Multipath Routing for Wireless Mesh Networks

(extended abstract of the MSc dissertation)

Cristina Neves Fonseca
Departamento de Engenharia Informática
Instituto Superior Técnico

Advisor: Professor Luís Rodrigues

***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.**

I. Introduction

Wireless Mesh Networks (WMN) are undergoing rapid progress and inspiring numerous deployments. As a result, WMNs are expected to deliver wireless services in a large variety of scenarios of different scale, including personal, local, campus, and metropolitan areas[1], [2].

As in any other network, when WMN are used in practice some routes are likely to become more heavily used than others. However, and in contrary to wired networks, in wireless networks, due to the limited spectrum, it is hard or even impossible to overdimension the links. Therefore, techniques that allow to take full benefit of all available routes become of paramount importance in this setting. Multipath routing is a promising approach to achieve such goals. By establishing multiple paths between a source and a destination, one can balance the load among multiple routes and also increase the bandwidth available to the applications.

There are a number of challenges in setting multiple paths in a WMN. One of these challenges is that multiple paths should not interfere at the radio level. Otherwise, the utilization of one path may disrupt the operation of a different path, which can easily reduce, instead of increasing, the total amount of bandwith provided to the applications. In fact, in a wireless medium, interferences among different channels are a limiting factor of network capacity.

We propose a new multipath routing protocol that considers multiple paths that do not interfere with each other, minimizing interferences between transmissions of neighbouring nodes. In order to simplify the discovery, maintenance, and deployment of such paths we rely on a clustering algorithm that elects cluster-heads in such a manner that transmissions from different cluster-heads do not interfere with each other.

We provide an extensive evaluation using the widely used NS-2 network simulator, and show that our solution offers an interesting tradeoff between the signaling cost required to setup and maintain paths and the resulting throughput.

The rest of the paper is organized as follows. In Section II we describe related work. Section III details the proposed protocol and Section IV presents its performance evaluation. Finally, Section V presents the conclusions and outlines the directions for future work.

II. Related Work

Most routing protocols that have been proposed for mesh and ad hoc networks consider the case where a single path is established between a source and a destination node. The main goal of multipath routing is to allow the use of several paths to reach destinations, not just the best path.

However, in order to be useful, multiple paths should not interfere with each other. In wireless transmissions it is known that the quality of packet transmission may be severely degraded if simultaneous transmissions occur in neighboring links; this is known as the *route coupling* problem [3]. It is trivial to observe that two routes that have nodes or links in common (other than the source and destination nodes) are coupled, as simultaneous transmissions are bound to interfere. Unfortunately, route coupling may occur even if paths have no nodes or links in common. In order to minimize interferences, low coupled paths (or zone-disjoint paths) must be used.

A. Reactive Multipath Routing

In the literature, several routing protocols that use multipath routing have been proposed. Typically, they are derived from pre-existing (single-path) routing protocols, that are augmented to collect and use more than one path between the source and the destination. These multiple paths may be used for increasing the bandwidth, load balancing or, more commonly, to have an available backup path to replace a primary path in case the primary path breaks due to failures or node movement. OLSR [4], AODV-BR AOMDV [5], MMESH [1] and MHRP [6] were designed to provide backup routing; MP-DSR [7] determines a set of paths that can satisfy a certain end-to-end reliability requirement and uses multiple paths at the same time; SMR [8] provides load balancing; finally there are multipath

routing protocols concerning error resilience [9] by using alternative paths to send data using *error-correcting code* (ECC) techniques. None of the protocols above adequately addresses the problem of avoiding route-coupling. Note that, in some scenarios route coupling is not an issue: if multipath is used only for backup, the primary and backup paths are never used simultaneously, and radio interference is not a concern.

To the best of our knowledge, AODV-DM [10] is the only multipath routing protocol that avoids interferences between multiple paths. The protocol operation includes a mechanism that is able to select zone-disjoint paths that work as follows. The protocol defines an insulate region around the primary path, to avoid interference between adjacent routes. The primary path is discovered in the following way: a RREQ is flooded to the entire network. 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 neighbour 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, neighbours 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 secondary 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 neighbours in order to remove themselves from their neighbours' tables. 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 rejection packet (RREJ) so the sRREP sender can try other entries in its table. The disadvantages of AODV-DM are the high-latency of the route discovery and the relatively low number of non-interfering paths that are found.

B. Proactive Multipath Routing

Concerning proactive multipath routing, there are some link-state protocols based in OLSR. MOLSR[11] computes multiple paths but uses only the best one at each moment. In QOLSR[12] 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[13] 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: collection of topology information and route computation. To get topology information of the network, nodes use topology sensing and topology discovery. Topology sensing includes link sensing and neighbour 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[13] an on-demand scheme is used to avoid the heavy computation of multiple routes to every destination. Altought 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.

C. Cluster-based Multipath Routing

More recently, the idea of using clustering algorithms to support multipath routing has been suggested in [14]. In that paper, a clustering algorithm is used to group nodes into clusters that may be used to find multiple low-coupled paths. However, cluster-based multipath routing (CBMPR)[14] does not discuss any specific clustering algorithm nor provides complete protocols to establish and maintain routes based on the existing clusters.

Clustering algorithms have been used to improve network performance parameters like routing delay, bandwidth consumption, energy consumption, throughput and to allow applications to scale easily [15]. A clustering algorithm splits the network into disjoint sets of nodes each centering around a chosen cluster-head. Efficient clustering protocols rely on different design goals, depending on the application they are designed to. For example LCA [16] or RCC [17] are not scalable as their convergence time is $O(n)$. The Lowest ID [18] algorithm guarantees that no cluster-head interferes with another, the ACE protocol [19] produces a highly uniform cluster formation, while others try to optimize the way cluster-heads are chosen, according to connectivity, node mobility, transmission power, among others [20], [21], [22].

Our work extends the work of [14] in multiple dimensions. First we show that the choice of the clustering algorithm is of paramount importance for the approach to work in practice, and propose a clustering algorithm suitable for this task. In particular, our clustering algorithm explicitly considers the following constraints: i) All nodes nodes in a cluster are in the interference range of the cluster-head and a node is at most k hops away of any other node in the same cluster; ii) Clusters are non-overlapping, which means that each node should be in the communication range of the minimal number of cluster-heads; iii) The clustering algorithm needs to be decentralized, have small overhead, and be fast and efficient. Also, we propose concrete algorithms to discover, deploy, and maintain multiple paths.

III. LOW-COUPLING CLUSTER-BASED MULTIPATH ROUTING

This section introduces Low-Coupling Cluster-Based Multipath Routing Protocol (*LoCoup*). 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 Section, one possible approach to achieve such goal is to use Cluster-based Multipath Routing (CBMPR) [14]. The idea behind this technique assumes that nodes in the network have to be grouped in clusters, each cluster having a cluster-head. 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.

A. Building Blocks

The *LoCoup* operation can be divided in two main functional steps. The first one is the clustering (which groups nodes) and 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.

1) Clustering as a Low-Coupling Technique: 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: i) the group the node belong to and its neighbouring groups (as well as the nodes that give access to that groups); ii) knowledge about the entire network represented by an overlay defined by groups.

In *LoCoup* paths are chosen according to clustering distribution of nodes. Our work extends the work of [14] in multiple dimensions. In particular, our clustering algorithm explicitly considers the following constraints: i) all nodes 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; ii) Clusters are non-overlapping, which means each node should be in the communication range of the minimal number of cluster-heads and cluster-heads do not interfere with each other.

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

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.

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 disjoint paths to route traffic.

The selection of paths is made by the source node that 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 having as base the information retrieved from the clustering operation

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.

B. 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, namely: i) every node has a unique ID and knows the ID of its 1-hop neighbours; ii) a message sent by a node is received correctly within a finite time by all its 1-hop neighbours. iii) the topology of the network does not change during the algorithm execution. We would like to design a protocol that own the following set of properties:

- 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 [19] algorithm, which provides a good cluster distribution but does not ensure that cluster-heads do not interfere. We have named our protocol ACE+.

The process of clustering can be described as a combination of two stages, cluster formation and cluster maintenance described below.

1) Cluster Formation: In ACE+, the cluster formation consists of two logical phases. 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. The

action each node takes when an iteration arrives 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 [19] 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 neighbours of both l and w and the ones who are in the interference range of w and are not neighbours 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 [19]. After the last iteration, all nodes execute the termination procedure described bellow, after which each node sends a DONE message to its neighbours with its final state and the ID of the cluster they belong to.

- 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 neighbours that are clustered to B stop following it, B sends an ABDICATE message.
- 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.

The final DONE message generated by cluster-heads is propagated in a three hop range, so that, at the end of the algorithm, all nodes know not only the cluster-heads they link to but the nodes that link them to that clusters.

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.

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.

C. 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.

1) Route Discovery: As we have previously noted, 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 neighbouring 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 responsible for 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 neighbouring 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 allways keep a routing table. Instead, it only computes the multiple paths when data packets need to be sent out.

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.

Path Validation: Although low-coupling paths share the source and destination clusters, they should not share other 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 list of interfering clusters and gateways updated with information of the new path.

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 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.

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) 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 DESTINATION-MOVED 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.

IV. EVALUATION

A. Introduction

In this section we present 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 the it represents the solution in the literature that 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 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*.

B. 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 [10] we implemented as well. Besides, a modified version of OLSR [4] (that we have named OLSR+) to discover multiple non-interfering paths between a given source and destination was also developed. As OLSR+ can build the network graph, the discovered paths are optimal in terms of hop count (considering that the shortest path is allways used) and the number of paths is allways the maximum for the network considered.

C. 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: length of discovered paths, signaling cost needed to discover such routes, quality of the discovered paths, and latency in the process of route discovery.

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

D. Optimal Number of Paths in a Multipath Solution

It is known that the increase of the bandwidth in a multipath solution is not directly proportional to the number of paths used [23]. 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 to observe the variation of the desired metric. These tests were performed using both UDP and TCP, as packet losses and re-ordering can have impact on the solution performance.

Parameter	Value
Data rate	11 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

Table I
PARAMETERS COMMON TO ALL SIMULATIONS.

1) UDP Protocol with CBR Traffic: Figure 1 reports delivery ratio results using different number of paths. In this case, using two paths improves the delivery ratio significantly but the presence of a third path does not provide improvements. When four paths are used the delivery ratio decreases notably. 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.

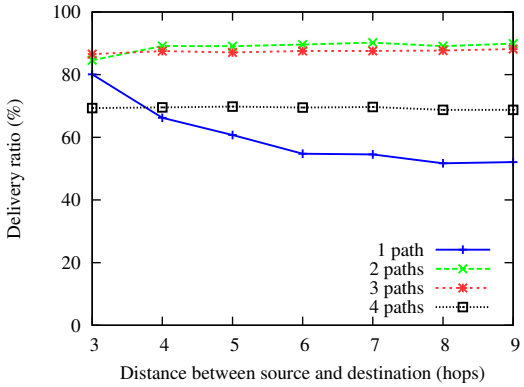

Figure 1. Determination of the optimal number of paths in a multipath solution

2) TCP Protocol with FTP Traffic: To evaluate the optimal number of paths using TCP as the transport protocol, we used FTP protocol to transfer a 10Mbyte file. The transfer time is taken to compute the average throughput. Additionally, in this scenario a reordering layer has been inserted between routing and TCP layers, to reorder incoming out-of-order packets from different paths at the destination node.

Figure 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 throughput decreases due to collisions at the MAC level and the mechanism that controls the access to the wireless medium. Another common aspect between the two experiments is that the use of two paths improves significantly the throughput, but a third path to route traffic between two nodes is not necessary as it increases the time

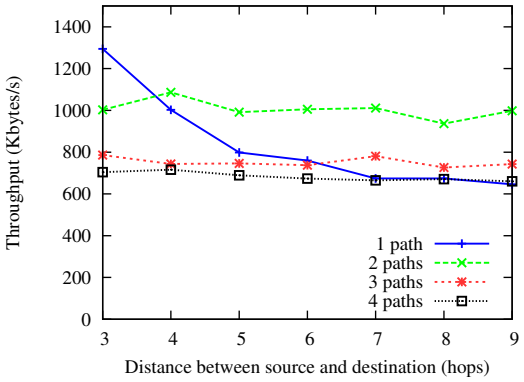

Figure 2. Throughput of a 10 Mbyte file transfer.

needed to transfer the file. This can be due to the RTT estimation problem [24].

E. Performance of Different Clustering Algorithms

Before evaluating the routing mechanism, we have made tests to assess how effective are different clustering algorithms when used by the CBMPR technique to determine low coupled paths. For that purpose, we have compared ACE+ with Lowest ID [18] and ACE [19]. The routing mechanism used is the one described in the previous section and the evaluation relates the distribution of clusters with the delivery ratio and number of paths that each solution can discover under the same conditions.

1) Distribution of Nodes by Clusters: As described in the previous section, ACE an LID algorithms represent different type of algorithms. In one hand, the ACE algorithm has the main goal to provide a good distribution of clusters and cluster-heads can interfere with each other. In this case, if two different paths linking the same pair of nodes are going to pass in 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, 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.

For a better understanding of the desired clustering distribution, we present for a random scenario, the result using ACE+, which combines the two main properties of the previous protocols (Figure 3). Nodes in black represent cluster-heads and the other 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

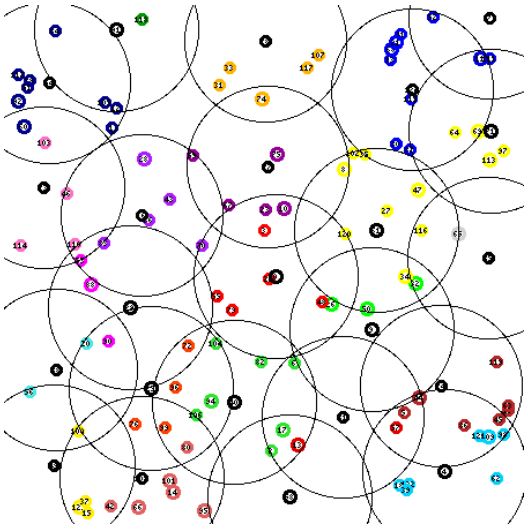

Figure 3. Distribution of clusters using ACE+ algorithm.

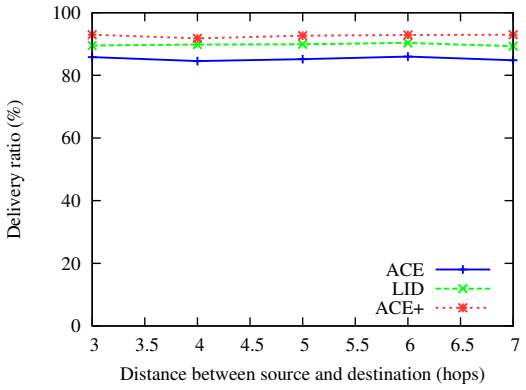

Figure 4. Delivery ratio using the different clustering protocols.

spatial coverage of a cluster.

2) Delivery Ratio: Figure 4 reports delivery ratio results using different number of paths in random scenarios where 2 paths are allways available and are discovered by all solutions. By comparing the analysed clustering schemes, we can conclude that best results are achieved when cluster-heads do not interfere with each other (such as in LID and ACE+). The small differences between LID and ACE+ are justified by the different distribution of nodes in each case, so paths can have different length from protocol to protocol.

3) Number of Paths Discovered: To understand the advantage of ACE+ over LID we also present in Table II the average number of paths discovered in the 3 solutions for the same experiments. While ACE+ discovers the highest number of paths, LID (and ACE) discover fewer paths due to the poor cluster distribution. The routing algorithm takes the clusters that are 1 hop away of nodes belonging to a path and excludes them, preventing those clusters 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.

	ACE	LID	ACE+
Nr. of paths (average)	1.37	1.29	1.52

Table II
NUMBER OF PATHS USING DIFFERENT CLUSTERING PROTOCOLS.

Based on the previous results, the protocol that is most suitable to apply to the CBMPR approach in the ACE+ as the bandwidth that can be available using the refereed algorithm is higher and the average number of paths found is also superior when compared with other solutions.

F. Efficiency of the Proposed Solution

In this section we present the results related with the efficiency of the protocol. In this analysis we consider only the two best paths reasoning that, as discussed in [23], in most of the networks the probability of finding a third path that does not interfere with the previous ones and having good length is relatively low. The refered study also states that with this number of paths the bandwidth offered to the application layer has more advantages (using a round robin distribution of traffic), which is a result consistent with what has been said previously.

1) Number of Hops of the Discovered Paths: Quality of the discovered paths is accounted using its hop count, as the bandwidth and reliability of a path are inversely proportional to its length.

Table III 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 allways 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 neighbours of nodes along this one. Exceptions happen when intermediate nodes make decisions based on outdated information that can result in paths a bit longer.For the *LoCoup* protocol, this is not always true as the restriction that the paths have to pass through cluster-heads can increase (1 or 2 hops) the length of the discovered paths. Moreover, as *LoCoup* does not guarantees the selection of the shortest

	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 III
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 IV
SIGNALING COSTS (NUMBER OF MESSAGES).

path, in some cases best paths can be selected, as in scenario 4.

2) 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. OLSR+ only has proactive component, AODV-DM is only reactive and *LoCoup* is both proactive and reactive. Furthermore, the cost of the reactive component depends on the path length as part of the execution in some protocols only involves nodes that will potentially take part in a path.

Results presented in Table IV consider 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. 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 5 presents the total approximate cost of signaling of protocols, where a crescent number of routes are established. This route requests must be performed between the refreshing period of the proactive information, 5 seconds in this case. This permits to identify which is the pattern of utilization of the network 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.

3) Delivery Ratio: From Figure 6 we can conclude that the three different protocols have similar delivery ratio.

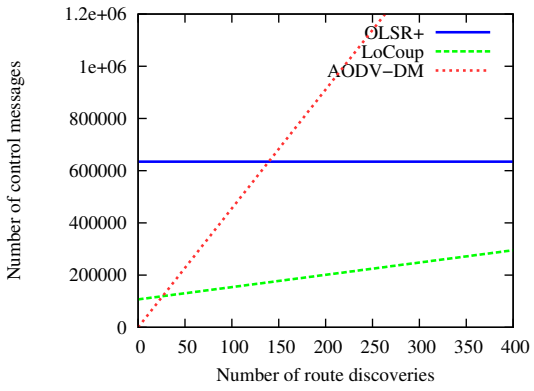

Figure 5. Signaling vs number of routes.

However, there is a little difference between the delivery ratio achieved in different paths which can be justifyed with the background traffic of each protocol. In OLSR+ and *LoCoup* there are background traffic being exchanged between nodes, due to the process of route discovery. As show in the previous section the signaling traffic is higher in OLSR+ which justifies the lower throughput achieved (more bandwith is used to forward control traffic). Although AODV-DM does not have a proactive component it exchanges messages periodically (when an active route exists) to detect route faillures. This amount of traffic supers the control traffic in *LoCoup*, which justify the differences observed.

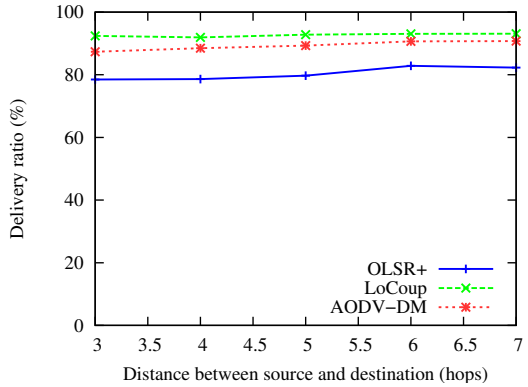

Figure 6. Delivery ratio of different protocols.

4) Latency in Route Discoveries: Finally, we have measured the time needed to discover two non-interfering paths, depending on the shortest distance between the source and the destination. In Figure 7 we only present the results for *LoCoup* and AODV-DM as OLSR+ compute routes locally. The advantage of *LoCoup* over AODV-DM is clear as *LoCoup* is more than 10 times faster to calculate routes than AODV-DM. This results are because the number os messages associated with the reactive component of the *LoCoup* protocol is smaller than the ones AODV-DM needs to exchange to discover a route. 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 and the number of messages needed to form the insulate region and guarantee that the secondary route does not include nodes in that region is also bigger.

G. Mobility

So far we always used static networks in the tests described, since it's expected that WMNs are stable in regard to mobility. Nevertheless, we also tested *LoCoup* in a mobile environment to investigate its effects in two key performance metrics: overhead and route recovery time.

1) Overhead: The overhead accounted here is due to the re-arrangement of clusters when several nodes move (or join, or leave the network). This procedure involves to run

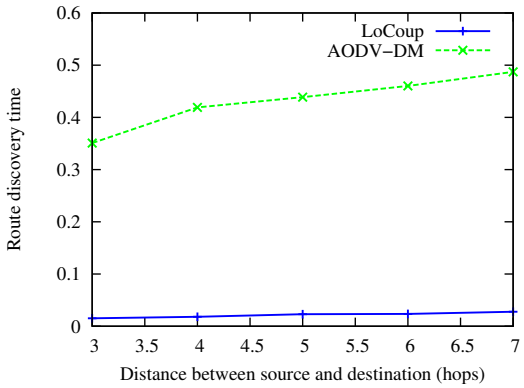

Figure 7. Time each protocol needs to discover a route with two paths.

clustering in order to update the state about interferences. As presented in Table V 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 V
OVERHEAD IN NUMBER OF MESSAGES.

2) Route Recovery Time: The results shown in Figure VI are related 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, this time is depends on the iteration length of the clustering protocol. As no number is suggested in the original protocol (ACE) and after doing some experiments we set it to 0.9, value that minimizes collisions. As a consequence, the minimum route recovery time would be 2.7s. Besides, time to propagate the clustering message that informs nodes that they have to rearrange and to update neighbouring nodes with new clustering information has to be considered.

	Minimum	Maximum	Average
Time (s)	3.22	9.53	4.85

Table VI
ROUTE DISCOVERY TIME.

V. CONCLUSIONS

In this work 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 established 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 algorithm has low signaling cost, and can effectively find multiple routes with low-coupling properties.

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.

ACKNOWLEDGMENTS

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, in particular, with José Mocito. I also want to thank Martijn Kuipers for his very usefull insights on the configuration of the NS-2 propagation model.

REFERENCES

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

[2] I. Akyildiz and X. Wang, "A survey on wireless mesh networks," *Communications Magazine, IEEE*, vol. 43, no. 9, pp. S23–S30, Sept. 2005.

[3] M. Pearlman, Z. Haas, P. Sholander, and S. Tabrizi, "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*, 2000, pp. 3–10.

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

[5] M. Marina and S. Das, "On-demand multipath distance vector routing in ad hoc networks," in *Network Protocols, 2001. Ninth International Conference on*, Nov. 2001, pp. 14–23.

[6] M. S. Siddiqui, S. O. Amin, J. H. Kim, and C. S. Hong, "Mhrp: A secure multi-path hybrid routing protocol for wireless mesh network," in *Military Communications Conference, 2007. MILCOM 2007. IEEE*, Oct. 2007, pp. 1–7.

[7] R. Leung, J. Liu, E. Poon, A.-L. Chan, and B. Li, "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*, 2001, pp. 132–141.

[8] S.-J. Lee and M. Gerla, "Split multipath routing with maximally disjoint paths in ad hoc networks," in *Communications, 2001. ICC 2001. IEEE International Conference on*, vol. 10, 2001, pp. 3201–3205 vol.10.

[9] A. Tsirigos and Z. Haas, "Multipath routing in the presence of frequent topological changes," *Communications Magazine, IEEE*, vol. 39, no. 11, pp. 132–138, Nov 2001.

[10] X. Hu and M. J. Lee, "An efficient multipath structure for concurrent data transport in wireless mesh networks," *Comput. Commun.*, vol. 30, no. 17, pp. 3358–3367, 2007.

[11] S. Xuekang, G. Wanyi, X. Xingquan, X. Baocheng, and G. Zhigang, "Node discovery algorithm based multipath olsr routing protocol," *Information Engineering, International Conference on*, vol. 2, pp. 139–142, 2009.

[12] H. Badis and K. Al Agha, "QOLSR, QoS routing for ad hoc wireless networks using OLSR," *European Transactions on Telecommunications*, vol. 16, no. 5, pp. 427–442, 2005.

[13] J. Yi, A. Adnane, S. David, and B. Parrein, "Multipath Optimized Link State Routing for Mobile ad hoc Networks," *Ad Hoc Networks*, 2010.

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

[15] S. Chinara and S. Rath, "A Survey on One-Hop Clustering Algorithms in Mobile Ad Hoc Networks," *Journal of Network and Systems Management*, vol. 17, no. 1, pp. 183–207, 2009.

[16] D. Baker and A. Ephremides, "The architectural organization of a mobile radio network via a distributed algorithm," *Communications, IEEE Transactions on [legacy, pre - 1988]*, vol. 29, no. 11, pp. 1694–1701, 1981.

[17] K. Xu and M. Gerla, "A heterogeneous routing protocol based on a new stable clustering scheme," in *MILCOM*, vol. 2. Citeseer, 2002, pp. 838–843.

[18] C. Lin and M. Gerla, "Adaptive clustering for mobile wireless networks," *IEEE Journal on Selected areas in Communications*, vol. 15, no. 7, pp. 1265–1275, 1997.

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

[20] A. Parekh, "Selecting routers in ad-hoc wireless networks," in *Proceedings SBT/IEEE Intl Telecommunications Symposium*, 1994, pp. 420–424.

[21] P. Basu, N. Khan, and T. Little, "A mobility based metric for clustering in mobile ad hoc networks," *icdcsw*, p. 0413, 2001.

[22] S. Basagni, "Distributed clustering for ad hoc networks," in *ispan*. Published by the IEEE Computer Society, 1999, p. 310.

[23] Y. Liaw, A. Dadej, and A. Jayasuriya, "Throughput performance of multiple independent paths in wireless multihop network," in *IEEE International Conference on Communications*, vol. 7, jun. 2004, pp. 4157 – 4161.

[24] H. Lim, K. Xu, and M. Gerla, "Tcp performance over multipath routing in mobile ad hoc networks," in *Communications, 2003. ICC '03. IEEE International Conference on*, vol. 2, May 2003, pp. 1064–1068 vol.2.