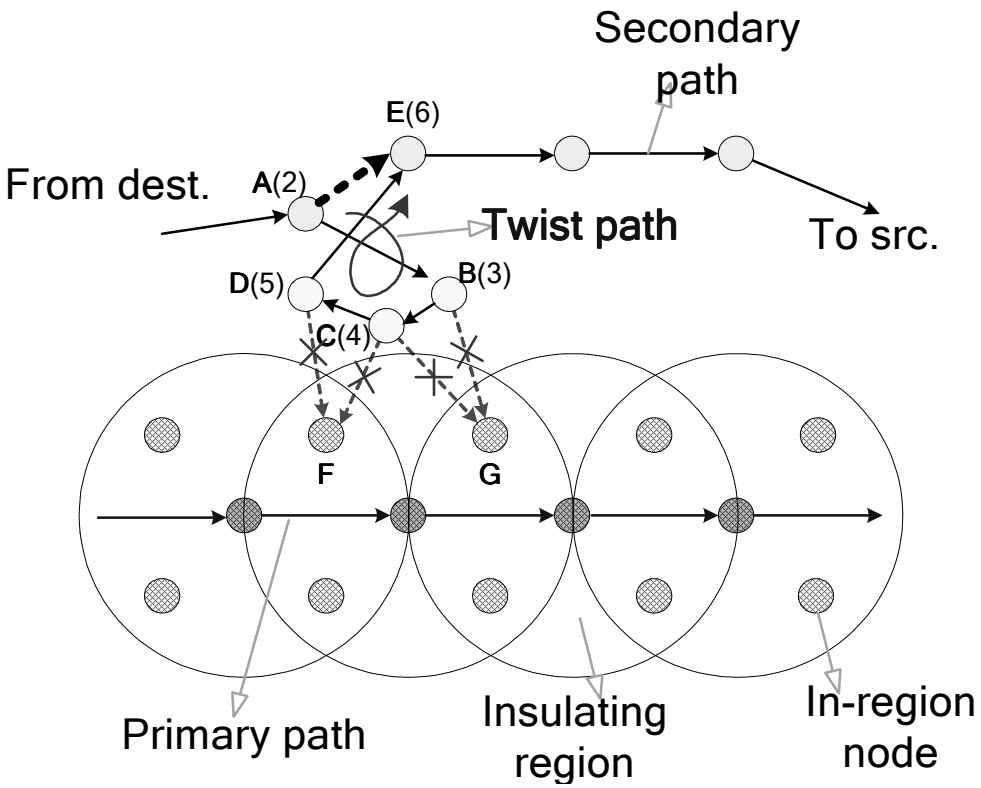


Figure 4-2. AODV-DM: Route-twisting Problem and the Solution

In this example, an sRREP propagates from the destination to node A. By checking its RREQ table, node A finds that node B is closest to the source. So A updates the "previous hop address" of the sRREP with B's address, and broadcasts the sRREP out. A's neighbors, when receiving the sRREP, delete A from their RREQ tables. Originally, B has a very short path to the source through node G. However, it deletes the entry corresponding to G from its RREQ table because G is an "in-region" node. B happens to have a very sparse neighborhood especially toward the source side. It has to deliver the sRREP to node C, which is the neighbor closest to the source in its record. Similarly, node C has to send the sRREP to node D, and D sends the packet to node E. In fact, since E is a one-hop neighbor of A, node A should have directly used E

as the next-hop relay instead of “twisting” the path all the way through B-C-D-E. The reason to cause this problem is that when a node, such as B, deletes some routing entries from its RREQ table, it does not notify its neighbors. Therefore, its neighbors, such as A, may make wrong choices based on the outdated routing information. Path twisting unnecessarily prolongs path length, and also causes serious intra-path interference.

One way to resolve the path-twisting problem is to ask every node to notify its neighbors whenever its optimal entry for the source is changed. However, such an update may cause chain-reaction and result in large-scale entry renewal, which incurs significant control overhead.

We propose to use a reactive path correction method to solve the path-twisting problem. In our method, a node that has relayed the sRREP should keep monitoring the sRREPs broadcast from its neighbors. If it finds that an sRREP carries a hop-count that is at least two-hop larger than its own hop-count to the destination, it shall sends a correction RREP (cRREP) to the sRREP sender to recommend a shorter path (through itself). If an sRREP sender receives two or more cRREPs, it shall select the cRREP sender that is closest to the destination. In the example of Figure 4-2, let’s assume node A is two-hop away from the destination. When B, C, D, and E receive the sRREP propagating along the twist path A-B-C-D-E, they record their hop-counts to destination as 3, 4, 5, 6, respectively. Node E sets the hop-count field of the sRREP as 6, and broadcasts the sRREP to its neighbors. Both A and B detect that the sRREP from E carries an excessively large hop-count value. Therefore, they send cRREPs to

node E to suggest route rectification. Since A has a shorter hop-count to the destination, node E accepts the cRREP from A and changes its routing entry accordingly. Eventually, the twist path is shortened by a direct link from A to E.

4.4 Multipath Congestion Control over AODV-DM

Transport layer protocols are used to provide reliable end-to-end data delivery and congestion control. As AODV-DM enables concurrent data transmission by decoupling multiple paths at the network layer, we attempt to exploit the path diversity obtained from network layer to improve end-to-end data delivery. Unfortunately, the traditional TCP is designed mainly for single-path data transmission. It is short of mechanism to support concurrent packet delivery over multiple paths. Specifically, the TCP of a source device cannot differentiate multiple paths created by the network layer. Whenever a path is congested, the TCP punishes all paths by reducing the overall transmission rate. This is unfair and inefficient to those non-congested paths. In order to make full use of the capacity provided by multiple paths, the transport layer shall also resolve the transport-layer route-coupling problem.

Recently, a new transport layer protocol, the standard Stream Control Transmission Protocol (SCTP) [48], was proposed to enhance end-to-end transmission performances of multi-stream and multi-path applications. SCTP inherits the major features of TCP, such as connection setup, reliable transmission, and congestion control etc. Besides, it introduces a few new features like multi-homing and multi-streaming, which makes it outperform TCP for many applications.

The multi-streaming feature of SCTP enables source-destination pairs to efficiently handle simultaneous transmission of different types of packets. Under TCP reliable transmission mechanism, all packets delivered through a TCP connection must arrive at the destination correctly and in sequence; in other words, TCP can only support one “stream” under one TCP connection. SCTP, however, can divide packets into multiple streams. Ordered flag enabled packets within the same stream have strong correlation and must be delivered in sequence. Packets from different streams, however, have no or very weak sequence relationship; thereby, they can be processed regardless of their arrival order. SCTP introduces a stream identifier field and a stream sequence number field to each data packet in order to separate the controls of different streams. The similar mechanism can be applied to the management of multiple paths.

The multi-homing feature of SCTP allows devices to maintain multiple IP addresses. This feature is especially beneficial to multipath data transmission, since it enables a source-destination pair to establish multiple paths under one SCTP connection, and bind different IP address pairs to different paths. Once a primary path is broken, the on-going traffic can be quickly switched to a backup path, which is associated with an alternative IP address pair.

The standard SCTP provides a single congestion control engine to each SCTP connection, despite how many streams and paths this connection maintains. A single control engine is good enough when multiple paths only serve for backup purpose. However, if multiple paths are used for concurrent data transmissions, the standard SCTP faces the same transport-layer route-coupling problem as TCP. In order to

resolve this problem, Al et al. [44] and Ye et al. [45] extend the original SCTP congestion control mechanism by providing one congestion control engine to each path. Two new parameters, Path ID (PID) and Path Sequence Number (PSN), are introduced to enable the path-based congestion control. In [44], sources attach PSN and PID to each data packet. From these path parameters, destinations can tell exactly which paths are normal, and which paths are problematic. Destinations report path statuses to sources by sending back Selective Acknowledgement packets (SACKs). Each source maintains an independent congestion window for every path. Based on the reports from the destination, it adjusts the congestion window size of every individual path, and tries to achieve the optimal throughputs at all paths. Al's approach needs to modify the SCTP packet format by adding two new fields. The authors in [45] propose a backward compatible method. In this method, each source stores the PSN and PID of all outstanding packets (the packets that have been sent by the source but have not been acknowledged), without transmitting these parameters to the corresponding destination. The destination has no idea of individual paths; thereby, it can only estimate and report the status of the SCTP connection based on the Transmission Sequence Number (TSNs) of received packets. When the source receives the SACKs from the destination, it derives the status of individual paths by checking the path parameters of the acknowledged and unacknowledged packets, and then the source is able to adjust the congestion window size of each path based on its most updated status.

We believe the path-aware SCTP and AODV-DM are perfect cross-layer partners to collaborate for concurrent data transmission in wireless mesh networks. AODV-DM provides a solid multipath routing structure to support the benefits of path-based

congestion control. The path-aware SCTP, by controlling the decoupled paths independently, makes the most of the multipath capacity. In order to integrate the path-decoupling mechanisms of these two layers, we bind path-based congestion control engines with decoupled paths.

4.5 Simulation Results

We implement AODV, AODVM and AODV-DM in OPNET and compare their performances under UDP, the standard SCTP and the path-aware SCTP traffic. The MAC layer we use is based on the distributed coordination function (DCF) of IEEE 802.11 for wireless LAN.

As we explained in chapter 4.2, in many cases, only a few overloaded source-destination pairs have high bandwidth requirements. Therefore, we can establish multipath structures between these endpoints and leave other lightly loaded pairs as single paths. In the simulation, we allocate one source-destination pair with various UDP traffic load and test the performances of different protocols. We also examine the performances of these protocols when TCP or SCTP traffic is applied to this pair. Influence of the lightly loaded background traffic on the overloaded pair is investigated as well.

The simulation scenario is a wireless network with 400 static nodes randomly deployed in a rectangle area of 4000m x 3000m. The transmission range of each node is 300m. The channel bandwidth is 1Mbps. One source-destination pair is randomly selected at the beginning of the simulation. 2000 data packets (except for Figure 4-5

and Figure 4-6) are transmitted from the source to the destination through the paths constructed with different routing protocols. In AODV case, the data are sent along one route; while in AODVM or AODV-DM case, the data are delivered along two routes concurrently.

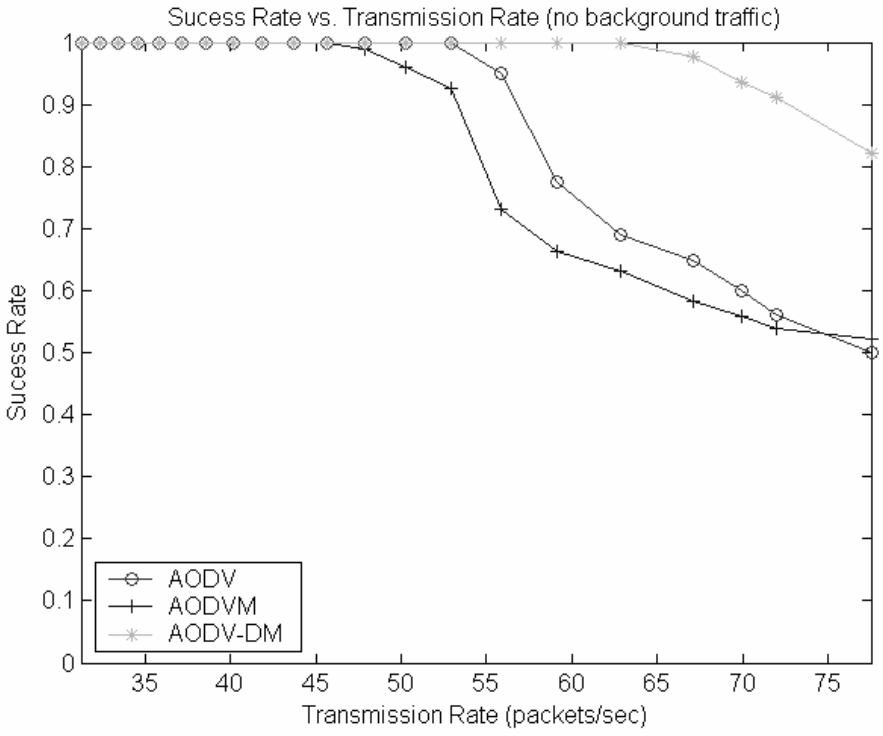
The source-destination pairs of the background traffic are also randomly decided. And the AODV single-path routing protocol is used to build single routes between them. The transmission rate of each of them is 28 packets per second, a lightly loaded traffic that can be handled by using single-path routing. The transmission process of each background traffic pair randomly lasts for 4 to 6 second and when it terminates, another background traffic pair is randomly selected and the transmission process continues. The number of the simultaneous background traffic pairs in each simulation round is fixed.

The performance metrics of the multipath for UDP traffic are success rate and average end-to-end delay of the overloaded source-destination pair. Constant bit rate (CBR) traffic is used in each UDP session. The performance metric for the transport layer traffic (TCP and SCTP) is goodput, the number of the ordered packets received by the application layer of the destination per second.

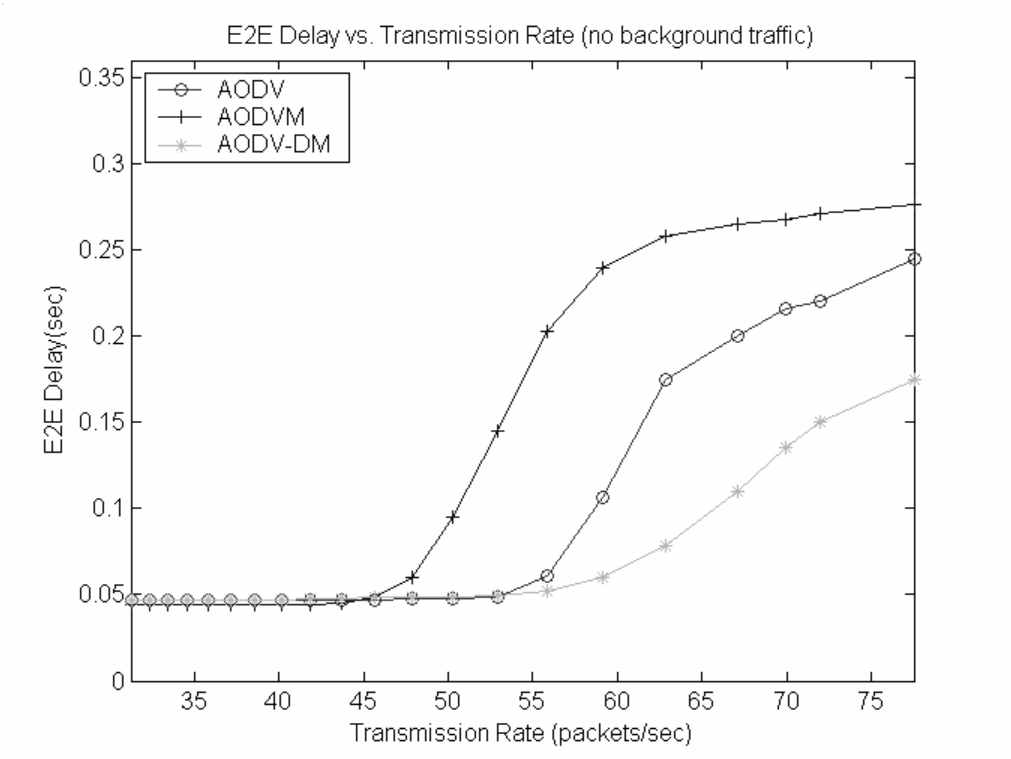
4.5.1 Performance Comparison under UDP Traffic

Figure 4-3 and Figure 4-4 demonstrate the efficiency of using AODV-DM under UDP traffic. The size of each UDP data packet is 4096 bits.

Figure 4-3 compares the performances of different protocols when transmission rate of the source changes. Background traffic is not considered here. Figure 4-3(a) shows that when the transmission rate increases, the success rate of AODV falls down quickly because the offered traffic is more than the single path can handle. AODVM multipath routes do not help to share the increased traffic. The reason is that since the node-disjoint routes found by AODVM are usually physically close, some nodes on the diverse routes are in the same collision domain. Hence, they will compete for the common channel and lose packets on the way. By using AODV-DM, two route-decoupled paths are built to forward data packets using unshared network resources. Now that the two paths do not interfere each other, the bandwidth they utilize is aggregated and the throughout capacity is increased. For this reason, the success rate of AODV-DM decreases much slower than AODV and AODVM.



(a) Success Rate Comparison



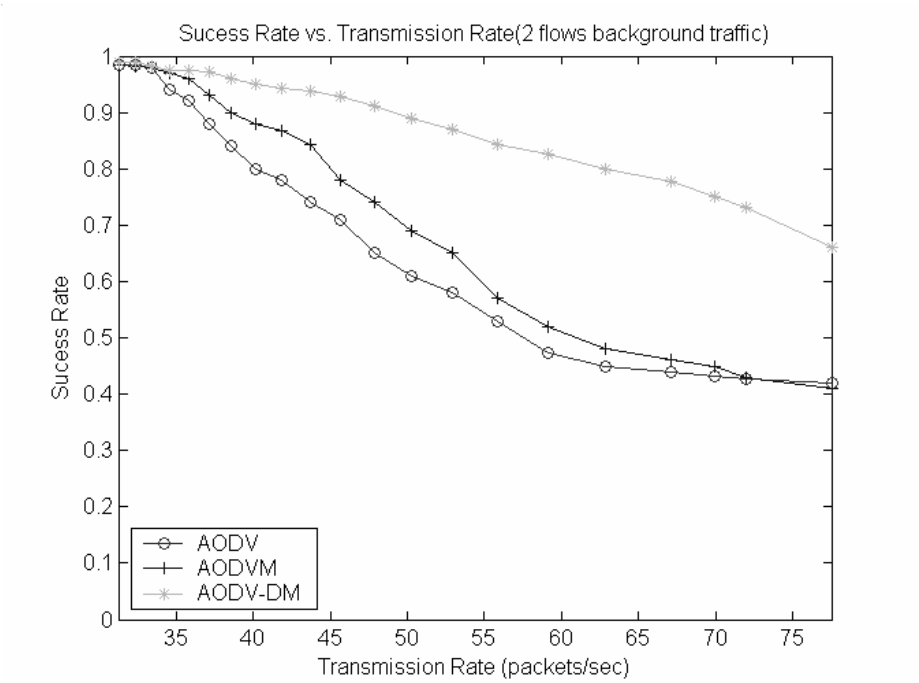
(b) E2E Delay Comparison

Figure 4-3. Impact of Transmission Rate on UDP performance (No Background Traffic)

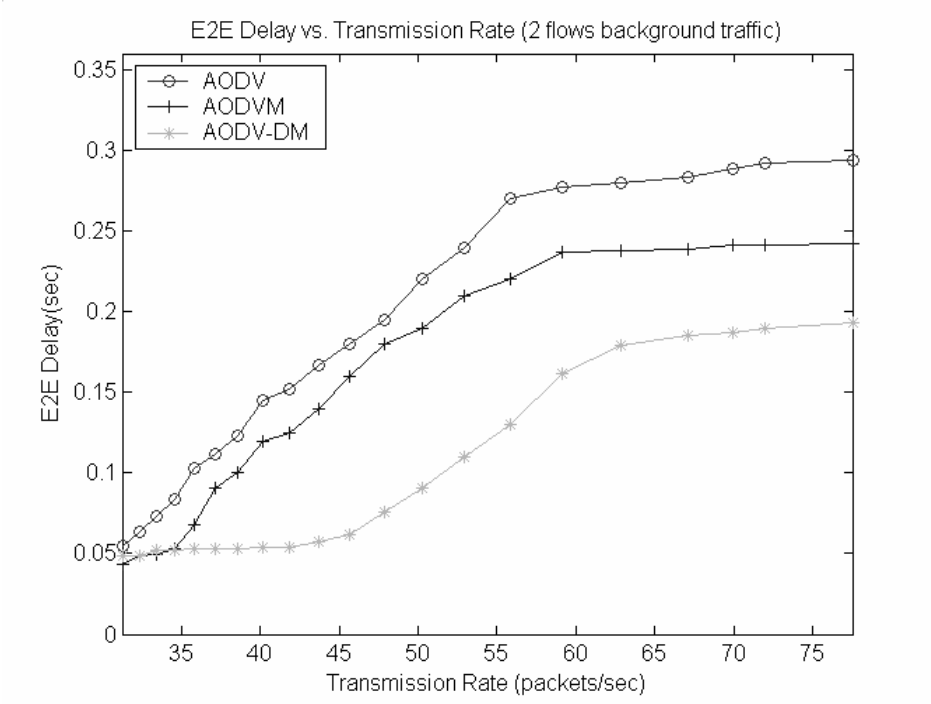
Figure 4-3(b) compares the average end-to-end delay of different routing protocols. Under low transmission rate, AODV performs slightly better than the other two approaches because AODV always selects the shortest route; while the secondary paths found with multipath routing protocols are usually longer than the shortest one. When the transmission rate of the source node goes up, the end-to-end delay of AODV increases quickly. It can be ascribed to the limited throughput capacity of the single path. When the transmission rate is large, the single path is congested and the data

packets have to be queued. Besides, the intermediate nodes on the single route experience more contention and thus defer or retransmit data more frequently under heavy traffic. The coupled multiple routes may not be able to reduce the delay. It is because the intermediate nodes within the interference range of both routes become the bottlenecks to block the traffic from being transferred immediately. In contrast, if the traffic is split along AODV-DM route-decoupled paths, more unoccupied network resources are utilized to carry traffic. As a consequence, the transmission burden on each path is relieved, which decreases queuing and processing delay.

Fig. 4-4 represents the effect of the transmission rate when two background traffic flows are introduced. It indicates that the performances of AODV-DM still surpass AODV and AODVM by a large margin. We also compare their performances under 4-flow and 6-flow background traffic, and get the similar conclusion. All these simulation results demonstrate the advantage of using AODV-DM for concurrent multipath transport and throughput enhancement. Due to the space limitation, we only list the success rate of these protocols under 4-flow and 6-flow background traffic in Table 1.



(a) Success Rate Comparison



(b) E2E Delay Comparison

Figure 4-4 Impact of Transmission Rate on UDP Performance (2 Flows Background Traffic)

Table 4-1. SUCCESS RATE OF UDP TRAFFIC WITH 4/6 FLOW BACKGROUND TRAFFIC

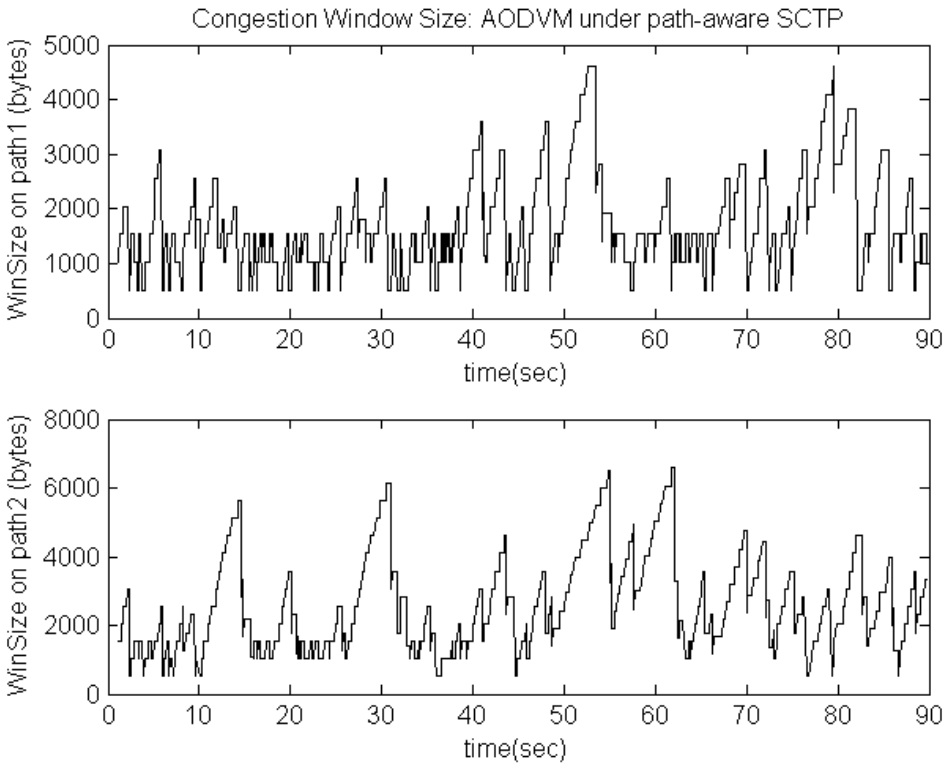
TxRate(pkt/sec) / Protocol Name	31.3	32.4	33.4	34.6
AODV, 4 flows	0.96	0.88	0.81	0.76
AODVM, 4 flows	0.91	0.85	0.83	0.77
AODV-DM, 4 flows	0.97	0.92	0.90	0.88
AODV, 6 flows	0.75	0.61	0.51	0.47
AODVM, 6 flows	0.76	0.74	0.72	0.65
AODV-DM, 6 flows	0.88	0.86	0.82	0.80

4.5.2 Performance Comparison under TCP and SCTP traffic

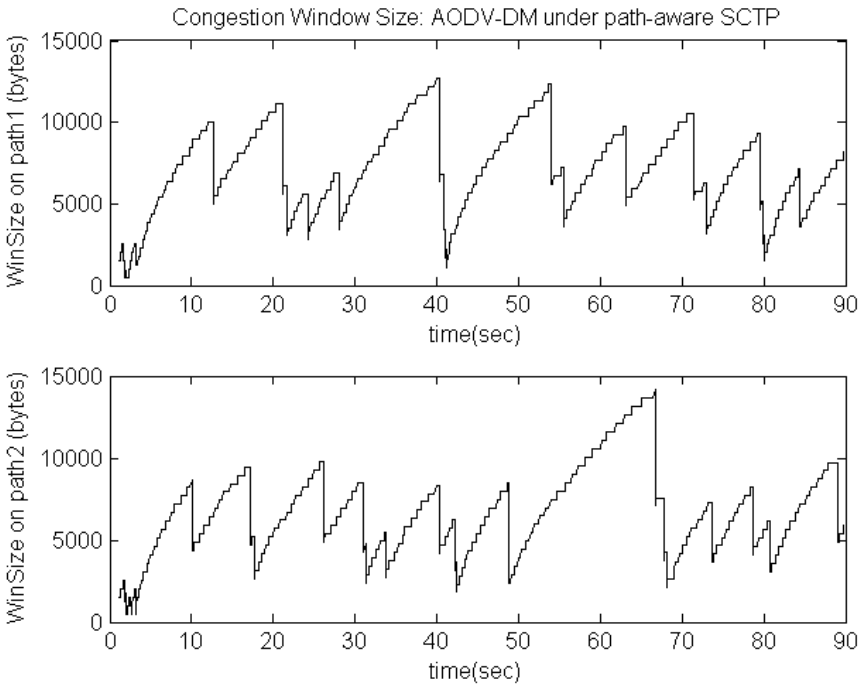
Now let us turn the attention to the performances of multipath under TCP and SCTP traffic. The MTU (Maximum Transmission Unit) of the TCP/SCTP traffic at each path is 4352 bits. FTP packets are sent down following the congestion control mechanism. That is, when TCP or the original standard SCTP are applied, traffics of two paths are controlled by the same congestion window. As for the path-aware SCTP traffic, each path has its own congestion control parameters, which are independent with the parameters of other paths. The source starts to send data packets along one path until the outstanding data (the data that has been sent by the source but has not been acknowledged) reaches the control window size of this path. And then the source

starts to send data packets along another path following the same policy. The path-aware SCTP approach we adopt is based on [45].

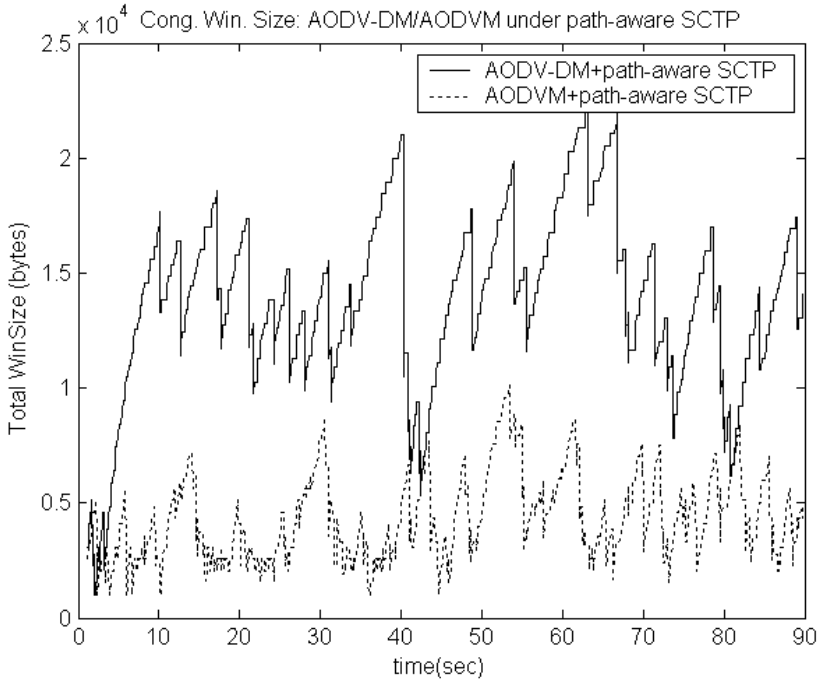
4.5.2.1 Congestion Window Size Analysis



(a) Path-aware SCTP over AODVM



(b) Path-aware SCTP over AODV-DM



(c) Total window size comparison: AODV-DM vs. AODVM

Figure 4-5 Congestion window size comparison: path-aware SCTP over AODV-DM/AODVM

In Figure 4-5, the congestion window sizes of the path-aware SCTP over AODVM and AODV-DM are compared. It expresses that the window size oscillation of the path-aware SCTP over AODVM (Figure 4-5(a)) is much more serious than that over AODV-DM (Figure 4-5(b)). This can be ascribed to the network-layer route-coupling problem. Due to the interference between two paths, the window size increase in AODVM case is not only disturbed by the channel contention among the intra-path nodes but also by that among the inter-path routes. Whereas, the window size increase in AODV-DM is mainly impacted by intra-path contention. Since the window sizes in AODVM case are frequently suppressed, the averages of them are greatly degraded. As a result, the total window size in AODVM case is very small compared with that in AODV-DM case (Figure 4-5(c)). The small window size results in the bad goodput of AODVM, as will be verified when we examine goodput performance later. From Figure 4-5, we see that the benefit of using the path-aware SCTP to control traffic of multipath separately will be impaired if the routes themselves are physically close, which verifies the need to introduce route-decoupled multiple paths to the path-aware SCTP.

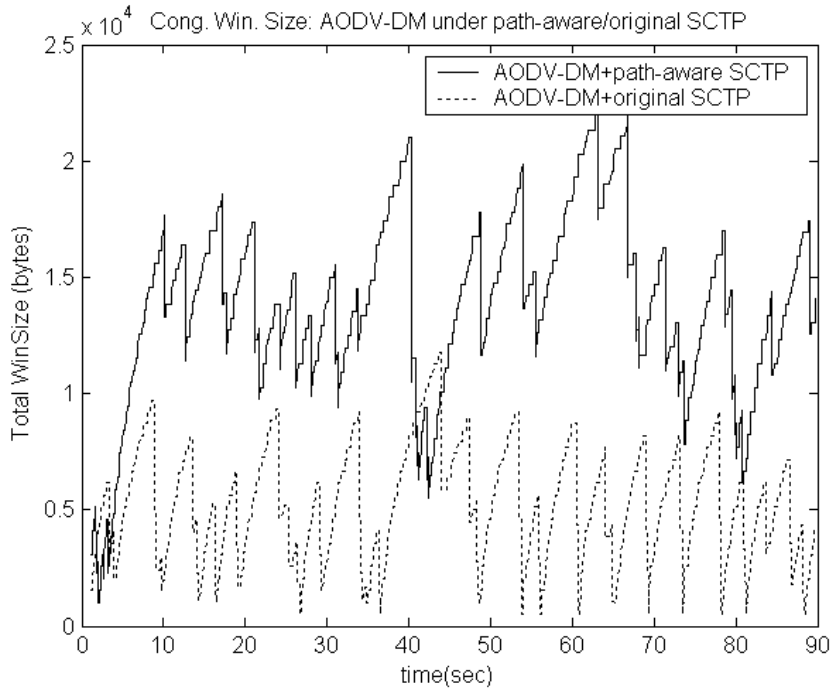


Figure 4-6. Congestion window size comparison: original/path-aware SCTP over AODV-DM

Figure 4-6 compares the congestion window sizes of the path-aware SCTP with the original standard SCTP when they are combined with AODV-DM. It indicates that the window size variation of the original SCTP is unstable compared with the path-aware SCTP. It is because the original SCTP has no mechanism to differentiate congestions from different paths. Hence, it simply decreases its window size whenever data loss of any path occurs. And the total window size of the original SCTP is smaller than that of the path-aware SCTP. Figure 4-6 demonstrates the advantage of the path-aware SCTP over the original standard SCTP.

4.5.2.2 Goodput Analysis

We integrate the original SCTP and the path-aware SCTP with AODVM and AODV-DM in order to examine under which integration can we obtain the best throughput enhancement. We also compare their throughputs with that of TCP over AODV. Figures 4-7 to Figure 4-9 inspect their goodput under different background traffics. From Figure 4-7 and Figure 4-8, we observe that for all tested transport layer protocols, the best improvement is attained whenever AODV-DM is employed. This is attributed to the route-decoupling feature of AODV-DM. Figure 4-9 compares the goodput of the path-aware SCTP and the original SCTP when AODV-DM is adopted and shows that the path-aware SCTP always outperforms the original SCTP. Having seen the congestion window performance in previous section, we expect the best performance to come from the combination of path-aware SCTP and AODV-DM, and the simulations affirmed our judgment.

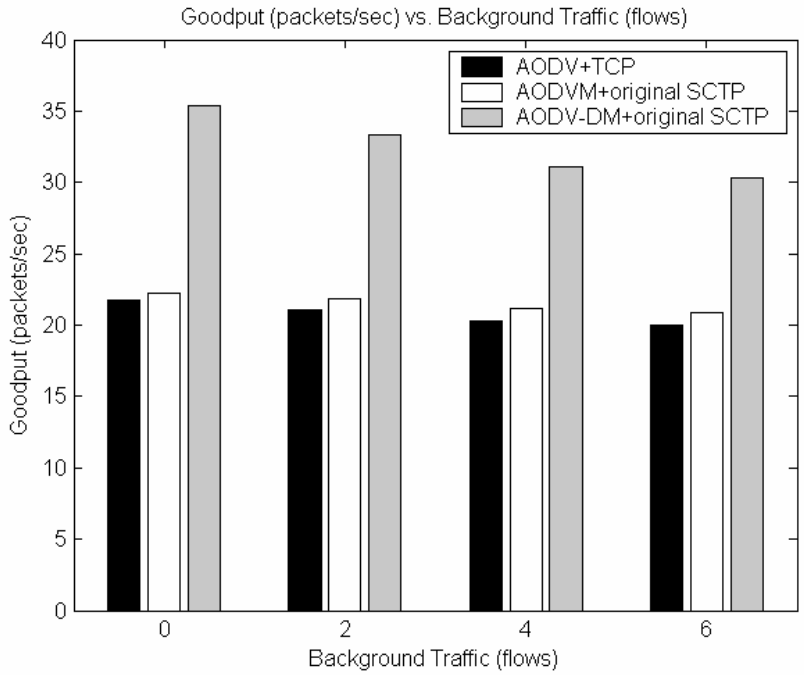


Figure 4-7 Routing protocol comparison under the original SCTP/TCP

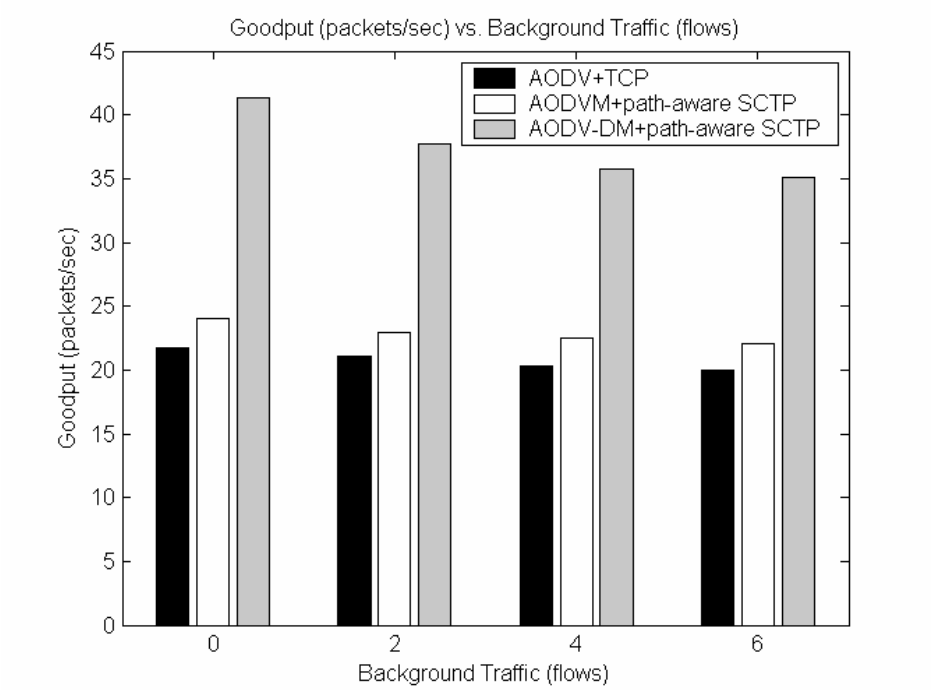


Figure 4-8 Routing protocol comparison under the path-aware SCTP/TCP

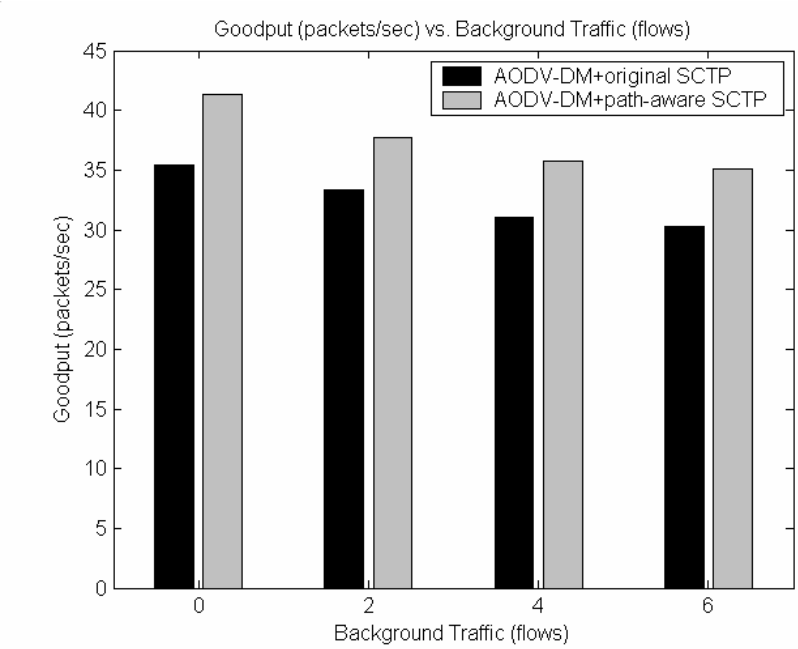


Figure 4-9 SCTP comparison when AODV-DM is applied

5 CONCLUSION AND FUTURE WORK

This thesis describes a set of progressive route discovery protocols, i.e. PRD. To establish good routes with less route discovery overhead, PRD divides the conventional route discovery into two stages: destination discovery and optimal route discovery. First, PRD can employ any efficient routing approach to roughly scan the entire network and constructs a preliminary route between a source node and its desired destination node. For the determination of the preliminary route, both proactive and reactive schemes can be used. Then, during the optimal route discovery stage, it concentrates on the route discovery activity within the local region around the preliminary route to find the optimal route. Both broadcast and unicast schemes are addressed to localize the optimal route discovery process. By using efficient routing techniques to sketchily probe the network and then progressively refine, and derive the optimal route within a small region, not only routing overhead is controlled to be very low, but route optimality is also assured. Both analytical and simulation results show that PRD is suitable for both static and dynamic network environments.

This thesis then extends PRD to route update and multipath discovery stages. At route update stage, a pre-routing region is directly established in the neighborhood of the old route. And a new route is built in the region just like the optimal route is established during the route discovery stage. At multipath discovery stage, an AODV decoupled multipath (AODV-DM) approach is proposed to separates multiple paths with insulating regions so that these paths are not only disjoint, but also separated far enough to avoid inter-path interferences. We describe the decoupled multipath

formation algorithm of AODV-DM as well as the way to optimize the paths. The decoupled feature of AODV-DM makes it particularly suitable to concurrently transfer real time and multimedia data through multiple paths in wireless mesh networks. Performance analysis and simulation results demonstrate the efficiency of our protocols.

Based on the work introduced in the thesis, several improvements can be made in the future:

Simulation study of PRD in mobility environments is to be performed to verify the feasibility of the progressive routing protocols in mobile ad hoc networks.

Continuous efforts can be made to optimize cross-layer designs among the data link layer, the network layer and the transport layer in order to examine the efficiency of the decoupled multipath structure on the wireless mesh networks.

The PRD can be improved to adapt to different application requirements and operational conditions based on predefined policies and the knowledge acquired over time.

APPENDIX: MAJOR ROUTING ALGORITHMS FOR WIRELESS MESH/AD HOC NETWORKS

This appendix introduces some well-known routing algorithms for wireless mesh or ad hoc networks, They are: DSDV (Destination-Sequenced Distance-Vector Routing Protocol) [1], OLSR (Optimized Link State Routing) [6], AODV (Ad Hoc On Demand Distance Vector) [8], DSR (Dynamic Source Routing) [12] and ZRP (zone routing protocol) [53]. Among the five routing algorithms, DSDV and OLSR belong to the proactive routing category, while AODV and DSR belong to the reactive routing category. ZRP is a hybrid routing scheme.

DSDV

The classical distance-vector algorithm is based on Bellman-Ford routing algorithm. In this algorithm, each router maintains a routing table indexed by, and containing one entry for, each route in the subnet. This entry contains two parts: the preferred outgoing line to use for each destination, and an estimate of the time or distance or other metric to that destination. The metric used might be number of hops, time delay in milliseconds, total number of packets queued along the path, or something similar. Each router sends to each of its neighbor a list of its estimated metric to all other destinations periodically or upon the update of the entry information. It also receives a similar list from each of its neighbor. Suppose that a router receives a list from neighbor X, with X_D being estimate of delay time for node X to get to destination D. If the router all knows the delay time from itself to X, then it will

calculate how long it takes for itself to get to destination D via node X. Thereby, by receiving and comparing routing information from each neighbor, a router can find out which estimate seems the best and use that estimates and the corresponding line in its new routing table. Note that the old routing table is not used in the calculation.

In theory, the distance-vector algorithm is computationally efficient, easy to implement and requires little storage space; however, it has a very serious drawback in practical. It reacts rapidly to good news, but leisurely to bad news. To specify the problem, let us consider the situation of Figure A-1, in which all the lines and routers are initially up. Router B, C, D and E have distances to A of 1, 2, 3, and 4, respectively. Suddenly, A goes down, or alternatively, the line between A and B is cut, which is efficiently the same thing from B's point of view. At the first packet exchange, B does not hear anything from A. Fortunately, C says "Do not worry. I have a path to A of length of 2". Little does B know that C's path runs through B itself. For all B knows, C might have ten outgoing line all with independent paths to A of length 2. As a result, B now thinks it can reach A via C, with a path length of 3. D and E do not update their entries for A on the first exchange.

On the second exchanges, C notices that each of its neighbors claims to have a path to A of length 3. It picks one of them at random and makes its new distance to A 4, as shown in the third row of Figure A-1. Subsequent exchanges produce the history shown in the rest of Figure A-1, which shows that the bad news travels slowly. The routing loop problem is known as the count-to-infinity problem. It is very clear that the primary reason of the count-to-infinity problem is that nodes choose their next-hops in

a completely distributed fashion based on information which can possibly be stale and, therefore, incorrect.

In wireless mesh networks, topology is changed more frequently than in wired networks. It is apparent that the classical distance-vector algorithm is not suitable to wireless networks.

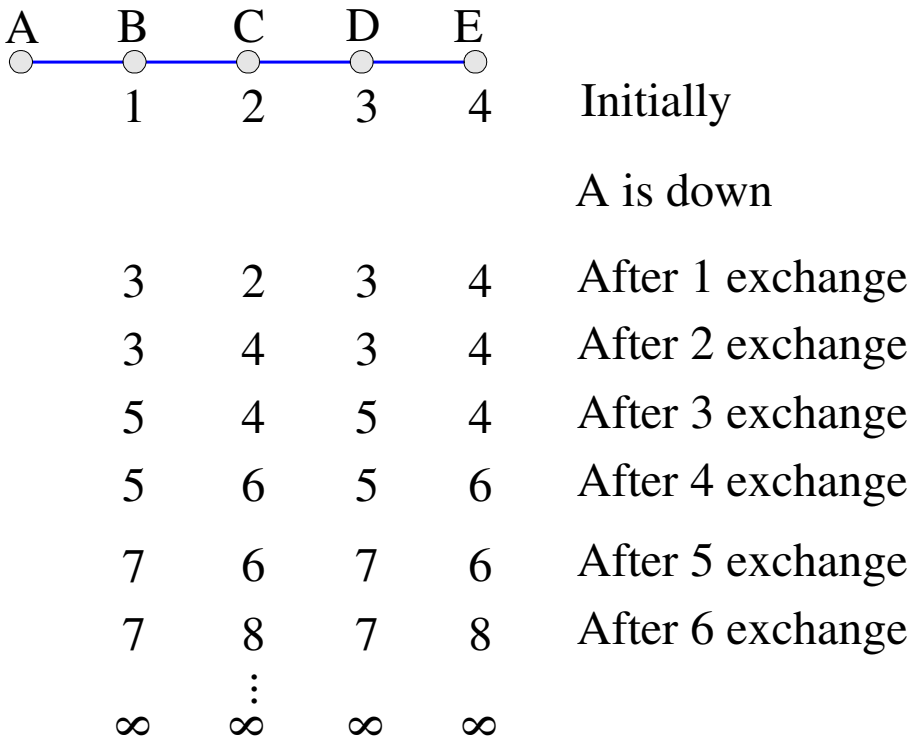


Figure A- 1 Count-to-infinity problem of Distance Vector protocol

To solve the count-to-infinity problem in distance vector algorithm, C. Perkins and P.Bhagwat developed Destination-Sequenced Distance-Vector Routing (DSDV) protocol. In DSDV, each entry in the routing table not only contains the destination address, the preferred outgoing line to the destination, the estimate of the metric to the destination, it also tags each route table entry with a sequence number so that nodes can

quickly distinguish stale routes from the new ones and thus avoid formation of routing loops. The sequence number is generated by the destination, and the emitter needs to send out the next update with this number. Routing information is distributed between nodes by sending *full dumps* infrequently and smaller incremental updates more frequently.

In Figure A-2, the initial sequence number (the number in parentheses) to destination A is zero. After A is down, B increases the sequence number to be one. In this way, B will never updates its routing entry towards A via C, since the routing information recorded by C is already stale. As can be seen from Figure A-2, it takes only 4 packet exchange for node B, C, D, E to update and obtain the latest and correct routing entry towards A, which is much faster than that in figure A-1.

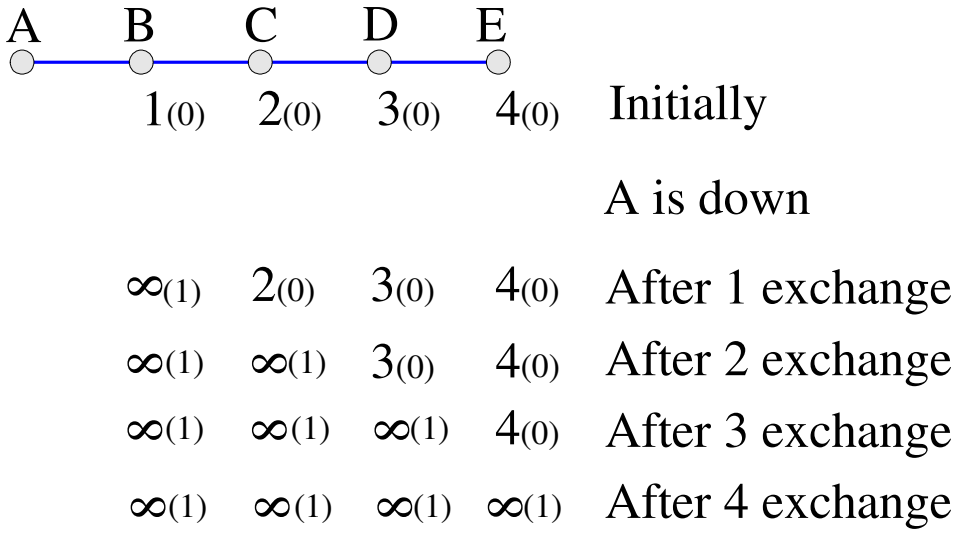


Figure A- 2 DSDV

DSDV is quite suitable for creating ad hoc networks with small number of nodes. However, like all other proactive routing protocol, DSDV requires a regular update of its routing tables, which uses up battery power and a small amount of bandwidth even when the network is idle. Besides, whenever the topology of the network changes, a new sequence number is necessary before the network re-converges; thus, DSDV is not suitable for highly dynamic networks.

OLSR

OLSR intends to reduce the topology update overhead in large and dense networks. It achieves this objective by using a Multipoint Relaying (MPR) technique [36][54].

In classical flooding mechanism, every node is required to relay a newly received flooding packet even if all its neighbors have already got such packet. The redundant packet relays not only waste precious network resources, but also cause packet collisions and delays.

The MPR scheme limits packet forwarder only to a few neighbors of each packet sender. To ensure full coverage, these neighbors should be able to reach all two-hop neighbors of the packet sender.

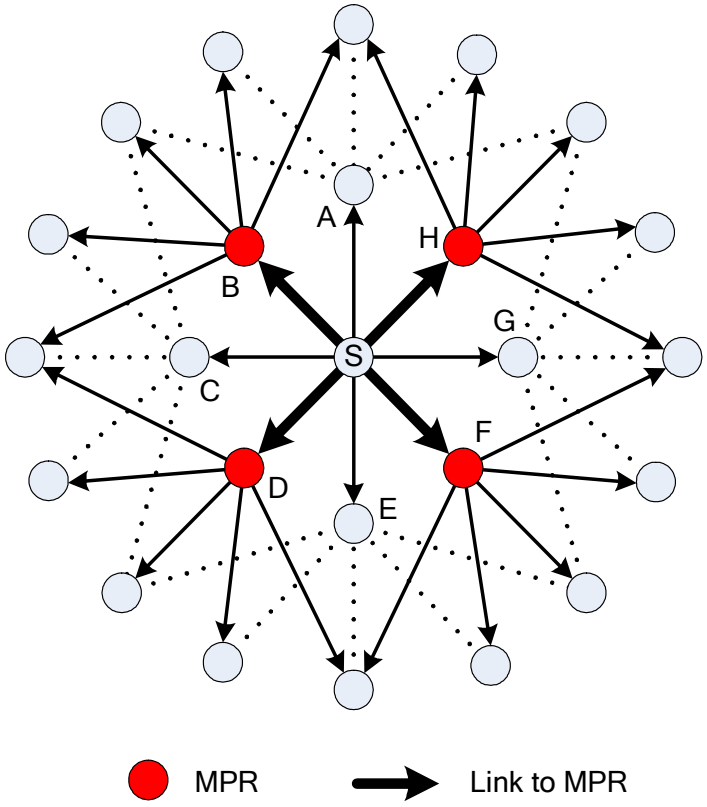


Figure A- 3 Multipoint relaying scheme

Figure A-3 is a typical illustration of multipoint relays. In this example, source S wants to flood a packet to the whole network. The classical flooding mechanism requires all neighbors of S to rebroadcast the packet. However, as the neighbors of A have already been covered by B and H, A does not need to rebroadcast the packet. Neither do C, E, and G. In the MPR scheme, every node maintains a MPR set, which includes the necessary neighbors covering the whole two-hop vicinage. For instance, S includes B, D, F, and H in its MPR set since these four nodes are sufficient to cover all its two-hop neighbors. S also informs the selected MPRs about their role, so that whenever S sends out a new flooding packet, only its MPRs participate in the packet

forwarding. Similarly, B, D, F, and H select their own MPRs to further relay the flooding packet.

To sum up, a node forwards a flooding packet when, 1) the packet has not been received before; 2) the node is an MPR of the last packet sender. Figure A-4 shows an interesting packet forwarding case. Node K is an MPR of H, but not an MPR of F. If K receives a flooding packet from H first, it needs to forward the packet. However, if K receives a flooding packet from F first, it shall not forward the packet even when it receives this packet later from H. The reason behinds this case is that when K receives a flooding packet from F and notices it is not an MPR of F, it knows that all its neighbors can be taken care of by the MPRs of F. So it does not need to forward this packet in any case.

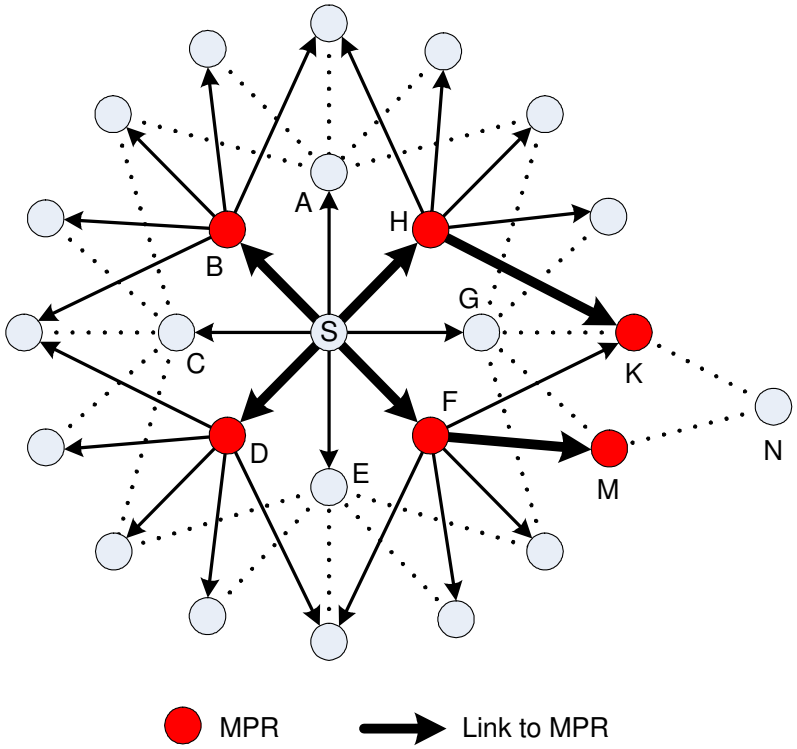


Figure A- 4 A particular packet forwarding case

MPR computation algorithm

Rules: 1) Any two-hop neighbor has to be covered by at least one MPR; 2) The size of the MPR set is minimized.

Definitions:
X : the node computing MPRs
N_X : set of one-hop neighbors of node X
M_X : set of two-hop neighbors of node X
MPR_X : MPR set of node X

1) Initialize the MPR_X as an empty set;
2) Derive a mapping $M_X \rightarrow N_X$, i.e. $\forall k \in M_X$, find all $i \in N_X$, which can reach k;
3) Find all one-to-one maps: $M_X^1 \rightarrow N_X^1$, $M_X^1 \subset M_X$, $N_X^1 \subset N_X$
   a) Subtract M_X^1 from M_X;
   b) Move nodes in N_X^1 from N_X to MPR_X;
4) Find $k \in N_X$ that covers the maximum nodes in M_X
   a) Move k from N_X to MPR_X;
   b) Remove the nodes covered by k from M_X;
5) If $M_X = \varnothing$, end; else, go back to 4).

Figure A- 5 MPR computation algorithm

A heuristic MPR computation algorithm is proposed in [36]. Figure A-5 summaries the major steps of this algorithm.

The MPR scheme enables OLSR to optimize the link-state update process in the following three aspects:

Only nodes selected by their neighbors as MPRs are required to generate link-state update messages (or topology control messages called in the OLSR algorithm).

Each MPR only reports the links between itself and its MPR selectors in topology control messages.

Topology control messages are only forwarded by MPRs of message senders.

By distributing partial topology information to the whole-network in a highly efficient way, OLSR significantly decreases the topology update overhead. The partial topology information is also good enough for each node to calculate the shortest path to every other node. However, it is not sufficient for nodes to derive the optimal routes when route metrics other than hop-count are considered.

One burden imposed by OLSR is to ask each node to collect two-hop neighbor information. To accomplish this, every node includes its one-hop neighbors and corresponding link information in periodically emitted HELLO messages.

AODV

AODV, like most reactive routing algorithms, consists of two processes: route discovery and route maintenance.

When an active source wants to contact a destination, but cannot find any fresh routing entry for the destination, it initiates the route discovery process by flooding a Route REQest (RREQ) packet to the whole network. Any intermediate node, when receiving the RREQ, has to check whether it has received the same RREQ before. If it does, the duplicated RREQ is discarded silently. Otherwise, the intermediate node checks whether it has an existing routing entry for the destination. If it does, the RREQ

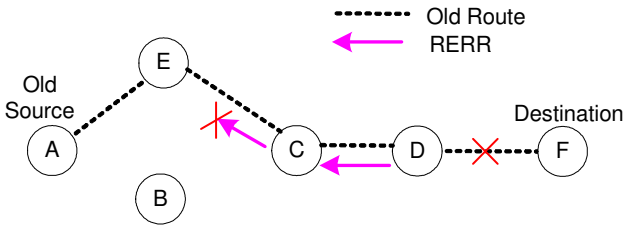
is discarded and a Route REPly (RREP) packet is unicast back to the source. Otherwise, it creates a backward routing entry for the source or updates an existing one, and forwards the RREQ to its neighbors. When the destination receives the RREQ for the first time, it sends a RREP to the source immediately to confirm the newly discovered route. For the later received RREQs, the destination replies with RREPs only if the duplicated RREQs propagate from shorter paths. When receiving a RREP, an intermediate node creates a forward routing entry to the destination. Once the RREP reaches the source, a route is immediately ready for data delivery.

In addition to the pure flooding approach, AODV proposed a "ring-search" scheme as an alternative way to spread RREQs. The "ring-search" scheme makes each active source gradually enlarge RREQ coverage by increasing TTL of the RREQ one by one. For close destinations, the "ring-search" can save a lot of overhead by controlling the RREQ flooding within a small range. However, for a distant destination, the gradual flooding adds extra overhead since repeated efforts are made to search the destination step by step from local area to distant area.

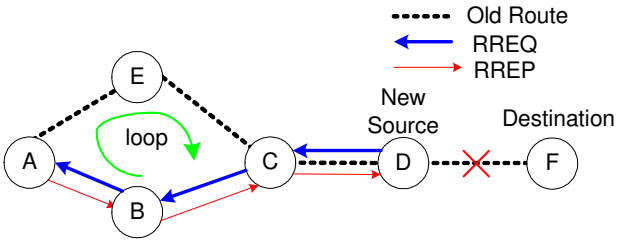
Nodes in active routes may periodically broadcast HELLO messages to provide connectivity information. The route maintenance process is called when a link breakage is detected in an active route. The upstream node of the broken link sends a Route ERRor (RRER) packet to the source to notify the link breakage. If the distance from the broken link to the destination exceeds half of the original route length, the source re-initiates a route discovery process to discover a new route. Otherwise, the upstream node of the broken link takes the responsibility to repair the problematic route by

flooding a RREQ in the local area between it and the destination. In the route repair process, either the source or the local repair initiator can effectively control the RREQ broadcast range based on the length of the old broken route.

In AODV, every entry has an expiration timer. The timer is renewed whenever the entry is consulted to deliver a packet. If this timer does not get any renewal for a long time and finally counts down to zero, the entry is deleted to avoid further maintenance. Compared to proactive routing algorithms, which require each node to store routing entries for every other node, AODV is much more memory efficient since every node only keeps the routing entries for active routes.



(a)



(b)

Figure A- 6 AODV Loop problem

AODV allows an intermediate node to terminate the RREQ propagation if it has an existing routing entry for the destination. Although this strategy accelerates route discovery and reduces RREQ transmission overhead, it may result in a loop route. Figure A-6 shows an example of the loop problem. In Figure A-6(a), node A has a fresh routing entry for F, which corresponds to the route A-E-C-D-F. Now link D-F breaks, and the RERR originated from D is lost during the transmission from C to E. As a result, A and E are not aware of the link change between D and F. In Figure A-6(b), D initiates a route discovery process to search for a new route to F. The RREQ from D is relayed by C and B and arrives at A. Node A finds that it has an existing routing entry for F, so it discards the RREQ and sends a RREP back to D. Node B, C, and D creates a new routing entry for F with the next-hop as A, B, and C respectively. A loop route C-B-A-E-C is thus formed.

AODV uses node sequence number to prevent the creation of loop routes. Every node maintains a sequence number, which is used to differentiate the outdated routing entries and the updated routing entries for this node. In the example of Figure A-7, originally all routing entries for F have a sequence number of 2. When D detects the breakage of link D-F, it increases the sequence number of F to 3 and includes the new sequence number in the RREQ. Once A receives the RREQ and finds the sequence number in its entry is less than that in the RREQ, it knows that its entry is an outdated one. Thereby, A needs to forward the RREQ to continue the route discovery. When the RREQ loops back to node C, it is discarded as a duplicated RREQ.

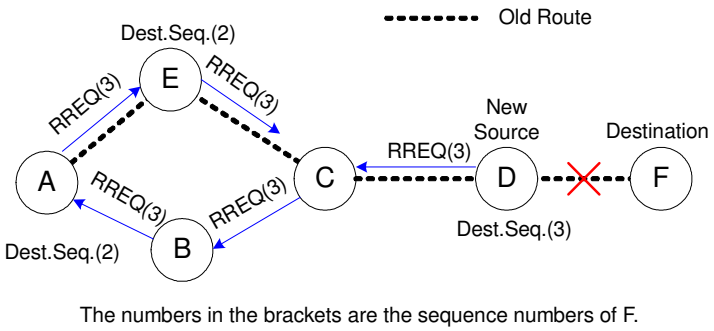


Figure A- 7 Sequence number is used to prevent loop

DSR

One common point of OLSR and AODV is that all of them establish routing tables at relay nodes to guide packet transmissions. DSR does not bother intermediate nodes to record routing information. Each active source stores the entire route, i.e. the so-called source route, in its route cache. This source route information is put in the header of every data packet to help relay nodes correctly forward the packet.

DSR also includes a route discovery process and a route maintenance process, which are very similar to those of AODV.

When an active source does not find any source route to a destination in its route cache, it floods a RREQ to the whole network. Each intermediate node receiving the RREQ appends its address into the RREQ and rebroadcasts the packet if it receives the RREQ for the first time and does not cache any route to the destination. If the intermediate node does cache a route to the destination, it is able to figure out a complete source route from the active source to the destination. It terminates the RREQ

propagation and sends a RREP back to the source with the complete source route. The destination, when receiving the RREQ, takes out the source route information from the RREQ and sends this information back to the source via a RREP.

Every relay node in an active route is responsible to monitor the status of the link between itself and the next-hop node. When link breakage happens, the upstream node of the broken link sends a RERR back to the source following the reverse source route. All nodes receiving the RERR delete the broken link from their route caches. If the source has a backup route in its route cache, it switches to that route immediately. Otherwise, it reinitiates a route discovery process to find a new route.

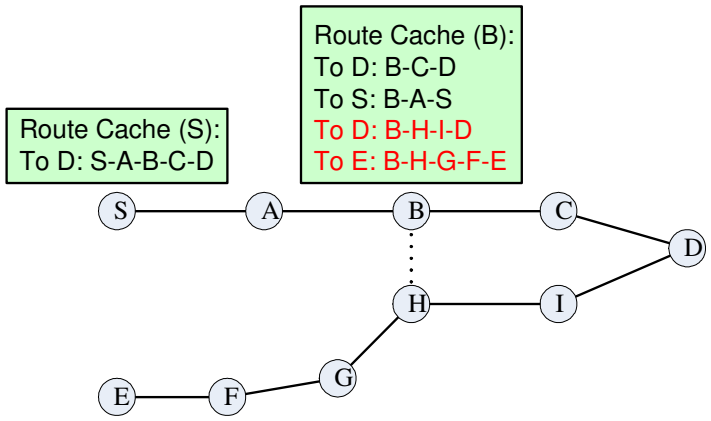


Figure A- 8 Example of route caching

DSR aggressively uses route caching to accelerate route discovery and reduce routing overhead. For example, relay nodes may caches source routes contained in any packets for potential usages in the future. Destinations reply all RREQs to make sources learn alternative routes. Most importantly, nodes are allowed to cache the routing information overheard from their neighbors' transmissions. Figure A-8 shows

an example of route caching. Node S delivers data packets to node D through S-A-B-C-D. Every data packet carries the source route (S-A-B-C-D) in its header. From these data packets, the relay node B learns the routes to S and D. It stores these routes, B-C-D and B-A-S, in its route cache for future usage. There is another active route E-F-G-H-I-D passing through B's neighborhood. The packets transmitted form H to I can be overheard by B. Based on the source route carried in these packets, B figures out another route to D (B-H-I-D) and a route to E (B-H-G-F-E). B can also keep these two routes in its route cache.

There are several optimizations that can be applied to the basic operations of route discovery and maintenance [55].

Packet salvage – When meeting a broken link, a relay node sends a RERR to the source to report the link change. At the same time, it consults its route cache to see whether it stores an alternative route. If it does, the blocked data packets are immediately forwarded through the cached route. In the example of Figure A-8, if link B-C is broken, B can relay the stuck packets through B-H-I-D, which is overheard from H's transmissions.

Gratuitous link update – When a source receives a RRER and reinitiates a route discovery process, it may piggyback the broken link information in the RREQ. So that the link change can be quickly spread to nodes which still have this link in their cached routes.

Dynamic route optimization – When a node overhears its neighbors' transmissions and finds it caches a better route (other than the route passing its neighbors). It may send a *gratuitous reply* packet to the source to notify this better choice. Figure A-9 shows an example of dynamic route optimization. Node S is sending data packets to D through S-A-B-C-D. Now node H moves to the neighborhood of A, B, C, and D, and overhears packets transmitted from all these nodes. From the packet headers, H learns that S is using S-A-B-C-D as the source route. However, it finds there is a shorter route through itself, i.e. S-A-H-D. So, node H notifies S about the shorter route through a *gratuitous reply* packet.

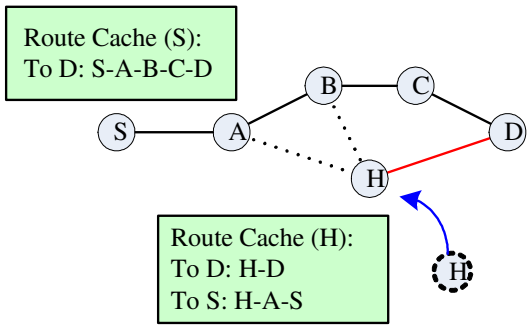


Figure A- 9 Example of dynamic route optimization

Ring search – This is similar to the ring search scheme used in AODV. Each active source gradually increases the RREQ coverage instead of flooding it at the beginning. If route caching and overhearing are widely applied, it is beneficial for sources to check their neighbors first for possible route caches.

ZRP

ZRP makes each node proactively maintain a local routing zone, so that it can calculate the optimal route to reach every other node within the zone. Reactive routing process is initiated only when one node wants to contact a destination outside its routing zone. Proactive maintenance of local zones enables nodes to quickly detect topology changes in their neighborhoods, while preventing topology updates spreading too far to cause huge overhead. Also, if every node stores more than one-hop local information, route discovery across the network can be efficiently performed without involving all network nodes.

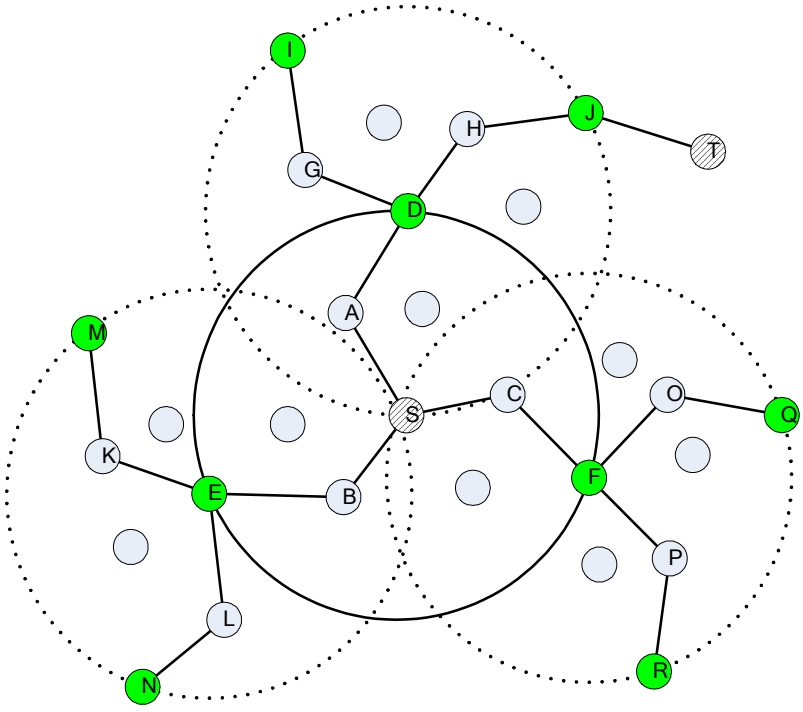


Figure A- 10 An example of bordercast

ZRP consists of three modules: intrazone routing protocol, interzone routing protocol, and bordercast resolution protocol. Any proactive routing algorithm can be employed in the intrazone routing module, providing that topology update activities are confined to the local zone of each node. For example, in a link-state based intrazone routing protocol, every node shall set the broadcast range of link-status messages equal to its local zone radius. The interzone routing protocol can be based upon any reactive routing algorithm, such as AODV and DSR. Instead of the traditional flooding based route discovery mechanism, the interzone routing protocol makes use of the local topology information stored at each node to efficiently spread RREQs. This efficient RREQ forwarding technique is called bordercast. Figure A-10 shows an example of bordercast. Source S initiates an interzone route discovery process to reach destination T, which is outside S's two-hop zone. Instead of flooding a RREQ out, S unicasts the RREQ to each border node of its local zone. In this way, only a fraction of nodes within its zone need to forward the RREQ. The border nodes of S's local zone, D, E, and F, further unicast the RREQ to the border nodes of their zones. When node J receives the RREQ from D, it finds out that destination T is inside its local zone. So it forwards the RREQ to T directly. T can then sends a RREP back to S to confirm the newly discovered route.

The bordercast mechanism can be further optimized to eliminate unnecessary or redundant network queries [56]. There are mainly three optimization methods:

Loop-back termination (LT): for a DSR based interzone routing protocol, it is easy to identify a loop-back RREQ by checking the accumulated source route stored in the packet. Once a node detects that the received RREQ contains its local zone member (except the immediate sender), it discards this packet right away.

Query detection (QD): Whenever a node bordercasts/relays a RREQ or overhears a RREQ relayed by its neighbors, it keeps a record of this RREQ. So that when a same RREQ arrives at a later time, it can detect the duplication and refuse to bordercast/relay the packet again.

Selective bordercasting (SBC): This idea is similar to the MPR scheme adopted in OLSR. In order to achieve this optimization, every node needs to collect the topology information of a larger zone, twice the size of its routing zone. So that when doing bordercast, a relay node only needs to send RREQs to a subset of its zone border nodes, which can completely cover all border nodes of the larger zone.

Figure A-11 shows an example of QD and SBC. Although node S has 8 two-hop neighbors, three of them (D, E, and F) are enough to cover all four-hop neighbors. So S only needs to send RREQs to these three border nodes. When S transmits a RREQ to D through A, node Z overhears the transmission. Node Z will refuse to relay the same RREQ sent from D to C since it knows that C has already been covered.

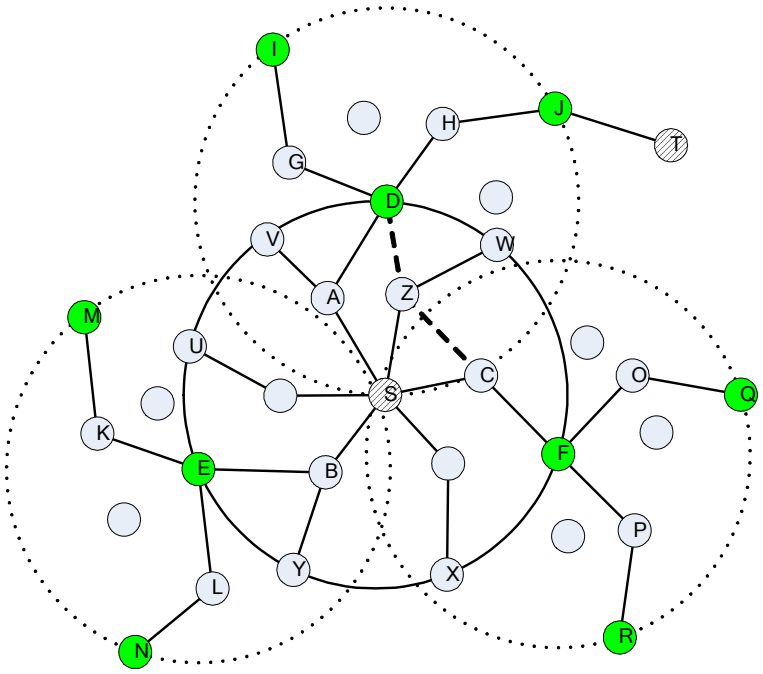


Figure A- 11 Bordercast optimization

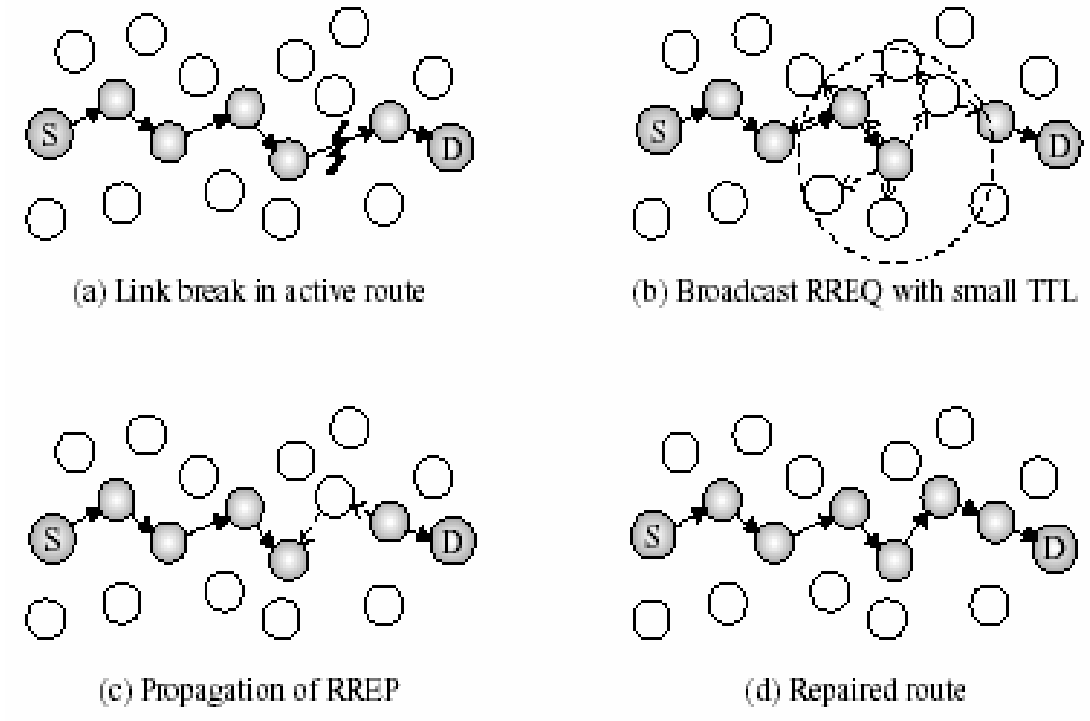
AODV Local Repair

To avoid the omni-directional broadcasts of the route discovery packets, AODV [8] and Query Localization (QL) [20] provide local repair mechanisms.

In the current AODV specification, when a link break in an active route occurs, the node upstream of the break creates a Route Error (RERR) message listing all the destinations which have become unreachable due to the break. It then sends this message to its upstream neighbors. If, instead of sending an error message to the source node, the upstream node attempts to repair the broken link itself, fewer data packets may be lost and the link can be repaired without the source node (and other upstream nodes) being disturbed.

A node upstream of a link break that attempts to repair the route does so by broadcasting a RREQ with a TTL set to the last known distance to the destination, plus an increment value. This TTL value is used so that only the most recent whereabouts of the destination will be searched, which prevents flooding the entire network. The upstream node places the sequence number of the destination, incremented by one, into the RREQ. This prevents nodes further upstream on the route from replying to the RREQ, which would form a loop. Figure 2-2 illustrates an example of a local repair.

If a route to the destination is not located on the first attempt, a RERR message is sent back to the source node, and route re-discovery continues as described in AODV expanding ring search [8].



(a) Link break in active route

(b) Broadcast RREQ with small TTL

(c) Propagation of RREP

(d) Repaired route

Figure A- 12 Example of AODV local repair

AODV local repair tries to fix a broken link locally by sending RREQ from the upstream node of the broken link with a small TTL. However, the updated route may become longer than the optimal path (shortest path) and with the continuous movement of the endpoint, the route is extended longer and longer. As an example, Figure A-13 illustrates the progress of route extension with local repair mechanism of AODV. When the sink keeps on moving, the route is extended accordingly (the gray solid path represents the extended parts of the route). By comparing the extended route with the optimal new route, we conclude that local repair greatly increases the overhead and the end-to-end delay of the data packets when used in the cases of mobile endpoints.

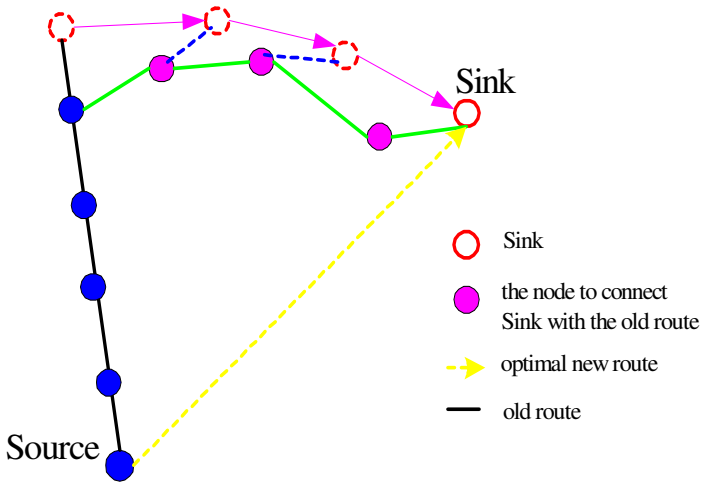


Figure A- 13 Route extension with local repair

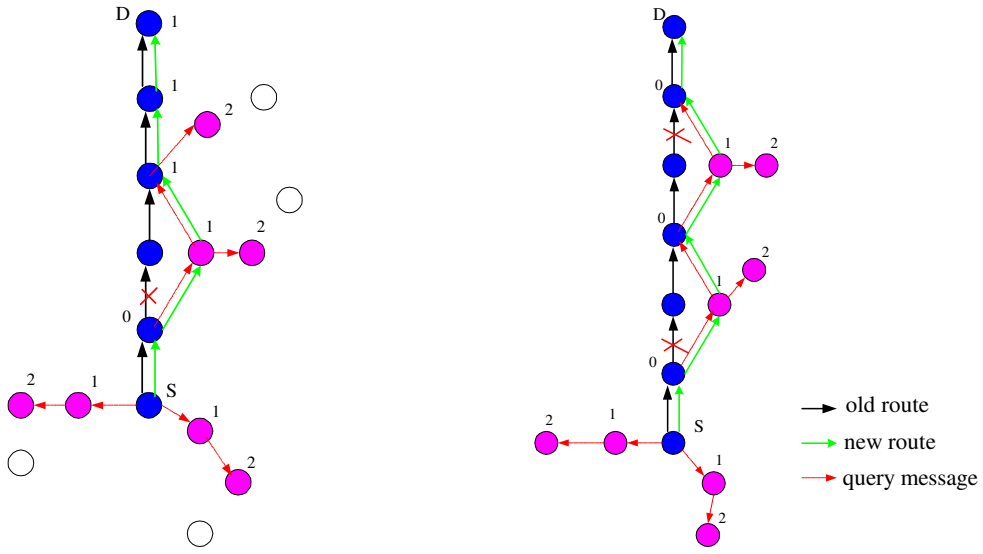
QL

In [20], a Query Localization (QL) scheme is proposed to utilize the directional information implied in the old route for efficient route repair. QL makes use of old

routing histories to localize the query flood to a limited region of the network. There are two independent algorithms in QL. In Algorithm 1, after a query which contains a counter initialized as zero is sent out to repair a broken link, if the node on the old route receives the query, it doesn't change the counter; when the node not on the old route receives the query, it increments the counter. The query is dropped once the counter exceeds a threshold. Algorithm 2 uses the same rules except that the counter is reset to be zero when the node on the old route receives the query. Figure A-14 illustrates two algorithms of QL with threshold set as 1. The black nodes are on the old route and the gray nodes are the nodes that qualify for forwarding the query. The value of the counter is shown at each node visited by the query after the route change. And the final new routes are found and shown as gray paths.

QL is very efficient to locally repair sporadic node or link failures. However, it compromises route optimization. The route prolonging problem appearing in AODV local repair case also happens here.

Another problem about QL is that it is very hard to use QL to discover a new route that is mostly disjoined with the old one. In sensor networks, when one route is used for a long time, most nodes along the route wear out; Then, a new fresh route should be found to replace the old one so that the whole network can maintain good energy-balancing and survive longer. Apparently, QL is not good at this kind of route-wide update.



a) Algorithm 1 b) Algorithm 2

Figure A- 14 A simple illustration of QL (threshold =1)

REFERENCES

[1] C. E. Perkins and P. Bhagwat. "Highly dynamic Destination Sequenced Distance-Vector routing (DSDV) for mobile computers," in *Proceedings of the SIGCOMM'94*, pages 234-244, August 1994

[2] T.-W. Chen and M. Gerla, "Global state routing: A new routing scheme for ad-hoc wireless networks," In Proceedings of IEEE ICC'98, Atlanta, GA, Jun. 1998, pp. 171-175.

[3] J. T. Moy, "OSPF: Anatomy of an Internet routing protocol," 3rd printing, Sept. 1998, ISBN 0-201-63472-4.

[4] L. Hester, Y. Huang, A. Allen, O. Andric, and P. Chen, "neuRFonTM Netform: A self-organizing wireless sensor network," Proceedings of the 11th IEEE ICCCN Conference, Miami, Florida, Oct. 2002.

[5] G. Pei, M. Gerla, and T.-W. Chen, "Fisheye state routing: A routing scheme for ad hoc wireless networks," Proc. ICC 2000, New Orleans, LA, Jun. 2000

[6] P. Jacquet *et al.*, "Optimized Link State Routing Protocol," draft-ietf-manetolsr-05.txt, Internet Draft, *IETF MANET Working Group*, Nov. 2000

[7] Standard draft of 802.11s tree routing.

[8] C. E. Perkins, E. M. Belding-Royer, and S. Das, "Ad Hoc On Demand Distance Vector (AODV) Routing," draft-ietf-manet-aodv-11.txt, *IETF MANET Working Group*, June 2002

[9] I. Chakeres and L. Klein-Berndt, "AODVjr, AODV simplified," ACM SIGMOBILE Mobile Computing and Communications Review, Jul. 2002.

[10] C.-K. Toh, "Associativity-based routing for ad-hoc mobile networks," Wireless Personal Communications Journal: Special Issue in Mobile Networking and Computing Systems, Kluwer Academic Publishers. Vol. 4, No.2, Mar. 1997, pp103-139.

[11] M, Günes et. al., "ARA - the ant-colony based routing algorithm for manets," In Stephan Olariu, editor, Proceedings of the 2002 ICPP Workshop on Ad Hoc Networks (IWAHN 2002), pages 79-85, IEEE Computer Society Press, Aug. 2002.

[12] D. Johnson, D. A. Maltz, and J. Broch, “The dynamic source routing protocol for mobile ad hoc networks,” IETF Internet draft, draft-ietf-manet-dsr-09.txt, Apr. 2003.

[13] [TORA] V. D. Park and M. S. Corson, “A highly adaptive distributed routing algorithm for mobile wireless networks,” Proc. InfoCom’97, Apr. 1997, 9 pages.

[14] I. Stojmenovic and J. Wu, "Broadcasting and Activity-Scheduling in Ad Hoc Networks," *Ad Hoc Networking*, IEEE Press, 2004.

[15] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network,” *ACM Wireless Networks*, Vol. 8, No. 2, March 2002.

[16] J. Cartigny, and D. Simplot, “Border node retransmission based probabilistic broadcast protocols in ad-Hoc networks,” *Telecommunication Systems* 22(1-4), 2003.

[17] M. Sun, W. Feng and T.H. Lai, "Location aided broadcast in wireless ad hoc Networks," *Proc. IEEE GLOBECOM 2001*, November 2001.

[18] Y.-B. Ko and N.H. Vaidya, “Location-aided routing (LAR) in mobile ad hoc Networks,” *Proc. ACM/IEEE MOBICOM’98*, Oct. 1998.

[19] Brad Karp H. T. Kung, *“GPSR: Greedy Perimeter Stateless Routing for Wireless Networks”,* MobiCom 2000.

[20] R. Castenada, S. R. Das, and M. K. Marina, “Query localization techniques for on-demand routing protocols in ad hoc networks,” *ACM/Kluwer Wireless Networks Journal*, Vol. 8, No. 2, March 2002

[21] M. R. Pearlman, Z. J. Haas, P. Sholander, and S. S. Tabrizi, “On the Impact of Alternate Path Routing for Load Balancing in Mobile Ad Hoc Networks,” Proc. ACM MobiHoc, 2000.

[22] D. Saha, S. Roy, S. Bandyopadhyay, T. Ueda and S. Tanaka, “An Adaptive Framework for Multipath Routing via Maximally Zone-Disjoint Shortest Paths in Ad hoc Wireless Networks with Directional Antenna”, Proc. IEEE GLOBECOM 2003.

[23] V. D. Park and M. S. Corson, “A highly adaptive distributed routing algorithm for mobile wireless networks,” Proc. InfoCom’97, Apr. 1997, 9 pages.

[24] G. Koh, D. Oh1 and H. Woo, "A graph-based approach to compute multiple paths in mobile ad hoc networks," Lecture Notes in Computer Science (LNCS), Springer-Verlag, Heidelberg, Germany, Number 2713, pages 323-331, Jun. 2003.

[25] S.-J. Lee and M. Gerla, "Split Multipath Routing with Maximally Disjoint Paths in Ad Hoc Networks," IEEE International Conference on Communications (ICC), Volume 10, June 2001.

[26] J. Wu, "An Extended Dynamic Source Routing Scheme in Ad HocWireless Networks", Proc. the 35th Hawaii International Conference on System Sciences, 2002.

[27] Z. Ye, S. V. Krishnamurthy, and S. K. Tripathi, "A Framework or Reliable Routing in Mobile Ad Hoc Networks," Proc. IEEE INFOCOM, 2003.

[28] D. Saha, S. Roy, S. Bandyopadhyay, T. Ueda and S. Tanaka, "An Adaptive Framework for Multipath Routing via Maximally Zone-Disjoint Shortest Paths in Ad hoc Wireless Networks with Directional Antenna", Proc. IEEE GLOBECOM 2003.

[29] K. Wu and J. Harms, "On-Demand Multipath Routing for Mobile Ad Hoc Networks", EPMCC 2001.

[30] S. De, C. Qiao H. Wu, "Meshed multipath routing with selective forwarding: An efficient strategy in wireless sensor networks," Computer Networks 43 (2003) p481–p497.

[31] D. Ganesan, R. Govindhan, S. Shenker, and D. Estrin, "Highly resilient, energy efficient multipath routing in wireless sensor networks," Mobile Computing and Communications Review (MC2R), Vol. 1, No. 2, 2002.

[32] A. Nasipuri and S. Das, "On-demand multipath routing for mobile ad hoc networks," In Proc. IEEE ICCCN'99, Oct. 1999.

[33] Y. Li, H. Man, J. Yu, Y.-D. Yao, "Multipath routing in ad hoc networks using directional antennas," Advances in Wired and Wireless Communication, 2004 IEEE/Sarnoff Symposium, Apr. 2004, Pages 119-122.

[34] A. Valera, W. K. G. Seah, S. V. Rao, "Cooperative packet caching and shortest multipath routing in mobile ad hoc networks," InfoCom 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Vol. 1, Pages: 260-269

[35] I. Stojmenovic, M. Seddigh, J. Zunic, Dominating sets and neighbor elimination based broadcasting algorithms in wireless networks, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 13, No. 1, January 2002.

[36] A. Laouiti, A. Qayyum et L. Viennot, "Multipoint Relaying: An Efficient Technique for Flooding in Mobile Wireless Networks," in *Proceedings of the 35th Annual Hawaii International Conference on System Sciences,* Big Island, Hawaii, Jan., 2002.

[37] QOLSR: Hakim Badis, Khaldoun Al Agha and Anelise Munaretto, "Quality of Service for Ad hoc Optimized Link State Routing Protocol (QOLSR)", IETF draft, draft-badis-manet-qolsr-00.txt, Washington, DC, USA, November 2004

[38] ZigBee Network Specification, V 1.0, Dec. 2004.

[39] Andrew S. Tanenbaum, Computer Networks, *Prentice Hall,* Third Edition.

[40] C. -K. Toh, "Maximum Battery life Routing to Support Ubiquitous Mobile Computing in Wireless Ad Hoc Networks," *IEEE Communication Magazine*, June 2001

[41] W. Ye, J. Heidemann and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks", *In Proceedings of the IEEE Infocom,* pp. 1567-1576. New York, NY, USA.

[42] Yong Liu, Xuhui Hu, Myung Lee, and Tarek Saadawi, "A Region-based Routing Protocol for Wireless Mobile Ad Hoc Networks," *IEEE Network Magazine*, Vol. 18, Issue 4, July/August, 2004.

[43] C.L. Fuller and J.J. Garcia-Luna-Aceves, " Solutions to hidden terminal problems in wireless networks," in *Proceedings of the ACM SIGCOMM'97*, Cannes, French Riviera, France, Sept. 1997

[44] Ahmed Abd El Al, Tarek Saadawi, Myung Lee, Improving throughput and reliability in mobile wireless networks via transport layer bandwidth aggregation, Computer Networks: The International Journal of Computer and Telecommunications Networking, Volume 46, Issue 5 (December 2004), Pages: 635 – 649.

[45] Guanhua Ye, Saadawi, T.N., Myung Lee, Independent per Path Congestion Control for Reliable Data Transmission between Multi-homed Hosts, 2004 IEEE Sarnoff Symposium on Advances in Wired and Wireless Communications, April 26- 27, 2004**,** Princeton, NJ.

[46] Haejung Lim, Kaixin Xu, Mario Gerla, "TCP Performance over Multipath Routing in Mobile Ad Hoc Networks", ICC '03, Vol. 2 (2003), pp. 1064-1068 vol.2.

[47] Zhenqiang Ye, Srikanth V. Krishnamurthy, Satish K. Tripathi, "Effects of Multipath Routing on TCP Performance in Ad Hoc Networks", Globecom 2004) , Dallas, Texas, USA, Nov. 29--Dec. 3, 2004, pp.4125-4131.

[48] R. Stewart, Q. Xie et al., Stream Control Transmission Protocol,. IETF RFC 2960,Oct. 2000.

[49] Jochen H. Schiller, "Mobile Communications," Addison Wesley, 2000

[50] Elizabeth M. Royer, Chai-Keong Toh, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks," *IEEE Personal Communications*, Vol. 6, No. 2, pp. 46-55, April 1999

[51] F. Ye, H. Luo, J. Cheng, S. Lu and L. Zhang, "A Two-tier Data Dissemination Model for Large-scale Wireless Sensor Networks," in *Proceedings of the MOBICOM'2002*, Atlanta, September 2002

[52] N. Nikaein, C. Bonnet and N. Nikaein, "HARP - Hybrid ad hoc routing protocol," In proceeding of IST, Iran, Tehran 2001.

[53] Z. J. Haas and M. R. Pearlman, "The zone routing protocol for ad hoc networks," IETF: draft-ietf-manet-zone-zrp-02.txt, Jun. 1999.

[54] P. Jacquet, A. Laouiti, P. Minet, and L. Viennot, "Performance of multipoint relaying in ad hoc mobile routing protocols," in *Proceedings of the Networking 2002*, Pise, Italy, 2002

[55] Johnson, D. and Maltz, D. "Dynamic source routing in ad hoc wireless networks," Mobile Computing (ed. T. Imielinski and H. Korth), Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996

[56] Z. Haas and M. Pearlman. "The Performance of Query Control Schemes for the Zone Routing Protocol," in *Proceedings of the ACM SIGCOMM'1998*, Vancouver, Canada, September 1998.

[57] B. S. Manoj, R. Ananthapadmanabha, and C. Siva Ram Murthy, "Link life based routing protocol for ad hoc wireless networks", Proc. of the 10th IEEE International Conference on Computer Communications 2001 (IC3N 2001), Oct. 2001.

[58] Aggelou G, Tafazolli R, "RDMAR: A bandwidth-efficient routing protocol for mobile ad hoc networks," In Proceedings of ACM MobiCom'99/WoWMoM99 , Washington, USA, Aug. 1999.

[59] R. Dube et al., "Signal stability based adaptive routing for ad hoc mobile networks," IEEE Pers. Comm., Feb. 1997, pp. 36-45.

[60] B. Awerbuch et al., "High throughput route selection in multi-rate ad hoc wireless networks," Johns Hopkins University Technical Report, Mar. 2003.

[61] D. S. J. De Couto, et al., "A high-throughput path metric for multi-hop wireless routing," Proceedings of the 9th ACM International Conference on Mobile Computing and Networking (MobiCom'03), San Diego, California, Sept. 2003.

[62] Comparison of Routing Metrics] R. Draves, J. Padhye and B. Zill, "Comparison of routing metrics for static multi-hop wireless networks," ACM SigComm, Portland, OR, Aug. 2004.

[63] R. Draves, J. Padhye and B. Zill, "Routing in multi-radio multi-hop wireless mesh networks," MobiCom 2004, Philadelphia, PA, Sept. 2004.

[64] C. Zhu, M. J. Lee and T. Saadawi, "RTT-based optimal waiting time for the best route selection in ad hoc routing protocols", Proceeding of IEEE MilCom 2003, Military Communication Conference, Boston, MA, Oct. 13-16, 2003.

[65] X.-Y. Li, et al., "Coverage in wireless ad hoc sensor networks," IEEE Trans. on Computers, Vol. 52, pp. 753-763, Jun. 2003.

[66] Jianliang Zheng, Myung Lee, "Will IEEE 802.15.4 make ubiquitous networking a reality?," *IEEE Communications Magazine,* vol. 42, no. 6, pp. 140 – 146, Jun 2004

[67] J. Li, C. Blake, D.S.J.De Couco, H. Lee, R. Morris, "Capacity of Ad Hoc Wireless Networks," in *Proceedings of the ACM SIGMOBILE'2001*, Rome, Italy, July 2001

[68] P. Karn, "MACA – A new channel access method for packet radio," in *Proceedings of the ARRL/CRRL Amateur Radio 9th Computer Networking Conf.,* pp134 –140,1990,

[69] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A media access protocol for wireless Lan's," in *Proceedings of the ACM SIGCOMM'1994*, pp. 212-225, London, Sept 1994

[70] Ajay Chandra V. Gummalla and John O. Limb, "Wireless Medium Access Control Protocols," *IEEE Communications Surveys*, Second Quarter 2000

[71] Tao Lin, Scott F. Midkiff and Jahng S. Park, "A Framework for Wireless Ad Hoc Routing Protocols," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, New Orleans, March 2003

[72] Xuhui Hu, Yong Liu, Myung Lee, and Tarek Saadawi, "Efficient Route Update Protocol for Wireless Sensor Networks," in *Proceedings of the Milcom'03*, Boston, MA, Oct. 2003

[73] Xuhui Hu, Yong Liu, Myung Lee, and Tarek Saadawi, "Route Update and Repair in Wireless Sensor Networks," in *Proceedings of the CCNC' 04*, Las Vegas, MA, Jan. 2004

[74] C. A. Santivanez, B. McDonald, I. Stavrakakis, and R. Ramanathan, "On the Scalability of Ad Hoc Routing Protocols," in *Proceedings of the IEEE INFOCOM 2002*, New York, 2002

[75] M. Gerla, X. Hong, L. Ma, and G. Pei, "Landmark routing protocol for large scale ad hoc networks," IETF: draft-ietf-manet-lanmar-03.txt, Dec. 2001.

[76] H. Frey, "Scalable geographical routing algorithms for wireless ad hoc networks," IEEE Network, Vol. 18, Issue 4, Jul.-Aug. 2004.

[77] Y. Huang, et al., "OPNET Simulation of A Multi-hop Self-organizing Wireless Sensor Network," in *Proceedings of the OPNETWORK 2002 conference*, Washington D.C., August 2002

[78] D. S. J. De Couto, et al., "A high-throughput path metric for multi-hop wireless routing," Proceedings of the 9th ACM International Conference on Mobile Computing and Networking (MobiCom'03), San Diego, California, Sept. 2003.

[79] R. Draves, J. Padhye and B. Zill, "Comparison of routing metrics for static multi-hop wireless networks," ACM SigComm, Portland, OR, Aug. 2004.

[80] A. Iwata, C-C. Chiang, G. Pei, M. Gerla, and T-W. Chen, "Scalable routing strategies for ad hoc wireless networks," IEEE JSAC, Aug. 1999, Vol. 17, No. 8, pp. 1369-79.

[81] Y. Ganjali and A. Keshavarzian, "Load Balancing in Ad Hoc Networks: Single-Path Routing vs. Multi-Path Routing," IEEE INFOCOM, 2004.