A New MPLS-Based Forwarding Paradigm for Multi-Radio Wireless Mesh Networks

Stefano Avallone and Giovanni Di Stasi

***Abstract*—Routing in multi-radio wireless mesh networks is a very challenging problem. In this paper, we propose a forwarding paradigm based on MPLS (Multi Protocol Label Switching) which makes use of a novel mechanism, denoted as MPLS splitting policy. Such mechanism allows to configure multiple next hops at an intermediate node, so that the incoming traffic is partitioned among the next hops according to predefined coefficients named split ratios. The MPLS splitting policy has been designed to allow for load balancing and fast local restoration. With such a mechanism, it is crucial to properly determine the set of split ratios, as they determine how the traffic is routed across the network. We present an approach to compute a set of split ratios that guarantee high performance under different traffic loads. To this end, we adopt the *hose* traffic model, according to which we only have knowledge of the maximum amount of traffic entering or leaving the network at each edge node. A thorough simulation study is conducted to show that our approach outperforms other routing protocols in terms of throughput and robustness against traffic load variations and single node failures.**

***Index Terms*—Multi-radio wireless mesh networks, MPLS, routing.**

I. Introduction

WIRELESS Mesh Networks (WMNs) have recently attracted a big interest thanks to their ability to wirelessly cover large areas with a low cost of deployment and maintenance. However, since wireless transmissions are involved, interference is a major concern. In order to alleviate the interference, mesh routers are being equipped with multiple radios which allow simultaneous transmissions on orthogonal channels. The availability of multiple radios per node leads to the channel assignment problem, i.e., the problem how to select a channel for each radio in the network. Channel assignment is a challenging problem and has been massively investigated in the recent years [1][2][3][4]. Routing in multi-radio wireless mesh networks is a very challenging problem as well, due to a number of reasons:

- due to interference, nodes cannot dispose of the full capacity of their links, because the channel capacity must be shared among all the interfering links. Thus, it does not suffice to ensure that each link is not allocated more flow than its capacity to guarantee a feasible routing. Instead, a routing protocol should be aware of which links interfere with each other and route flows in such a way that the amount of flow allocated on each set of interfering links does not exceed the channel capacity;
- wireless transmissions are affected by fading, propagation losses, environment noise, etc. Such phenomena may cause frequent (and temporary) unavailability of links. Therefore, a routing protocol should quickly react to such failures by providing an alternate path to the destination;
- given the decrease in the available bandwidth caused by interference, it is necessary to fully exploit the network resources. To this end, it is advisable that a routing protocol support the ability to route flows over multiple paths between the same ingress-egress pair, in order to balance the traffic load across the whole network;
- the traffic load offered to the network may vary dynamically. A routing protocol should not be tailored for a particular traffic matrix, but it should ensure a high performance despite variations in the traffic load.

To our knowledge, the routing protocols proposed so far for multi-radio wireless mesh networks fail to address all the issues mentioned above (see Section II). In this paper, we present a novel forwarding paradigm for multi-radio WMNs based on Multi-Protocol Label Switching (MPLS) [5] with the purpose to address all the aforementioned issues. The first contribution of this paper is the definition of an MPLS *splitting* policy, a new, standard-compliant, MPLS mechanism that enables each intermediate node to split the incoming traffic belonging to a specific Forwarding Equivalence Class (FEC) among a predefined set of neighbors according to predefined *split ratios*. As a result, different packets of a given FEC follow distinct paths between the ingress and egress nodes. The proposed mechanism has the potential for addressing the second and the third of the above issues. Indeed, the traffic between an ingress-egress pair can be balanced across multiple paths (third issue). Also, the availability of multiple next hops for a given destination enables a fast local restoration in case of single node/link failures (second issue). The proposed splitting policy is presented in Section III.

Enforcing the MPLS splitting policy requires to: (i) identify a suitable set of paths for each ingress-egress pair and (ii) compute the set of split ratios. We address the former task by proposing RDAS (Resilient Directed Acyclic Graph), an algorithm that finds, for a given ingress-egress pair, a set of paths that guarantee protection against single node/link failures (Section V-C). To accomplish the latter task, we present an approach to compute, given the current channel assignment, a set of split ratios that ensure high performance despite variations in the traffic load (Section V-B). To this end, rather than sticking to a given traffic matrix, we adopt the *hose*

Manuscript received October 4, 2012; revised March 4, 2013; accepted June 18, 2013. The associate editor coordinating the review of this paper and approving it for publication was A. Abouzeid.

S. Avallone and G. Di Stasi are with the Department of Computer Engineering, University of Naples, 80125 Italy (e-mail: {stefano.avallone, giovanni.distasi}@unina.it).

Digital Object Identifier 10.1109/TWC.2013.071113.121529

traffic model [6], according to which we only have knowledge of the maximum amount of traffic entering and leaving the network at each edge node, but we do not have knowledge of the actual traffic matrix. Based on such a model, we formulate a convex optimization problem whose objective is to optimize the *average* performance over all the *possible* traffic matrices (thus addressing the fourth of the above issues). The performance of a given solution is measured in terms of its ability to route flows such that the amount of flow allocated on each set of interfering links does not exceed the channel capacity (thus addressing the first of the above issues).

The proposed convex optimization problem requires the knowledge of the network topology and of the maximum amount of traffic entering and leaving the network at each edge node (hose model). Given that mesh nodes are typically stationary, changes in the network topology that require a re-computation of the split ratios only occur due to the addition/removal of mesh nodes (since temporary node/link failures are handled by the splitting mechanism). Also, the maximum amount of traffic entering (leaving) the network at an edge node can be set to a value close to the sum of the transmission rates of the radio interfaces used for receiving (sending) the incoming (outgoing) traffic [6], hence such values are rather stable as well. Thus, the computed set of split ratios can be held for a long time. Hence, though the proposed approach is centralized, it entails a low communication overhead because both the retrieval of topology information and the transmission of a new set of split ratios to the network nodes do not need to be performed frequently.

The rest of the paper is structured as follows. In Section II we give an overview of the related work. The MPLS splitting policy is presented in Section III. In Section IV we formalize the problem to find a proper set of split ratios for the use with the MPLS splitting policy, while in Section V we present our approach to solve such a problem. In Section VI we present the results of the simulation study we conducted to show that our approach achieves high throughput and is robust against variations in the traffic load and against single node failures. Finally, Section VII concludes the paper.

II. Related Work

Most of the work related to routing in wireless mesh networks focused on link or path metrics proposed as improvements upon the hop count metric. Among the first link metrics to be introduced, the expected transmission count (ETX) [7] estimates the number of transmissions required to successfully send a packet to a neighbor. The authors in [8] introduce two metrics, the expected transmission time (ETT) and the weighted cumulative ETT (WCETT). MIC (metric of interference and channel switching) [9] takes the inter-flow interference into account in addition to the intra-flow interference. A link metric based on the estimated available bandwidth is instead proposed in [10]. The above mentioned link metrics (and many others) are intended to be used with a single path destination-based routing protocol. Very often, the routing protocol used to test the proposed routing metric is one of those designed for ad hoc networks, like AODV [11] or OLSR [12]. The routing protocol specified in the IEEE 802.11s amendment [13], too, is basically a modified version of AODV that uses the Airtime link metric to associate each link with an estimate of the amount of time needed to successfully transmit a packet across that link. These routing protocols, being single path, have limited capabilities in terms of load balancing and require some time to discover alternative routes in case of link/node failures. A number of proposals extend the above mentioned single-path routing protocols to use multiple paths between a source and a destination, such as AODV-BR [14], AOMDV [15] and AODV-DM [16].Such proposals define some measures of interference, but do not consider the available bandwidth resulting from the channel assignment. Also, in case of failures, repairing a path requires the exchange of routing messages and hence some time is needed to have consistent routing tables.

An adaptive load-aware routing scheme is proposed in [17]. The network is divided into multiple clusters and each cluster head estimates the traffic load in its cluster. If the estimated load gets higher, the cluster head increases the routing metrics of the routes passing through the cluster so that the traffic avoids overloaded clusters. This scheme requires a continuous adaptation of the link costs to the offered traffic load, which might lead to instabilities, and does not account for link/node failures. A number of approaches exploits the broadcast nature of the wireless medium. ExOR [18] is an opportunistic approach where a node broadcast a packet and the nodes that received it correctly agree on which of them has to further forward the packet, based on the distance to the destination. However, the protocol used to reach such agreement introduces some overhead. ROMER [19] builds a "forwarding mesh" around the minimum cost path and each packet is allowed to travel along one of the paths in the forwarding mesh based on the current conditions. GATOR [20] is another opportunistic approach which exploits the knowledge of the geographic coordinates of the nodes while selecting the receiver in charge of retransmitting a packet. A drawback of the opportunistic approaches, however, is that they are less effective in multi-radio WMNs because only the neighbors listening on the channel used by the sender can receive the packet.

The anypath routing paradigm [21] generalizes the opportunistic approach. Each node is pre-configured with a set of next-hops, each having a different priority. A packet is further forwarded only by the next-hop with the highest priority that correctly received the packet. An anypath is composed by all the possible paths that a packet can take from the source to the destination. In [21] the goal is to find the least cost anypath, while in [22] additive constraints to be satisfied are considered. While sharing some concepts with anypath routing, our proposal is deeply different. Firstly, anypath routing requires a modified MAC to determine which next-hop has to forward the packet. Secondly, the degree of load balancing achieved depends on the outcome of the packet transmissions: if all the transmissions were successful, the anypath routing would reduce to single-path routing. Thirdly, like other opportunistic approaches, anypath routing is less effective in multi-radio networks, where neighbors are reachable via different channels.

Finally, we mention our previous approach known as Layer-2.5 forwarding paradigm (L2.5) [4]. In L2.5, forwarding decisions are not taken by looking up the routing table, but

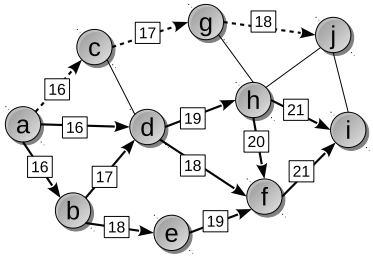


Fig. 1. Example to illustrate the splitting mechanism.

are based on two objectives: i) balance the traffic among the outgoing links in proportion to their available bandwidth; ii) guarantee that all the packets reach the destination in a predetermined maximum number of hops. L2.5 suffers from a loose control over the paths taken by packets and fails to ensure that they are cycle-free. Also, the performance of L2.5 is dependent on the available bandwidth values associated with the network links, and hence it may degrade if they are not suited to the actual traffic load.

III. The MPLS Splitting Policy

According to RFC 3031 [5], MPLS nodes use three tables to forward packets: NHLFE (Next Hop Label Forwarding Entry), ILM (Incoming Label Map) and FTN (FEC-to-NHLFE Map). An entry of the NHLFE specifies the next hop, the operation to perform on the packet's label stack and (optionally) any additional information needed in order to properly dispose of the packet. RFC 3031 provides the following three operations: pop the label at the top of the stack, push a new label onto the stack, swap the label at the top of the stack and possibly push other labels. The ILM is used when forwarding packets that arrive as labeled packets. An entry of the ILM maps an incoming label to a *set* of NHLFE entries. The FTN, instead, is used when forwarding packets that arrive unlabeled. An entry of the FTN maps a FEC (Forwarding Equivalence Class) to a *set* of NHLFE entries. Normally, each FEC in a FTN entry is associated with a *single* NHLFE entry and each label in an ILM entry is associated with a *single* NHLFE entry. In such a way, there is a unique next hop for a given FEC or label at each node and thus all the packets of a FEC follow the same path (e.g., the dashed path from a to j in fig. 1).

As noted above, RFC 3031 explicitly mentions the possibility that a label or a FEC may be associated with a set of NHLFE entries, in order to perform, e.g., some sort of load balancing. We exploit such a possibility to allow the packets of a FEC to follow a predefined set of paths (as opposed to a single path) between the ingress and egress nodes. Such an approach is illustrated by fig. 1, where a continuous arrow departing from a node denotes a possible next hop (as specified in an NHLFE entry) for the packets that entered the network at a and are destined to i. It can be observed that nodes have multiple possible next hops from among they select the one which a given packet is forwarded to. Consequently, different packets of the same FEC may follow distinct paths (e.g., a-d-h-i or a-b-d-f-i). All the possible paths taken by the packets of a given FEC are determined a priori and can be enforced by properly configuring the MPLS tables on the nodes. For instance, the behavior of node b is achieved by configuring an entry in its ILM that associates the incoming label 16 with two entries in the NHLFE: one that replaces label 16 with label 17 and sends the packet to node d and the other one that replaces label 16 with label 18 and sends the packet to node e. When an incoming packet with label 16 arrives, node b has to select either of the two NHLFE entries.

MPLS SPLITTING($\{\rho_i\}$, $\{\overline{\rho}_i\}$, B, p)
1 **for** each NHLFE i in decreasing order of $\rho_i - \overline{\rho}_i$
2 **if** the next hop in NHLFE i is reachable
3 select NHLFE i
4 $\overline{\rho}_i = \dfrac{\overline{\rho}_i \cdot B + p}{B + p}$
5 $\overline{\rho}_j = \dfrac{\overline{\rho}_j \cdot B}{B + p} \quad \forall j \neq i$
6 $B = B + p$
7 **if** $B > B_{max}$
8 $B = B_{min}$

Fig. 2. Pseudo-code of the MPLS splitting policy.

In case a FEC or a label is associated with multiple NHLFE entries, the procedures to choose an NHLFE entry among the given set are beyond the scope of RFC 3031. Here, we define a policy to select one of multiple NHLFE entries that fits our goal to balance the traffic among the outgoing links according to predefined proportions, while ensuring a fast reaction to node/link failures. We assume that each NHLFE entry also specifies, as an additional information, a *split ratio*, which is a value between 0 and 1. The split ratios associated with a set of NHLFE entries that correspond to the same FEC or to the same label must sum to 1. The goal of the splitting policy is to balance the traffic matching a given FEC or a given label among the neighbors specified by the corresponding NHLFE entries in proportion to the specified split ratios. For this purpose, the algorithm shown in fig. 2 is used to select an NHLFE entry (and hence a next hop) from among the set of NHLFE entries associated with a given FEC or with a given label. The procedure shown in fig. 2 is given the set of the split ratios $\{\rho_i\}$ associated with the set of NHLFE entries, the set of the actual utilizations $\{\overline{\rho}_i\}$ of each NHLFE entry (i.e., the ratio of the amount of traffic transmitted as specified by an NHLFE entry to the total traffic matching the FEC or the label), a counter B that records the amount of traffic matching the given FEC or label, and the size p of the packet for which an NHLFE entry must be selected. Before the traffic starts flowing, all the actual utilizations and the counter B are set to zero. Then, every time a packet matches a given FEC or label, the associated NHLFE entries are sorted and visited in decreasing order of the gap between the split ratio and the actual utilization. If the next hop neighbor included in the i-th NHLFE entry is marked as unreachable, then the NHLFE entry is skipped. To this end, we assume that a feedback is provided by the lower layers informing on the unavailability of a neighbor. Otherwise, the packet is sent as specified by the i-th NHLFE entry and the actual utilization of all the NHLFE entries and the total amount of traffic B are updated (lines 4–6). To avoid that B grows indefinitely, it is reset to a value B_{min} once it exceeds a given threshold B_{max}. B_{min} should be

a value greater than zero to avoid that all the actual utilizations are reset after receiving the next packet (from line 5, $\overline{\rho}_j$ would be null if B were zero). Also, a node that is marked as unreachable can be included among the active neighbors again after a configurable amount of time.

Thus, the proposed MPLS splitting policy enables to balance the traffic matching a given FEC or label among the neighbors specified in the associated NHLFE entries in proportion to the corresponding split ratios. Also, by having multiple NHLFE entries (and hence next hop neighbors) already configured, our splitting policy enables a fast restoration against single node/link failures, as another NHLFE entry can be readily used to send the packet. We note here that it also makes sense to have NHLFE entries with an associated null split ratio. Such entries are not used to forward packets in normal conditions, but they are only used in case all the other entries with non-null split ratios have been disabled due to the corresponding next hops being unreachable. Thus, NHLFE entries with a null split ratio (that may be present in a solution returned by our approach proposed in Section V-B) can be usefully configured since they serve as backup routes in case of failures.

IV. Problem Statement

Interfering links and collision domains: We assume that each mesh router is equipped with multiple radio interfaces, each of which is assigned one of the $|\mathcal{C}|$ available channels and transmits at a fixed transmission power. We also assume the availability of a set of transmission rates. Given that a radio may serve multiple links and the ability of commodity hardware to set the transmission rate on a per-packet basis, we will assign a rate to links rather than radios.

We model the WMN as a directed graph $G = (V, E)$, where V is a set of nodes each representing a mesh router. Given two nodes $u, v \in V$, a directed edge $u \rightarrow v$ belongs to E iff u and v share at least a common channel and, in the absence of transmissions on other links, there exists a rate r such that a transmission from u to v is successful. We assume that a transmission from u to v is successful if the Signal-to-Interference and Noise Ratio (SINR) at the receiver is sufficiently high to decode the signal. The SINR at receiver v when a signal is transmitted by u is defined as $SINR_{uv} = \frac{G_{uv}P(u\rightarrow v)}{\sum_{x\rightarrow y\neq u\rightarrow v} G_{xv}P(x\rightarrow y)+n_v}$, where $P(u \rightarrow v)$ is the power emitted by u to transmit to v, G_{uv} is the gain of the radio channel between u and v, and n_v is the thermal noise at receiver v. If u transmits at rate r, the receiver v can correctly decode the signal if $SINR_{uv} \geqslant \gamma_r$, where γ_r denotes the minimum SINR required to correctly decode a signal modulated at the rate r. Thus, a link $u \rightarrow v \in E$ iff there exists a rate r such that $\frac{G_{uv}P(u\rightarrow v)}{n_v} \geqslant \gamma_r$. The highest rate r for which such inequality holds is selected as the capacity of the link and denoted by $c(u \rightarrow v)$. In case u and v share multiple channels, the set E may include as many links between the two nodes as the number of common channels. To differentiate among those links and stress that a link has been assigned channel c, we use the notation $u \xrightarrow{c} v$.

A link $x \xrightarrow{c} y \in E$ interferes with $u \xrightarrow{c} v \in E$ if a transmission on $x \xrightarrow{c} y$ prevents a simultaneous transmission on $u \xrightarrow{c} v$. We assume that happens when i) the two links share the same transmitter or receiver, ii) the transmitter of a link is the receiver of the other (since a single radio cannot transmit and receive simultaneously), or iii) a transmission on $x \xrightarrow{c} y$ makes the SINR at v too low to correctly decode the signal from u. We define the set of all the links that interfere with $u \xrightarrow{c} v$ as its collision domain and denote it by $\mathcal{D}(u \xrightarrow{c} v) = \left\{x \xrightarrow{c} y \in E \mid \{x, y\} \cap \{u, v\} \neq \emptyset \vee \frac{G_{uv}P(u\rightarrow v)}{G_{xv}P(x\rightarrow y)+n_v} < \gamma_{c(u\xrightarrow{c}v)}\right\}$. In other words, none of the links in $\mathcal{D}(u \xrightarrow{c} v)$ can be active at the same time as $u \xrightarrow{c} v$. Finally, we define the *total utilization* of the collision domain of link e as $U_{tot}(e) = \sum_{e_0\in\mathcal{D}(e)} \frac{f(e_0)}{c(e_0)}$, where $f(e_0)$ denotes the amount of flow routed on link e_0.

The MPLS splitting-based routing problem: Without loss of generality, we consider a set $V_e = \{n_1, ...n_N\} \subseteq V$ of N edge nodes acting as both ingress and egress nodes. We denote by $\{I_s^{max}\}_{s\in V_e}$ and $\{O_d^{max}\}_{d\in V_e}$, respectively, the sets of the maximum amount of incoming and outgoing traffic at each edge node. According to the hose traffic model, we only have knowledge of these values and we know neither the actual amount of traffic entering at each edge node nor what portion of traffic entering at a given edge node is destined to each of the other N-1 edge nodes. A set of incoming flows $\mathcal{I} = \{I_s\}_{s\in V_e}$ is said to be *feasible* if $I_s \leqslant I_s^{max} \quad \forall s \in V_e$.

Our goal is to route the (unknown) traffic matrix, using MPLS and the splitting policy, in such a way to minimize the cost function defined in Section V-A. A routing *solution* consists of a set of split ratios $\{\rho_{u\rightarrow v}^{s,d}\}_{u\rightarrow v\in\mathcal{E}^{s,d}}^{s\in V_e, d\in V_e-\{s\}}$, where $\rho_{u\rightarrow v}^{s,d}$ represents the ratio of the flow between the ingress-egress pair (s, d) entering node u that is forwarded to node v and $\mathcal{E}^{s,d}$ represents the set of links along which the flow between the ingress s and the egress d is routed. Clearly, the equation $\sum_{v \mid u\rightarrow v\in\mathcal{E}^{s,d}} \rho_{u\rightarrow v}^{s,d} = 1$ must hold for each u and for each ingress-egress pair (s, d). The set of split ratios determine how the (unknown) traffic matrix is routed across the network. Specifically, they determine, for each ingress-egress pair (s, d), a directed subgraph of G, $\mathcal{S}^{s,d} = (\mathcal{V}^{s,d}, \mathcal{E}^{s,d})$, where $\mathcal{V}^{s,d}$ is the set of nodes belonging to the links in $\mathcal{E}^{s,d}$. Given how the splitting policy works, it turns out that the packets flowing from ingress node s to egress node d can follow any of the paths between s and d in the subgraph $\mathcal{S}^{s,d}$. A routing solution is said to be *admissible* if, for every ingress-egress pair (s, d), the set of links $\mathcal{E}^{s,d}$, or, equivalently, the directed subgraph $\mathcal{S}^{s,d}$, meets the following constraints:

- t') to avoid that packets take excessively long paths, the length of every path in $\mathcal{S}^{s,d}$ must be at most α times the length of the shortest path between s and d in G
- t'') every path in $\mathcal{S}^{s,d}$ must be cycle-free
- t''') the set of paths in $\mathcal{S}^{s,d}$ must guarantee protection against single node/link failures, i.e., if a single node/link fails, the upstream node must have an alternative path to the egress node d

We denote by $f_{u\rightarrow v}^{s,d}$ the variable representing the amount of flow between the ingress-egress pair (s, d) that is routed on link $u \rightarrow v$. The total amount of flow routed on a link $u \rightarrow v$ is $f_{u\rightarrow v} = \sum_{s\in V_e} \sum_{d\in V_e-\{s\}} f_{u\rightarrow v}^{s,d}$. We denote by $\varphi_{u\rightarrow v}^{s,d} = \frac{f_{u\rightarrow v}^{s,d}}{I_s}$ the variable representing the amount of flow on link $u \rightarrow v$

contributed by node s and destined to node d, *normalized* to the actual (unknown) amount of traffic I_s entering source node s. We observe that $\varphi_{u\to v}^{s,d}$ is independent of the actual amount of traffic entering source node s and only depends on how traffic flows are routed. As shown in Section V-D, the set $\Phi = \{\varphi_{u\to v}^{s,d}\}_{u\to v\in\mathcal{E}^{s,d}}^{s\in V_e, d\in V_e-\{s\}}$ suffices to determine the set of split ratios, and hence it can be considered as representative of a particular routing solution.

We denote by $\Gamma(\Phi,\mathcal{I})$ the cost of a particular configuration where a routing solution Φ is used to route a feasible set of incoming flows $\mathcal{I} = \{I_s\}_{s\in V_e}$ (see Section V-A for its definition). The *average* cost of a routing solution Φ, i.e., the average of $\Gamma(\Phi,\mathcal{I})$ over all the feasible sets $\mathcal{I}$, is denoted by:

$$\Gamma(\Phi) = \frac{1}{\prod_{s\in V_e} I_s^{max}} \int_0^{I_{n_1}^{max}} \cdots \int_0^{I_{n_N}^{max}} \Gamma(\Phi,\mathcal{I}) dI_{n_1} \cdots dI_{n_N} \quad (1)$$

Given the maximum amount of traffic entering or leaving the network at each edge node, the MPLS splitting-based routing problem is to find a *feasible* admissible routing solution Φ that minimizes $\Gamma(\Phi)$. A routing solution is said to be feasible, given $\{I_s^{max}\}_{s\in V_e}$ and $\{O_d^{max}\}_{d\in V_e}$, if it obeys the following constraints:

- f') the amount of flow routed on each link must not exceed the link capacity, for *every* feasible set of incoming flows
- f'') the amount of flow routed towards each egress node must not exceed the maximum amount of outgoing traffic of that egress node, for *every* feasible set of incoming flows

The feasible admissible routing solution that minimizes $\Gamma(\Phi)$ has the minimum cost on the average and hence it can be considered as the most robust routing solution against variations of the traffic matrix.

V. Solving the MPLS Splitting-Based Routing Problem

In this section, we show how we solve the MPLS splitting-based routing problem defined in the previous section:

- we first define the cost $\Gamma(\Phi,\mathcal{I})$ of routing a given set $\mathcal{I}$ of actual incoming flows according to a particular routing solution Φ. Then, we compute the average cost $\Gamma(\Phi)$ of a routing solution Φ over all the feasible sets of incoming flows (Section V-A)
- then, we address the problem to find a feasible admissible routing solution minimizing $\Gamma(\Phi)$. Requiring that the returned routing solution be admissible makes the problem to find a feasible routing solution minimizing $\Gamma(\Phi)$ hard to solve. Hence, our approach is to decouple the problem to find a set of directed subgraphs that make a routing solution admissible from the problem to find a feasible routing solution minimizing $\Gamma(\Phi)$ subject to the constraint that the flow between each pair of ingress and egress nodes can only be routed along the links of predefined subgraphs. We present a convex optimization problem to find an optimal solution to the latter problem (Section V-B) and propose a heuristic to solve the former problem (Section V-C)
- finally, we show how the set of split ratios can be derived from the values of the $\varphi_l^{s,d}$ variables (Section V-D)

The main notations and symbols used in this section are summarized in Table I.

A. Computing the average cost of a routing solution

In this section, we define the cost $\Gamma(\Phi,\mathcal{I})$ of a particular configuration where a given routing solution $\Phi = \{\varphi_{u\to v}^{s,d}\}_{u\to v\in\mathcal{E}^{s,d}}^{s\in V_e, d\in V_e-\{s\}}$ is used to route a given feasible set of incoming flows $\mathcal{I} = \{I_s\}_{s\in V_e}$. The proposed cost function penalizes configurations where an excessive amount of flow is routed on the links of a collision domain (which need to share the available channel capacity). To this end, we exploit the results of a theoretical analysis [23] that shows that a set of flows routed on the links of a collision domain satisfy the constraint on the channel capacity if the total utilization of such collision domain is lower than a certain threshold λ_0 (which depends on the overhead of the PHY layer and has been determined in [23] for the case of IEEE 802.11a). Also, it has been shown through extensive simulations that: i) as long as the total utilization of all the collision domains is less than λ_0, there is a high probability ($\sim$85%) that a high percentage (>95%) of the traffic load offered to the network is delivered to the destination; ii) if the total utilization exceeds that threshold for some collision domains, the percentage of the traffic load that is delivered to the destination is a decreasing function of the average total utilization. Such results suggest that, in order to maximize the portion of the traffic demands that is satisfied, we should strive to keep the total utilization of all the collision domains below a given threshold λ_0 or, in case that is not feasible, to minimize the average total utilization. Hence, we define the cost of a particular routing configuration as the average total utilization over all the collision domains. However, in order to further penalize the solutions leading to high values for the total utilization of some collision domains, we consider a weighted average of the total utilizations. In particular, we consider the weighting function:

$$w(x) = \frac{e^x - 1}{e^{\lambda_0} - 1}$$

and define the cost $\Gamma(\Phi,\mathcal{I})$ of a particular configuration as the average of $w(U_{tot}(e))$ over all the links $e \in E$:

$$\Gamma(\Phi,\mathcal{I}) = \frac{1}{|E|}\sum_{l_0\in E} w(U_{tot}(l_0)) = \frac{1}{|E|}\sum_{l_0\in E} \frac{e^{U_{tot}(l_0)} - 1}{e^{\lambda_0} - 1} \quad (2)$$

The weighting function is such that $w(x) \leqslant x$ if $x \leqslant \lambda_0$ and $w(x) > x$ if $x > \lambda_0$, i.e., it decreases the weight of the total utilizations below λ_0 and increases the weight of the total utilizations above λ_0. The goal is thus to penalize the configurations with total utilizations larger than λ_0.

For conciseness, we define $\varphi_l^s = \sum_{d\in V_e-\{s\}} \varphi_l^{s,d}$, i.e., φ_l^s is the amount of flow on link l originated at node s (independently of the destination node) and normalized to the actual (unknown) amount of traffic I_s entering node s. It follows that the actual amount of flow routed on link l can

be expressed as $\sum_{s\in V_e} \varphi_l^s I_s$. Hence:

$$\Gamma(\Phi,\mathcal{I}) = \frac{1}{|E|(e^{\lambda_0}-1)} \sum_{l_0\in E} \left[e^{\sum_{l\in\mathcal{D}(l_0)} \sum_{s\in V_e} \frac{\varphi_l^s I_s}{c(l)}} - 1 \right]$$

$$= \frac{1}{|E|(e^{\lambda_0}-1)} \sum_{l_0\in E} \left[e^{\sum_{s\in V_e} \sum_{l\in\mathcal{D}(l_0)} \frac{\varphi_l^s I_s}{c(l)}} - 1 \right]$$

Now, our goal is to compute $\Gamma(\Phi)$ by integrating $\Gamma(\Phi,\mathcal{I})$ over the region of all the feasible sets of incoming flows (eq. 1):

$$\Gamma(\Phi) = \frac{1}{|E|(e^{\lambda_0}-1)\prod_{s\in V_e} I_s^{max}} \cdot$$

$$\sum_{l_0\in E} \int_0^{I_{n_1}^{max}} \cdots \int_0^{I_{n_N}^{max}} \left[e^{\sum_{s=1}^{N} \sum_{l\in\mathcal{D}(l_0)} \frac{\varphi_l^s I_s}{c(l)}} - 1 \right] dI_1 \cdots dI_N$$

$$= \frac{1}{|E|(e^{\lambda_0}-1)\prod_{s\in V_e} I_s^{max}} \cdot$$

$$\sum_{l_0\in E} \left[\int_0^{I_{n_1}^{max}} \cdots \int_0^{I_{n_N}^{max}} \prod_{s=1}^{N} e^{\sum_{l\in\mathcal{D}(l_0)} \frac{\varphi_l^s I_s}{c(l)}} dI_{n_1} \cdots dI_{n_N} - \prod_{s=1}^{N} I_s^{max} \right] \quad (3)$$

The integrating function is the product of N functions each depending on a distinct integration variable. Hence, the multiple integral can be decomposed as the product of N integrals:

$$\int_0^{I_{n_1}^{max}} \cdots \int_0^{I_{n_N}^{max}} \prod_{s\in V_e} e^{\sum_{l\in\mathcal{D}(l_0)} \frac{\varphi_l^s I_s}{c(l)}} dI_{n_1} \cdots dI_{n_N}$$

$$= \prod_{s\in V_e} \left[\frac{1}{\sum_{l\in\mathcal{D}(l_0)} \frac{\varphi_l^s}{c(l)}} \cdot e^{\sum_{l\in\mathcal{D}(l_0)} \frac{\varphi_l^s I_s}{c(l)}} \right]_0^{I_s^{max}}$$

$$= \prod_{s\in V_e} \frac{e^{\sum_{l\in\mathcal{D}(l_0)} \frac{\varphi_l^s I_s^{max}}{c(l)}} - 1}{\sum_{l\in\mathcal{D}(l_0)} \frac{\varphi_l^s}{c(l)}}$$

Hence, the average cost $\Gamma(\Phi)$ of a routing solution Φ over all the feasible sets of incoming flows is:

$$\Gamma(\Phi) = \frac{1}{e^{\lambda_0}-1} \left[\frac{1}{|E| \prod_{s\in V_e} I_s^{max}} \cdot \sum_{l_0\in E} \prod_{s\in V_e} \frac{e^{\sum_{l\in\mathcal{D}(l_0)} \frac{\varphi_l^s I_s^{max}}{c(l)}} - 1}{\sum_{l\in\mathcal{D}(l_0)} \frac{\varphi_l^s}{c(l)}} - 1 \right] \quad (4)$$

B. Finding an optimal feasible routing solution

The average cost $\Gamma(\Phi)$ of a particular routing solution Φ over all the feasible sets of incoming flows is expressed by equation (4). Here, we formulate a convex optimization problem, denoted as METER (Minimum avErage cosT fEasible Routing), to find a routing solution that is feasible given the maximum amount of traffic entering or leaving the network at each edge node and minimizes $\Gamma(\Phi)$, subject to the constraint that the flow between each pair of ingress and egress nodes can only be routed along a predefined set of links. Solving such a problem provides the normalized amount of flow $\varphi_l^{s,d}$ routed on each link l and belonging to each ingress-egress pair (s,d). From such information, as illustrated in section V-D, we can derive the set of split ratios that each node needs to enforce our MPLS splitting policy.

TABLE I
NOTATIONS AND SYMBOLS USED IN THIS PAPER

Notation	Explanation
$\mathcal{D}(l)$	collision domain of link $l \in E$
$c(l)$	capacity of link $l \in E$
V_e	subset of nodes acting as ingress/egress nodes
I_s^{max}	maximum amount of incoming traffic at $s \in V_e$
O_d^{max}	maximum amount of outgoing traffic at $d \in V_e$
I_s	actual (unknown) amount of incoming traffic at $s \in V_e$
$\mathcal{I}$	$\{I_s\}_{s\in V_e}$
$\mathcal{E}^{s,d}$	set of links along which flow between s and d is routed
$f_l^{s,d}$	flow between edge nodes s and d that is routed on link l
f_l	total flow routed on link l
$\varphi_l^{s,d}$	flow between s and d on link l ($f_l^{s,d}$) normalized to I_s
φ_l^s	flow originated at s and routed on link l, normalized to I_s
Φ	routing solution (set of $\varphi_l^{s,d}$ for each $s,d,l \in \mathcal{E}^{s,d}$)
$\Gamma(\Phi,\mathcal{I})$	cost of routing a given set of incoming flows according to Φ
$\Gamma(\Phi)$	average of $\Gamma(\Phi,\mathcal{I})$ over all feasible sets of incoming flows $\mathcal{I}$
$\rho_{u\to v}^{s,d}$	ratio of the flow from s to d entering u that is forwarded to v

The formulation of the METER problem is shown in fig. 6. Besides the set of normalized variables $\{\varphi_l^{s,d}\}_{l\in\mathcal{E}^{s,d}}^{s\in V_e,\, d\in V_e-\{s\}}$, we also consider a set of auxiliary variables $\{F^{s,d}\}^{s\in V_e, d\in V_e-\{s\}}$, each representing the amount of flow routed between an ingress node s and an egress node d, normalized to the actual incoming traffic at node s. The objective of METER is to minimize $\Gamma(\Phi)$. Constraints 1) represent the usual (normalized) flow conservation constraint that must be enforced at each node for every pair of ingress-egress nodes. Constraints 2) ensure that all the actual amount of incoming flow at each edge node is split among the other edge nodes. Constraints 3) ensure that the amount of flow routed towards each egress node does not exceed the maximum amount of outgoing traffic of that egress node, for *every* feasible set of incoming flows (constraint f'' of Section IV). Indeed, if the incoming set of flows is feasible, then $\sum_{s\in V_e-\{d\}} F^{s,d} I_s \leqslant \sum_{s\in V_e-\{d\}} F^{s,d} I_s^{max}$, where the left hand side is the actual amount of flow routed towards egress node d. Hence, if constraint 3) holds, constraint f'') holds as well. Constraints 4) prevent the incoming (outgoing) flow at an edge node to be re-routed back to the ingress node (from the egress node). Constraints 5) ensure that the amount of flow routed on each link does not exceed the link capacity, for *every* feasible set of incoming flows (constraint f' of Section IV). Finally, constraints 6) ensure that the flow between edge nodes s and d is only allocated on links that belong to the predefined set $\mathcal{E}^{s,d}$. In such a way, if the predefined set of links are properly computed, the routing solution returned by the optimization problem is guaranteed to be admissible. In Section V-C we present an algorithm that finds, for a given ingress-egress pair (s,d), a directed subgraph that meets

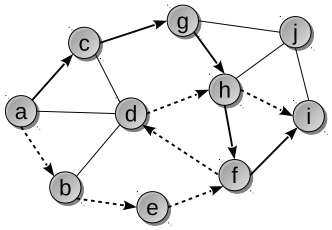


Fig. 3. Example to illustrate loops in the directed subgraph

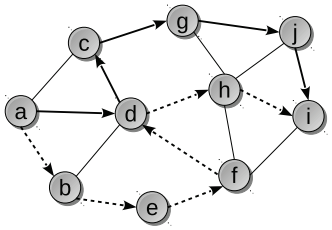


Fig. 4. Example to illustrate paths exceeding the maximum length in the directed subgraph

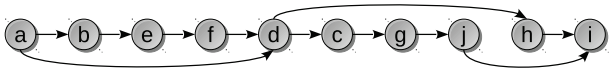


Fig. 5. Topologically sorted nodes of the DAG in fig. 4.

variables

$$\varphi_l^{s,d} \in [0,1] \quad \forall l \in E, \forall s \in V_e, \forall d \in V_e - \{s\}$$
$$F^{s,d} \in [0,1] \quad \forall s \in V_e, \forall d \in V_e - \{s\}$$

minimize $\Gamma\left(\left\{\varphi_l^{s,d}\right\}_{l \in \mathcal{E}^{s,d}}^{s \in V_e, d \in V_e - \{s\}}\right)$

subject to

1) $$\sum_{u \to v \in \mathcal{E}^{s,d}} \varphi_{u \to v}^{s,d} - \sum_{v \to u \in \mathcal{E}^{s,d}} \varphi_{v \to u}^{s,d} = \begin{cases} 0 & \text{if } u \neq s \wedge u \neq d \\ F^{s,d} & \text{if } u = s \\ -F^{s,d} & \text{if } u = d \end{cases}$$
$$\forall u \in V,\ \forall s \in V_e, \forall d \in V_e - \{s\}$$

2) $$\sum_{d \in V_e - \{s\}} F^{s,d} = 1$$
$$\forall s \in V_e$$

3) $$\sum_{s \in V_e - \{d\}} F^{s,d} I_s^{max} \leqslant O_d^{max}$$
$$\forall d \in V_e$$

4) $$\varphi_{u \to v}^{s,d} = 0$$
$$\forall s \in V_e,\ \forall d \in V_e - \{s\},\ v = s \vee u = d$$

5) $$\sum_{s \in V_e} \sum_{d \in V_e - \{s\}} \varphi_l^{s,d} I_s^{max} \leqslant c(l)$$
$$\forall l \in E$$

6) $$\varphi_l^{s,d} = 0$$
$$\forall s \in V_e,\ \forall d \in V_e - \{s\},\ l \notin \mathcal{E}^{s,d}$$

Fig. 6. Formulation of the METER problem.

constraints t'), t'') and t''') of Section IV.

METER is a convex optimization problem, because the objective function is convex (see the Appendix) and the constraint functions are linear. Convex optimization problems have the property that a locally optimal point is also globally optimal. Hence, we use an interior point method [24] to find an optimal solution to the problem.

C. Finding a directed subgraph for an ingress-egress pair

We now address the problem to find, for a given ingress-egress pair (s, d), a set of links $\mathcal{E}^{s,d}$ or, equivalently, a directed subgraph $\mathcal{S}^{s,d}$ that guarantees that a routing solution is admissible. The approach we follow is to find a set of paths between s and d (in G) and insert all of their links into $\mathcal{E}^{s,d}$. Given the constraint on the maximum allowed length of any path between s and d in $\mathcal{S}^{s,d}$, a possible approach would be to use a k-shortest loopless path algorithm [25] to find all the loopless paths between s and d in G having a length less than the maximum one. A k-shortest loopless path algorithm indeed returns all the shortest paths in increasing order of length. However, the following two issues should be considered:

- *only adding loopless paths to $\mathcal{E}^{s,d}$ does not ensure that all the possible paths in the subgraph $\mathcal{S}^{s,d}$ are loopless*. An example is illustrated in fig. 3, where the links of two loopless paths (a-c-g-h-f-i and a-b-e-f-d-h-i) are added to $\mathcal{E}^{s,d}$. The resulting subgraph includes paths (e.g., a-b-e-f-d-h-f-i) containing a cycle (f-d-h-f)
- *only adding paths with length less than the maximum allowed one does not ensure that all the possible paths in the subgraph have a length less than the maximum allowed one*. An example is illustrated in fig. 4, where the maximum length is fixed to 6 hops. If we add the links of two paths satisfying the constraint on the maximum path length (a-d-c-g-j-i and a-b-e-f-d-h-i), the resulting subgraph includes a path (a-b-e-f-d-c-g-j-i) having a length (8 hops) exceeding the maximum allowed length

Therefore, we present an algorithm that finds, for a given ingress-egress node pair, a directed subgraph of the network topology that meets constraint t'), t'') and t''') of Section IV. Basically, the subgraph is initialized to contain the shortest path between the ingress and egress nodes and then it is augmented with other paths to fulfil constraint t'''). Every time we attempt to add a path to the subgraph, we check whether the augmented subgraph contains a cycle or a path having a length exceeding the maximum one. Fortunately, the subgraph we seek is a *directed acyclic graph* (DAG) and hence performing the above checks is as simple as running a Depth-First-Search (DFS) [26]. Also, a DFS in a DAG allows to sort the nodes in a topological ordering, which is such that if a link $u \to v$ exists in the DAG, then u precedes v in the ordering. Thus, the ingress (egress) node is the first (last) node in this ordering. As an example, fig. 5 shows a topological sort ordering of the DAG resulting from the two paths highlighted in fig. 4.

Figure 7 shows the pseudo-code of the Depth-First-Search visit used by our algorithm. The DFS procedure initializes the attributes (color, predecessor and maximum distance from the egress node) of all the nodes of the subgraph D, clears the list that will contain the nodes sorted in a topological ordering, sets the boolean variable $acyclic$ to true and calls DFS_VISIT on node s. The DFS_VISIT procedure performs the classic DFS visit starting from node u. In addition, if a *back* edge is detected (lines 6–7), the $acyclic$ variable is set to false. Indeed, a directed graph is acyclic if and only if a depth-first-search yields no back edges. To obtain a list of the nodes of the DAG sorted in a topological ordering, we can push

```
DFS(D = (V_D, E_D), s)
1  for each u ∈ V_D
2       color[u] = WHITE
3       pred[u] = NIL
4       maxdist[u] = 0
5  sortedList = ∅
6  acyclic = TRUE
7  DFS_VISIT(D, s)

DFS_VISIT(D = (V_D, E_D), u)
 1  color[u] = GRAY
 2  for each v ∈ V_D | u → v ∈ E_D
 3       if color[v] = WHITE
 4            pred[v] = u
 5            DFS_VISIT(D, v)
 6       else if color[v] = GRAY
 7                 acyclic = FALSE
 8  color[u] = BLACK
 9  if sortedList ≠ ∅
10       maxdist[u] = max_{v | u→v∈E_D} maxdist[v] + 1
11  sortedList = u, sortedList
```

Fig. 7. Pseudo-code of the depth-first-search.

nodes on the front of such a list as soon as they are marked as black (line 11). Also (lines 9–10), when inserting a node u in such a list (but the first node being inserted, which is the egress node), we compute the maximum distance in the DAG between u and the last node of the sorted list (the egress node) by increasing by 1 the maximum distance of each neighbor of u (at this point, all the neighbors v of u such that $u \rightarrow v$ exists in the DAG have been already inserted into the sorted list, by definition of topological ordering). Therefore, a DFS on a subgraph D checks whether D is a DAG and, in that case, returns a sorted list of nodes in a topological ordering and the length of the longest path between each node in D and the last node in the topological ordering (the egress node).

We now present our algorithm, denoted as RDAS (Resilient Directed Acyclic Subgraph), to find, for a given pair (s,d) of ingress and egress nodes, a directed subgraph of a graph G that meets constraint t'), t'') and t''') of Section IV. Basically, RDAS (fig. 8) initializes the subgraph D to the shortest path in G between the ingress and egress nodes and then explores the nodes in D in a reverse topological order, starting from the penultimate node. To satisfy constraint t'''), an attempt is made to ensure that the explored node u has two distinct next hops in D. To this end, a path between u and the egress node d is sought that does not include the current next hop of u and satisfies t') and t''). If such a path is found, it is added to D and the exploration restarts from the penultimate node in the new topological ordering of the nodes in D. An explored node is marked as *done*, so that it is explored just once. The algorithm ends when the ingress node is marked as done.

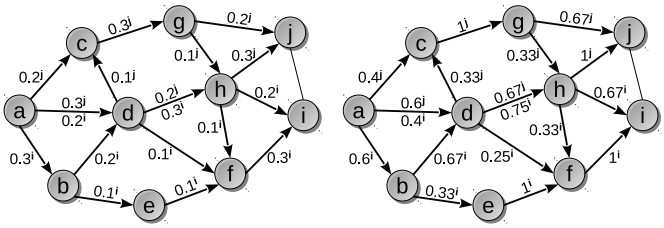
We now describe the exploration of a node in more details. If the explored node has been already marked as done, we continue by exploring its predecessor in the topological ordering of the nodes in D (lines 9–11). If the explored node has already more than one next hop in D, it is marked as done and its predecessor in the topological ordering is then explored (lines 34–35). If the explored node u has a single neighbor

```
RDAS(G = (V, E), s, d, α)
 1  D = ∅
 2  SP = SHORTEST_PATH(G, s, d)
 3  PATH_ADD(D, SP)
 4  DFS(D, s)
 5  for each u ∈ V
 6       done[u] = FALSE
 7  u = previous[back[sortedlist[D]]]
 8  while u <> NIL
 9       if done[u] = TRUE
10            u = previous[u]
11            continue
12       if |Adj(D, u)| = 1
13            v = front[Adj(D, u)]
14            G_Pruned = G
15            if v = d
16                 REMOVE_EDGE(G_Pruned, u → d)
17            else REMOVE_VERTEX(G_Pruned, v)
18            found = FALSE
19            L^su = MAX_DIST_FROM_SOURCE(D, u)
20            L^ud = 0
21            while !found AND L^su + L^ud ⩽ α · length[SP]
                    AND KSP_HAS_NEXT(G_Pruned, u, d)
22                 P = KSP_NEXT(G_Pruned, u, d)
23                 D_Augm = D
24                 PATH_ADD(D_Augm, P)
25                 DFS(D_Augm, s)
26                 if IS_ACYCLIC(D_Augm) AND
                      MAX_DIST_TO_DEST(D_Augm, s)
                      ⩽ α · length[SP]
27                      found = TRUE
28                 L^ud = length[P]
29            if found
30                 done[u] = TRUE
31                 D = D_Augm
32                 u = previous[back[sortedlist[D]]]
33                 continue
34       done[u] = TRUE
35       u = previous[u]
```

Fig. 8. Pseudo-code of the RDAS algorithm.

(v) in D, we attempt to find an alternative path to the egress node. For this purpose, we consider a copy (G_{pruned}) of the input graph G and prune the link $u \rightarrow v$, in case v is the egress node, or the vertex v otherwise. Then, we look for a path between u and the egress node d in the pruned graph. A k-shortest loopless path algorithm [25] provides, one-by-one and in increasing order of length, the shortest paths between u and d in the pruned graph. The path P returned by the k-shortest path algorithm is tentatively added to a copy (D_{Augm}) of the subgraph D. A DFS of D_{Augm} is run, which determines whether D_{Augm} is acyclic, finds the length of the longest path between s and d and topologically sorts the nodes. If D_{Augm} is acyclic and the length of the longest path is less than the maximum allowed path length, the path P is actually added to D, node u is marked as done and the exploration of the nodes restarts from the penultimate node in the new topological ordering (lines 29–33). Otherwise, a new path returned by the k-shortest path algorithm is considered.

To avoid that a number of shortest paths between u and d in the pruned graph are uselessly considered, we compute the length L^{su} of the longest path between the ingress node s and u in D (line 19). Given that we already have a topological



(a) Sample values for the $\varphi_l^{s,d}$ variables (b) Corresponding set of split ratios

Fig. 9. Deriving the set of split ratios from the set of $\varphi_l^{s,d}$ variables.

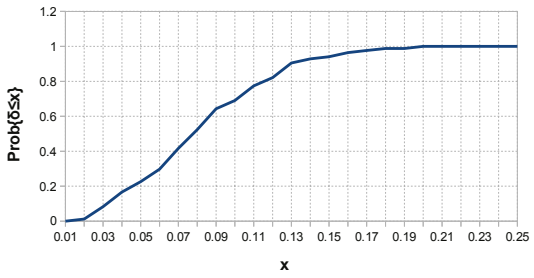


Fig. 10. Empirical CDF of δ.

ordering of the nodes in D, computing such a value only requires to relax all the edges of D (whose weights must be set to -1) [26]. Thus, as soon as the k-shortest path algorithm returns a path with a length L^{ud} such that $L^{su} + L^{ud}$ exceeds the maximum allowed path length, we can stop processing the shortest paths between u and d. Indeed, since shortest paths are returned in increasing order of length, none of the following shortest paths can be added to the subgraph without violating the constraint on the maximum path length. In such a case, node u is marked as done (despite it only has one neighbor) and the predecessor of u in the topological ordering is explored. The algorithm ends when the ingress node s is marked as done and returns the directed subgraph D.

The complexity of RDAS is dominated by the inner while loop (lines 21–28). In the worst case (since the nodes in D are a subset of those in G), a DFS on D requires $O(|V| + |E|)$, while obtaining the next path from the k-shortest path algorithm requires $O(|V|(|E| + |V| \log |V|))$. The outer while loop is repeated at most $|V|$ times (once for each node in D), hence the complexity of RDAS is $O(|V|^2(|E| + |V| \log |V|))$.

D. Computing the optimal set of split ratios

Solving the METER problem (fig. 6) provides the values for the set of variables $\{\varphi_l^{s,d}\}_{l \in \mathcal{E}^{s,d}}^{s \in V_e, d \in V_e - \{s\}}$ representing the amount of flow routed on each link and associated with each pair of ingress-egress nodes, normalized to the actual incoming flow at the ingress node. Figure 9a shows a network with some sample values for the flows routed between a and i and between a and j (the superscript next to a value indicates the destination of the flow) normalized to the incoming flow at a. Since the splitting policy balances the traffic among the outgoing links in proportion to the split ratios, to achieve the normalized flows given by the set of variables $\{\varphi_l^{s,d}\}_{l \in \mathcal{E}^{s,d}}^{s \in V_e, d \in V_e - \{s\}}$ we need to set the split ratios as follows:

$$\rho_{u \to v}^{s,d} = \frac{\varphi_{u \to v}^{s,d}}{\sum_{u \to w \in \mathcal{E}^{s,d}} \varphi_{u \to w}^{s,d}}$$

Figure 9b shows the set of split ratios corresponding to the normalized flow values of fig. 9a.

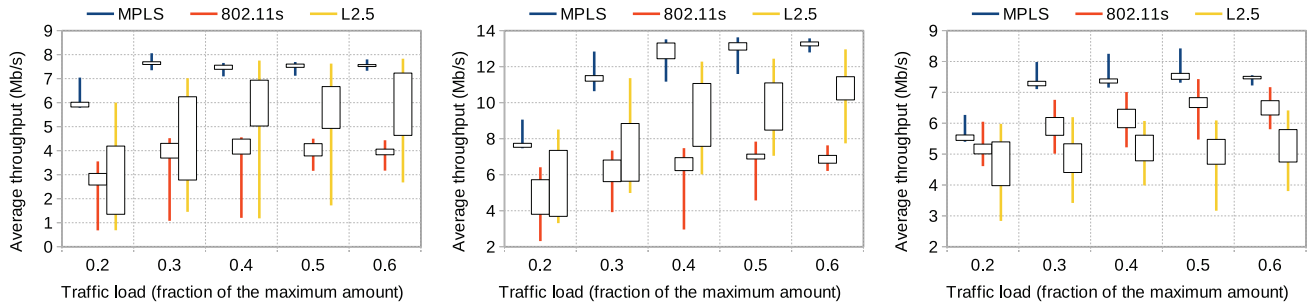
VI. Performance Evaluation

We conducted a number of simulation studies to evaluate the performance of the proposed MPLS-based forwarding paradigm. For this purpose, we implemented a software tool that uses the planar coordinates of the mesh nodes to build the network topology based on the interference model described in Section IV. We assume the gain G_{uv} of the radio channel between u and v to be the reciprocal of the square of the distance between u and v and the thermal noise to be -20dbm. The SINR thresholds are set to allow a rate of $54Mbps$ when the nodes are within $30m$, $48Mbps$ when within $32m$, $36Mbps$ when within $37m$, $24Mbps$ when within $45m$, $18Mbps$ when within $60m$, $12Mbps$ when within $69m$, $9Mbps$ when within $77m$ and $6Mbps$ when within $90m$. The channel assignment is performed by FCRA [4]. Since FCRA is a traffic-aware channel assignment algorithm, different channel assignments can be obtained by feeding FCRA with different sets of link flows. We consider two topologies with 30 nodes and two radios per node. In the *dense* topology, nodes are placed in a $193 \times 215m^2$ area and the maximum link rate is $54Mbps$. In the *sparse* topology, nodes are placed in a $261 \times 265m^2$ area and the maximum link rate is $36Mbps$. We consider 4 edge nodes and hence 12 ingress-egress pairs. METER, the convex problem described in Section V-B, is solved by using the open source software *Ipopt* (Interior Point OPTimizer). The next subsections (but the first one) aim to evaluate the performance of our approach in terms of network throughput. To this end, experiments were carried out with the network simulator *ns-3*. We contributed to the implementation of the MPLS module in ns-3 and added the MPLS splitting policy[1] In such experiments, we compare our approach (simply labelled as MPLS) based on the MPLS splitting policy with the split ratios determined as shown in Section V to our previous Layer-2.5 forwarding paradigm (L2.5) and to the routing protocol specified in IEEE 802.11s [13]. Unless otherwise stated, RDAS is run with α=3 in order to consider paths that are much longer than the shortest path, which likely consists of links between distant nodes utilizing low bit rates. The threshold λ_0 is set to 0.5. In the ns-3 experiments, TCP traffic is generated (for a duration of 60 seconds) according to the on-off model, with $T_{on} \sim U(0.5s, 1.5s)$ and $T_{off} \sim U(0.05s, 0.15s)$.

A. Comparing METER-RDAS to a lower bound

Our approach to solve the MPLS splitting-based routing problem is to solve the METER-RDAS problem, where RDAS is used to compute the set of directed subgraphs that are

[1]As of this writing, the MPLS module has not been merged yet into the mainline ns-3 code. The MPLS code is available at http://code.google.com/p/ns-3-shop.



(a) Dense topology, 3 channels (b) Dense topology, 11 channels (c) Sparse topology, 11 channels

Fig. 11. Average throughput achieved under different traffic loads.

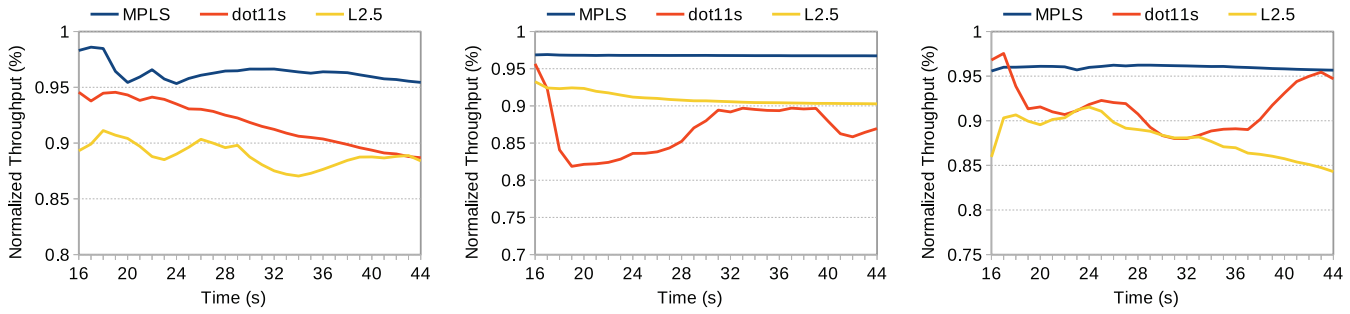
required to formulate the METER problem. Thus, our approach might not return the optimal feasible admissible routing solution of the MPLS splitting-based routing problem, because the returned routing solution is constrained to allocate flow on the links of the directed subgraphs computed by RDAS. Though the average cost of the optimal routing solution Φ^{opt} is difficult to find, it is straightforward to compute a lower bound to such value. To this end, we denote by $\mathcal{E}_{KSP}^{s,d}$ the set of links of all the paths between s and d in G whose length is at most α times the length of the shortest path. Such a set can be easily computed by using a k-shortest loopless path algorithm to find all the paths between s and d with length less than the maximum one. $\mathcal{E}_{KSP}^{s,d}$ is not guaranteed to satisfy constraints t'), t'') and t'''), but it certainly includes any set $\mathcal{E}^{s,d}$ leading to an admissible routing solution. Thus, if we solve the METER problem with $\mathcal{E}_{KSP}^{s,d}$ as the set of links that are allowed to carry the flow between s and d (we denote such a problem by METER-KSP), the obtained objective value, denoted as $\Gamma(\Phi^{KSP})$, represents a lower bound to the average cost of the optimal feasible admissible routing solution.

In order to compare the average cost of the routing solution returned by METER-RDAS, denoted as $\Gamma(\Phi^{RDAS})$, to the lower bound to the minimum average cost, we performed 100 experiments with varying values for the maximum amount of traffic entering and leaving the network at the edge nodes, different channel assignments and α values (ranging from 1.5 to 3). For each experiment, we solved both METER-RDAS and METER-KSP and computed $\delta = \frac{\Gamma(\Phi^{RDAS})-\Gamma(\Phi^{KSP})}{\Gamma(\Phi^{KSP})}$, i.e., the percentage increase with respect to the lower bound. The empirical CDF (Cumulative Distribution Function) of the values of δ resulting from our experiments is shown in fig. 10. It can be observed that the percentage increase of the average cost of the routing solution returned when using RDAS is always below 20%, while the percentage increase is below 10% in the 70% of the cases and below 6% in the 30% of the cases. If we consider that such results refer to a comparison with a lower bound and that $\mathcal{E}_{KSP}^{s,d}$ is unlikely to lead to an admissible routing solution, we can assert that our approach achives an average cost very close to the minimum one.

B. Robustness against variations in the traffic load

Our approach to solve the MPLS splitting-based routing problem has been designed to provide a set of split ratios ensuring high performance under different traffic loads. The experiments described in this section aim to show that our approach is actually more robust against variations in the traffic load than other routing protocols such as the default routing protocol specified in IEEE 802.11s and our previous approach L2.5. We report the results obtained for the two topologies mentioned earlier, where the maximum amount of traffic entering each edge node is uniformly distributed between $6Mbps$ and $8Mbps$. We also considered the availability of 3 and 11 orthogonal channels. For each topology, we performed different experiments where the actual traffic load entering the network was, on the average, a percentage of the maximum amount ranging from 20% to 60%. For each such cases, we considered 4 uniform random variables with a mean equal to the required percentage (e.g., $U(0.4, 0.6), U(0.3, 0.7), U(0.2, 0.8), U(0.1, 0.9)$ in case the actual traffic load is, on the average, the 50% of the maximum amount). Each of such uniform random variables was used to derive the fraction of the maximum amount of traffic entering each edge node that actually entered the network. Then, the actual traffic entering each edge node was split among the destination nodes in 5 different ways, thus leading to a total of 20 experiments for each topology and for each given percentage of the maximum amount of traffic load.

Figure 11 summarizes the distribution of the throughput (average over the whole duration of a simulation) achieved by each algorithm in the 20 experiments carried out for each given percentage of the maximum amount of traffic load. In particular, a vertical line spans from the minimum to the maximum values, while a white box spans from the first quartile to the third quartile. It can be observed that our approach outperforms 802.11s and L2.5 in all the considered scenarios. The average throughput achieved by our approach is indeed from 20% to 200% higher than that of the other routing protocols. The poor performance of 802.11s in some experiments can be explained by considering that 802.11s is a single path routing protocol and therefore, in case of a high traffic demand between an ingress-egress pair, the selected path may be easily congested, thus leading to a decrease in the throughput. The poor performance of L2.5 in some experiments can be explained by considering that each node attempts to utilize each link in proportion to predefined flow rates, that are returned by a traffic aware channel assignment algorithm. Thus, if the channel assignment has been computed based on a traffic load which is different than the actual offered



(a) Dense topology, 3 channels (b) Dense topology, 11 channels (c) Sparse topology, 11 channels

Fig. 12. Transferred data loss due to a single node failure.

load, the performance of L2.5 may decrease.

Figures 11a-11c also show that our approach ensures higher robustness against variations in the traffic load. Indeed, the ratio of the maximum average throughput to the minimum average throughput achieved in the 20 experiments associated with a given topology and a given percentage of the maximum amount of traffic load ranges from 5% to 21% for our approach, from 23% to 400% for 802.11s and from 50% to 600% for L2.5. Such results prove that the performance in terms of throughput of our approach keeps firmly at high values despite variations in the traffic load, while the performance of the other routing protocols oscillates between low and moderate to high values depending on the offered traffic load.

C. Robustness against a single node failure

The experiments described in this section aim at evaluating the behavior of the routing protocols in the presence of a single node failure. We consider the same topologies as in the previous section and an actual traffic load equal to 20% of the maximum traffic load. Each experiment lasts 45 seconds and, after 15 seconds from the beginning, a node failure is simulated by increasing the noise level at every radio interface of that node to the point that the node is not able to send or receive packets. For each topology, we perform 26 experiments, each involving the failure of a different node (except the edge nodes), and compute, for each $1s$ interval, the average throughput over all the experiments. The average throughput over a $1s$ interval represents the amount of data transferred to the destination nodes during that interval. Figures 12a-12c show, for each second following the failure, the ratio of the average (over the 26 experiments) amount of data transferred to the destination nodes since the failure to the amount of data transferred in the same interval in the absence of failures. It can be observed that our approach based on the MPLS splitting policy outperforms both 802.11s and L2.5. Indeed, our approach keeps the data loss in case of a node failure below 5% in all the considered scenarios. Instead, 802.11s and L2.5 experience higher data losses, ranging from 5% to 18%. Such results thus prove that our approach is able to better react to single node failures than 802.11s and L2.5.

VII. Conclusion

We addressed the problem to develop a routing strategy for multi-radio wireless mesh networks which: i) takes the constraint on the channel capacity into account; ii) is able to quickly recover from node/link failures; iii) supports multi-path routing; iv) ensures high performance with a wide range of traffic matrices. For this purpose, we presented a novel mechanism, the MPLS splitting policy, which consists in allowing multiple candidate next hops at each intermediate node for a given FEC and partitioning the traffic of that FEC among such next hops in proportion to predefined split ratios. The MPLS splitting policy enables to balance the traffic load across multiple paths and allows for a fast local restoration. We then developed a technique to compute the set of split ratios in order to ensure high throughput despite variations in the traffic load. This goal is achieved by properly defining a cost function, computing directed subgraphs along which the flow of each ingress-egress pair can be routed and solving a convex optimization problem. Finally, we performed a thorough simulation study which confirmed that our approach outperforms other routing protocols in terms of network throughput and robustness against load variations and single node failures.

Appendix

We first show that $f : \vec{x} \in \mathbf{R}^N \rightarrow \prod_{i=1}^{N} \frac{e^{x_i}-1}{x_i}$ is log-convex, i.e., $\log f(\vec{x}) = \sum_{i=1}^{N} \log \frac{e^{x_i}-1}{x_i}$ is convex. To this end, we denote by $g_i(\vec{x}) = \log \frac{e^{x_i}-1}{x_i}$ the i-th term in the expression of $\log f(\vec{x})$. Since $\frac{\partial^2}{\partial x_j \partial x_k} g_i(\vec{x}) = 0$ if $j \neq i$ or $k \neq i$, it follows that the Hessian of $g_i(\vec{x})$ is a diagonal matrix:

$$\nabla^2 g_i(\vec{x}) = \mathrm{diag}\left(\frac{1}{x_i^2} - \frac{e^{x_i}}{(e^{x_i}-1)^2}, 0, ...0\right)$$

The unique non-null eigenvalue is always positive (it can be easily seen by plotting its graph) and hence the Hessian is positive semidefinite, which means that $g_i(\vec{x})$ is a convex function. Being a sum of convex functions, $\log f(\vec{x})$ is convex, i.e., $f(\vec{x})$ is log-convex, which implies that $f(\vec{x})$ is convex.

We now consider the column vector $\vec{\varphi}_{l_0} \in \mathbf{R}^{N \cdot |\mathcal{D}(l_0)|}$:

$$\vec{\varphi}_{l_0} = [\varphi_{l_1}^1 \dots \varphi_{l_d}^1 \, \varphi_{l_1}^2 \dots \varphi_{l_d}^2 \dots \varphi_{l_1}^N \dots \varphi_{l_d}^N]^T$$

where l_0 is a link, $d = |\mathcal{D}(l_0)|$ and $\{l_i\}_{i=1}^{d} = \mathcal{D}(l_0)$, and the matrix $A_{l_0} \in \mathbf{R}^{N \times N \cdot |\mathcal{D}(l_0)|}$:

$$A_{l_0} = \begin{bmatrix} a_{1,1} & \dots & a_{1,d} & 0 & \dots & 0 & \dots & 0 & \dots & 0 \\ 0 & \dots & 0 & a_{2,d+1} & \dots & a_{2,2d} & \dots & 0 & \dots & 0 \\ \vdots & & \vdots & \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \dots & 0 & 0 & \dots & 0 & \dots & a_{N,(N-1)d+1} & \dots & a_{N,Nd} \end{bmatrix}$$

where

$$a_{i,j} = \begin{cases} \dfrac{I_i^{max}}{c(l_{(j-1) \bmod d+1})} & \text{if } (i-1)d < j \leqslant id, \\ 0 & \text{otherwise} \end{cases}$$

with $i = 1, ...d$ and $j = 1, ...Nd$. Then, the function:

$$h(\vec{\varphi}_{l_0}) = f(A_{l_0}\vec{\varphi}_{l_0}) = \prod_{s=1}^{N} \frac{e^{\sum\limits_{l \in \mathcal{D}(l_0)} \frac{\varphi_l^s I_s^{max}}{c(l)}} - 1}{I_s^{max} \sum\limits_{l \in \mathcal{D}(l_0)} \frac{\varphi_l^s}{c(l)}}$$

is convex because obtained from a convex function ($f(\vec{x})$) through composition with an affine mapping. Hence the average cost of a routing solution (eq. 4) can be expressed as:

$$\Gamma(\Phi) = \frac{1}{e^{\lambda_0} - 1} \left[\frac{1}{|E|} \sum_{l_0 \in E} h(\vec{\varphi}_{l_0}) - 1 \right]$$

and therefore it is convex as well, because sum of convex functions. Hence, the objective function of the optimization problem defined in section V-B is convex.

ACKNOWLEDGMENT

This work was partially funded by the Italian Ministry of Education, University and Research (MIUR) within the framework of project PRIN 2009 "Software router to Improve Next-Generation Internet" (SFINGI).

REFERENCES

[1] A. Raniwala, K. Gopalan, and T. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks," *ACM Mobile Computing and Commun. Review*, vol. 8, no. 2, pp. 50–65, Apr. 2004.

[2] M. Alicherry, R. Bhatia, and E. Li, "Joint channel assignment and routing for throughput optimization in multiradio wireless mesh networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 11, pp. 1960–1971, Nov. 2006.

[3] A. Subramanian, H. Gupta, S. R. Das, and J. Cao, "Minimum interference channel assignment in multi-radio wireless mesh networks," *IEEE Trans. Mobile Comput.*, vol. 7, no. 12, pp. 1459–1473, 2008.

[4] S. Avallone, I. F. Akyildiz, and G. Ventre, "A channel and rate assignment algorithm and a layer-2.5 forwarding paradigm for multi-radio wireless mesh networks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 1, pp. 267–280, Feb. 2009.

[5] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," IETF, RFC 3031, Jan. 2001.

[6] N. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. Ramakrishnan, and J. van der Merwe, "A flexible model for resource management in virtual private network," in *Proc. 1999 ACM SIGCOMM*, pp. 43–57.

[7] D. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "High-throughput path metric for multi-hop wireless routing," in *Proc. 2003 ACM MobiCom*, pp. 134–146.

[8] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *Proc. 2004 ACM MobiCom*, pp. 114–128.

[9] Y. Yang, J. Wang, and R. Kravets, "Designing routing metrics for mesh networks," in *Proc. 2005 IEEE WiMesh*.

[10] T. Liu and W. Liao, "Interference-aware QoS routing for multi-rate multi-radio multi-channel IEEE 802.11 wireless mesh networks," *IEEE Trans. Wireless Commun.*, vol. 8, no. 1, pp. 166–175, 2009.

[11] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," IETF, RFC 3561, July 2003.

[12] T. Clausen and P. Jacquet, "Optimized link state routing protocol (OLSR)," IETF, RFC 3626, Oct. 2003.

[13] "IEEE Standard for Information Technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 10: Mesh Networking," *IEEE Std 802.11s-2011*, Sept. 2011.

[14] S.-J. Lee and M. Gerla, "AODV-BR: backup routing in ad hoc networks," in *Proc. 2000 IEEE WCNC*, vol. 3, pp. 1311–1316.

[15] M. Marina and S. Das, "On-demand multi-path distance vector routing in ad-hoc networks," in *Proc. 2001 IEEE ICNP*.

[16] X. Hu and M. J. Lee, "An efficient multipath structure for concurrent data transport in wireless mesh networks," *Computer Commun.*, vol. 30, pp. 3358–3367, Nov. 2007.

[17] K.W.Choi, W. Jeon, and D. Jeong, "Efficient load-aware routing scheme for wireless mesh networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 9, pp. 1293–1307, Sept. 2010.

[18] S. Biswas and R. Morris, "ExOR: opportunistic multi-hop routing for wireless networks," in *Proc. 2005 ACM SIGCOMM*, pp. 133–143.

[19] Y. Yuan, H. Yang, S. Wong, S. Lu, and W. Arbaugh, "ROMER: resilient opportunistic mesh routing for wireless mesh networks," in *Proc. 2005 IEEE WiMesh*.

[20] B. Choi, T. Wong, and J. Shea, "Geographic transmission with optimized relaying (GATOR) for the uplink in mesh networks," *IEEE Trans. Wireless Commun.*, vol. 11, no. 6, pp. 2095–2105, 2012.

[21] R. Laufer, H. Dubois-Ferriere, and L. Kleinrock, "Polynomial-time algorithms for multirate anypath routing in wireless multihop networks," *IEEE/ACM Trans. Netw.*, vol. 20, no. 3, pp. 742–755, June 2012.

[22] X. Fang, D. Yang, and G. Xue, "MAP: multi-constrained anypath routing in wireless mesh networks," *IEEE Trans. Mobile Comput.*, 2013, to appear.

[23] S. Avallone, G. Di Stasi, and A. Kassler, "A traffic-aware channel and rate reassignment algorithm for wireless mesh networks," *IEEE Trans. Mobile Comput.*, vol. 12, no. 7, pp. 1335–1348, 2013.

[24] A. Wächter and L. T. Biegler, "On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[25] Y. Yen, "Finding the K shortest loopless paths in a network," *Management Science*, vol. 17, no. 11, pp. 712–716, July 1971.

[26] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. MIT Press, 2001.

Stefano Avallone received the M.S. degree in Telecommunications Engineering (2001) and the PhD degree in Computer Networks (2005) from the University of Napoli "Federico II". He is currently an Assistant Professor with the Department of Computer Engineering at the University of Napoli. His research interests include wireless mesh networks, traffic engineering, QoS routing. He was a visiting researcher at the Delft University of Technology (2003-04) and at the Georgia Institute of Technology (2005). He is on the editorial board of *Elsevier Ad Hoc Networks*.

Giovanni Di Stasi received the Laurea degree in Computer Engineering (2007) and the PhD degree in Computer Networks (2011) from the University of Napoli Federico II. He was a visiting researcher at INRIA Sophia Antipolis, France (2009) and at the Karlstad University, Sweden (2010). His current research interests include experimental reserch infrastructures and testbeds, routing and channel assignment algorithms for wireless mesh networks and peer-to-peer traffic optimization.