INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

Multipath Routing for Wireless Mesh Networks

Cristina Neves Fonseca

Dissertação para obtenção do Grau de Mestre em
Engenharia de Redes de Comunicações

Júri

Presidente:	Prof. Doutor Rui Jorge Morais Tomaz Valadas
Orientador:	Prof. Doutor Luís Eduardo Teixeira Rodrigues
Vogal:	Prof. Doutor Luís Filipe Lourenço Bernardo

October 2010

Acknowledgements

My first acknowledgement goes to my advisor Professor Luís Rodrigues. I am really grateful for having the pleasure to work with such a wise person. Thank you for our enlightening discussions, for the incessant motivation in the most difficult moments and for all the help along this sinuous path.

I also thank to all my colleagues from the GSD group at INESC-ID for all the support and their seamless help in keeping me motivated.

Thanks to Martijn Kuipers from IT who was a key part in the evaluation process.

I would like to express my profound gratitude to my parents for their unconditional love and support, and to my brother César and my sister Carina for their endless friendship.

Last but not least, I want to manifest my most affective thanks to Tiago, among other reasons, for being always there.

This work was partially supported by FCT in the context of project "Redico" (PTDC/EIA/71752/2006) and by INESC-ID multiannual funding, through the PIDDAC Program funds.

Lisboa, October 2010

Cristina Neves Fonseca

For my parents,

Maria da Purificação and Américo.

Resumo

Esta tese aborda o problema do encaminhamento multi-rota em redes sem fios em malha. Neste contexto, propomos um algoritmo de agrupamento de nós que facilita a escolha de múltiplas rotas que não interfiram mutuamente ao nível rádio, assim como um algoritmo que combina facetas pró-activas e reactivas para determinar e manter estas rotas. O protocolo resultante é avaliado através de simulações no NS-2, mostrando que a nossa solução obtém um equilíbrio interessante entre o custo de sinalização, o tempo necessário para descobrir as rotas e as características das rotas encontradas.

Abstract

This thesis addresses the problem of multipath routing in wireless mesh networks. We study the use of clustering algorithms to facilitate the discovery and deployment of non-interfering multipath routes in these settings. In this context we propose a novel clustering algorithm and complementary protocols to discover and maintain routes in an efficient manner, aiming at minimizing interferences between transmissions of neighboring nodes. We provide an evaluation using the NS-2 network simulator and show that our solution offers an interesting tradeoff between the signaling cost, the time required to set up and maintain paths, and the properties of the discovered paths.

Palavras Chave
Keywords

Palavras Chave

Redes Sem Fios em Malha

Encaminhamento Multi-caminho

Agrupamento de Nós

Interferência Rádio

Keywords

Wireless Mesh Networks

Multipath Routing

Node Clustering

Radio Interference

Índice

List of Figures

List of Tables

Acronyms

ACE Algorithm for Cluster Establishment

ACK Acknowledge message

AODV Ad hoc On-demand Distance Vector

AODV-BR AODV Backup Routing

AODV-DM AODV-based Decoupled Multipath

AOMDV Ad hoc On-demand Multipath Distance Vector

ATM Asynchronous Transfer Mode

BER Bit Error Rate

CBMPR Cluster-Based Multipath Routing

CST Carrier Sensing Threshold

DCA Distributed Clustering Algorithm

DMAC Distributed Mobility-Adaptive Clustering

DSR Dynamic Source Routing

ENT Effective Number of Transmissions

ETT Expected Transmission Time

ETX Expected Transmission Count

LCA Linked Cluster Algorithm

LID Lowest ID Algorithm

LoCoup Low-Coupling Cluster-Based Multipath Routing Protocol

MAC Media Access Control

MIC Metric of Interference and Channel-switching

ML Minimum Loss

MOBIC Mobility Based Clustering

MOLSR Multipath Optimized Link State Routing

MP-DSR Multipath Dynamic Source Routing

MPR Multipoint Relay

OLSR Optimized Link State Routing

OSPF Open Shortest Path First

PNNI Private Network-to-Network Interface

WCETT Weighted Cumulative ETT

WMN Wireless Mesh Network

WLAN Wireless Local Area Network

WPAN Wireless Personal Area Network

WMAN Wireless Metropolitan Area Network

RCC Random Competition-based Clustering

RERR Route Error message

RREP Route Reply message

RREQ Route Request message

RTT Round Trip Time

RXT Receiving Threshold

SCTP Stream Control Transmission Protocol

SMR Split Multipath Routing

SNR Signal-to-noise Ratio

TCP Topology Control message

TCP Transmission Control Protocol

1 Introduction

This thesis addresses the problem of multipath routing in wireless mesh networks. We study the use of clustering algorithms to facilitate the discovery and deployment of non-interfering multipath routes in these settings. Our aim is create a solution that offers an interesting tradeoff between the signaling cost, the time required to set up and maintain paths, and the properties of the discovered paths.

1.1 Motivation

A Wireless Mesh Network (WMN) is formed by a set of gateways, mesh routers, and mesh clients. Gateways and mesh routers form the backbone of the network, where mobility is reduced. Mesh clients can be cell phones, laptops or other wireless devices. Routers communicate with the external network (e.g. the Internet) by forwarding each other's traffic (including clients traffic) towards the gateway nodes, which are directly connected to the wired infrastructure. In a WMN, each router forwards packets on behalf of other nodes (that may not be within direct wireless transmission range of their destinations). Moreover, the gateway functionalities enable the integration of WMNs with various existing wireless networks such as Wi-Fi, cellular networks, WiMax, among others.

In this type of networks, the nodes automatically establish and maintain mesh connectivity among themselves (creating an ad hoc network). Therefore, a WMN is self-organized, self-configured and redundant (if one node fails, the other ones are still able to communicate). This brings many advantages, such as low up-front cost, easy network maintenance, robustness, resilient and reliable service coverage. Moreover, and comparing meshes with traditional ad hoc networks, routers in meshes are not limited in terms of resources, and thus can be exploited to perform more resource intensive functions.

WMNs are expected to improve significantly the performance and circumvent the limitations

of many ad hoc networks, including wireless local area networks (WLANs), wireless personal area networks (WPANs), and wireless metropolitan area networks (WMANs). They are undergoing rapid progress and inspiring numerous deployments. WMNs will deliver wireless services in a large variety of scenarios of different scale, including personal, local, campus, and metropolitan areas. Mesh technology finds many applications in wireless multi-player gaming, campus connectivity, military communication, municipal networks, etc.

Given that these networks are usually multi-hop, in order to forward data to other nodes each node has to make routing decisions. Intuitively, routing algorithms designed for a WMN aim at choosing the path with the best links from each router to the gateways. All the traffic coming from mesh routers and mesh clients must be forwarded to mesh gateways. Consequently, certain nodes or links can be heavily loaded while some nodes/links can be seldom used. This may lead to an undesirable situation in which the best paths eventually degrade due to excessive load, consequently resulting in suboptimal performance. This scenario may be avoided if mutipath routing is used to balance traffic among multiple paths that may exist to reach the gateways. Multipath routing can also be used as a fault tolerance mechanism or to provide error resilience. The goal of our research is to analyze the existing solutions for multipath routing in Wireless Mesh Networks and explore new techniques to discover and maintain multiple non-interfering paths with a favorable trade-off between the latency in the path setup and the signaling cost required to establish those paths.

1.2 Contributions

This work addresses the problem of providing support to maximize the bandwidth utilization in backbone Wireless Mesh Networks. More precisely, the thesis analyzes, implements and evaluates techniques to discover and maintain routes in an efficient manner, aiming at minimizing interferences among transmissions on neighboring nodes. As a result, the thesis makes the following contributions:

- It proposes a novel clustering algorithm with the necessary properties to group nodes of a network in groups so that the overall distribution is closest to hexagonal with reduced overlap.

- It proposes *LoCoup*- Low-Coupling Cluster-Based Multipath Routing Protocol, a multipath routing protocol that consider multiple paths that do not interfere with each other, minimizing interferences between transmissions of neighboring nodes.

1.3 Results

The results produced by this thesis can be enumerated as follows:

- A study to determine the best set of properties a clustering algorithm should have in order to be applied to multipath routing without interferences between paths.
- A specification of the *LoCoup* multipath protocol and its implementation for the NS-2 simulation platform.
- An experimental evaluation of the implemented *LoCoup* comparing it with AODV-DM, a multipath protocol that is able to find non-interfering paths, as well as with the optimal approach.

1.4 Research History

The initial goal of this work was to study and analyze techniques to support multi-path routing in wireless mesh networks. From the works surveyed, AODV-DM emerged as a protocol able to find non-interfering routes with a reasonable signaling cost. Unfortunately, the latency in the discovery of the second route seemed very large. Our first idea was to modify the protocol in an attempt to speed up the route discovery process. Eventually, this effort lead to the design of a cluster-based algorithm for route discovery and maintenance.

This work was performed at INESC-ID in the context of the FCT project Redico: "Reconfiguração Dinâmica de Protocolos de Comunicação" (PTDC/EIA/71752/2006) project. During my work, I benefited from the fruitful collaboration with the remaining members of the GSD team working on Redico, namely José Mocito. I also want to thank Martijn Kuipers for his very usefull insights on the configuration of the NS-2 propagation model.

1.5 Structure of the Document

The rest of this document is organized as follows. Chapter 2 provides an introduction to the different technical areas related to this work. Chapter 3 introduces *LoCoup* and Chapter 4 presents the results of the experimental evaluation study. Finally, Chapter 5 concludes this document by summarizing its main points and future work.

2 Related Work

2.1 Introduction

Although mesh networks are a relatively new subject, multipath routing is a concept that was introduced in 1984, when it was applied in the telecommunication industry (Marc Mosko 2005). Known as alternate path routing in traditional circuit switched telephone networks, the technique has been used to decrease the call blocking probability. In this scheme, the shortest path between two entities is used until it fails, after which calls are routed through another backup path. Multipath has also been used in ATM networks, specifically in the PNNI signaling protocol, during the reservation process to find multiple alternate paths. Other known protocols, such as OSPF (Moy 1998) have proposed techniques using multipath, although with some restrictions (the paths must have equals cost) (Mueller, Tsang, & Ghosal 2004).

This section starts by addressing mesh networks characteristics and routing protocols. Then, we survey multipath routing including protocol operation, main multipath routing problems, evaluation of routing protocols and some examples. We also discuss low-coupling techniques.

2.2 Wireless Mesh Networks

Wireless Mesh Networks combine static mesh routers (mesh routers and gateways) operating in ad hoc mode with mobile wireless nodes (mesh clients). These networks are, for instance, appropriate to expand network connectivity in regions where access to an infra-structured network is limited. Gateways and mesh routers communicate with the external network (e.g. the Internet) by forwarding each other's traffic (including clients traffic) towards the gateway nodes, which are directly connected to the wired infrastructure. They form the backbone of the network, where mobility is reduced. Mesh clients can be cell phones, laptops or other wireless devices. Our work focus on the backbone of this networks.

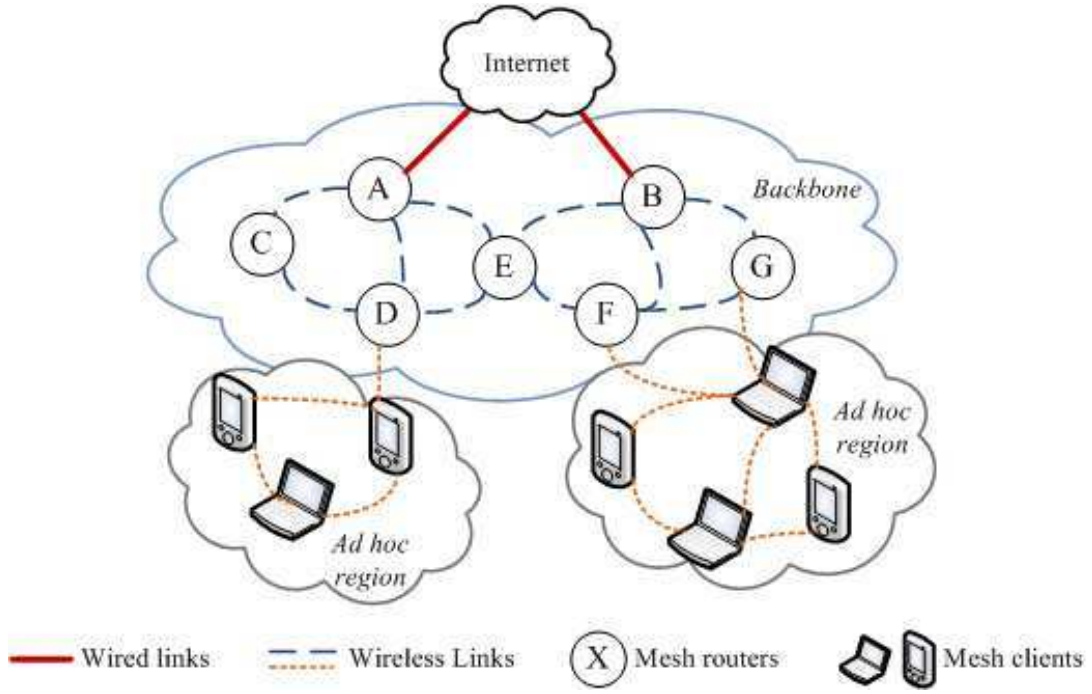


Figure 2.1: Representation of a Hybrid Mesh Network.

A mesh network can be characterized according to its architecture in one of the following types (Akyildiz & Wang 2005):

- *Infrastructure or backbone mesh:* is formed by a set of mesh routers connected by self-configuring and self-healing wireless links. Some of the routers have gateway function, providing Internet connectivity for other routers, and consequently for clients that connect to them.

- *Client mesh:* is a pure mobile ad hoc network, since each mesh client (mobile clients) act as an independent router with no centralized routing control. Clients are able to perform network functionalities like routing and forwarding.

- *Hybrid mesh:* is the most generic and interesting type of architecture (and the focus of this work), as it combines both infrastructure and client mesh, as represented in Figure 2.1. While infrastructure provides connectivity to other networks such as Internet, clients provide a dynamic extension of the network.

Wireless Mesh Networks can be seen as a particular case of an ad hoc network, characterized by the following set of unique properties(Nandiraju, Nandiraju, & Agrawal 2006; Akyildiz &

Wang 2005):

- Mesh routers are relatively static, so links used to support communication in the backbone are relatively static too.
- Mesh routers are not power constrained. Since mesh routers are typically connected to the electrical grid, mesh routing protocols do not have energy consumption restrictions.
- The traffic model is different and may concentrate in certain paths, predominantly between mesh routers and gateways. In other words, the backbone of the network, specially the zones near the gateways, are the ones where links are most used.
- The traffic volume and the number of users in a WMN can be high, as it aims to provide broadband connections for Internet access to large communities.

2.3 Routing in Wireless Mesh Networks

Routing protocols are used to find and maintain routes between source and destination nodes, in order to forward traffic. To perform well in Wireless Mesh Networks, a routing protocol must be tailored to deal with the characteristics enumerated before. More precisely, WMNs must consider:

- Transmission errors: the unreliability of the wireless medium may lead to transmission errors.
- Link and node failures: nodes and links may fail at any time due to different types of hazardous conditions in the environment.
- Incorrect routes: due to node/link failures or additions to the network, routes may become obsolete or based on an incorrect system state.
- Congested nodes or links: due to the topology of the network and the nature of the routing protocols, certain nodes or links may become congested, which will lead to higher delay or packet loss.

When considering route creation process, routing protocols can be classified in three main categories: proactive, reactive and hybrid, as described below.

2.3.1 Proactive Routing.

Each node maintains a routing table, containing routes to all other nodes in the network. Thus, routes are computed and stored, even when they are not needed, incurring in a considerable overhead and bandwidth consumption due to the number of messages that have to be exchanged to keep routing information up-to-date. Proactive protocols may be impractical for large and dynamic networks.

2.3.2 Reactive Routing.

Also called on-demand, reactive protocols only compute routes when they are needed. The process of finding a suitable route requires the transmission of route requests and the wait for replies with a path to the destination. Due to the delays incurred in this process, this approach is not suitable for operations that require immediate route availability.

2.3.3 Hybrid routing.

Neither proactive nor reactive protocols provide an optimal solution for the hybrid WMNs we aim at addressing. Ad hoc regions, the ones formed by clients, have some mobility and thus reactive protocols are most suitable because route updates are frequent. On the other side, the backbone has reduced mobility, thus proactive routing allows to maintain routes with low overhead.

Hybrid approaches aim at providing an optimal solution by combining the best properties of both proactive and reactive protocols. Hybrid routing uses different routing protocols in different parts of the hybrid WMN: reactive protocols for ad hoc zones and proactive ones in the backbone.

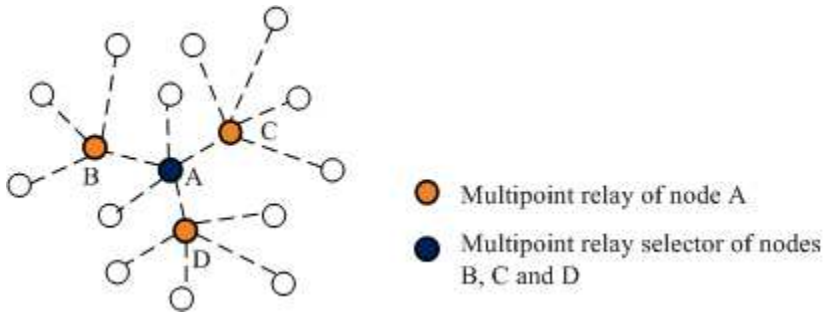


Figure 2.2: Multipoint Relays and Multipoint Relay Selectors.

2.4 Single-path Routing Protocols for Wireless Mesh Networks

2.4.1 Proactive Protocols

2.4.1.1 OLSR

In the next paragraphs we describe some relevant routing protocols used in wireless mesh networks. We start by giving a brief overview of OLSR, AODV, and DSR, given that many of the multipath routing protocols discussed in this report are an extension of one of these three protocols.

Optimized Link State Routing (Jacquet, Mühlethaler, Clausen, Laouiti, Qayyum, & Viennot 2001) is a proactive protocol designed for large and dense networks, where communication is assumed to occur frequently. OLSR uses two key concepts to compact the amount of control information sent in the messages and to reduce the number of retransmissions required to propagate them: multipoint relay and multipoint relay selectors.

The purpose of *multipoint relays* (MPRs) is to optimize the flooding of packets and reduce duplicate retransmissions in the same region. As represented in Figure 2.2, each node has various multipoint relays, selected among its one-hop neighbors in such a manner that the set covers all the nodes that are two hops away. For instance, the multipoint relays of node A are nodes B, C and D given that via one of these nodes A can reach all nodes that are two hops away from it (note that only relevant links are represented in the figure). Symmetrically, B, C and D will have A in its multipoint relay selector set.

For a node to choose its multipoint relay selectors, it needs first to detect the set of neighbors with which it has a bidirectional link. This is done by broadcasting periodic HELLO messages,

to all one hop neighbors. By receiving HELLO messages from an one-hop neighbor N, a node can record the following information about N: i) the status of the link to/from N; ii) a list of the one-hop neighbors that N gives access to. Based on this information, a node can choose its multipoint relay selectors. The selected nodes are listed in HELLO messages (so the multipoint relays know they have been selected). This information is refreshed when a change is detected in a one or two-hop neighborhood.

OLSR maintains two tables to support its operation: the topology table and the routing table. The topology table is constructed with the information obtained from periodic TC (Topology Control) messages, sent by each node in the network. A TC message contains the list of nodes that have selected the source as a multipoint relay, so a node receiving this message has information of two-hop links, through the TC sender. Each entry in the table contains the address of a potential destination (a MPR selector received in the TC message) and the address of the last hop to that destination (the source of the TC message).

The routing table is the one used to forward traffic. Its construction is based on the topology table, by tracking the connected pairs in a descending order. For example, let S be the source and D the destination; consider that the route S to D is a 2-hop route via an intermediate node R; if a link (R,D) connecting R to D is already known, then one needs to search for a connected pair (S,R) in order to get a complete route from source to destination. Each node will select only the connected pairs on the minimal path, and create an entry in the routing table with the following information: the destination address, the next hop address, and the distance to destination.

2.4.2 Reactive Protocols

2.4.2.1 AODV

Ad hoc On-Demand Distance Vector (Perkins & Royer 1999) is a reactive protocol. Therefore it consists of two main phases: route discovery and route maintenance. Route discovery is the process to find a route between two nodes. It is initiated only when a node wants to communicate with another node and does not have the required routing information in its routing table. Route maintenance consists of repairing a broken route or finding a new one, and is initiated when a route failure occurs.

During the route discovery, two paths have to be considered, the forward path and the reverse path. According to the way protocols record these paths, we can consider two different approaches:

- Source routing: the list of hops traversed are stored in the messages directly. In source routing, more overhead is added to data packets, as the entire route must be specified in the packet header.
- Hop-by-hop routing: the reverse path is stored in a table (routing table) in the nodes along the path. In hop-by-hop routing, the header overhead is replaced by the need to maintain routing tables in the intermediate nodes, with forwarding information.

AODV is based on hop-by-hop routing, i.e., it maintains routing table entries at intermediate nodes, which means it uses hop-by-hop routing to forward traffic.

Route discovery. The source node broadcasts a route request packet (RREQ) to its neighbors, which is uniquely identified by the pair (source address, broadcast id). When a node receives a RREQ, it can act the following way:

- If the RREQ was already received, it is dropped;
- If the RREQ has not been received and the node does not have a path to the destination, the RREQ is re-broadcasted (with an increased hop count);
- If the RREQ has not been received and the node is the destination or has a route to the destination, a RREP (route reply) is sent to the source of RREQ.

When the RREQ is forwarded in the network, each intermediate node stores the previous hop, thus forming the route to the sender (i.e. the reverse path). Control messages have two additional control fields worth of notice, namely a destination sequence number and a source sequence number. Destination sequence numbers (also stored in RREP and routing tables) are used to maintain freshness information about routes; a larger sequence number indicates a more recent route. Source sequence numbers are used to maintain freshness information about the required reverse route to the source, specifying how fresh a route to the destination must be. A route is only accepted by the source if it has a sequence number greater or equal to the source sequence number contained in the RREQ packet.

Route Maintenance. Due to mobility or other factors, sometimes routes have to be reconstructed. If a node is detected to be unreachable by one of its neighbors, the node sends an unsolicited RREP with special characteristics that erases all the routes using the link along the way. This message is broadcasted until it reaches all the source nodes. If the route is still needed, a new RREQ is sent to build a new fresh route. Additionally, there are also other mechanisms used to manage routes, like timeouts to remove temporary or obsolete paths in routing tables.

2.4.2.2 DSR

Dynamic Source Routing (Johnson & Maltz 1996) is, like AODV, a reactive protocol. However, as the name implies, it is a source routing protocol: the full path is included in the packet header, and this information is used to forward traffic.

Route discovery. Is initiated by a route request packet (RREQ) containing the destination address. When an intermediate node receives a RREQ, it can act in one of the following ways:

- If it has already received the RREQ, or if the node is still in the route contained in the packet header, the RREQ is dropped (to prevent routing loops);
- If it has a route to the destination in its cache, it may append that route to the route record in the RREQ header and send a RREP (route reply) back to the source. Although this mechanism helps to reduce the overhead caused by the flood of RREQ, if the cache is out-of-date, it can provide a wrong route;
- Otherwise, it re-broadcasts the RREQ packet.

RREP packets can follow the reverse advertised route, or can be forwarded through another route, if the destination has another route to the source in its route cache (this allows DSR to operate when links are not bidirectional).

Route maintenance. When a link is detected to be broken, all routes containing the link must be removed from the route cache, and a route error packet (RERR) should be sent back to the source with an indication of the broken link. When a node receives a RERR, all the routes containing the broken link are deleted.

2.5 Multipath Routing

Most of the routing protocols that have been proposed for mesh and ad hoc networks are unipath, which means only a single route is used between a source and a destination node.

The main goal of multipath routing is to allow the use of several good paths to reach destinations, not just the best path. This should be achieved without imposing excessive control overhead in maintaining such paths. The availability of multiple paths between a source and a destination can be used to achieve the following benefits:

- *Fault tolerance:* introducing redundancy in the network (Amir, Danilov, Kaplan, Musaloiu-Elefteri, & Rivera 2008) or providing backup routes to be used when there is a failure (Lee & Gerla 2000), are forms of introducing fault tolerance at the routing level in mesh networks. To this end, some techniques may be applied like packet salvaging (Nasipuri & Das 1999; Valera, Seah, Rao, & Rao 2002), which consists in modifying the route of a packet if the actual route is broken.

- *Throughput enhancement:* in a mesh network, some links can have limited bandwidth. Routing along a single path may not provide enough bandwidth for a connection. Therefore, using simultaneously multiple paths to route data can be a good approach to satisfy the bandwidth requirement of some applications. By increasing the throughput, a smaller end-to-end delay is achieved and quality of service is improved (Leung, Liu, Poon, Chan, & Li 2001).

- *Load balancing:* as traffic distribution is not equal in all links in the network, spreading the traffic along multiple routes can alleviate congestion in some links and bottlenecks (Lee & Gerla 2001).

- *Error resilience:* multipath protocols can be used to provide error resilience by distributing traffic (for instance, using data and error correction codes) over multiple paths. In (Tsirigos & Haas 2001) a *M-for-N diversity coding scheme* is proposed, which consists of using M additional links to send traffic, coded in a way that the system can tolerate $M - 1$ simultaneous link failures at any time.

- *Security:* with single-path routing protocols, it is easy for an adversary to launch routing attacks, but multipath offers attack resilience (Siddiqui, Amin, Kim, & Hong 2007).

2.5.1 Routing Metrics

The route establishment phase includes the choice of paths among all the available to forward traffic. If various paths are available we may want to choose a limited number: to use only the one with the best metric or to choose the n best ones. The path evaluation and selection according to certain metrics are discussed in this section. Hop count is a widely used metric since it allows quick path discovery in presence of mobility.

However, in WMNs, the stationary topology benefits from *quality-aware routing* metrics (Koksal & Balakrishnan 2006), because radio communication is often unpredictable.

Path reliability and link quality are performance metrics used by a considerable number of quality-aware routing schemes. We can define path reliability as the probability of having a successful data transmission between two mobile nodes within a certain amount of time (Tsai & Moors 2006). This success is dependent of many factors, that have motivated the use of the following metrics that can be used to evaluate mesh routing protocols (Campista, Esposito, Moraes, Costa, Duarte, Passos, De Albuquerque, Saade, Rubinstein, & Rubinstein 2008; Yang, Wang, & Kravets 2005):

- *ETX (Expected Transmission Count)* is the expected number of transmissions a node needs to do to successfully transmit a packet to a neighbor. It is based on the delivery ratio of a number of packets, in a certain time interval.
- *ML (Minimum Loss)* refers to the route with the lowest end-to-end packet drop probability.
- *ETT (Expected Transmission Time)* considers link quality as a function of the time a data packet takes to be successfully transmitted to each neighbor.
- *mETX (modified ETX)* works at bit level, by computing the bit error probability. This metric aims to solve a problem related to fast link-quality variation in wireless networks (that makes ETX, ETT, among others less suitable). Metrics based on a time-window interval may not follow the link-quality variations or may produce excess in overhead.
- *ENT (Effective Number of Transmissions)* measures the number of successive retransmissions per link, considering the variance. It was also defined to consider link-quality variations.

- *iAWARE* estimates the average time the medium is busy because of transmissions from each interfering neighbor, considering aspects like medium instability, data-transmission time, and interferences.

- *WCETT (Weighted Cumulative ETT)* reflects the interference among links (inter-flow interference) that operate on the same channel. Reduction of the throughput is a reflection of the considered interference.

- *MIC (Metric of Interference and Channel-switching)* improves WCETT, considering not only inter-flow interference but also intra-flow. MIC consists of two principal components: the first one captures the potential for the path to interfere with other paths and the second one captures the potential for the path to interfere with itself.

Although there are so many metrics to evaluate these protocols, and some of them more complex and complete than others, most of routing protocols implementations prefer metrics with simpler designs such as ETX or ETT.

Considering multipath routing protocols, the following metrics are most suitable:

- *Degree of route coupling (Pearlman, Haas, Sholander, & Tabrizi 2000a):* defines the grade of interference between paths in multipath routing. In wireless transmissions, it is known that packet transmission may result in degraded quality of a simultaneous transmission on a neighboring link. If two routes have nodes or links in common, they are highly coupled, but this effect may occur even if there are no shared nodes or links. Low coupling links are the best in terms of transmission quality, which means that disjointness between links must be achieved.

- *Path correlation (Wu & Harms 2001):* correlation factor describes the interference of traffic between two node-disjoint paths which is relevant when all the nodes use the same radio spectrum and compete for the radio channel. The total correlation factor of a set of multiple paths (we are considering multipath protocols) is defined as the sum of the correlation factor of each pair of paths. The lower the path correlation, the better.

Other approaches use combinations of some metrics, especially in QoS routing, where a subset of paths is selected only if the combined metric satisfies the QoS requirements.

2.5.2 Proactive Multipath Routing

Proactive multipath routing protocols have two operation phases: network setup and network maintenance.

Network Setup This phase consists of the steps required to build the routing table needed to forward traffic.

Network Maintenance This phase is executed when the routing table has already been created and consists of the steps required to maintain and repair the existing routes in face of topology changes.

2.5.3 Reactive Multipath Routing

To the best of our knowledge, most of the multipath routing protocols are reactive. The operation of these protocols can be split in three components: route establishment, route maintenance, and traffic allocation. In the following paragraphs we discuss each of these components.

Route Establishment. Consists in finding multiple routes between a source and a destination node. This is performed by flooding a route request (RREQ) with unique sequence number. To limit the message cost, nodes drop a RREQ they have already received. New RREQs are rebroadcasted until they reach the destination that, in turn, will answer to the RREQ by sending one or more route replies (RREP). As the destination has collected information about multiple existing paths, it can apply one of the following criteria to choose the ones to use:

- *Minimum cost paths (Rangarajan 2007):* are minimum cost paths amongst all available paths (shortest paths according to piggybacked information on the RREQ).
- *Non-disjoint paths (Mueller, Tsang, & Ghosal 2004):* paths that can have nodes and links in common. Non-disjoint routes can be more easily discovered (as no restrictions are imposed, more disjoint routes may be discovered). Moreover, it was shown that a network becomes more reliable and better amortizes the cost of on-demand path discovery over many links, by exploiting rich mesh connectivity, in spite of using disjoint

paths (Marc Mosko 2005). As the distance between nodes increases, the probability of finding node and link-disjoint routes decreases, so non-disjoint paths have to be used. But this scheme has also disadvantages: since we are considering wireless communication, it can be more difficult to choose routes that do not interfere with each other.

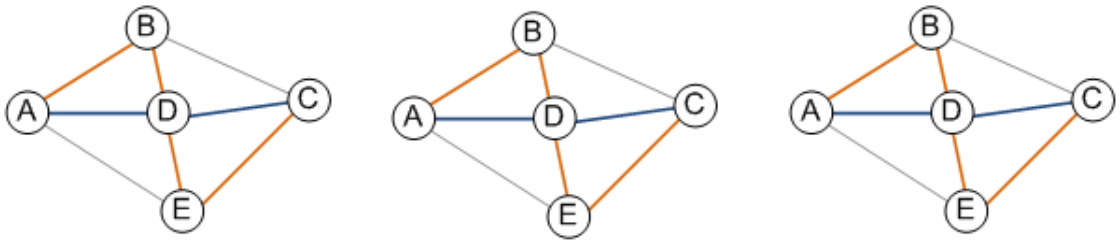
- *Link disjoint paths:* paths that have no links in common but may have nodes in common.
- *Node disjoint paths:* paths that have no nodes (and, consequently, no links) in common. In principle, node disjoint routes offer a better use of network resources, because neither links nor nodes are shared between two paths. For fault-tolerance, node disjoint routes offer the highest availability, because when using link disjoint routes, a failure of a node may cause the failure of several routes.
- *Zone-disjoint paths (Roy, Saha, Bandyopadhyay, B, Tanaka, & Ueda 2003):* paths are said to be zone-disjoint when data communication over one path does not interfere with other paths, meaning that route coupling (Pearlman, Haas, Sholander, & Tabrizi 2000b) (interference) between the considered paths is zero.

When the paths have been selected, the corresponding RREPs have to be sent back to the source. For this purpose, the information about the reverse path has to be set up during the RREQ forwarding, using one of the schemes referred in Section 2.4: source routing or hop-by-hop routing. Note that multiple RREQ can reach the destination, and the destination can only reply to a subset of them, according to the criterion mentioned above.

As an optimization, if an intermediate node receives a RREQ and it has a route to the destination, it can send a RREP with the known route. However, in order to let the destination select disjoint or minimum cost paths, sometimes multipath routing protocols have to inhibit the reply to RREQ by intermediate nodes (Mueller, Tsang, & Ghosal 2004).

Route Maintenance. The purpose of route maintenance is to validate existing routes and find suitable replacements when one of the existing routes fail. There are multiple ways to maintain routes, from which we highlight two:

- *Periodic beacons (*HELLO *messages):* each node periodically transmits a HELLO message. If a node does not receive a HELLO from a neighbor after a certain amount of time, it



(a) Node disjoint (b) Link disjoint (c) Non-disjoint

Figure 2.3: Route establishment

assumes that the link connecting the node to the neighbor is broken and sends an error notification to the predecessor nodes affected by the failure. As HELLO messages consume bandwidth that could be better utilized for data traffic, this is not a desirable scheme.

- *MAC layer acknowledgements:* the procedure is almost the same, but no additional messages are sent. A node that does not receive an ACK for a data packet for a certain number of retries, assumes the link to the neighbor to be broken, and an error notification is sent.

If any of the previous mechanisms suggests that there is a route failure, in order to find a new route, route discovery process must be performed. In the specific case of multipath, route discovery can be triggered each time one of the routes fails or only when all the routes have failed. None of the schemes is best for all scenarios, as waiting for all the routes to fail before performing a route discovery would result in a delay before new routes are available, and initiating a route discovery every time a path fails may incur in excessive overhead. An alternative approach that can be a compromise between these two extreme options consists in initiating route discovery only when a threshold of n paths fail.

To prevent routes from being removed as a result of inactivity, route utilization becomes important, as on-demand routing protocols tend to drop routes if they have not been utilized for a certain amount of time.

Traffic Allocation Once the source node has selected a set of paths to the destination, it can begin to send data to the destination along these paths. To choose the way data is distributed among the existing paths, an allocation strategy is needed. There are two relevant aspects of an allocation strategy: granularity and scheduling. The granularity specifies the smallest unit of information allocated to each path (Tsai & Moors 2006):

- *Per source-destination pair:* consists in using the same path to forward all traffic belonging to a certain pair of source and destination nodes.
- *Per-connection:* consists in allocating all traffic for the same connection to a single path.
- *Per-packet:* consists in distributing the packets from multiple connections among the existing paths.
- *Per-segment:* consists in splitting a packet into segments and forward each segment using a different path.

A finer granularity allows more efficient load balance, as it allows for a better control over the network resources. At the same time, granularity per-packet or per-segment requires reordering at the destination.

Paths obtained in the route discovery process can be scheduled using the following strategies:

- *Round robin:* in this strategy a node just sends each packet using a different path. In spite of being simple, this scheme suffers from setback due to the possibility of out of order delivery of packets belonging to the same flow. Out of order delivery leads to the maintenance of a large buffer by TCP and can lead to unnecessary loss. Nevertheless it can be used for effective load balancing because load will be uniformly distributed.
- *Congestion aware:* this scheduling scheme proposes to send traffic using non-congested links to avoid losses and additional delays. Congested routes can be measured, for example, by the average queue length and improving the performance significantly (Nandiraju, Nandiraju, & Agrawal 2006).
- *Backup path:* during the route discovery procedure, a source can establish a primary path as well as several backup paths to the desired destination. Multipath can be used to reduce the delay in recovering from a failure, thus using the backup paths to switch the on-going traffic to these alternative paths, instead of shooting down the end to end connection, when the primary path fails.
- *Unequal cost scheme:* proposes to distribute traffic based on a joint measurement of distance and link quality. The path with best metrics is selected according to a probability distribution (Boltzmann distribution used in (Marc Mosko 2005)). This becomes the most

used path, but sometimes other routes are selected to forward traffic. The advantage of this approach is that, as all routes are used, they are not removed from routing tables, which prevents route discovery procedures.

- *Concurrent delivery:* refers to sending traffic at the same time in more than one path. This scheme is used, for example, to enhance throughput (Lee & Gerla 2001).

2.6 Multipath Routing Protocols for Wireless Mesh Networks

In the next paragraphs we describe some significant routing protocols that use multipath.

2.6.1 Proactive Protocols

2.6.1.1 MOLSR

Multipath Optimized Link State Routing Protocol (MOLSR) (Xuekang, Wanyi, Xingquan, Baocheng, & Zhigang 2009) is a proactive protocol based on OLSR that aims to achieve lower delay and packet drop by using multipath routing. As seen in Section 2.4.1.1, OLSR uses multipoint relays and multipoint relay selectors to limit broadcasts. The operation of MOLSR is almost the same as OLSR, although the network setup phase has some differences. In this section we explain the differences between the two protocols.

OLSR maintains two tables: the topology table and the routing table. The topology table records the organization of the whole network, but does not store any link state parameter, so precise routing selection is hard to be made based on this table only. In MOLSR, SNR and Delay are added to the TC (Topology Control) message and these parameters, representing the link state between the MPR selector and the TC originator, are also stored in the topology table. The routing table is constructed based on the topology table and stores no more than two routes to every destination. These routes represent the best ones at that moment. A node can choose the best route to transmit data, but if it fails the other one can be used without route discovery.

The main differences between OLSR and MOLSR lie in the procedure to compute the routing table. In MOLSR, more than one route is calculated and two best routes are chosen

according to the link metrics announced in TC messages. Routes with more than 2 nodes in common are not considered (node disjoint paths).

2.6.1.2 OLSR-based multipath routing

Concerning multipath proactive routing, there are some link-state protocols based in OLSR. MOLSR(Xuekang, Wanyi, Xingquan, Baocheng, & Zhigang 2009), as we have seen, computes multiple paths but uses only the best one at each moment. In QOLSR(Badis & Al Agha 2005) multiple paths are used to satisfy certain bandwidth and delay requirements. Such paths have minimum correlation factor but they are not zone disjoint. Another multipath proactive protocol is MP-OLSR(Yi, Adnane, David, & Parrein 2010) which computes multiple node or link disjoint paths according to different cost functions. All the mentioned solutions use OLSR as base, so its core functionality has two parts: topology information acquirement and route computation. To get topology information of the network, nodes use topology sensing and topology discovery. Topology sensing includes link sensing and neighbor detection and allows each node to collect information about its neighbors, based on the periodic exchange of HELLO messages. Topology discovery is based in TC messages and gives each node enough information to enable routing. Route computation is performed everytime a TC is received. Routes to all the destinations in the network are computed and saved in the routing table. However, in MP-OLSR(Yi, Adnane, David, & Parrein 2010) an on-demand scheme is used to avoid the heavy computation of multiple routes to every destination. Although the network topology update operation may benefit from the use of multipoint relays, as every node is required to send link-state updates the signaling cost required to maintain this information is high, specially in large and dense networks.

2.6.1.3 MMESH

The MMESH protocol (Nandiraju, Nandiraju, & Agrawal 2006) is essentially a proactive routing protocol for wireless meshes to balance the traffic load uniformly over multiple paths.

Network Setup. In the initial setup phase, nodes discover routes to gateways by hearing advertise messages of Internet connectivity. Based on performance metrics that are included in these advertisements, mesh routers select the routes that are acceptable to them and add them to the routing table. Then, each node notifies the mesh router that has announced the route

about the chosen routes. Mesh routers update the routes to forward traffic to and from the child mesh router, and notify other routers along the route, including the gateway. To limit the amount of possible paths to reach a node, each router can only announce its n best disjoint path routes (where n represents a tradeoff between load balancing capacities and complexity in route finding). To avoid additional overhead, when routing tables are already established, each route is labeled and is assigned a sequence number to identify its freshness.

Network Maintenance. As WMNs are not static, when new mesh routers join the network or paths degrade to a level that they cannot be used to forward packets, network maintenance is needed. Each router continuously monitors the performance of paths to the gateway and propagates this information to the neighboring nodes. By analysing changes in the observed performance, new routes can be found or existing ones suspended. If new routes are found, routers update the routing table and propagate these changes to their neighbors. In order to adapt the algorithm to the changes that can occur in the network (mobility even if reduced, new mesh routers, reboot), if one node fails and one of its neighbors detects the failure, it suspends the routes that include the node. It then forwards the traffic through another node. This mechanism makes the protocol more stable, by only removing routes when a timeout occurs.

MMESH applies a congestion aware approach by using link metrics and variance to select the best path to forward traffic. If the product of variance by the metric of the best path is no less than the metric of the selected path, the packet is sent through the selected path. Otherwise, the route is temporarily suspended and a different path is selected to send the packet.

2.6.2 Reactive Protocols

2.6.2.1 AODV-DM

AODV-based Decoupled Multipath (Hu & Lee 2007) proposes an algorithm that defines an insulate region around the primary path, to avoid interference between adjacent routes, trying to minimize the route coupling problem (Pearlman, Haas, Sholander, & Tabrizi 2000b). Moreover, it uses SCTP (Ong & Yoakum 2002) instead of TCP because the latter is designed for single-path routing and cannot adapt to the network layer multipath structure and SCTP can independently control the traffic rate of each path.

Route establishment. The primary path is discovered in the following way: a RREQ is flooded to the entire network.

Each node upon receiving a message, stores the information about it in its routing table. Multiple RREQs may reach the destination, coming from different paths. The destination first responds to the request that has followed the shortest path (this is also called primary path), by sending a primary route reply (pRREP).

The packet follows the shortest path; intermediate nodes along the route broadcast the packet to neighbor nodes, which mark themselves as "in-region" nodes. When the pRREP reaches the source, the primary path formation is complete and an insulating region is established. To prevent future RREPs entering this region, neighbors outside the insulating region remove the "in-region" nodes from their table.

After waiting a period of time to allow the insulating region to be established, the destination responds to other RREQs. The response packet is called second route reply (sRREP) and the propagation process of this message is almost the same as the pRREP. The exception is that, a node receiving a broadcast sRREP does not mark itself as an "in-region" node. Intermediate nodes shall broadcast the packet to their neighbors in order to remove themselves from their neighbors' tables, as is represented in Figure 2.4. If an "in-region" node receives one of these packets, but has no available entry in its table to forward the packet, it sends a route reply rejection packet (RREJ), so the sRREP sender can try other entries in its table. This procedure is represented in Figure 2.4.

Route maintenance. Is done in the same way as in the base protocol, AODV.

Traffic distribution. The protocol may use a single path or multiple paths, when the available bandwidth between a source/destination pair in not enough. When multiple paths are used, traffic is distributed concurrently.

The decoupled features, and the use of path-aware SCTP scheme, make the protocol suitable for concurrent data transfer in dense networks, otherwise, the delay can compromise the efficiency of the protocol.

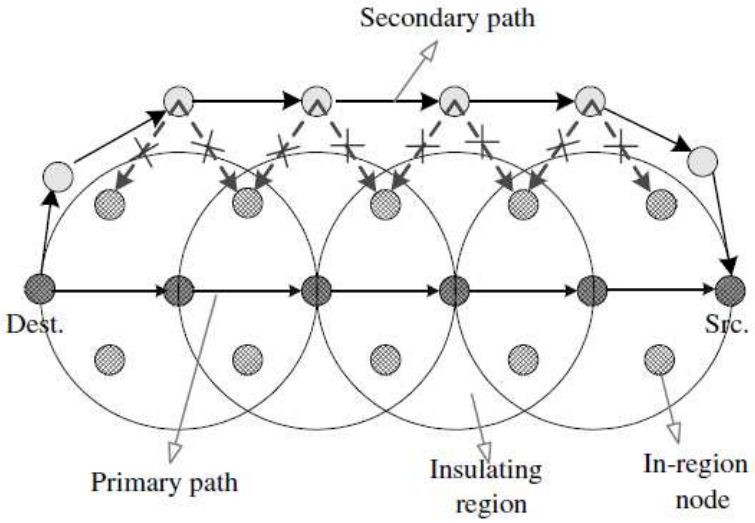


Figure 2.4: AODV-DM illustration (extracted from(Hu & Lee 2007)).

2.6.2.2 AODV-BR

AODV-Backup Routing (Lee & Gerla 2000) proposes a backup route mechanism to improve the performance of existing on-demand protocols that discover routes through a query/reply procedure.

Route construction. Route construction is done almost in the same way as in AODV (Perkins & Royer 1999), but multiple paths are formed in the following way. When a node receives a RREP not directed to itself transmitted by a neighbor, it adds the neighbor as the next hop to the destination in its alternate route table (if it receives more than one it chooses the best). The resulting structure resembles a fishbone, as depicted in Figure 2.5. The primary path is used until a failure occurs.

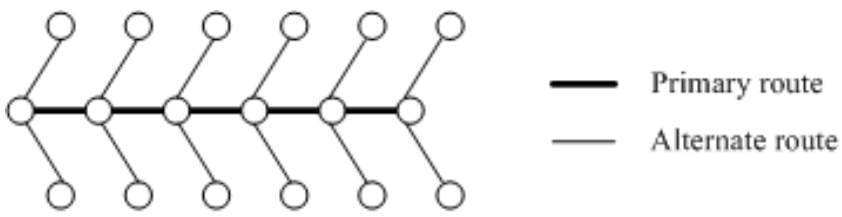


Figure 2.5: Multiple routes forming a fish bone structure(Lee & Gerla 2000).

Route maintenance. Applies when a node detects a link failure. In this case, the node performs a one hop data broadcast to its intermediate neighbors, classifying the packet for alternate routing. Also, the node that detects the failure sends a route error (RERR) packet to the source to initiate a route rediscovery (in order to build a fresh and optimized path).

Traffic distribution. Only one path at a time (the primary path) is used. When it breaks, a backup route is used to forward traffic.

AODV-BR was shown to have improved throughput and protocol effectiveness, in mobile scenarios. Moreover, as the number of data sessions is increased, the protocol becomes less efficient due to collision and contention problems, so it does not perform well under heavy traffic.

2.6.2.3 AOMDV

The Ad-hoc On-demand Multipath Distance Vector protocol (Marina & Das 2001) is a AODV-based protocol, proposed to reduce routing overhead when a route fails. It is able to discover multiple routes with a single route discovery procedure.

Route establishment. As in AODV (Perkins & Royer 1999), route discovery procedure is triggered when a node wants to communicate with a destination to which a path is not known. The route establishment procedure is the same as in the base protocol with the following change: to form multiple routes, all duplicates of the RREQ arriving at a node are examined (but not propagated), as each one defines an alternate route.

The protocol can find node-disjoint or link-disjoint routes. To find node-disjoint ones, intermediate nodes do not reject duplicate RREQs. To get link-disjoint routes, the destination replies to duplicate requests even if they have the same last hop. To ensure link-disjointness in the first hop of the RREP, the destination only replies to RREQs arriving via unique neighbors. After the first hop, the RREPs follow the reverse paths, which are node-disjoint and thus link-disjoint.

Route maintenance. To preserve connectivity information, each node executing AOMDV can use link-layer feedback or periodic HELLO messages to detect broken links to nodes that it

considers as its immediate neighbors. As in AODV, in case a broken link is detected, a RERR message is sent to the active neighbors that were using that particular route.

Traffic distribution. With multiple redundant paths available, the protocol switches routes to a different path when the path in use fails. Thus a new route discovery is avoided. Route discovery is initiated only when all paths to a specific destination fail. For efficiency, only link disjoint paths are computed so that the paths fail independently of each other.

2.6.2.4 MP-DSR

The MP-DSR is a DSR-based protocol that addresses path reliability requirements. MP-DSR (Leung, Liu, Poon, Chan, & Li 2001) first collects application path reliability requirements. Then it determines the number of paths needed and the lowest path reliability requirement each path must provide.

Route establishment. The source sends RREQ messages, containing information regarding the requirements, to the destination node via its immediate neighbors. The RREQ message also contains the traversed path (as this is a source routing protocol), and the accumulated path reliability. The intermediate nodes use the information included on the RREQ message to check if reliability requirements are still satisfied. If so, the node updates the accumulated path reliability based on the availability of the link just traversed, and re-broadcasts the message to its neighbors. Otherwise, the RREQ message is dropped. The destination waits until it receives all the RREQ packets, sorting them according to the path reliabilities. Then, a set of disjoint paths are selected according to the reliability required. A RREP is sent to the source, via each selected path.

Route maintenance. The route maintenance procedure differs for each of the following scenarios: when all routes fail or when one of the used paths fails. If all routes fail, the route establishment process is simply re-initiated. If only one route fails, the source sends a route check messages along the paths to collect the path reliabilities. The destination replies to the route check messages. The source collects all the replies, and checks to see if the paths still meet

the reliability requirement. If not, route discovery is performed. MP-DSR collects QoS characteristics using local information available at intermediate nodes, which means global knowledge is not required.

Traffic distribution. Traffic is sent along more than one path if multiple paths are required to meet QoS requirements.

2.6.2.5 SMR

Split routing (Lee & Gerla 2001) is protocol similar to DSR, and is used to construct maximally disjoint paths, to be used concurrently.

Route establishment. The RREQ is flooded in order to find routes. Intermediate nodes forward RREQ without replying, even if they have routes to the destination (this is to allow the destination to select disjoint paths). Intermediate nodes do not need to discard duplicate RREQs. Instead, they forward RREQs that are received through a different incoming link, and whose hop count is no larger than the previously received RREQs. The proposed algorithm selects two routes, but it can be easily extended to return more routes. The selection procedure is done the following way. The destination node replies to the first RREQ, which represents the shortest path. Then, it waits to receive more RREQs and selects the path that is maximally disjoint from the shortest delay path. If more than one exists, the shorter is selected. Then, a RREP for the selected path is sent.

Route maintenance. When a link is detected to be failed, a RERR packet is sent to notify nodes using the link that is broken. They can then delete the entry from their routing tables. When there is a failure, a new route discovery procedure may be triggered, or this can be done only when both routes of the session are broken.

Traffic distribution. The protocol considers splitting the traffic into two available routes, using a simple per-packet allocation scheme.

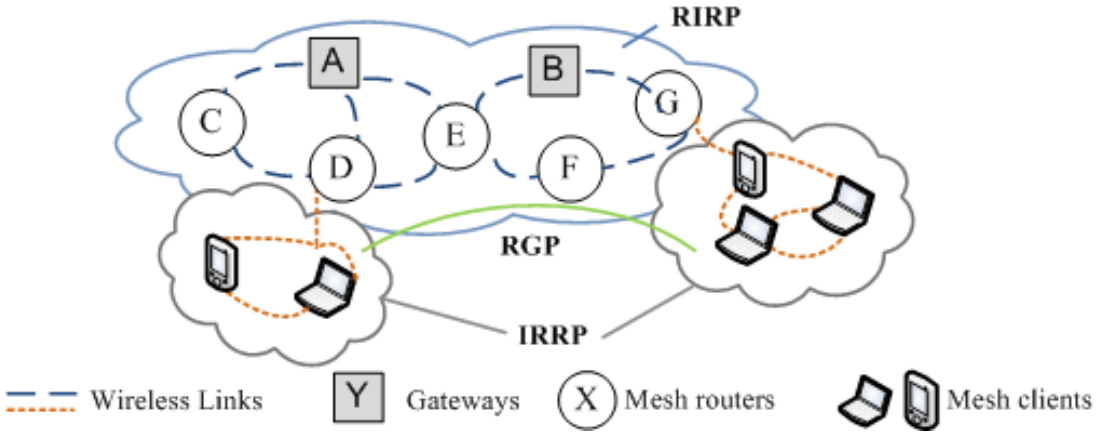


Figure 2.6: Protocol architecture of MHRP.

2.6.3 Hybrid Protocols

2.6.3.1 MHRP

MHRP (Siddiqui, Amin, Kim, & Hong 2007) is a hybrid routing protocol for wireless mesh networks, where multipath is used as a backup mechanism. It combines proactive and reactive approaches, and makes use of the four following sub-protocols, each one used in a different zone, as depicted in Figure 2.6:

- *Router Infrastructure Routing Protocol (RIRP):* as the routes in the infrastructure mesh are relatively static, this is a proactive based protocol. To keep the routing table fresh and accurate, periodic HELLO packets are sent by each router to its neighbor routers at a constant interval. Each node maintains all the possible paths to the other nodes and uses the best one to communicate.

- *Intra-region Protocol (IRRP):* since it is used in the ad hoc region, it is reactive, based on AODV, and uses forward selection to find an alternate route when link is down. Multiple reverse paths are created when a RREQ flows in the network. These paths are stored for future use, decreasing the latency of finding a new route.

- *Region Gateway Routing Protocol (RGP):* is used when routes between two ad-hoc regions are required. The protocol is based on the information provided by RIRP and IRRP. When a node requires a route, it sends a route request to the router responsible for its ad hoc region (who runs the RGP protocol). The router gets the information of the destination

node from other protocols and constructs the available multiple routes, sending them to the source node. As messages pass to and from clients to routers, routing tables are updated.

- *Route maintenance Protocol:* maintains the routing table and provides alternate routes whenever required.

The route discovery mechanism, performed when a node wants to communicate with another and does not have a route to it, can be divided in three stages. The first one is the route request, during which a request is sent to the neighbors of the client, using IRRP. They forward the request if they do not know the route to the destination. During route formation, after the destinations have been found, routes are formed even if the destination node is in another ad hoc region and the operation of RGP is needed. The last phase is the route reply in which a message containing the whole route is sent back to the source. One node-disjoint path is used at a time.

2.6.4 Low-Coupling Techniques

To design a multipath routing protocol concerning interferences between paths (which means guarantee a low degree of route coupling), there are two different techniques that can be applied to select those paths.

- The first one is used in protocols like AODV-DM (Hu & Lee 2007), where the protocol operation includes a mechanism that is able to select zone-disjoint paths. This is done by saving state about interferences in the nodes along the network.
- The other one is called Cluster-based Multipath Routing (CBMPR) (Jie Zhang 2008) which is not fully explored and no concrete studies exist concerning this technique. It was proposed and only preliminary test were made, using scenarios manually configured with limited number of nodes. It states that using a clustering algorithm to group nodes, and then choosing routing paths through different clusters, that do not interfere with each other.

The first approach is more specific, protocol-dependent and in some cases the complexity to maintain state about the interferences in the network is high. Besides, as explained in AODV-

DM (Hu & Lee 2007) (to the best of our knowledge, the unique protocol that uses zone-disjoint paths) more time is needed to discover the paths (two REEQ-RREP cycles).

The second one implies the use of a clustering algorithm, whose goal is to define an abstract structure for the network, where nodes are grouped and typically each group has a leader called cluster-head. This can lead to a higher overhead due to the addition of another component to the algorithm.

Note that, although many clustering algorithms have been proposed in the literature, CBMPR paper does not specify any concrete clustering algorithm. In the next section we survey some clustering algorithms and discuss their applicability to CBMPR.

2.7 Clustering Algorithms

Cluster algorithms may used to improve network performance in metrics such as routing delay, bandwidth consumption, energy consumption, throughput, and scalability (Chinara & Rath 2009). A clustering algorithm divides the network into (usually) disjoint sets of nodes, each centring around a chosen cluster-head. Efficient clustering protocol rely on different design goals, depending on the application they are designed to. For the specific case of cluster-based multipath routing, there are a set of properties that must be considered when selecting a clustering algorithm (stability, convergence time, stability, among others).

Since different clustering algorithms have different properties, each algorithm is suitable for a different application. Some clustering algorithms have $O(n)$ convergence time, where n represents the number of nodes in the network. These algorithms, such as LCA (Baker & Ephremides 1981), RCC (Xu & Gerla 2002), or CLUBS (Nagpal & Coore 1998), are are not scalable and impractical to implement in networks with a large number of nodes.

Considering the scalable algorithms, there are different choices concerning the applications and the formation of the cluster-heads. The Lowest ID (Lin & Gerla 1997) algorithm guarantees that no cluster-head can be in the range of another and its maintenance scheme leads to higher number of re-elections comparing with other approaches. In the Highest Connectivity (Parekh 1994), which aims to use highest connectivity nodes as cluster-heads, the goal is trying to reduce the number of clusters in the network. The mobility of the nodes changes the connectivity and, consequently there are frequent cluster-head re-elections.

Other type of clustering algorithms calculate the weight of each node according to a specific metric. Then, typically, the node with the highest weight between neighboring nodes is selected to be the cluster-head. Examples include MOBIC (Basu, Khan, & Little 2001) whose metric is node mobility, DCA and DMAC (Basagni 1999; Basagni, Chlamtac, & Farago 1997), that assign weight to nodes in function of its transmission range or node mobility, and WCA (Chatterjee, Das, & Turgut 2002) that chooses its cluster-heads depending on the node battery, degree of connectivity or transmission power according to the scenario. ACE (Chan & Perrig 2004) is another clustering protocol that results in a highly uniform cluster formation, close to hexagonal.

To successfully use a clustering algorithm to discover low-coupling paths, the cluster-formation must consider the following aspects:

- 1-hop clustering where all nodes nodes are in the interference range of the cluster-head and a node is at most two hops away of any other node ate the same cluster.
- Clusters formed must be non-overlapping clusters, which means that in a perfect world nodes should be in the interference range of only a cluster-head (as this is not possible, nodes should hear the minimal number of cluster-heads).
- Protocol should have reduced overhead and should be fast and efficient. For the protocol to be used in large networks it must be scalable and the overhead in terms of additional traffic should be minimal.

The criteria used to elect cluster-heads in some of the referred algorithms consider mobility-related metrics to reduce frequent re-elections of cluster-heads. However, to solve the problem of interferences in routing protocols, these are not the main concerns of the clustering protocol. In the next subsections we describe two clustering protocols in detail that have two important characteristics to reach our goal, cluster-heads do not interfere (Lowest ID) and clusters are non-overlapping (ACE).

2.7.1 Lowest ID

Lowest ID (Lin & Gerla 1997) partitions the network into non-overlapping clusters, as the name suggests, based on the nodes IDs.

This algorithm assumes that each node is assigned a distinct ID. Periodically, the node broadcasts the list of nodes he can hear, including itself. A node which only hears nodes with ID higher than itself is a cluster-head, unless it specifically gives up its role. Consequently, the lowest ID a node hears is its cluster-head. The grouping process is done the following way.

Consider that the set of IDs including its one-hop neighbors and itself is called *NodeSet*. If the ID of the node is the minimum in the *NodeSet*, the node becomes cluster-head and removes its ID from the *NodeSet*. Then, it broadcasts a CLUSTER message with its ID and the ID of the cluster-head (in this case they are the same).

When a node receives a CLUSTER message it deletes the cluster ID advertised from its *NodeSet*. Then, if the cluster ID of the receiver is unknown or its node ID is greater than the advertised cluster-head, the node sets its cluster ID to the one advertised.

After this, if the node is still unclustered and its ID is minimal, the cluster ID of the node becomes its own ID. One more time the node broadcasts a CLUSTER message and deletes the node from the *NodeSet*. When the *NodeSet* becomes empty the algorithm ends. During the algorithm execution a node broadcasts one cluster message before the algorithm stops when it decides its cluster ID.

2.7.2 ACE

The ACE algorithm (Algorithm for Cluster Establishment (Chan & Perrig 2004)) consists in two logical parts, the first controls how clusters can spawn and the second rules how clusters migrate dynamically to reduce overlap.

As nodes can start the protocol in different times, this is an asynchronous protocol. Nodes only initiate actions at random intervals to avoid collisions. However, they respond to other nodes messages when they are received. During the protocol execution a node can have three states, namely unclustered (not a follower of any cluster), clustered (a follower of one or more clusters) or it may be a cluster-head (cluster leader).

The formation of clusters is done in a self-elective process. Each node waits for its next iteration and then it takes an action depending on its state.

unclustered: if a node A is in this state when the next iteration arrives, it counts the number

of *loyal followers* (a loyal follower is a follower of only one cluster) that in this case will be the number of unclustered neighbors of the node. If this number is above a certain threshold, the node will spawn a new cluster by generating a random cluster ID and by sending a RECRUIT message to its neighbors. Nodes receiving this message will follow the advertised cluster.

cluster-head: in the case a node A is a cluster-head it prepares to migrate its cluster if a new cluster-head could improve the distribution of nodes. A sends a POLL message to its neighbors asking them how many followers they have, to find the best candidate for the new cluster-head. This one will be the node with the largest potential number of loyal followers in its neighborhood. If a better candidate to cluster-head is found (node B), A will migrate the cluster leadership to the new cluster-head by sending to B a PROMOTE message. When B receives this message it will issue a RECRUIT message with A cluster ID for the clusters that were not part of the cluster to become clustered. Once A receives this message it will issue an ABDICATE message to its neighbors. Nodes that are in the range of A (received an ABDICATE) but are not in the range of B (have not received the RECRUIT message) will leave the cluster.

clustered: in this state the node does nothing.

After a pre-defined number of iterations (typically 3, a number experimentally determined to provide good results) the algorithm is ready to terminate. If a node is a cluster-head it terminates immediately and informs its neighbors. If a node is clustered it waits until its cluster-heads have terminated and then it chooses one randomly.

2.8 Cross-layer Issues

Routing is just one of the needed functions for communication to take place between two nodes. Additionally, data link and transport functions are needed, among others. The way these other functions are implemented may affect the performance of a routing scheme. In this section we address some issues that multipath routing must consider.

2.8.1 Link Layer Issues

The wireless medium is normally shared among all the nodes that are within radio range of each other. When a node wants to transmit and the channel is being used by another node, it has to wait until the medium is free (if they could hear the transmission) or, in the worst case, if it cannot hear the on-going transmission, it transmits blindly and may cause a collision (this is known as the hidden terminal problem).

The quality of transmissions within the same radio neighborhood may degrade due to interferences, even if multiple channels are used. Nodes that affect each other by transmissions, are said to be in the same collision domain. This means that, even using multiple paths to route traffic, transmissions may interfere if they share the same collision domain. Therefore, the use of multipath routing to achieve higher bandwidth may not be as effective as it is in wired networks, as radio interference must be taken into account. Therefore, it is preferable to choose paths that are as independent as possible to avoid this interference (Mueller, Tsang, & Ghosal 2004).

This problem is known as the route-coupling problem (Pearlman, Haas, Sholander, & Tabrizi 2000b). The mutual interference of routes in a common wireless channel causes a dependence between a given flow and every other flow using the same radio region. The cost of maintaining low-coupled routes in an on-demand protocol is high (Marc Mosko 2005).

2.8.2 Transport Layer Issues

As we have seen, multipath routing can provide some robustness to link failures, and facilitate the transmission of packets along paths, avoiding congestion regions. As a consequence, we would expect that the network performance would be better in terms of achieved throughput, when multiple paths are used simultaneously. Unfortunately, this may not be the case, especially if we use TCP as transport protocol. In (Lim, Xu, & Gerla 2003; Ye, Krishnamurthy, & Tripathi 2004) it has been shown that the use of multiple paths simultaneously degrades TCP performance.

In fact, although TCP is presently used for all networks, the optimizations for its operation were developed considering wired networks. Therefore, when operating over an unreliable radio medium, in face of external interference, multiple access contention and some mobility, its

performance is not stable.

We can consider two main features of the TCP protocol that cause its poor performance in these scenarios:

- *RTT estimation problem:* TCP uses average Round Trip Time (RTT) to set a timeout and decide when packet loss occurs. When we have multiple paths, the RTT in the longest one can be much shorter than the average RTT (used for the estimation). As a result, TCP can prematurely timeout packets taking the longest path.

- *Out of order problem:* if multiple paths are used to forward traffic from the same connection, packets going through different paths may arrive at the destination out of order, which generates duplicate ACKs. TCP protocol reacts to this by reducing the congestion window, and consequently limiting the throughput, which is an action contrary to the one we are trying to achieve with multipath routing.

Another issue has been described in (Ye, Krishnamurthy, & Tripathi 2004): sometimes long TCP connections steal some bandwidth from short TCP connections (as bandwidth resources are shared between the nodes in an ad hoc network). It is commonly assumed that multipath routing is good to be used in long connections, in terms of hop count (not considering the higher volatility on this routes) where recovery using the default TCP mechanisms is longer than recovery from failures with multipath.

However, as proposed in (Lim, Xu, & Gerla 2003), if multipath routing is used only as a way of providing backup routing it brings advantages, because using one path at a time reduces the probability of out of order packets. In the same paper, it was shown that two paths offer the best TCP performance, but they have to be chosen carefully in order to achieve good results.

It is also possible to use SCTP (Ong & Yoakum 2002), whose multi-streaming function allows data to be partitioned into multiple streams that have the property of independent sequenced delivery, so that message loss in any stream will only initially affect delivery within that stream, minimizing TCP problems referred above.

Table 2.1: Summary of the protocols presented.

Protocol	MOLSR	MMESH	AODV-DM	AODV-BR	AOMDV	DSR-MP	SMR	MHRP
Base protocol	OLSR	AODV	AODV	AODV	DSR	DSR	-	-
Motivation	Lower delay and packet loss	Deal with high bandwidth require-ments	Improve on-demand protocols	Reduce routing overhead	Balance the traffic load	Deal with reliability require-ments	Use network resources efficiently	Security in mesh networks
Proposed function	Backup routing	Congestion avoidance / backup routing	Avoid route-coupling	Backup routing	Backup routing	Reliability	Throughput enhance-ment	Backup routing
Source or hop-by-hop	Source	Source	Hop-by-hop	Hop-by-hop	Hop-by-hop	Source	Source	
Proactive, reactive or hybrid	Proactive	Proactive	Reactive	Reactive	Reactive	Reactive	Reactive	Hybrid
Traffic distribution	Single-path	Single-path	Multipath/ Single-path	Single-path	Single-path			Single-path
Route rediscovery	-	-	-	When all/one path fails	When all/one path fails	When all/one path fails	-	-
Route relations	Node-disjoint	n best disjoint paths	Node-disjoint	Non-disjoint/Best path	Link/node-disjoint	-	Disjoint paths	Node-disjoint

2.9 Discussion

Table 2.1 summarizes the routing protocols presented. Comparing multipath routing protocols with unipath routing protocols, first we can observe that they add complexity and additional overhead. Also, the maintenance of routing tables in intermediate nodes results in larger routing tables. In routing there is also an additional phase needed in the protocols, which is traffic allocation, resulting in more time needed to establish routes.

2.9.1 Key Aspects in Multi-Path Routing

Based on the survey presented, we consider that a well designed routing protocol should address the following aspects:

- *Multiple paths:* in a limited number, multiple paths should be used in order to establish a balance between the number of used paths and negative aspects like interferences or overhead.
- *Low overhead:* as we referred previously, one of the goals of routing is to discover and use multiple paths, thus benefiting from this, but with lower additional overhead.
- *Good performance:* when multiple paths are used there are more routes and state to maintain. Sending traffic over the paths in function of its quality is a way of maintaining

paths alive (Marc Mosko 2005). Having mechanisms to manage routing tables avoiding unnecessary information is also needed to achieve a well designed protocol.

- *Low degree of route coupling:* as in wireless medium interferences among different channels are a conditional factor in transmissions, in an optimal protocol, these interferences must be minimized.

2.9.2 Throughput Performance of Multiple Independent Paths

Most of the studies existing in the literature concerning low coupling only consider 2 paths (Liaw, Dadej, & Jayasuriya 2004; Waharte & Boutaba 2006; Maimour 2008). Performance of multiple independent paths (2 is the number used in the studies mentioned) is said to allow much better performance than the use of a large number of interfering paths. Using two nodes, it is shown that multipath routing has a maximum improvement factor of 50% over a single-path and in the majority of the networks, the probability of finding a third non-interfering path whose length offers advantages is low (Liaw, Dadej, & Jayasuriya 2004). Obviously this can not be the case, specially if the network is sparse and the minimum distance connecting two nodes is not high in terms of hop number. It is also important to say that, multipath is only good for paths with 3 or more hops in length. As the rate of transmissions over the last hop decreases with the increasing in path length, the probability of a packet colliding on the last hop decreases.

One of the refered works (Liaw, Dadej, & Jayasuriya 2004) also studies the limiting factor in multipath routing, saying that the optimal number of paths is limited by the degree of spatial reuse achievable at the source node, and not by the number of paths used. Although TCP is used in the study to evaluate the impact of packet losses and packet-reordering with two paths, both TCP and UDP present similar results (Waharte & Boutaba 2006) in terms of throughput achieved. However, experiments with TCP can have a higher delay due to packet reordering.

Summary

This chapter introduced some fundamental concepts which will be central to the discussion in the next chapter. A survey of the main protocols and applications of multipath routing in

Wireless Mesh Networks was presented, however, there are only a solution that uses multipath to improve throughput, using low coupling routes to forward data. To achieve this goal in an effective way cluster-based multipath routing can be used, as discussed before. Some clustering algorithms were presented as well as its appliance to this technique. Moreover, main challenges and upper bounds in this field were also discussed.

As none of the existing multipath solutions is built on top of a clustering algorithm, the next Chapter introduces a protocol that addresses this gap.

3 Low-Coupling Cluster-Based Multipath Routing Protocol

3.1 Introduction

As discussed in the previous chapter, the use of multiple paths at the same time to route traffic can degrade the total amount of bandwith , instead of improving it. This Chapter introduces Low-Coupling Cluster-Based Multipath Routing Protocol (*LoCoup*), a multipath routing algorithm designed to improve bandwith usage between a given source and destination. This improvement is achieved by using multiple low-coupled paths to communicate. This key feature of the protocol, as its name suggests is build on top of a clustering algorithm. A description of the system and its components can be found in the following sections.

The goal of the proposed protocol is to find non-interfering paths, in order to diminish interferences when concurrent transmissions are used to improve bandwidth. As described in the previous Chapter one possible approach to achieve such goal is to use Cluster-based Multipath Routing (CBMPR) (Jie Zhang 2008). The *LoCoup* protocol uses this approach as the other one (used by AODV-DM) leads to higher delays and limited number of paths (otherwise, the complexity to manage state in intermediate nodes would be too high).

The idea behind this technique is represented in Figure 3.1. First of all, nodes in the network have to be grouped in clusters (represented in the figure by circles), each cluster having a cluster-head (nodes in orange). Cluster-heads from different clusters do not interfere with each other, so by selecting paths that pass in this nodes, non-interfering paths are selected. For example in Figure 3.1 there are three non-interfering paths between nodes *SRC* and *DST*.

3.2 Building Blocks

According to the intuition of the protocol previously described, *LoCoup* operation can be divided in two main functional steps. The first one is the clustering which groups nodes and

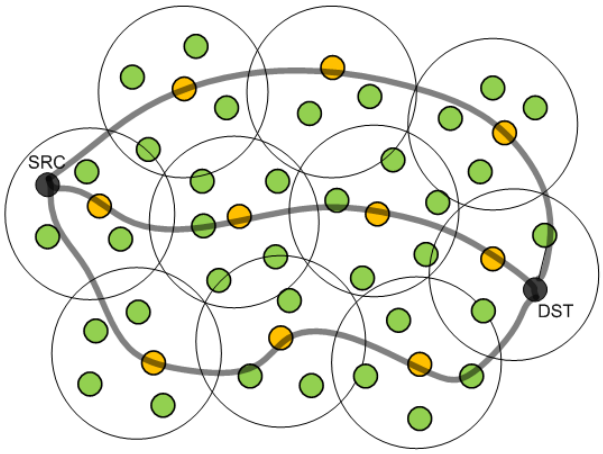


Figure 3.1: Use of clustering to choose non-interfering paths.

the second one is an hybrid routing protocol, responsible for finding and maintaining routes. A novel aspect of *LoCoup* is that this step explicitly takes into account the potential interferences among nodes in the network. This composes the main challenge of the protocol: to ensure that paths are totally disjoint.

The two protocol components relate with each other as follows. First of all, information about clustering has to be available. Precisely after the clustering phase, each node in the network is able to retrieve the following information:

- the group the node belongs to and its neighboring groups (as well as the nodes that give access to that groups);
- knowledge about the entire network represented by an overlay defined by groups.

Referring to the routing component, when a node wants to communicate with another one and the reactive part determines the destination group of the destination node, the proactive part of the algorithm at the source node is able to calculate the valid paths, according to the algorithm specification and using the information retrieved from the clustering operation.

Both building blocks of the protocol have to ensure certain properties in order to accomplish this goal, namely the choice of the clustering algorithm is of paramount importance for the approach to work in practice and the routing protocol has to be suitable for this task. In the next subsections we make an overview of clustering and routing components.

3.2.1 Clustering as a Low-Coupling Technique

In *LoCoup* paths are chosen according to clustering distribution of nodes. Our work extends the work of (Jie Zhang 2008) in multiple dimensions. In particular, our clustering algorithm explicitly considers the following constraints:

- All nodes in a cluster are in the interference range of the cluster-head and a node is at most 2 hops away of any other node in the same cluster;
- Clusters are non-overlapping, which means that each node should be in the communication range of the minimal number of cluster-heads;

Also, we propose a concrete algorithm to discover, deploy, and maintain multiple paths using clustering information. The overview of this component is presented below.

3.2.2 The Routing Algorithm

Our algorithm combines ideas from proactive link-state protocols and reactive source-based routing protocols in order to exploit the advantages of both approaches. As scalability is a requirement of our solution, in a large network it would be impractical to have a proactive protocol that was able to maintain paths to any other destination in the network. At the same time, using a reactive solution to achieve this goal would not be possible without using a large amount of traffic, as it is really expensive to discover all the paths in the network (this would be a way to manage interferences), or to maintain state in intermediate nodes along the network, and the overhead added to the messages would also be high. Besides, we aim at solving some of the AODV-DM limitations, namely the high route discovery delays and limited number of paths found by the later algorithm.

According to its functionality, *LoCoup* is a reactive protocol, as it finds and maintains paths to route traffic between a source and a destination when they are needed. A Route Request (RREQ) is broadcasted when a path is requested by a source node and the destination replies to the request by sending a Route Reply (RREP) containing the location of the destination node. Then the source node can use proactive information and according to the protocol established rules select the disjoint paths that can be used to route traffic.

The decision about which paths should be used to route traffic is made by the source node but to make this decision it needs to know about the distribution of nodes in the network in order to select non-interfering paths. This information is collected in a proactive way, using a link-state approach.

As *LoCoup* combines proactive and reactive solutions, we can say it is an hybrid protocol. Hybrid routing protocols are commonly used in heterogeneous networks to enhance the performance of routing protocols in scenarios with mobile and static environments. In this case, we use an hybrid protocol whose core functionality (routing) is based on a reactive solution and the proactive one is needed to achieve a specific functionality - choose non-interfering paths in a short amount of time.

In the next sections the proactive component (clustering) and reactive component (routing mechanism) are described in detail. The solution consists of formation and maintenance activities, for both clustering and routing.

3.3 Node Clustering

To achieve efficient multipath routing, we make the following assumptions that are the same as LID and other algorithms in the literature do.

- Every node has a unique ID and knows the ID of its 1-hop neighbors.
- A message sent by a node is received correctly within a finite time by all its 1-hop neighbors.
- The topology of the network does not change during the algorithm execution.

First of all, it is important to have in mind the requirements of the protocol we propose to build, which are the following.

- The algorithm should result in a highly uniform cluster formation that can achieve a packing efficiency closest to hexagonal with reduced overlap.
- A cluster-head cannot interfere with another cluster-head, as the goal of this mechanism is to form zones that do not interfere with each other.

- When the clustering protocol finishes, all the nodes are clustered (belong to a cluster) or are cluster-heads (cluster leaders).

To the best of our knowledge, there is no previous protocol in the literature that meets all the requirements listed above. Therefore, we adapted the ACE (Chan & Perrig 2004) algorithm, which provides a good cluster distribution but does not ensure that cluster-heads do not interfere.

The process of clustering can be described as a combination of two stages, cluster formation and cluster maintenance. The cluster formation stage deals with the logical partition of the mobile nodes into groups and the election of suitable nodes to act as leaders in every group. The objective of cluster maintenance is to preserve the existing clustering structure as much as possible.

3.3.1 Cluster Formation

In our algorithm (we named it ACE+), the cluster formation consists of two logical parts. The first one controls how clusters are formed (by having a node that elects itself as leader) and the second controls how nodes dynamically migrate to reduce cluster overlap.

The algorithm does not require node synchronization, therefore nodes can start the protocol in different times. During the protocol, nodes respond immediately to messages from other nodes but will only initiate actions at random intervals to avoid collisions. Each of these actions is called an iteration. Across protocol iterations, a node can be in one of the following states: *cluster-head*, *unclustered*, or *clustered*. At the beginning of the protocol each node is unclustered which means it is not associated with any cluster-head. Cluster-heads are cluster leaders and when a node associates with a cluster-head it becomes clustered. A node can be clustered to more than one cluster-head during the protocol operation.

During cluster formation new clusters are spawned by letting nodes self-elect as candidate cluster-heads. The procedure that leads to this election runs every iteration. Each node waits for its next iteration a random amount of time, after which its action depends on its current state, as described below:

unclustered: in this state, the node polls its neighbors to determine how many *loyal followers* it has. A *loyal follower* is a follower of only one cluster. Each node determines its loyal

followers based on the periodic exchange of HELLO messages during the clustering phase. If the number of *loyal followers* is above a given threshold (Chan & Perrig 2004) the node will declare itself as a cluster-head, generating an ID for the new cluster. Then, the cluster-head broadcasts a RECRUIT message and the neighbors that receive this message become followers of the new cluster.

cluster-head in this state, the node checks if a cluster-head migration could improve the distribution of clusters. A POLL message is sent by the cluster-head to its neighbors to determine which has the highest number of *loyal followers*, in order to activate the following migration process. Let l be the loosing cluster-head that demotes itself and w be the winner cluster-head that will adopt l and some of its followers (namely the l node, those who are neighbors of both l and w and the ones who are in the interference range of w and are not neighbors of l). Migration is initiated by having l send a PROMOTE message to w, which in turn issues a RECRUIT message, so that nodes that hear the new cluster will follow it. Finally, when l receives this message, it sends an ABDICATE message such that all the nodes who are not in the range of w may find another cluster-head.

clustered in this state, the node does nothing.

The clustering algorithm repeats this procedure for a pre-defined number of iterations i. This number must be experimentally determined by executing the algorithm in multiple scenarios. In our case, we use 3 interactions, which is consistent with the values obtained with the original ACE algorithm (Chan & Perrig 2004). After the last iteration, all nodes execute the termination procedure described bellow, after which all nodes send a DONE message to its neighbors with its final state and the ID of the cluster they belong to (neighboring update).

- If the node is a cluster-head, it will terminate and send the DONE. If a cluster-head (B) receives a message of this type from another cluster-head (A), B will give up its role as a cluster-head and become follower of A. For the neighbors that are clustered to B stop following it, B sends an ABDICATE message. In the case two cluster-heads terminate concurrently, the one with higher ID remains cluster-head.

- If the node is clustered, it waits until all the cluster-heads it hears have terminated and selects the closest one as its associated cluster-head. The distance between nodes and cluster-heads is determined using the signal strength of the received transmissions.

- If the node is unclustered (which means no cluster-head is in its neighborhood), it declares itself as a cluster-head.

This procedure is different from the original ACE as it does not ensure that cluster-heads interfere with each other and nodes hearing more than a cluster become clustered to a random cluster-head, not the closest one. For example, if we consider a network of 16 nodes, disposed in a 4×4 grid where nodes are in the interference range of its direct neighbors and not the ones in diagonal, the ACE protocol would choose the 4 central and the corner edge nodes to be cluster-heads, as the central are the ones with the highest number of loyal followers. It is clear that this protocol does not ensure the properties we are looking for.

The DONE message generated by cluster-heads when they terminate is propagated in a three hop range, as detailed in Algorithm 1. So that, at the end of the algorithm, all nodes know not only the cluster-heads they link to and the nodes that link them to that clusters. After this final step, all nodes have information about the surrounding interferences (stored in a table called clusters table).

3.3.2 Cluster Maintenance

In WMNs the topology can change due to nodes failing, leaving, joining or simply moving. Therefore, the cluster configuration may change over time and a maintenance procedure is required.

Algorithm 1 ACE+ neighboring update procedure

```
Input: update //message that arrived

nHops ← update.getHops()
nChs ← update.getNrOfCHsTraversed()
senderState ← update.getSenderState()
senderCluster ← update.getSenderCluster()

if (nHops ⩽ 3 ∧ nChs == 1 ∧ senderState == CLUSTER_HEAD) ∨ (nHops == 1 ) then
  CHTable.insert(myCluster, senderCluster, nHops)
end if
if nHops ⩽ 2 ∧ nChs == 1 ∧ senderState == CLUSTER_HEAD then
  propagateUpdate(update)
end if
```

The ACE protocol that was used as a base to the clustering protocol we use, runs when the set up of the network occurs and then the information provided by the algorithm is static and does not change.

For updating information about clusters there are two options concerning when the information is updated: i) the algorithm can be allways running in background, generating the migration of the clusters when reaching a certain threshold or condition; ii) clustering information can be updated periodically, when an error in a path occures due to a faillure or node movements.

In one end, the first option is an adaptive process and so, in each moment the clusters are updated and form groups with the best characteristics. On the other end, such an approach generates too much traffic and is allways looking for the "perfect configuration" of the network, which is not necessary for the protocol to perform well. Having this in mind, the second option is much more suitable for choosing the best time to rearrange clusters, reducing the overhead associated with clustering.

Since these changes are expected to be infrequent in wireless mesh networks, we opted to re-apply the cluster formation algorithm whenever a change is detected. The conditions under which it is done are stated bellow.

To detect changes in the topology we employ a reactive strategy, where broken links are identified during the propagation of application data, therefore saving on the signaling cost. The last node in the path that received but was not able to forward the data message is responsible for initiating the cluster formation algorithm. To this end it floods the network with an RECLUSTER message that, when received, returns every node to its initial (unclustered) state and restarts the cluster formation procedure.

3.3.3 Clustering Improvements on *LoCoup*

The protocol we propose uses link-state for the nodes to have an overview of the entire network. It is known that this type of protocols produce a large amount of overhead to achieve this goal. However, in our case, by using clustering we reduce the overhead by: i) minimizing flooding; ii) minimizing the length of the data messages; iii) minimizing the processing effort needed to find the maximum number of non-interfering paths.

Flooding Minimization Information flooding occurs when a node needs to send a message to all nodes in the network. Specifically, when a node performs a route discovery by sending a RREQ message or a TOPOLOGY message with link-state information. As all nodes in a cluster are neighbors of the cluster-head, the following procedure can be applied to reduce the number of messages needed to flood information along the network.

- If the packet arrives at a cluster-head, it will save the information contained in the message (if it is the case) and then broadcasts the packet including the address of the gateways it knows are a way to reach other clusters and the clusters that are reached through that gateways.
- If a clustered node receives a TOPOLOGY message it saves the information it advertises and then discards it. If it receives a RREQ message it is dropped immediately.
- In the case a node receives a broadcast packet (TOPOLOGY or RREQ) and its address is included in the gateway list, it retrieves the next cluster from the message and searches for its cluster-head or the next gateway to reach the cluster.
- If the packet arrives at a cluster-head node and it had already received the packet, it is dropped.

Minimization of Messages Length Source routing has the disadvantage to increase packet size as the nodes need to include in its header the path through which the data should pass. However, in this case when the source node decides that a path is valid according to the protocol rules, the data has to follow a specific path and a hop-by-hop approach can lead to the wrong operation as a intermediate node can have more than a path to the same destination node. So that, source routing has to be used in order to guarantee that paths used are the correct ones. Clustering also helps to minimize the path information included in the messages as in spite of all the nodes that belong to the path only the cluster-heads are stored in message header. As between two cluster-heads there are at most two hops, the source information can be reduced by 1/3, comparing with a pure link-state approach.

Minimization of the Processing Effort By having the graph that represents the network (information about the nodes and the links that connect them), it is possible to compute all the

available paths between a source and destination and determine the non-interfering paths. Of course this approach is not practical as the information needed is too much and the resources to compute paths would be high. Clustering can also diminish the processing effort necessary to calculate these paths as the network is represented by an overlay network defined by cluster-heads and it is less expensive to analyse them instead of analyse all the nodes.

3.4 Routing - Route Discovery and Maintenance

To allow the establishment of the multipath routes nodes execute a specially designed route discovery algorithm whose resulting paths are guaranteed to not interfere with each-other. Additionally, a maintenance procedure is executed whenever a path is disrupted.

In this subsection the operation of the reactive component of the protocol is described, namely the requirements the protocol needs to accomplish, its characteristics and steps.

The requirements that the reactive component of this protocol should accomplish are the following ones.

- Non-interfering paths have to be selected to meet the algorithm requirements. Even so, the first and the last nodes of the path all have to be in the range of the sender/receiver (probably in the same cluster), which means that first and last hops of a path can interfer with each other. Although this is true no node belongs to more than a path and no link is used in two different paths, which means the protocol is node-disjoint and link-disjoint at the same time.
- In order to reduce the overhead the multipath routing protocol adds natively, the state that has to be maintained in each node/for each route should be minimal.
- As refered in the previous chapter, protocols can classify in two different categories, according to the way they record paths: source or hop-by-hop. Each one has advantages and disadvantages. As multiple non-interfering paths have to be selected, and during the reply an intermediate node can have more than a path to the same source node, the criteria to choose the path to use cannot be the shortest or the latest, but the one that the receiver have determined to be the one that does not interfer. So, the solution is to indicate the path specifically in the packets headers (source protocols).

3.4.1 Route Discovery

As we have previously refered, the protocol proposed is hybrid. The proactive component consists of letting each node maintain a link-state database of the overlay defined by the cluster-heads that result from the clustering algorithm described before. Thus, like in any link-state protocol, all nodes maintain (some) topology information. As cluster-heads do not interfere with each other, in the worst case two nodes will be needed to link two cluster-heads, so the information maintained in each node has to include three hop neighboring information. In our case, this information forms the topology table and refers exclusively to the cluster-head nodes and the gateway nodes that link two clusters. Unlike flat link-state protocols, not every node is required to send link-state updates, only cluster-heads need to do so; this highly reduces the signaling cost of maintaining link-state information.

To support this operation, when the clustering algorithm terminates, cluster-head nodes (and those nodes alone), broadcast a link-state message containing the identifier of their "next-hop" cluster-heads as well as the gateway node to reach the advertised cluster-heads. These link-state messages are flooded on the network and stored by each node in a local link-state database. Using this database each node is able to maintain information about the topology of the overlay defined by the cluster-heads.

Node that, in this way, any node can always find the available low-coupling routes between itself and a given target cluster ID. These paths are all paths that do not share cluster-heads, other than the destination cluster-head or the cluster-head of the source node.

The reactive component of the algorithm is associated with the discovery of the cluster-head associated with the destination node and with the calculation of the paths. Since link-state information is only maintained about cluster-head nodes, one still needs to find the location of the destination node in the cluster overlay. This is performed using a simple RREQ/RREP protocol, where the route-reply includes the cluster ID of the destination node and the set of neighboring clusters, as well as the gateways to reach that clusters. According to the way routes are calculated, the protocol is also classified as reactive, as it does not always keep a routing table. Instead, it only computes the multiple paths when data packets need to be sent out.

3.4.1.1 Path Computation

Possible paths are found linking clusters from the topology table. By combining clusters the source node gives access to, with clusters the destination can reach directly, possible paths are found. However, they have to be validated in order to choose non-interfering ones.

3.4.1.2 Path Validation

Although low-coupling paths share the source and destination clusters, they should not share links or nodes. In order to ensure that accepted paths do not interfere with each other, additional information needs to be maintained during the path validation procedure.

- The set of clusters (represented by its Cluster ID) that the source node links to. This is called the *SourceLinkClusterSet*
- The set of clusters that the destination node gives access to (they are transmitted in the RREP message). This is called the *DestinationLinkClusterSet*
- The set of clusters that interfere with the path that has been accepted. This set is called the *ExcludeClusterSet*.
- The set of gateways that interfere with the path that has been accepted. This set is called the *ExcludeGatewaySet*.

The information above is used by the source in order to accept or discard a new path as follows. The path is accepted if the first and the last hop of the path belongs to the *SourceLinkClusterSet* and *DestinationLinkClusterSet*, respectively, and clusters of the path are not in the *ExcludeClusterSet* nor gateways interfering with the path are in *ExcludeGatewaySet*.

If the new path is considered to be valid, the first and last hops are removed from the *SourceLinkClusterSet* and *DestinationLinkClusterSet* and the clusters of the path are updated with clusters of the new path. This procedure is described in Algorithm 2 in more detail. Note that info related with gateways and clusters are retrieved from the topology table.

Moreover, to ensure the minimum possible length of the selected paths, two paths can be selected and have the possibility to use the same gateway to link two clusters (although they

Algorithm 2 *LoCoup* protocol - path validation

```
Input: listPaths
for path ∈ listPaths do
  firstHop ← path.first()
  lastHop ← path.last()
  if SourceLinkClusterSet.contains(firstHop) ∧ DestinationLinkClusterSet.contains(lastHop)
  then
    pathValid ← true
    x ← 0
    while x < path.size() do
      if ExcludeClusterSet.contains(path[x]) then
        pathValid ← false
      end if
      x ← x + 1
    end while

    x ← 0
    while x < path.size() -1 do
      cluster1 ← path[x]
      cluster2 ← path[x+1]
      if ExcludeGatewaySet.contains(getGateways(cluster1,cluster2)) then
        pathValid ← false
      end if
      x ← x + 1
    end while

    if pathValid then
      listFinal.insert(path)
      SourceLinkClusterSet ← SourceLinkClusterSet - firstHop
      DestinationLinkClusterSet ← DestinationLinkClusterSet - lastHop
      ExcludeClusterSet.insert(path[x])
      while x < path.size() do
        ExcludeClusterSet.insert(path[x+1])
        cluster1 ← path[x]
        cluster2 ← path[x+1]
        gwList ← getGateways(cluster1,cluster2) + getGateways(cluster2,cluster1)
        for gateway ∈ gwList do
          ExcludeGatewaySet.insert(gateway)
          clList ← getAllClustersInterferingWith(gateway)
          for cluster ∈ clList do
            ExcludeClusterSet.insert(cluster)
          end for
        end for
        x ← x + 1
      end while
    end if

  end if
end for
return listFinal
```

have other possibilities). To prevent the use of the same gateway by two paths, a list of the common gateways are included in the packet header.

Note that it would be possible to convert the algorithm to apply a pure proactive approach, by letting cluster-heads disseminate also the entire membership of nodes in their cluster. This would avoid the need for the reactive component described above. However, this would increase the cost of the link-state protocol, not only because the link-state messages would be much larger, but also because updates would need to be much more frequent to account for node mobility. With our approach, the link-state information is not only smaller but also much more stable.

3.4.2 Multipath Routing

As described above, as soon as the source knows the cluster-head of the destination, it is capable of, locally, finding the existing low-coupling routes to the destination. These routes are characterized by the set of cluster-heads that the route transverses. Thus, source routing can be used to transmit individual packets via one of these paths. As a result, no additional route information is stored in intermediate nodes between the source and the destination.

In all our experiments, multiple paths are used by letting the source send packets in a round-robin fashion among all the selected paths. Obviously, our protocol puts no constraints on the way the multiple paths are used by the application.

3.4.3 Route Maintenance

We consider three different events that may cause routes to be recomputed, namely:

- the source node moves;
- the destination node moves;
- the topology of the cluster-head overlay changes.

If the source nodes moves, but the cluster-head overlay remains unchanged, the source node only needs to locally recompute the paths. No other action is required.

If the destination node moves and becomes clustered to another cluster-head, it sends to the source a DestinationMoved message, indicating its new cluster-head. When the source received this message, it uses its local link-state information to compute new routes.

Finally, if the topology of the cluster-head overlay changes in such a way that one of the paths used by the source becomes invalid, this is detected when source-routing is being applied. When this occurs, the error is notified to all nodes through the flooding of an Recluster message, which causes the clustering algorithm to be re-executed. When the clustering formation procedure ends source node simply recomputes the paths locally.

3.5 Traffic Balancing

As refered in the previous chapter, when more than a path exist, traffic can be distributed according to different distribution schemes. As our goal is to improve the throughput in the network, we want to consider the possibilities that apply to concurrent transmissions.

There are two possibilities for this case, the first one distributes the traffic equally for the existing paths and the other one uses a non-uniform distribution.

Uniform distribution of traffic (round robin). Consists of sending packets in a round robin fashion. In other words in a certain period of time, the same number of packets are sent out through each path is the same and the traffic is scheduled circularly.

Non-uniform distribution of traffic. This way of distributing traffic is also known as unequal cost scheme. The idea is to select a metric and then select paths according it. As a result, best paths are more used than the ones that offer worst quality of service.

Although the second option may be better in terms of results, it adds complexity as good metrics have to be found and a distribution mechanism implemented. As a result we decided to use round-robin, not excluding the possibility of considering the second approach in the future.

3.6 NS-2 Implementation

The proposed protocol was implemented in the widely used NS-2[1] simulator. Having been under constant investigation and enhancement for years, NS-2 has become the most widely used open source network simulator, and one of the most used network simulators. It is popular in academia for its extensibility (due to its open source model) and plentiful online documentation.

3.6.1 The NS-2 Simulator

NS-2 was the simulator the result of an on-going effort of research and development. It is a discrete event simulator targeted at networking research. NS-2 provides modules for numerous network components such as routing, transport or application. Moreover, it allows the definition of new modules.

NS was built in C++ and provides a simulation interface through OTcl, an object-oriented dialect of Tcl. The user describes a network topology by writing OTcl scripts, and then the main NS program simulates that topology with specified parameters.

3.6.2 *LoCoup* Implementation

Performance evaluation of higher layer protocols are affected by a set of factors in the physical layer and in the MAC layer, and different simulators can lead to different results (Takai, Martin, & Bagrodia 2001). Computation of interference and noise determines the probability of successful signal reception for a given frame. There are essentially two methods of doing this estimation, the SNR threshold based and the BER based. The last one is considered to be more realistic and accurate, however the BER based (the only available in NS-2) requires less computational effort and can be a good abstraction if each frame length is long.

Another important factor is the path loss, which defines the average signal power loss of a path. While the free space model is the default model in some simulation platforms, the two-ray model (the one we have used that is the default in NS-2 simulator) is more realistic (Takai, Martin, & Bagrodia 2001). Moreover, we have set the value of the carrier sense threshold (CST)

[1] http://www.isi.edu/nsnam/ns/

equal to the value of the receiving threshold (RXT), similarly to other studies in this field. As a result, only nodes within the interference range of a node can hear the signal.

Some extensions have been proposed to improve the modeling of the lower layers, namely the dei80211mr multirate extension[2] or the mac802.11Ext extension, already included in the last versions of NS-2. Unfortunately, these extensions are poorly documented and the pre-defined configuration parameters do not include scenarios compatible with our goals.

Summary

In this chapter we have described *LoCoup*, a cluster-based multipath routing protocol that uses a clustering algorithm to calculate non-interfering paths to route traffic. As we have seen, the protocol has two building blocks, clustering and routing. For the first one, a clustering algorithm was designed specifically as the solutions we have found in the literature were considered to be not suitable for our goals. The proposed routing mechanism uses link-state information to compute paths and to validate them. When the protocol terminates, paths that pass through cluster-heads are selected, which means that concurrent transmissions using this paths should improve the throughput. In the following chapter an extensive evaluation of the solution is presented.

[2] http://www.dei.unipd.it/wdyn/?IDsezione=5090

4 Evaluation

4.1 Introduction

In this Chapter we describe the experiments that were carried out in order to evaluate the proposed solution and compare *LoCoup* with other routing schemes used for the same purpose. To accomplish this goal we executed a series of experiments that compare the performance of *LoCoup* against AODV-DM (as it also explicitly considers interferences between paths). We also compared *LoCoup* with the an algorithm that provides optimal route discovery, i.e., returns the maximum number of routes possible in any given network. This algorithm makes use of complete knowledge of the network topology, and by no means represents a realistic solution to the multipath routing problem. Its sole purpose is to provide a reference value, that captures the best possible results for a given topology.

We begin by describing the experimental settings and the criteria used in the evaluation. First of all we present and discuss the performance of multipath routing. Then, as our solution requires a clustering algorithm, we present results obtained with different clustering algorithms. These results show that ACE+ has advantages over other algorithms. Finally, and using ACE+, the chapter provides results for the signaling costs, and latency and gains of *LoCoup*.

4.2 Experimental Settings

To evaluate the performance of *LoCoup* we developed a testing environment using the NS-2 simulation tool. A 802.11b network was simulated using IEEE 802.11 MAC layer operated in DCF mode at 11Mbps.

We implemented *LoCoup* and used, for comparison purposes, a version of AODV-DM (Hu & Lee 2007) that we have implemented as well. Furthermore, a modified version of OLSR (Jacquet, Mühlethaler, Clausen, Laouiti, Qayyum, & Viennot 2001) (called OLSR+) to discover multiple

non-interfering paths between a given source and destination was also developed. As OLSR+ can build the complete network graph, the discovered paths are optimal in terms of hop count (considering that the shortest path is always used) and the number of paths is always the maximum for the network considered.

4.3 Evaluation Criteria

We begin by evaluating the optimal number of paths in a multipath solution by measuring delivery ratio in different circumstances. The delivery ratio is also used (in addition to the number of paths found in each case) to determine the suitability of different clustering algorithms to support multi-path routing.

Moreover, as our solution intends to build a protocol that uses multipath to perform well under heavy traffic scenarios, the evaluation considers the following aspects: number of discovered paths and length of those paths, signaling cost needed to discover such routes, quality of the discovered paths measured one more time by the delivery ratio and latency in the process of route discovery.

To evaluate such metrics, different random scenarios have been generated. Some parameters of the scenarios vary according to the specific evaluation requirements, nevertheless most of the simulation experiments share the same network layout and traffic patterns whose parameter values are presented in Table 4.1.

Parameter	Value
Data rate	11 Mbps
Basic rate	1 Mbps
RTS/CTS	disabled
Transmission range	250m
Packet size	1500 bytes
Sending interval	0.005 packets/s
Simulation area	1500 × 1500 m
Number of nodes	125
Simulation time	100 s
Propagation model	TwoRayGround

Table 4.1: Parameters common to all simulations.

The evaluation results are provided and discussed in the next sections.

4.4 Optimal Number of Paths in a Multipath Solution

Multipath has been proposed for several purposes, but the use of multipath for increase available bandwidth between a source and a destination has been mainly studied using two paths at the same time (Liaw, Dadej, & Jayasuriya 2004; Waharte & Boutaba 2006; Maimour 2008). In this case, typically the paths are used by allocating to each path the same amount of traffic. It is known that the increase of the bandwidth provided by a multipath solution is not directly proportional to the number of paths used (Liaw, Dadej, & Jayasuriya 2004). There are several reasons that explain this behavior. Firstly, the contention both in the source and in the destination nodes limits the number of paths that can bring advantages. Secondly, it has been observed that the degree of spatial reuse in the source node is the limiting factor (Waharte & Boutaba 2006). Third, considering that only the source and destination nodes are common in different paths and all other nodes do not interfere with each other, the maximum number of paths that can be established considering this constraint would be 4 due to the following reasons. Consider that nodes are disposed as regular polygons with 4 or more edges (i.e., a square, pentagon, hexagon, etc), where the cluster-head is in the center and there is a node in each vertex. These polygons can be decomposed in triangles with one vertex being the center of the polygon. However, only in the square topology the triangles are not equilateral, which means the distance between the cluster-head (the center) and the nodes is smaller than the distance between the two nodes (corresponding to the other vertex of the triangle). Therefore, it is possible to place 4 disjoint nodes around a given node, but it is not possible to do it with 5 or more nodes. To experimentally assess how many paths still bring advantages, we have used OLSR+ to compare delivery ratio of a single scenario where 4 paths between the source and the destination nodes are available and nodes are distributed in a regular grid. In each case, we limit the number of paths, in order to observe the variation of the desired metric.

4.4.1 Concurrent Delivery using Round Robin

These tests were performed using both UDP with CBR traffic and TCP with an FTP file transfer, as packet losses and re-ordering can have impact on the solution performance. We start

by distributing the traffic among the available paths using a round-robin schedule (later we will show results using other scheduling strategies).

4.4.1.1 UDP Protocol with CBR Traffic

For the results depicted in Figure 4.1(b), obtained in experiments using a low send rate, we can conclude that the use of two paths improves the delivery ratio significantly, and that the use of three paths also brings advantages, but the difference between two and three paths is minimal.

Figure 4.1(a) reports the delivery ratio results, using different number of paths with a higher send rate. In this case, the same pattern is observed when two paths are used, but the presence of a third path no longer offers improvements.

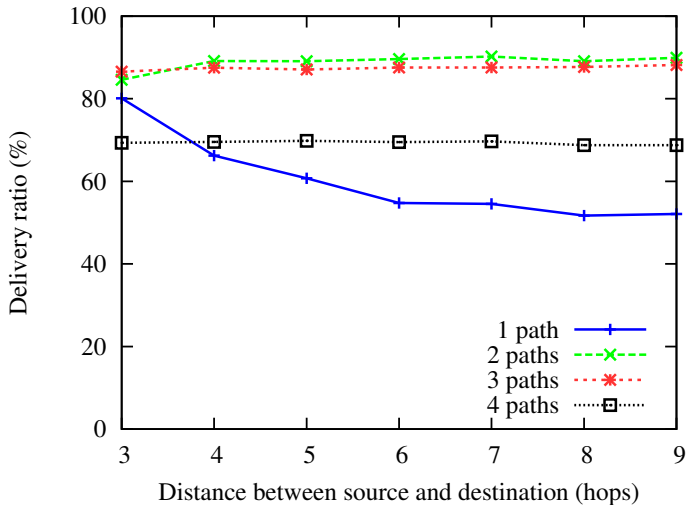
In summary, we can conclude that with only one path, as the path length increases, the delivery ratio decreases, and using multiple paths the delivery ratio achieved is similar for different path length. Although a third path can bring some advantages, this is not always the case, as results are different in the two experiments. When four paths are used the delivery ratio decreases notably.

4.4.1.2 TCP Protocol with FTP Traffic

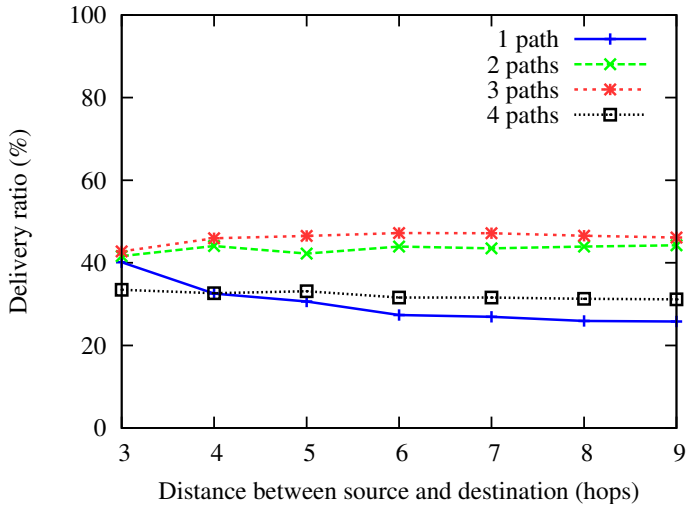
To evaluate the optimal number of paths using TCP as the transport protocol (the default TCP version of the simulator was used), we used the FTP protocol to transfer a 10Mbyte file; its transfer time is used to compute the average throughput.

In this scenario, a reordering layer needs to be placed between the routing and the TCP protocols, to reorder incoming out-of-order packets from different paths at the destination node. This layer stores out-of-order packets until an in-order packet arrives. After 6 interarrival periods (calculated based on the average interarrival time) without receiving a missing packet, the packets in the re-ordering queue are passed up to the TCP layer, activating the fast retransmit and recovery mechanisms.

Figure 4.2 presents the throughput of the TCP connection using this setting. As in the previous scenario, when the distance between the source and destination increases, the overall



(a) Lower rate (sending interval = 0.005)



(b) Higher rate (sending interval = 0.0025)

Figure 4.1: Determination of the optimal number of paths in a multipath solution

throughput decreases due to collisions at the MAC level and due to the mechanism that controls the access to the wireless medium. Another common aspect to both experiments is that the use of two paths improves the throughput, but a third path is no longer beneficial, as it increases the time needed to transfer the file. In Chapter 2, we have enumerated a number of problems derived from the operation of TCP that could limit the impact of multipath routing, namely the RTT estimation problem (the out of order problem is mitigated by the multipath module we have introduced). This problem can trigger timeouts prematurely, due to the difference in path delays, generating unnecessary retransmissions. This phenomenon may justify the difference between UDP and TCP results using three paths.

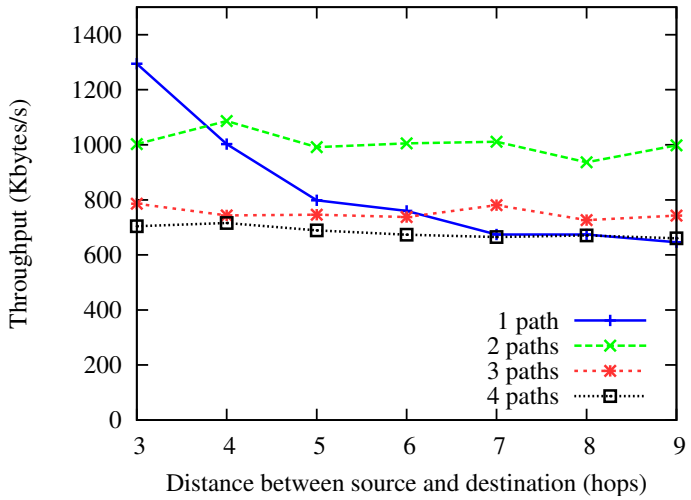


Figure 4.2: Throughput of a 10 Mbyte file transfer.

4.4.2 Concurrent Delivery Using Non-uniform Distribution of Traffic

Previous experiments have shown that the use of two paths can certainly improve the amount of data that can be transferred in a given time window. Furthermore, in UDP experiments, the use of three paths is not always beneficial, even if sometimes it provides improvements. Some of the limiting factors of multipath (contention or spatial reuse) can explain the results. However, the uniform distribution of traffic among paths may also be a cause for sub-optimal result. However, to make a distribution dependent on the path length (or of another metric of path quality) a different scheduling mechanism (other than round-robin) needs to be deployed. As this is not an easy task, we made some preliminary tests using the same scenario as in the previous tests, and tested different distributions for a path with a minimum of 6 hops in length.

Experimentally, we have determined the maximum send rate that can be imposed on each path while still ensuring a delivery ratio of 100%. Then, we distribute the traffic among the different paths proportionally to the maximum capacity of each path. In this case, when two paths are used, one with 6 hops and the other one with 10, the traffic load in each one is 52% and 48%, respectively. When the traffic is distributed for 3 paths the distribution is 43.3% (path with 6 hops), 40% (path with 10 hops) and 16.7% (path with 22 hops).

The results presented in Figure 4.3 depict the best result achieved in each case. We can conclude that a non-uniform distribution provides advantages in all scenarios. However, even with a non-uniform load distribution, having a third path does not improve the results signifi-

cantly; the best result using non-uniform distribution with three paths is only 2.73% above the best result achieved using two paths.

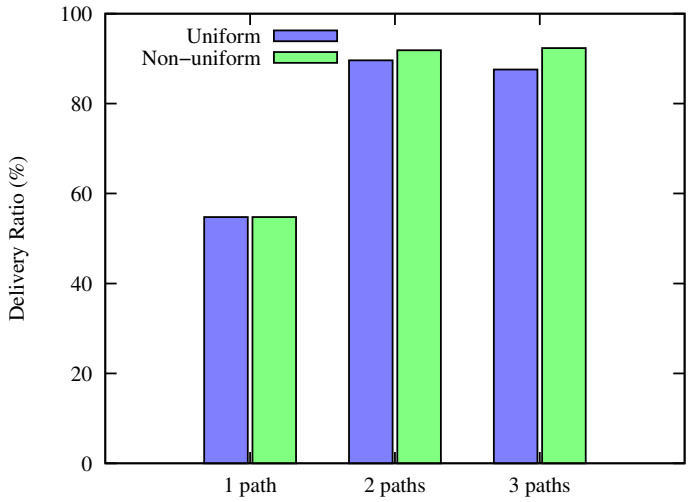


Figure 4.3: Delivery ratio using uniform and non-uniform distributions of traffic.

These results are consistent with the ones presented in (Liaw, Dadej, & Jayasuriya 2004). The results of that paper indicate that the limiting factor in multipath routing is the degree of spatial reuse in the source node.

In the analysis that follow we consider only the two best paths. This is based on the results presented above and also on the discussion presented in (Liaw, Dadej, & Jayasuriya 2004), that observes that in most networks the probability of finding a third non-interfering path of appropriate length is relatively low. The same paper also states that two paths usually maximize the bandwidth offered to the application (using a round robin distribution of traffic), which is also consistent with the results that we have presented this section.

4.5 Performance of Different Clustering Algorithms

Before evaluating the routing mechanism, we have made tests to assess how effective are the different clustering algorithms when used by the CBMPR technique to determine low coupled paths. For that purpose, we have compared ACE+ with Lowest ID (Lin & Gerla 1997) and ACE (Chan & Perrig 2004). The routing mechanism used is the one described in Chapter 3 and the evaluation criteria consists in relating the distribution of clusters with: i) the delivery ratio and; ii) the number of paths that each solution can discover under the same conditions.

4.5.1 Distribution of Nodes by Clusters

For a better understanding of the difference between the clustering solutions we are comparing, we now present, for a random scenario, the different distribution of clusters provided by each protocol. In each case, the nodes in black represent cluster-heads and the remaining nodes of a cluster are marked with a different color. The transmission range of the cluster-head is represented by circles, each circle being the maximum spatial coverage of a cluster.

As described in Chapter 2, ACE an LID are representative of different type of algorithms. On one hand, the main goal of the ACE algorithm is to provide a good distribution of clusters. As depicted in Figure 4.4, we can observe that there are cluster-heads that interfere with each other. In this case, if two different paths use two interfering cluster-heads, the achieved throughput will decrease, affecting the protocol performance. Moreover, we can see that the distribution of nodes in clusters is not optimal as, sometimes, when nodes know two cluster-heads, they become clustered to the most distant.

On the other hand, when considering the distribution provided by the LID algorithm (Figure 4.5), we can observe that the interference of cluster-heads is not a problem, but the distribution of nodes suffers from the same problem than ACE. This can lead to the exclusion of clusters adjacent to the nodes that already belong to the first path and, as a consequence, it can severely constrain the number of nodes that can belong to a second path, resulting in longer paths or less discovered paths, which is not desirable.

The ACE+ protocol mixes the good characteristics of the two previous algorithms by combining their two main properties. As a result, and as can be observed in Figure 4.6, within a circle corresponding to the transmission range of a node there is only a cluster-head and within this range the distribution of nodes is done according to its distance to the cluster-head. In the next sections we will discuss how this property affects the performance of the routing algorithm.

4.5.2 Delivery Ratio

The delivery ratio gives information about the quality of the routes used. The following set of experiments evaluate the delivery ratio a connection can provide by using multiple paths at the same time.

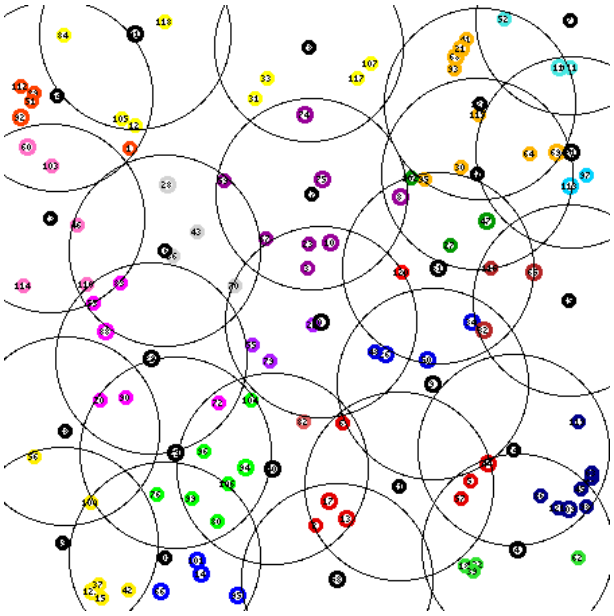


Figure 4.4: Distribution of clusters using ACE algorithm.

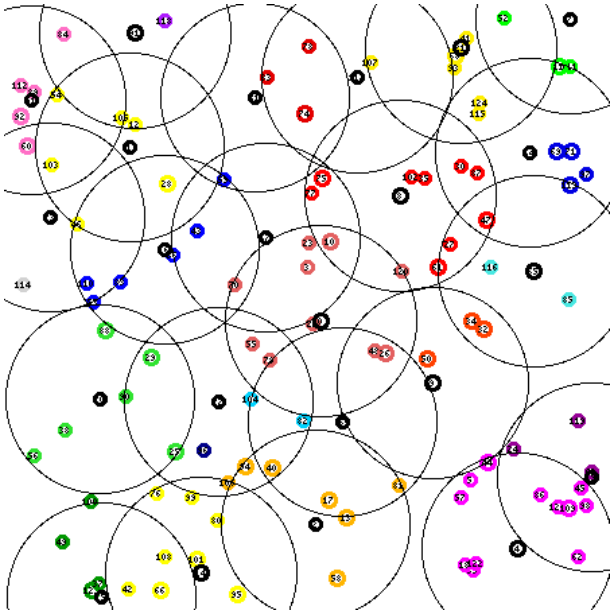


Figure 4.5: Distribution of clusters using LID algorithm.

Figure 4.7 reports delivery ratio results using different number of paths in random scenarios where 2 paths are always available and are discovered by all solutions. Comparing the analysed clustering schemes, lead us to the conclusion that if the cluster-heads do not interfere with each other, the chosen paths to route traffic are low coupled, as it was expected. This property explains the best results in terms of percentage of delivery achieved, namely if we compare the LID and ACE+ with ACE protocol, where cluster-heads can be neighbors and paths can

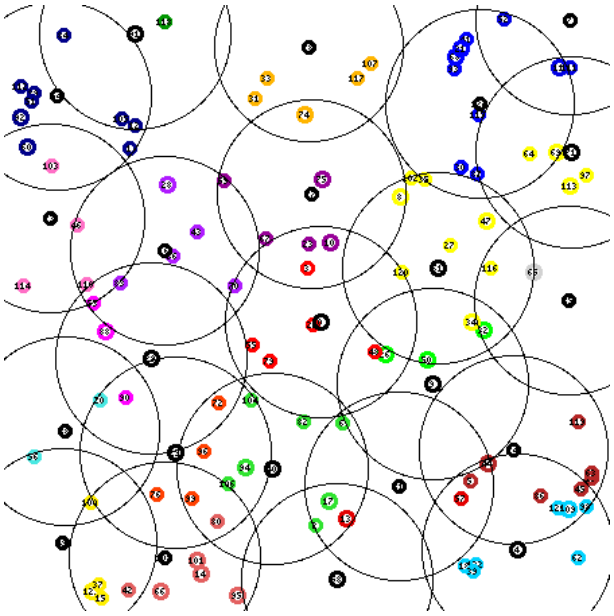


Figure 4.6: Distribution of clusters using ACE+ algorithm.

interfere with each other. The small differences between LID and ACE+ are justified by the different distribution of nodes in each case (given that paths can have different length from protocol to protocol).

4.5.3 Number of Paths Discovered

To understand the advantage of ACE+ over LID, we also present in Figure 4.8 the average number of paths discovered in the 3 solutions for the same experiments. While ACE+ can discover the highest number of paths, LID and ACE discover a smaller number due to the poor cluster distribution. The routing algorithm excludes the clusters that are 1 hop away of nodes belonging to a path, preventing them from being used in subsequent paths. Therefore, a better distribution of clusters reduces the probability of excluding clusters that are not part of a path.

Each algorithm has its own strengths and weaknesses, depending on the network conditions and requirements. As our goal is to improve the available bandwidth between a given source-destination pair, the ACE+ algorithm provides better overall performance when this metric is considered. Based on the previous results, the protocol that is most suitable to apply to the CBMPR approach is the ACE+, as the bandwidth that can be achieved when using this algorithm is higher, and the average number of paths found is also superior when compared with the two other solutions.

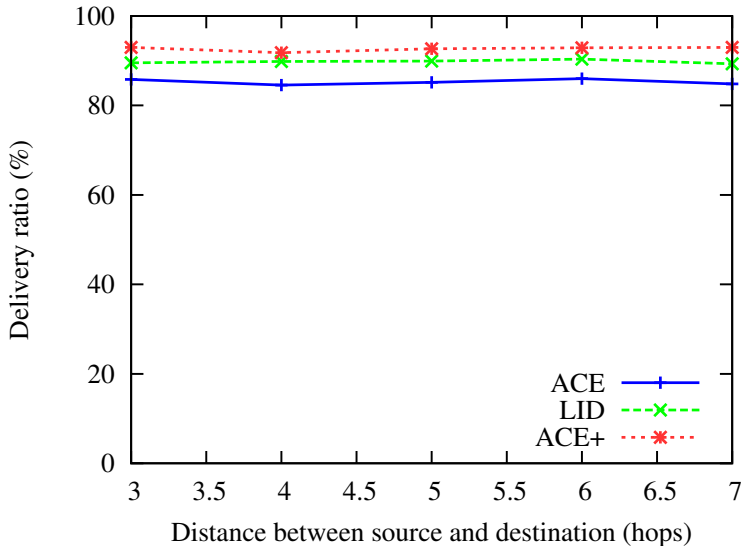


Figure 4.7: Delivery ratio using the different clustering protocols.

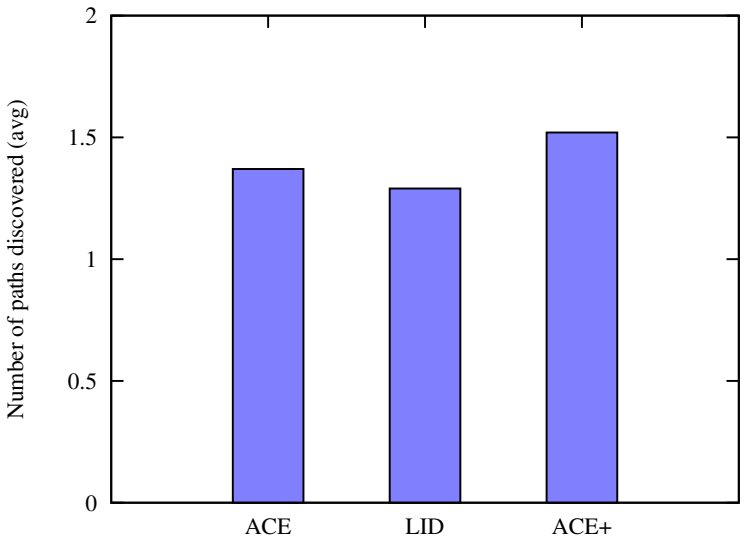


Figure 4.8: Number of paths using the different clustering protocols.

4.6 Efficiency of the *LoCoup* Protocol

In this section we present the results related with the efficiency of the *LoCoup* protocol when compared with OLSR+ and AODV-DM, namely the number of paths discovered by each protocol for the same scenario, the length of the discovered paths, the signaling cost, the delivery ratio achieved, and latency of the route discovery procedure.

4.6.1 Number of Paths Discovered

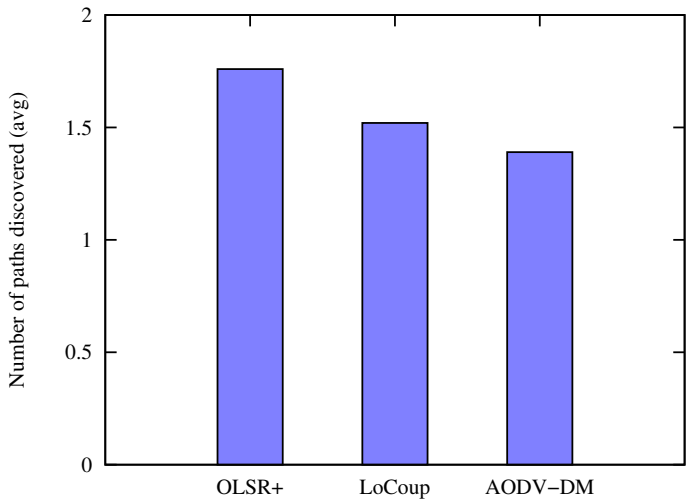


Figure 4.9: Average number of paths discovered by each protocol.

Figure 4.9 depicts the average number of paths discovered in the tests that we have made. As can be observed, the number of paths discovered is not the same in the three protocols. This number is higher for OLSR+ because this protocol calculates all the possible paths between the desired nodes: then it first chooses the shortest one and, afterwards, one of the shortest non-interfering paths. The other two protocols, *LoCoup* and AODV-DM, are not able to find the same number of paths, each for different reasons. *LoCoup* uses clusters to group nodes and chooses paths that pass through cluster-heads. This can limit the number of paths depending on the way nodes are grouped. Finally, the AODV-DM protocol has the lowest number of paths among the three protocols. The reason for this problem is that when a node deletes some routing entries from its RREQ table (to prevent future RREP enter in the insulating region) it does not notify its neighbors. Therefore, neighbors may take wrong decisions based on the outdated routing information, considering that a certain node offers the shortest path to the source node

when this is not true. As a result, RREPs can die in zones of the network with few nodes or where the existing ones have already been used (note that a RREP can never go back as when a node receives a message of this type it deletes the last hop from the RREQ table).

4.6.2 Number of Hops of the Discovered Paths

The hop count can be considered a measure of path quality, as the bandwidth and reliability of a path are inversely proportional to its length. Therefore, we are interested in showing that the routes discovered by *LoCoup* do not include paths significantly longer than OLSR+ or AODV-DM.

Table 4.2 presents the hop count of each discovered path, for different protocols in 5 different scenarios. There are no significant differences between the length of the two routes found by *LoCoup* and the other protocols. It is important to say that OLSR+ has information about the entire topology, which means it finds always the shortest path first (without restrictions) and then, the shortest path that does not interfere with this one. Besides, AODV-DM discovers the same paths because the shortest path is used to eliminate nodes that are neighbors of nodes in the shortest path. Therefore, the second path found has the minimum number of possible hops. Exceptions happen when intermediate nodes make decisions based on outdated information that can result in paths a bit longer (although the protocol considers a mechanism that in certain situations corrects the path if a shorter is found). For the *LoCoup* protocol, this is not always true as the restriction that the paths have to pass through cluster-heads can increase the length of the discovered paths (1 or 2 hops). Moreover, as *LoCoup* does not enforce the selection of the shortest path, in some cases, better combined paths can be selected (for example in scenario 4 paths discovered in OLSR+ and AODV-DM have 6 and 9 hops and paths determined by *LoCoup* have 7 hops each).

4.6.3 Signaling Traffic

To measure signaling costs, all the control traffic to discover routes is accounted for, by summing up the proactive and reactive components of each protocol. Note that, OLSR+ only has a proactive component, AODV-DM is only reactive, and *LoCoup* has both proactive and reactive components. Furthermore, the cost of the reactive component depends on the path length, given

	OLSR+		*LoCoup*		AODV-DM	
Path	1	2	1	2	1	2
Scenario 1	3	7	5	5	3	7
Scenario 2	4	5	5	5	4	6
Scenario 3	5	8	5	10	5	8
Scenario 4	6	9	7	7	6	9
Scenario 5	7	10	8	12	7	10

Table 4.2: Hop count of each discovered path in different protocols.

Fixed costs			Variable costs			
OLSR+	*LoCoup*	AODV-DM	*LoCoup*		AODV-DM	
			4	6	4	6
748600	61525	0	271	364	4555	6771

Table 4.3: Signaling costs (number of messages).

that part of the execution for some protocols only involves nodes that may potentially take part in a path.

Results presented in Table 4.3 have been obtained in a scenario where a route discovery was performed between nodes that are 4 and 6 hops away from the source node. For each route, 2 paths were discovered. Columns on the left present the fixed cost of the protocols and columns on the right the variable costs, depending on the length of the discovered paths. As can be seen, *LoCoup* has a much lower cost than OLSR+ and AODV-DM, in fixed and variable costs respectively.

A proactive protocol is only effective if the fixed costs may be amortized by various path discoveries. Figure 4.10 presents the total approximate cost of signaling of protocols, where an increasing number of routes are established. These route requests must be performed within the refreshing period of the proactive information, 5 seconds in this case. This permits to identify which is the network utilization pattern for which each protocol is most appropriate. Based on the results we can conclude that after the discover of 20 routes (or 4 routes per second), *LoCoup* is better than AODV-DM. Also, due to its high fixed cost, OLSR+ is only advantageous if the number of route discoveries exceed 1319 in the considered interval.

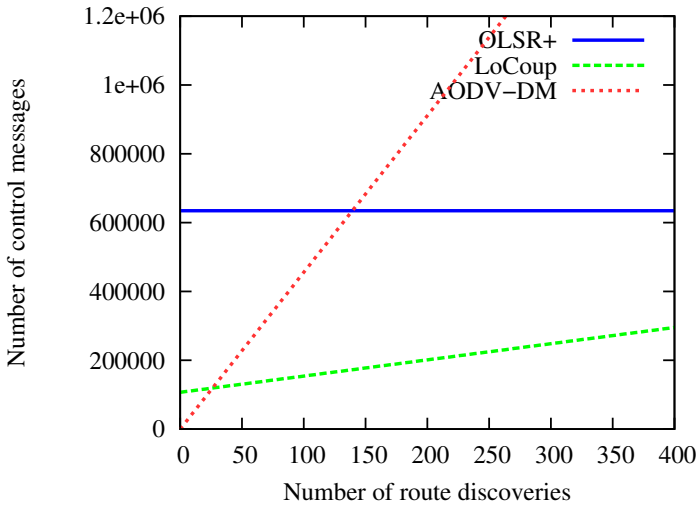


Figure 4.10: Signaling vs number of routes.

4.6.4 Delivery Ratio

To evaluate *LoCoup* according to the percentage of traffic that reaches the destination, we have run a number of experiences where the different protocols always discover two paths and have measured the resulting delivery ratio in each scenario. The values presented in Figure 4.11 depict the average of delivery ratio values: for each data point the average of 4 different experiences (at minimum) is considered.

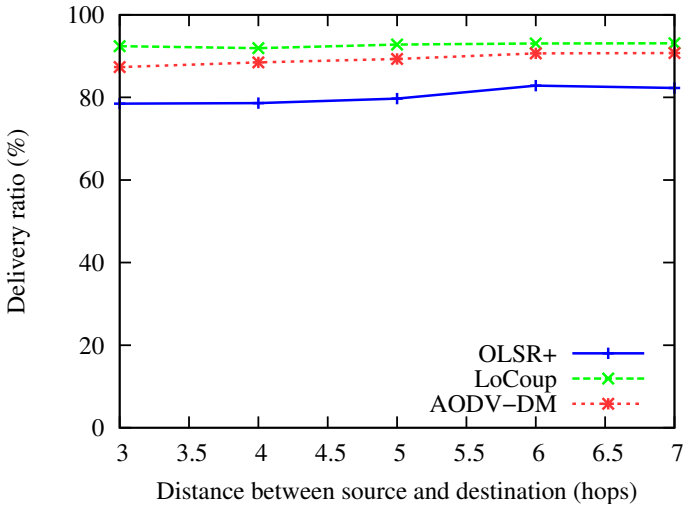


Figure 4.11: Delivery ratio of different protocols.

As we can observe, the three different protocols have similar delivery ratio which means the paths discovered in the three protocols are non-interfering paths. However, there is a little

difference between the delivery ratio achieved in different paths which can be justified with the background traffic of each protocol. In OLSR+ and *LoCoup*, there is background traffic being exchanged among nodes, by the proactive component of these protocols. As show in the previous section, the signaling traffic is higher in OLSR+ which justifies the lower throughput achieved (more bandwidth is used to forward control traffic). Although AODV-DM does not have a proactive component, it also exchanges messages periodically (when an active route exists) to detect route failures. This amount of traffic supersedes the control traffic in *LoCoup*, as the latter uses the overlay network defined by cluster-heads to reduce the amount of control traffic it generates in the network. This relation between the control traffic of the three protocols, justifies the small differences concerning this metric.

4.6.5 Latency in Route Discoveries

Finally, we have measured the time needed to discover two non-interfering paths, depending on the shortest distance (in number of hops) between the source and the destination. Obviously, the OLSR+ protocol, as it stores all the needed information to compute routes locally, only needs computation time which we consider negligible when compared with the time spent in the message exchanges required by the reactive components of the other protocols. Therefore, in Figure 4.12, we only present the results for *LoCoup* and AODV-DM. The advantage of *LoCoup* over AODV-DM is clear as *LoCoup* is more than 10 times faster to calculate routes than AODV-DM. This is due to the fact that number of messages associated with the reactive component of the *LoCoup* protocol is much smaller than that of AODV-DM. In *LoCoup*, RREQs are flooded using the structure defined by cluster-heads and only a RREP message is sent back, to respond to the request. Besides, AODV-DM floods RREQs using all nodes in the network. Also the number of messages needed to form the insulate region and guarantee that the secondary route does not include nodes in that region is larger.

4.7 Mobility

So far, we have always used static networks for our tests, since it is expected that WMNs are relatively stable in regard to mobility, although allowing for infrequent changes to the topology. Nevertheless, we also tested *LoCoup* in a mobile environment to investigate its effects in two

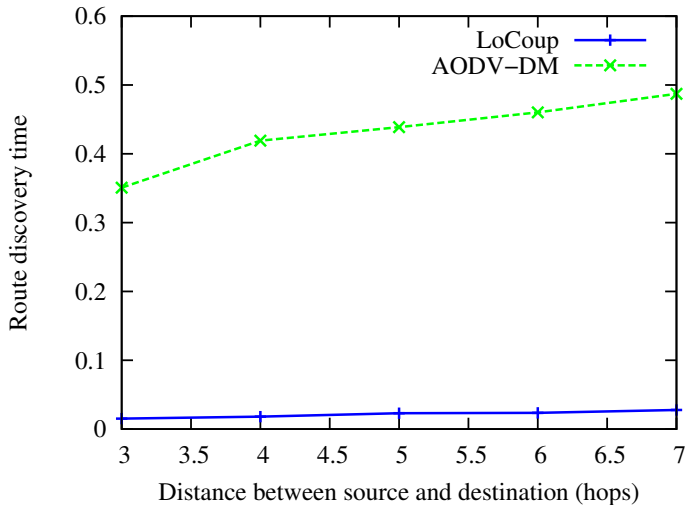


Figure 4.12: Time each protocol needs to discover a route with two paths.

key performance metrics: overhead and route recovery time. Mobility can produce changes in the topology, which in turn invalidates already established routes, which forces *LoCoup* to recover and produce additional control traffic, therefore we expect to observe increases both in the overhead and in route recovery time.

4.7.1 Overhead

We now measure the protocol overhead in the presence of mobility. The overhead presented here is due to the re-arrangement of clusters when several nodes move (or join, or leave the network) and it is necessary to group nodes to preserve the clustering structure. This update procedure involves to run clustering in order to update the state about interferences, as well as propagating the information to the neighboring nodes for them to update their tables. As depicted in Table 4.4, there is a considerable amount of information that needs to be exchanged to update the state of the network. Moreover, the variation of this value, from scenario to scenario, is significant. This number is dependent on the distribution of nodes along the network.

	Minimum	Maximum	Average
Overhead (messages)	10800	28359	16879.94

Table 4.4: Overhead in number of messages.

4.7.2 Route Recovery Time

The last performance metric we have analysed is the time a protocol takes to perform the recovery process under the presence of mobility. The results are shown in Figure 4.5. These numbers are correlated with the ones presented in the previous section, as more messages need to be exchanged to perform the route recovery, more time is needed to send that messages. Moreover, the resulting time is a function of the iteration length defined in the clustering protocol. As no concrete value is suggested in the original ACE paper, after doing some experiments to determine the best value, we set it to 0.9: this value allows the transmission of messages by nodes, minimizing collisions. As a consequence, the minimum route recovery time would be 2.7s. Finally, the time to propagate the clustering message that informs nodes that they have to rearrange and to update neighboring nodes with new clustering information also has to be considered.

	Minimum	Maximum	Average
Time (s)	3.22	9.53	4.85

Table 4.5: Route discovery time.

Although recovery from errors is not an efficient task in *LoCoup*, this procedure should occur infrequently in Wireless Mesh Networks. Improvements in this aspect are left for future work, as it would be desirable to perform the recovery locally.

Summary

In this chapter we have experimentally evaluated the operation of *LoCoup* using simulations. We have compared its performance against a proactive (OLSR+) and reactive (AODV-DM) protocols. *LoCoup* proved to offer an interesting tradeoff between the signaling cost, the time required to set up paths, and the properties of the discovered paths.

5 Conclusions

5.1 Conclusions

Nowadays, as the amount of traffic generated by applications increase, network mechanisms to improve the bandwidth available between two end nodes are required. Although some studies exist concerning this question in Wireless Mesh Networks there are few protocols targeting this problem.

In this thesis we have presented *LoCoup*, a novel protocol to find multiple paths between two nodes in Wireless Mesh Networks. The paths have low interference and are determined using an overlay of cluster-heads computed by a clustering algorithm adapted to our goals. The resulting algorithm combines proactive components (associated with the maintenance of the clustering overlay) and reactive components (associated with the selection of multiple paths between a given source and destination).

Experimental results show that the clustering algorithm that serves has a base to the protocol should have certain properties to guarantee that paths are low-coupled and the maximum number of paths are discovered for the protocol in each case. We have also concluded that no more than two paths are necessary to improve the capacity of the network, using concurrent transmissions with round robin scheduling. The proposed algorithm has low signaling cost, and can effectively find multiple routes with low-coupling properties.

5.2 Future Work

The current solution is targeted to networks with low mobility and where nodes crashes are infrequent. Therefore, the overlay defined by cluster-heads is assume to be stable and its recovery has not been optimized. As future work we plan on improving the stability of routes in face of failures and mobility of cluster-heads. Solutions updating only a subset of the network

could be used to reduce the time to recover from a failure.

It would also be interesting to explore our solution to improve the performance of WMN, even when multipath routing is not necessary to increase the available bandwidth (for example to reduce the interference between independent paths).

Finally, a mechanism to distribute traffic according the capacity of each different path could further improve the results obtained.

References

Akyildiz, I. & X. Wang (2005, Sept.). A survey on wireless mesh networks. *Communications Magazine, IEEE 43*(9), S23–S30.

Amir, Y., C. Danilov, M. Kaplan, R. Musaloiu-Elefteri, & N. Rivera (2008, June). On redundant multipath operating system support for wireless mesh networks. In *Sensor, Mesh and Ad Hoc Communications and Networks Workshops, 2008. SECON Workshops '08. 5th IEEE Annual Communications Society Conference on*, pp. 1–6.

Badis, H. & K. Al Agha (2005). QOLSR, QoS routing for ad hoc wireless networks using OLSR. *European Transactions on Telecommunications 16*(5), 427–442.

Baker, D. & A. Ephremides (1981). The architectural organization of a mobile radio network via a distributed algorithm. *Communications, IEEE Transactions on [legacy, pre - 1988] 29*(11), 1694–1701.

Basagni, S. (1999). Distributed clustering for ad hoc networks. In *ispan*, pp. 310. Published by the IEEE Computer Society.

Basagni, S., I. Chlamtac, & A. Farago (1997). A generalized clustering algorithm for peer-to-peer networks. In *Workshop on Algorithmic Aspects of Communication*.

Basu, P., N. Khan, & T. Little (2001). A mobility based metric for clustering in mobile ad hoc networks. *icdcsw*, 0413.

Campista, M. E. M., P. M. Esposito, I. M. Moraes, L. H. M. K. Costa, O. C. M. B. Duarte, D. G. Passos, C. V. N. De Albuquerque, D. C. M. Saade, M. G. Rubinstein, & M. G. Rubinstein (2008). Routing metrics and protocols for wireless mesh networks. *Network, IEEE 22*(1), 6–12.

Chan, H. & A. Perrig (2004). Ace: An emergent algorithm for highly uniform cluster formation. In *in Proceedings of the First European Workshop on Sensor Networks (EWSN*, pp. 154–171.

Chatterjee, M., S. Das, & D. Turgut (2002). WCA: A weighted clustering algorithm for mobile ad hoc networks. *Cluster Computing 5*(2), 193–204.

Chinara, S. & S. Rath (2009). A Survey on One-Hop Clustering Algorithms in Mobile Ad Hoc Networks. *Journal of Network and Systems Management 17*(1), 183–207.

Hu, X. & M. J. Lee (2007). An efficient multipath structure for concurrent data transport in wireless mesh networks. *Comput. Commun. 30*(17), 3358–3367.

Jacquet, P., P. Mühlethaler, T. Clausen, A. Laouiti, A. Qayyum, & L. Viennot (2001). Optimized link state routing protocol for ad hoc networks. In *Proceedings of the 5th IEEE Multi Topic Conference (INMIC 2001)*.

Jie Zhang, Choong Kyo Jeong, G. Y. L. H. J. K. (2008). Cluster-based multi-path routing algorithm for multi-hop wireless network. *International Journal of Future Generation Communication and Networking*.

Johnson, D. B. & D. A. Maltz (1996). Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, pp. 153–181. Kluwer Academic Publishers.

Koksal, C. & H. Balakrishnan (2006, Nov.). Quality-aware routing metrics for time-varying wireless mesh networks. *Selected Areas in Communications, IEEE Journal on 24*(11), 1984–1994.

Lee, S. J. & M. Gerla (2000). Aodv-br: backup routing in ad hoc networks. Volume 3, pp. 1311–1316 vol.3.

Lee, S.-J. & M. Gerla (2001). Split multipath routing with maximally disjoint paths in ad hoc networks. In *Communications, 2001. ICC 2001. IEEE International Conference on*, Volume 10, pp. 3201–3205 vol.10.

Leung, R., J. Liu, E. Poon, A.-L. Chan, & B. Li (2001). Mp-dsr: a qos-aware multi-path dynamic source routing protocol for wireless ad-hoc networks. In *Local Computer Networks, 2001. Proceedings. LCN 2001. 26th Annual IEEE Conference on*, pp. 132–141.

Liaw, Y., A. Dadej, & A. Jayasuriya (2004, jun.). Throughput performance of multiple independent paths in wireless multihop network. In *IEEE International Conference on Communications*, Volume 7, pp. 4157 – 4161.

Lim, H., K. Xu, & M. Gerla (2003, May). Tcp performance over multipath routing in mobile ad hoc networks. In *Communications, 2003. ICC '03. IEEE International Conference on*,

Volume 2, pp. 1064–1068 vol.2.

Lin, C. & M. Gerla (1997). Adaptive clustering for mobile wireless networks. *IEEE Journal on Selected areas in Communications 15*(7), 1265–1275.

Maimour, M. (2008). Maximally radio-disjoint multipath routing for wireless multimedia sensor networks. In *WMuNeP '08: Proceedings of the 4th ACM workshop on Wireless multimedia networking and performance modeling*, New York, NY, USA, pp. 26–31. ACM.

Marc Mosko, J. G.-L.-A. (2005). Multipath routing in wireless mesh networks. In *in Proc. IEEE Workshop on Wireless Mesh Networks (WiMesh)*.

Marina, M. & S. Das (2001, Nov.). On-demand multipath distance vector routing in ad hoc networks. In *Network Protocols, 2001. Ninth International Conference on*, pp. 14–23.

Moy, J. (1998, April). OSPF Version 2. RFC 2328 (Standard). Updated by RFC 5709.

Mueller, S., R. Tsang, & D. Ghosal (2004). Multipath routing in mobile ad hoc networks: Issues and challenges. In *In Performance Tools and Applications to Networked Systems, volume 2965 of LNCS*, pp. 209–234. Springer-Verlag.

Nagpal, R. & D. Coore (1998). An algorithm for group formation in an amorphous computer. In *Proc. 10th International Conference on Parallel and Distributed Computing Systems*.

Nandiraju, N. S., D. S. Nandiraju, & D. P. Agrawal (2006). Multipath routing in wireless mesh networks. In *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, pp. 741–746.

Nasipuri, A. & S. R. Das (1999). On-demand multipath routing for mobile ad hoc networks. In *Computer Communications and Networks, 1999. Proceedings. Eight International Conference on*, pp. 64–70.

Ong, L. & J. Yoakum (2002, May). An Introduction to the Stream Control Transmission Protocol (SCTP). RFC 3286 (Informational).

Parekh, A. (1994). Selecting routers in ad-hoc wireless networks. In *Proceedings SBT/IEEE Intl Telecommunications Symposium*, pp. 420–424.

Pearlman, M., Z. Haas, P. Sholander, & S. Tabrizi (2000a). On the impact of alternate path routing for load balancing in mobile ad hoc networks. In *Mobile and Ad Hoc Networking and Computing, 2000. MobiHOC. 2000 First Annual Workshop on*, pp. 3–10.

Pearlman, M. R., Z. J. Haas, P. Sholander, & S. S. Tabrizi (2000b). On the impact of alternate path routing for load balancing in mobile ad hoc networks. pp. 3–10.

Perkins, C. E. & E. M. Royer (1999). Ad-hoc on-demand distance vector routing. In *IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90–100.

Rangarajan, S. (2007). On demand loop free multipath routing in ad hoc networks using source sequence numbers.

Roy, S., D. Saha, S. Bandyopadhyay, S. B, S. Tanaka, & T. Ueda (2003). Improving end-to-end delay through load balancing with multipath routing in ad hoc wireless networks using directional antenna. In *in Proc. IWDC 2003: 5th International Workshop, LNCS v2918*, pp. 225–234.

Siddiqui, M. S., S. O. Amin, J. H. Kim, & C. S. Hong (2007, Oct.). Mhrp: A secure multi-path hybrid routing protocol for wireless mesh network. In *Military Communications Conference, 2007. MILCOM 2007. IEEE*, pp. 1–7.

Takai, M., J. Martin, & R. Bagrodia (2001). Effects of wireless physical layer modeling in mobile ad hoc networks. In *MobiHoc '01: Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, New York, NY, USA, pp. 87–94. ACM.

Tsai, J. & T. Moors (2006). A review of multipath routing protocols: From wireless ad hoc to mesh networks.

Tsirigos, A. & Z. Haas (2001, Nov). Multipath routing in the presence of frequent topological changes. *Communications Magazine, IEEE 39*(11), 132–138.

Valera, A., W. K. G. Seah, S. Rao, & S. Rao (2002). Champ: A highly-resilient and energy-efficient routing protocol for mobile ad hoc networks. *IEEE MWCN*, 79–85.

Waharte, S. & R. Boutaba (2006, jun.). Totally disjoint multipath routing in multihop wireless networks. Volume 12, pp. 5576 –5581.

Wu, K. & J. Harms (2001). Performance study of a multipath routing method for wireless mobile ad hoc networks. *Modeling, Analysis, and Simulation of Computer Systems, International Symposium on 0*, 0099.

Xu, K. & M. Gerla (2002). A heterogeneous routing protocol based on a new stable clustering scheme. In *MILCOM*, Volume 2, pp. 838–843. Citeseer.

Xuekang, S., G. Wanyi, X. Xingquan, X. Baocheng, & G. Zhigang (2009). Node discovery algorithm based multipath olsr routing protocol. *Information Engineering, International Conference on 2*, 139–142.

Yang, Y., J. Wang, & R. Kravets (2005). Designing routing metrics for mesh networks. *Proceedings of the IEEE Workshop on Wireless Mesh Networks (WiMesh). IEEE Press*.

Ye, Z., S. Krishnamurthy, & S. Tripathi (2004, Nov.-3 Dec.). Effects of multipath routing on tcp performance in ad hoc networks. In *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, Volume 6, pp. 4125–4131 Vol.6.

Yi, J., A. Adnane, S. David, & B. Parrein (2010). Multipath Optimized Link State Routing for Mobile ad hoc Networks. *Ad Hoc Networks*.