Abstract—This paper presents MARA, a joint mechanism for automatic rate selection and route quality evaluation in Wireless Mesh Networks. This mechanism targets at avoiding the problems of lack of synchronization between metric and rate selection decisions, and inaccurate link quality estimates, common to main existing proposals of multi-hop wireless routing metrics and automatic rate adaptation. In this proposal, the statistics collected by the routing protocol are used by the rate adaptation algorithm to compute the best rate for each wireless link. This coordinated decision aims at providing better routing and rate choices. In addition to the basic MARA algorithm, two variations are proposed: MARA-P and MARA-RP. The first considers the size of each packet in the transmission rate decision. The second variation considers the packet size also for the routing choices. For evaluation purposes, experiments were conducted on both real and simulated environments. In these experiments, MARA was compared to a number of rate adaptation algorithms and routing metrics. Results from both environments indicate that MARA may lead to an overall network performance improvement.

Index Terms—Routing Metrics, Automatic Rate Selection, Wireless Mesh Networks.

I. INTRODUCTION

WIRELESS Mesh Networks (WMNs) [1] are composed by mesh routers and client nodes. The mesh routers compose a mesh of wireless links which is used for multi-hop communication by client nodes. Each mesh router may act as an access point, serving as an Internet gateway for client nodes, or only as a part of the backbone, forwarding packets from other routers.

For the past few years, mesh networks have experienced a huge increase in popularity, mostly due to its potentially low cost of deployment and maintenance. Many companies already provide WMN solutions [2]–[4], although their costs are, usually, still elevated. Nevertheless, there are low cost commercial solutions targeted specifically at end-users [5], [6]. Moreover, there are also consolidated open solutions often applied to digital inclusion projects [7], [8].

To cope with low cost requirements, WMNs usually employ off-the-shelf hardware based on the IEEE 802.11 standard. These devices are capable of operating at multiple transmission rates, varying from 1 Mbps to 54 Mbps. However, selecting the most suitable transmission rate is not trivial, since there is a trade-off between link capacity and transmission rate. Typically, PER (Packet Error Rate) increases with the transmission rate, considering the same SNR conditions, because lower rates tend to use more robust modulations and code rates.

Furthermore, WMN nodes must implement a dynamic route discovery mechanism. One of the core elements of this mechanism is the routing metric. Even though there are a number of metrics with very coherent formulations, they all face the obstacle of obtaining consistent and reliable information about the quality of wireless links [9].

It is important to notice that these two problems, automatic rate adaptation and routing metrics assignment, are strongly related. The characteristics of a wireless link, as evaluated by routing metrics, are dependent on the chosen transmission rate. For instance, if the routing metric evaluates packet error rates, it is important to know what is the current transmission rate, when the rate will be modified and what impact this change will have on the link quality. However, despite this strong dependency, historically these two problems have been studied separately, leading to suboptimal solutions [10]–[19].

This paper presents a new mechanism based on a joint approach for solving these two important problems in WMNs: automatic rate adaptation and routing metric assignment. The idea is to use a cooperative cross-layer method, called Metric Aware Rate Adaptation (MARA). With this method, rate adaptation decisions are based on statistics provided by the routing metric. Conversely, aware of the chosen rate, the metric may provide better estimates for link costs.

In addition to the main mechanism, two variations are also evaluated. The first one, named MARA-P, considers the impact of packet size on rate adaptation choices. Similarly, the second variation, called MARA-RP, considers the impact of packet size on both rate adaptation and routing metric choices.

To evaluate the performance of MARA and its variations, comparative experiments have been conducted in both simulated and real environments. In these experiments, MARA has been compared with every combination of a number of rate adaptation algorithms and routing metrics of the literature. The results show that MARA is consistently superior to these other proposals in terms of throughput and end-to-end delay, specially in scenarios with large number of hops.

II. RELATED WORK

Since MARA is a mechanism that combines both routing metrics and automatic rate adaption principles, there are three
lines of related work: routing metrics, rate adaptation algorithms and some recent joint approaches. In the next three sections we summarize some proposals in each of these lines.

A. Routing Metrics

In terms of routing metrics [9], the very first proposal is the Hop Count. This metric considers the best path to be the one with lowest number of hops. This approach, however, does not take into account the differences between wireless links, since it considers all network links to be equally good.

In practice the Hop Count metric does not perform well, because the quality of a wireless link depends on a number of factors, such as length and interference sources. Therefore, different wireless links tend to present different levels of quality.

Other proposals, known as quality-aware metrics, improve performance by dynamically evaluating characteristics of links. For instance, the Expected Transmission Count metric (ETX) [10] tries to estimate the number of layer-2 transmissions necessary to successfully transmit a packet between two nodes. The ETX metric works by periodically broadcasting control packets to infer a Packet Error Rate (PER) on every link. Since the transmission of a packet in the link-layer usually can be modeled using a geometric distribution (the sender repeats the transmission until it succeeds), the expected number of link-layer retransmissions is the reciprocal of the PER.

A number of proposals in the literature are based on the ETX PER estimation method, or even in the complete ETX proposal. The Expected Transmission Time metric (ETT) [11] tries to minimize the end-to-end delay, considering the cost of each link to be its ETX multiplied by the transmission delay of a packet using the links’ current transmission rate. The Minimum Loss metric (ML) [12] tries to minimize the end-to-end PER, by defining the link cost as its PER value. For deeper discussion on these and other metrics, please refer to [9].

Although these metrics take different approaches quantifying the quality of wireless links, they are all based on the same PER estimation method. This process retrieves statistical data from periodical broadcast probes, which, according to the 802.11 standard, are always sent at a robust rate. However, since nodes usually apply a rate adaptation algorithm, data packets are typically sent at higher rates in which packet error rates are possibly higher. Therefore, these metrics apply their models to possibly inaccurate statistics, leading to sub-optimal performance.

There are works in the literature that suggest using a variation of the ETT metric based on unicast probes [13]. In other words, each node has to send one control packet per neighbor to collect statistical data for computing PER. Despite yielding more precise statistics, this approach has serious scalability implications, especially on dense networks.

B. Rate Adaptation Algorithms

As with the routing metrics, there is a vast literature on rate adaptation algorithms. Perhaps the simplest and most widely adopted [19] mechanism is the Auto Rate Fallback (ARF) [14]. This algorithm keeps track of sequences of frame transmission successes and failures. If ARF detects that the number of consecutive failures has reached a given threshold (by default, 2), the current rate is decreased. Conversely, if the number of consecutive successes transpasses its given threshold (by default, 10), ARF increases the transmission rate.

The SampleRate algorithm [15] is based on delay. For every available rate, it keeps an statistic on the average frame transmission time in terms of the average number of retransmissions and the IEEE 802.11 protocols’ overhead, considering data from the past 10 seconds. If a rate has had 4 consecutive losses in this period, it is marked as unfeasible. Before each transmission, SampleRate loops through the feasible rates and chooses the one with lowest transmission time estimate.

The SNR (Signal-to-Noise-Ratio) algorithm is often considered to be optimal. The idea is to choose the highest possible rate in the transmitter such that the SNR in the receiver is high enough to decode the frame with an error probability lower than a given threshold. This idea, however, is not feasible in practice because it depends on future information. Nevertheless, the SNR algorithm is usually employed on simulation-based evaluations as a baseline. There are, also, other proposals which use simplified versions of the SNR algorithm, such as Receiver Based Auto Rate (RBAR) [16].

There is yet another class of rate adaptation algorithms targeted at differentiating frame losses caused by collisions from the ones caused by channel degradation. On dense environments under heavy traffic loads, nodes may experience an increasing number of frame collisions, which may cause automatic rate adaption algorithms to misinterpret losses as a sign of channel quality degradation. Among the proposals on this class, one can cite the Snoopy Rate Adaptation (SRA) [17], the Robust Rate Adaptation Algorithm (RRAA) [18] and the Collision-Aware Rate Adaptation (CARA) [19].

For further details on the subject, please refer to [20].

Although many proposals have been presented in the literature, they face the problem of dealing with small and non-uniform statistical samples. Since these proposals base their statistics on data packets, they all depend on the existence of network traffic. When there are no packets to be transmitted, these rate adaptation algorithms cannot collect information about the quality of a link. Given the fast variability of quality faced by wireless links, after some time the statistics available to these algorithms may not be a valid representation of the current link state. Therefore, when a new packet arrives to be transmitted, the algorithm needs a new convergence period to find the optimal rate. Moreover, proposals based on monitoring the packet error rate face the challenge of normalizing their statistics according to the size of each packet, since packets with different sizes have different loss probabilities. Usually, though, these proposals do not apply any normalization procedure.

C. Joint Approaches

To the best of our knowledge, only one joint approach has been proposed to this date. In [21], authors evaluate the
Multirate Anypath Routing Problem, in which they take into consideration links’ rates in the path choice. They propose a modification of the ETT routing metric, which computes a link cost for each available transmission rate. Since they work with anycast routing, the link’s PER is considered to be the probability that every single neighbour fails to correctly decode the packet.

The problem with this proposal is that the PER must be inferred at each available transmission rate using probes. Thus, the overhead associated with this solution is considerably higher than with the original ETT metric.

III. THE MARA PROPOSAL

The MARA (Metric-Aware Rate Adaptation) mechanism has two major components: the routing metric, which evaluates and assign costs for network links, and the rate adaptation, which chooses the most suitable transmission rate for each link. These two components share information and make coordinated decisions.

The metric component of MARA evaluates routes according to the expected end-to-end delay. To do so, it assigns each link a cost given by the following expression:

\[
MARA_{ab} = \min_i \left( \frac{ETX^i_{ab} \cdot ps}{R_i} \right),
\]

where \( R_i \) represents the \( i \)-th available transmission rate, \( ETX^i_{ab} \) is the ETX of the link \( a \rightarrow b \) using rate \( R_i \) and \( ps \) is the size of the probe packet used to infer ETX. The physical meaning of \( MARA_{ab} \) is the total transmission delay of the link \( a \rightarrow b \), considering all expected retransmissions.

The cost computed using Equation (1) is associated with a transmission rate \( R_i \), which is the best possible rate in the proposed model (i.e., the rate that minimizes the link transmission delay, considering the average number of transmissions). As such, the rate adaptation component of MARA selects \( R_i \) as unicast transmission rate of link \( a \rightarrow b \).

A. Link Quality Estimation

According to Equation (1), in order to compute the metric \( MARA_{ab} \) it is necessary to know the value of ETX for \( a \rightarrow b \) in every available transmission rate. Since the original formulation of ETX relies on broadcast probes, this value is only computed at one rate (the basic rate used in broadcast transmissions, according to the IEEE 802.11 Standard [22]). Therefore, MARA needs a different approach for computing ETX.

The most straightforward solution is to simply manipulate the transmission rate for broadcast frames, so that MARA can send the ETX probes at every available rate. This way, MARA would have statistical data in order to compute ETX in all rates. However, considering, for instance, the IEEE 802.11b/g mixed mode (widely used in commercial devices), there are twelve available rates. Hence, this strategy would considerably increase network overhead.

To avoid such an overhead increase, MARA adopts a different approach, shown in Algorithm 1. This approach is based on a process of conversion of the links success probabilities. This conversion happens in two steps:

1) the average SNR of the link is estimated using the information provided by probe packets;
2) the average SNR is used to estimate the link success probability in every rate, which is later used to compute ETX for each rate.

Both steps require the knowledge of a function which relates SNR and the success probability of a link. While defining a closed expression for such a function is not trivial, previous works have collected data through experiments and simulations for all transmission rates used in the IEEE 802.11b/g mixed mode [23], [24]. These data can be used to build a table relating four physical quantities: SNR, transmission rate, frame size and PER. Such table is used, for instance, by the DEI802.11-mr [25] module, an enhanced implementation of the IEEE 802.11 standard for the ns-2 simulator [26].

Using this pre-computed table, it is possible to infer the SNR of a link from the packet error rate computed with probe packets at the basic rate (as done by the ETX metric). The transmission rate and the frame size (probe size) are known from the transmission of the probe packets. The PER can be approximated by the value measured using the probes. Therefore, the table can be consulted in order to find a value for the SNR. Conversely, using this SNR estimate and choosing a target rate for a new table consult returns an estimate for the link’s PER.

Algorithm 1 Compute ETX at any rate for a link \( a \rightarrow b \).

Input: \( P_{ab}, P_{ba}, \) sourceRate, targetRate and probeSize.

Output: ETX for the link \( a \rightarrow b \) at targetRate.

1) Conversion Issues: The proposed method for converting link’s PER presents an issue. The function that relates SNR and PER (for a given rate and a given frame size) has an asymptotic behavior for both low and high values of SNR. As SNR increases, PER approaches 1. Conversely, as SNR...
decreases, the value of PER approaches 0. The extreme cases (PER equals to 0 or 1), however, are never reached, because, independently of how high (or low) the SNR is, there is always a chance of failure (or success).

Nevertheless, in practice, PER can reach values so close to 0 or 1 that it is not possible to estimate it with the necessary precision. Due to limitations in available memory and bandwidth, it is not feasible for routing protocols to use a very large sample of probes for computing PER. Most likely, the number of probes considered is in the order of hundreds of packets. In this case, if a link has a very low PER, for example, chances are no probes will be lost during the window considered by the routing protocol and PER will be estimated as 0. Therefore, for practical purposes, the function that relates SNR and PER is not injective, because for the extreme values of PER, there may be many associated SNR values. In other words, it is impossible to properly evaluate the SNR of a link if its estimated PER is equal to 0 (or 1).

To avoid this issue, MARA uses probe packets in four different transmission rates: 1 Mb/s, 18 Mb/s, 36 Mb/s and 54 Mb/s. These rates were chosen because, according to the data on the conversion table, their $PER \times SNR$ curves intersect on useful intervals. The points plotted in Fig. 1 show the data available on the conversion table for these four transmission rates. For instance, the lowest SNR value which results in a PER value of 0 for the 1 Mb/s transmission rate is associated to a PER lower than 1 for the 18 Mb/s transmission rate, as show in Fig. 1.

Algorithm 2 Selection of the most useful statistic.

Input: $lossProb$.
Output: $usedProb$ and $usedRate$.

if $lossProb[3] < 1$ then
  $usedProb \leftarrow lossProb[3]$
  $usedRate \leftarrow 54$ Mbps
else
  if $lossProb[2] < 1$ then
    $usedProb \leftarrow lossProb[2]$
    $usedRate \leftarrow 36$ Mbps
  else
    if $lossProb[1] < 1$ then
      $usedProb \leftarrow lossProb[1]$
      $usedRate \leftarrow 18$ Mbps
    else
      $usedProb \leftarrow lossProb[0]$
      $usedRate \leftarrow 1$ Mbps
  end if
end if
return $usedProb, usedRate$

When MARA has to compute the metric for a link, it first choses one of the four probabilities using Algorithm 2. In the code, $lossProb$ is an array, containing the link error rates estimated at each of the four transmission rates. This simple algorithm chooses as the most appropriated statistics for the current link the one associated with the higher probe rate, such that the PER is lower than 1. Therefore, extreme values (0 and 1) are avoided, improving the precision of the SNR estimate.

To provide a higher degree of scalability to the method, the periodical probes are sent in broadcast. This guarantees that the overhead does not increase with the number of neighbors. Moreover, instead of sending the probes of each rate all at once, MARA sends only one probe per period. In other words, in the first period MARA sends the probe at 1 Mb/s. Then, in the second period it uses 18 Mb/s and so forth. With this policy, the overhead is even lower than the one caused by the original formulation of the ETX metric, since probes at 18 Mb/s, 36 Mb/s and 54 Mb/s use the wireless medium for less time.

2) Table Limitations: The conversion table used by MARA associates four physical quantities: SNR, transmission rate, frame size and PER. Since this data is organized as a table, there are, at least, two limitations:

1) Two physical quantities stored in the table assume continuous values: the SNR and the PER. Hence, to represent all possible values of both physical quantities, the table would require an infinite number of entries. Since this is impossible, there must be a minimal granularity for these two physical quantities.

2) If the table has a low granularity, it can become too extensive, causing the consults to be computationally expensive.

To avoid both limitations, MARA does not use directly the table, but instead a set of functions we propose for interpolating data contained in the table. In this work, these functions were derived from the values in the table as follows. For each combination of transmission rate and frame size on the original table, we computed the values of the parameters $\alpha$ and $\beta$ that best fit the curve:

$$PER = \frac{1 - erf \left( \frac{SNR - \alpha}{\beta \cdot \sqrt{2}} \right)}{2},$$  \hspace{1cm} (2)

where $erf(x)$ denotes the Error Function. Table I shows the values of $\alpha$ and $\beta$ found for each combination of transmission

![Fig. 1: $PER \times SNR$ curves for 1 Mb/s, 18 Mb/s, 36 Mb/s and 54 Mb/s (for 1500 byte packets). Comparison between actual data and curve fitting.](image-url)
rate and frame size of the original table. Fig. 1 compares the original data and the obtained curves for four different rates, considering 1500-byte packets. Even tough this work considers the transmission rates available for the IEEE 802.11b/g standard, we argue that the method described here can be applied to transmission rates available in other standards such as IEEE 802.11n.

B. Variations

The expression used by MARA to define the cost of a link, presented in Equation (1), is parametrized by the size of the packet. The proposal presented so far supposes this value is always a constant, namely the probe packet size. However, the size of data packets transmitted by each node varies according to the network traffic. Therefore, the following questions arise:

1) Is the optimum transmission rate dependent on the size of the packet we wish to transmit?
2) Does the optimum route depend upon the size of the packet we wish to transmit?

In order to answer these questions, in this paper we propose two variations of MARA: MARA-P and MARA-RP. In the first variation, MARA-P, we consider the size of packets in order to choose the best rate, whereas in the second, MARA-RP, the packet size is also taken into consideration for choosing the best route. Notice again that packets with different sizes present different PER.

1) MARA-P: The idea of MARA-P is to compute the optimal rate for each link (according with MARA’s mathematical model), considering the size of the packet to be transmitted. This can be done either online (whenever a packet has to be transmitted through a link, the best rate is computed) or offline (the best rate for each possible size of computed periodically and stored in a table for later consults). Both options are computationally expensive: the first, in terms of time, and the second, in terms of space.

An alternative, adopted by MARA-P, is to take the offline approach, but computing and storing the rates only for a small set of packet sizes. MARA-P defines classes of packet sizes (for instance, from 1 to 300 bytes) and computes the best rate for the highest value of the interval using Equation (1). MARA-P considers this rate to be also the best rate for every other size within the same class. Whenever a packet has to be transmitted through a link, MARA-P finds the packet size class to which the packet belongs and consults the rate table.

For assigning the link cost, MARA-P proceeds exactly as the original proposal of MARA. In other words, it computes the expression of Equation (1) considering the size of the probe packet.

2) MARA-RP: The rate selection algorithm of MARA-RP works exactly as the one used by MARA-P. The difference of this variation lies on the cost assignment process. While MARA and MARA-P use the probe packet size as a constant value for Equation (1), MARA-RP computes the expression for every packet size class (using the highest value of the interval as the packet size, just as MARA-P does for rate selection).

The result of this approach is a set of topology graphs, i.e., if the number of defined packet size classes is \( k \), then there are \( k \) different topology graphs, one for each class of packet sizes. Applying a shortest path algorithm to each graph results in \( k \) routing tables. Whenever a node has to decide how to forward a packet, it first decides to which size class it belongs and then consults the appropriate routing table.

IV. IMPLEMENTATION ISSUES

Two different implementations of MARA and its variations were developed in this work, for evaluation purposes. The first implementation was in the form of a module for the ns-2 simulator [26], whereas the second was a practical implementation based on the OpenWRT Linux distribution [27]. In both cases, the following packet size classes were defined for MARA-P and MARA-RP: \([1, 300], [301, 750], [751, 1300], [1301, 1520] \]. This section gives details on both implementations, discussing some of the challenges found during the development.

A. Simulation Environment

The implementation developed for simulation purposes is based on the well-known ns-2 simulator [26]. The metric component of MARA and its variations were implemented as an extension of the OLSR protocol [28], which runs throughout the simulations, dynamically choosing the best routes (according to each evaluated metric). OLSR is a proactive protocol developed for ad hoc networks. Specifically, our implementation is based on the code developed by [29] and modified by [30]. In addition to MARA, the ML and ETT metrics were also implemented. The Hop Count and ETX metrics were already part of the original code. In this work we implement the variation of ETT proposed in [13]. For all metrics, except for Hop Count (which does not need this information), the link delivery ratios are estimated using OLSR Hello packets as probes, as suggested by [10]. These packets are sent every 2 seconds and the last 25 packets are considered for computing the ratios. The metric for each link is recomputed at every change in the value of link delivery ratio. Every 5 seconds, OLSR sends a Topology Control packet, containing the metric for all links of the node. Routes are recomputed upon the reception of this kind of packet.

One issue of using OLSR as a routing protocol when evaluating routing metrics is the use of Multi-Point Relays (MPR). The MPRs of a node \( a \) are defined as a set of direct neighbors through which \( a \) can reach each 2-hop neighbor (in at most two hops). OLSR uses the concept of MPR to reduce the overhead associated with the diffusion of control messages, increasing the scalability of the protocol. However, as a side effect of its MPR usage, OLSR may not find the optimal network paths, when using metrics other than Hop Count. The problem happens because the topology graph generated by OLSR contains only links between nodes and its MPRs. If the used metric is Hop Count, using only these links is enough for finding the optimal paths. However, when using a different metric, this property does not hold. Therefore, allowing OLSR to use its MPR mechanism would not be fair for evaluating routing metrics, because links with good quality which might
be used in one or more paths may be discarded before the shortest path selection. With this in mind, we disabled the usage of MPRs by OLSR during our simulations.

Since MARA also has a rate selection component, the original ns-2 implementation of the IEEE 802.11 standard was not applicable to our evaluation, because it does not allow dynamic rate adaptation nor it considers the effect of different rates in PER. To cope with such needs, in this work we adopted the DEI-802.11-mr module [25] instead. The DEI-802.11-mr implementation not only allows dynamic rate adaptation, but also includes better models for wireless medium phenomena such as self-interference and capture effect.

### B. Real Environment

The implementation of MARA in an real environment was developed using one of the wireless mesh networks of the ReMoTE project [7]. This network is composed by Linksys WRT54G commercial routers using a customized firmware based on the OpenWRT Linux distribution [27].

Since MARA manipulates both rate adaptation and routing information, its implementation has been separated in two modules: a routing module and a rate selection module. The routing module is responsible for collecting information about links and assigning them costs. This module also selects the optimal transmission rates for each link and informs them to the rate selection module. In turn, the rate selection module stores the optimal rates for each link using a table and analyzes each packet before transmission to configure the wireless interface to the proper rate.

The metric component of MARA can be implemented over any routing protocol based on link state or distance vector. Although there are many protocols optimized for multihop wireless networks, in this work we chose to develop a simple protocol, called SLSP (Simple Link State Protocol), which just implements the basic functionalities of the link state algorithm. This protocol was written in C and implements the metrics Hop Count, ML, ETX, besides MARA and its variations.

The goal of the rate selection module is to create a communication interface between the routing protocol and the wireless interface, in order to provide the necessary functionalities for MARA. This module is needed due to the configuration inflexibility of the drivers for the wireless network interfaces available in the market. For instance, on most interfaces, it is not possible to configure different transmission rates for different neighbors. Specifically, that is the case with the routers used in our testbed. There is not a mechanism on the interface driver which allows SLSP to inform the transmission rate for each neighbor. In fact, the driver only allows the specification of one transmission rate.

The solution found to this issue was the creation of a Loadable Kernel Module (LKM) for the Linux Kernel. This LKM, called PPRS (Per-Packet Rate Selection), maintains a table of transmission rates and monitors the packet transmission process within the kernel. Whenever a packet is about to be sent to the network interface, PPRS consults the table looking for a match for the current packet’s parameters (e.g., next hop and packet size). PPRS then requests the network interface to change its transmission rate to the one matching the packet.

Fig. 2 gives an overview of the complete implementation. When PPRS is loaded, it creates two files in the \(/proc/pprs\) directory. On Linux, files in this directory work as a communication channel between the user space and the kernel space. Specifically for PPRS, these two files allow the definition of packet size classes and rules for choosing transmission rates. These definitions are done by simply writing to \(/proc\) files (this is done by SLSP). Besides creating the \(/proc\) files, PPRS also intercepts packets in their regular transmission path within the wireless interface and puts itself as an intermediary. Using this strategy, PPRS is able to process packets exactly before they are transmitted by the network interface and to configure

### Table I: Values of the parameters \(\alpha\) and \(\beta\) obtained for the curve fitting.

<table>
<thead>
<tr>
<th>Size (B)</th>
<th>(\alpha)</th>
<th>(\beta)</th>
<th>(\alpha)</th>
<th>(\beta)</th>
<th>(\alpha)</th>
<th>(\beta)</th>
<th>(\alpha)</th>
<th>(\beta)</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>0.34</td>
<td>0.26</td>
<td>0.19</td>
<td>0.21</td>
<td>0.14</td>
<td>0.32</td>
<td>0.11</td>
<td>0.62</td>
</tr>
<tr>
<td>256</td>
<td>2.96</td>
<td>0.01</td>
<td>2.78</td>
<td>0.60</td>
<td>3.15</td>
<td>0.66</td>
<td>3.54</td>
<td>0.59</td>
</tr>
<tr>
<td>512</td>
<td>5.48</td>
<td>0.83</td>
<td>5.96</td>
<td>0.78</td>
<td>6.42</td>
<td>0.76</td>
<td>6.85</td>
<td>0.70</td>
</tr>
<tr>
<td>1024</td>
<td>-0.71</td>
<td>1.07</td>
<td>-0.34</td>
<td>1.07</td>
<td>0.04</td>
<td>1.08</td>
<td>0.37</td>
<td>1.10</td>
</tr>
<tr>
<td>1500</td>
<td>3.01</td>
<td>1.70</td>
<td>3.44</td>
<td>1.71</td>
<td>3.81</td>
<td>1.74</td>
<td>4.14</td>
<td>1.75</td>
</tr>
<tr>
<td>122</td>
<td>8.99</td>
<td>1.03</td>
<td>8.68</td>
<td>0.98</td>
<td>9.25</td>
<td>0.96</td>
<td>9.83</td>
<td>1.00</td>
</tr>
<tr>
<td>122</td>
<td>2.25</td>
<td>1.10</td>
<td>2.64</td>
<td>1.10</td>
<td>3.03</td>
<td>1.10</td>
<td>3.39</td>
<td>1.12</td>
</tr>
<tr>
<td>122</td>
<td>6.03</td>
<td>1.70</td>
<td>6.38</td>
<td>1.71</td>
<td>6.81</td>
<td>1.76</td>
<td>7.23</td>
<td>1.77</td>
</tr>
<tr>
<td>122</td>
<td>7.64</td>
<td>1.16</td>
<td>8.10</td>
<td>1.15</td>
<td>8.55</td>
<td>1.18</td>
<td>8.93</td>
<td>1.17</td>
</tr>
<tr>
<td>36</td>
<td>11.92</td>
<td>1.67</td>
<td>12.41</td>
<td>1.70</td>
<td>12.85</td>
<td>1.75</td>
<td>13.21</td>
<td>1.78</td>
</tr>
<tr>
<td>48</td>
<td>15.04</td>
<td>1.30</td>
<td>15.55</td>
<td>1.28</td>
<td>16.00</td>
<td>1.27</td>
<td>16.40</td>
<td>1.28</td>
</tr>
<tr>
<td>54</td>
<td>16.97</td>
<td>1.62</td>
<td>17.40</td>
<td>1.63</td>
<td>17.96</td>
<td>1.66</td>
<td>18.43</td>
<td>1.69</td>
</tr>
</tbody>
</table>

![Diagram](image-url)
its transmission rate. Once the correct rate is configured on the interface, PPRS puts the packet back on the normal transmission path in order to complete the process, as shown in the lower part of Fig. 2.

Another challenge faced when implementing MARA was the MARA-RP variation. This variation generates multiple routing tables, one for each packet size class. Fortunately, such feature is available in the advanced routing implementation of the Linux Kernel [31]. To use multiple routing tables, it is necessary to create routing rules which associate packet characteristics to each table. Among the possible characteristics is the logical mark of the packet (a field internal to the kernel of the local node), which can be created through the iptables tool [32].

In summary, MARA-RP’s implementation works as follows. SLSP defines the packet size classes it will use and informs them to PPRS by writing to the file /proc/pprs_classes (see Fig. 2). It then computes the routing tables and inform them to the Linux Kernel using system calls. After that, iptables rules are created to mark the packets before the routing decisions, according to their size. Also, routing rules are created by SLSP, associating the marks with their respective routing tables. These rules are informed to PPRS, through the file /proc/pprs. After these configurations, the operating system takes charge of doing the correct forwarding.

V. PERFORMANCE EVALUATION

In this section, we present a performance evaluation of MARA and its variations. This evaluation was conducted on both simulated and real environments, using the implementations discussed in Section IV.

The evaluation methodology consists in comparing MARA with different combinations of routing metrics and rate adaptation algorithms from the literature. Each experiment consists of a 300 seconds TCP flow between a specific pair of nodes and the performance metrics considered were throughput, end-to-end delay and end-to-end packet loss.

The motivation behind conducting both real and simulated experiments is to take advantage of the qualities of both methods. Simulations are completely reproducible, very flexible and make it possible to extract information about every network event. On the other hand, a real environment provides all the complexity of wireless communication systems and one can trust its behaviors are not a result of simulation artifacts. With this in mind, we opted for including in our simulations a topology modeled after the real testbed.

A total of five topologies representing different kinds of scenarios (with different characteristics) were used during the simulations. Because of space restrictions, though, only the most relevant results are presented here.

A. Simulation Results

During the simulations, MARA was compared with all possible combinations of the following routing metrics and automatic rate adaptation algorithms:

- Routing metrics: Hop Count, ETX, ML and ETT.
- Automatic rate adaptation algorithms: ARF, SampleRate and SNR.

The first simulation topology used, hereinafter referred to as Indoor Topology, is a representation of a real mesh network, based on the indoor mesh network of the ReMoTE project. The network is composed of 10 nodes placed in rooms of a building, as shown in Fig. 3. The actual network was also used for the real experiments (see Section V-B).

In this topology, the destination for the TCP flow was always node 0. In the actual network, this node works as a gateway for the Internet. The source of the flow was varied from nodes 1 to 9. For each pair of nodes, the experiment was repeated 6 times. The 95% confidence interval is plotted in all graphs.

Fig. 4 shows throughput results for the indoor topology for three different sources: nodes 1, 6 and 9. As expected, as the geographic distance between source and sink nodes increases, the throughput decreases. For the closest destinations, MARA’s performance is good, but very close to the other proposals. Indeed, when the destination is node 1, 10 out of the 13 combinations had quite comparable performances. When the destination is node 6, the number of combinations with similar performances drops to 4. For destinations farther than node 6, MARA’s performance decreases much slower than the others. When the destination is node 9, this difference becomes evident, with MARA achieving 4 times over the throughput of any of the other proposals. Throughout all our topologies, this tendency was verified: MARA’s performance increases (with respect to the other proposals) with the increase of the geographic distance between source and sink nodes. As we select farther destination nodes, we increase the complexity of rate adaptation and routing metric problems, since more alternative paths and heterogeneous links are available. With diversity, the advantages of MARA are accentuated.

In summary, MARA-RP’s implementation works as follows. SLSP defines the packet size classes it will use and informs them to PPRS by writing to the file /proc/pprs_classes (see Fig. 2). It then computes the routing tables and inform them to the Linux Kernel using system calls. After that, iptables rules are created to mark the packets before the routing decisions, according to their size. Also, routing rules are created by SLSP, associating the marks with their respective routing tables. These rules are informed to PPRS, through the file /proc/pprs. After these configurations, the operating system takes charge of doing the correct forwarding.

V. PERFORMANCE EVALUATION

In this section, we present a performance evaluation of MARA and its variations. This evaluation was conducted on both simulated and real environments, using the implementations discussed in Section IV.

The evaluation methodology consists in comparing MARA with different combinations of routing metrics and rate adaptation algorithms from the literature. Each experiment consists of a 300 seconds TCP flow between a specific pair of nodes and the performance metrics considered were throughput, end-to-end delay and end-to-end packet loss.

The motivation behind conducting both real and simulated experiments is to take advantage of the qualities of both methods. Simulations are completely reproducible, very flexible and make it possible to extract information about every network event. On the other hand, a real environment provides all the complexity of wireless communication systems and one can trust its behaviors are not a result of simulation artifacts. With this in mind, we opted for including in our simulations a topology modeled after the real testbed.

A total of five topologies representing different kinds of scenarios (with different characteristics) were used during the simulations. Because of space restrictions, though, only the most relevant results are presented here.

A. Simulation Results

During the simulations, MARA was compared with all possible combinations of the following routing metrics and automatic rate adaptation algorithms:

- Routing metrics: Hop Count, ETX, ML and ETT.
- Automatic rate adaptation algorithms: ARF, SampleRate and SNR.

The first simulation topology used, hereinafter referred to as Indoor Topology, is a representation of a real mesh network, based on the indoor mesh network of the ReMoTE project. The network is composed of 10 nodes placed in rooms of a building, as shown in Fig. 3. The actual network was also used for the real experiments (see Section V-B).

Besides modeling nodes’ positions within the topology, we also conducted an adjustment phase to find the values for the parameters of the propagation model (the Shadowing model, in this case) which best represent the actual network performance. Table II summarizes the parameters used in this work.

TABLE II: Parameters for the Shadowing propagation model used during the simulations.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path Loss Exponent</td>
<td>2.15</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>1.4</td>
</tr>
<tr>
<td>Reference Distance</td>
<td>0.5</td>
</tr>
</tbody>
</table>

In this topology, the destination for the TCP flow was always node 0. In the actual network, this node works as a gateway for the Internet. The source of the flow was varied from nodes 1 to 9. For each pair of nodes, the experiment was repeated 6 times. The 95% confidence interval is plotted in all graphs.

Fig. 4 shows throughput results for the indoor topology for three different sources: nodes 1, 6 and 9. As expected, as the geographic distance between source and sink nodes increases, the throughput decreases. For the closest destinations, MARA’s performance is good, but very close to the other proposals. Indeed, when the destination is node 1, 10 out of the 13 combinations had quite comparable performances. When the destination is node 6, the number of combinations with similar performances drops to 4. For destinations farther than node 6, MARA’s performance decreases much slower than the others. When the destination is node 9, this difference becomes evident, with MARA achieving 4 times over the throughput of any of the other proposals. Throughout all our topologies, this tendency was verified: MARA’s performance increases (with respect to the other proposals) with the increase of the geographic distance between source and sink nodes. As we select farther destination nodes, we increase the complexity of rate adaptation and routing metric problems, since more alternative paths and heterogeneous links are available. With diversity, the advantages of MARA are accentuated.
this by sacrificing the packet loss rate. Therefore, MARA could keep a good balance between these two performance metrics.

Fig. 6: Grid and Random Topologies used during simulations.

Another topology used in our simulations was the Grid Topology, depicted in Fig. 6a. It is composed of 49 nodes, arranged in a square grid. A variation of this topology is the Random Topology, shown in Fig. 6b. In this topology we kept the same number of nodes and the total area (a square with 120 meters of side), but chose randomly the position of each node.

Fig. 7: Throughput for the TCP flows between the most distant nodes of the Grid and Random Topologies.

The simulation results obtained in these topologies, shown in Fig. 7, were very similar to those obtained in the Indoor Topology. For instance, in the Grid Topology (Fig. 7a), using a TCP flow between nodes 0 and 48 (the white nodes in Fig. 6a), MARA’s throughput was 8 times better than the second best combination. The analysis of the routes chosen by each routing metric yielded new interesting insights, though. The most frequent route chosen by each metric is shown in Fig. 8. The ETT and ML metrics opted for very long paths, which increases the amount intra-flow interference. The poor performance of ETT in this topology, however, is due not only to the choice for long paths, but mainly to the overhead needed for measuring
link qualities in all transmission rates, as proposed in [13]. MARA, ETX and Hop Count chose shorter paths, along the grid diagonal. The difference between the three choices was the length of the selected links: MARA selected shorter links than the other two proposals. The selection of these shorter links (which have better quality in terms of SNR) along with the joint rate adaptation selection allowed MARA to perform better than the other proposals.

Besides comparing MARA with other proposals in the literature, we also performed simulations to compare MARA and its variations. To do so, we substituted the TCP flow with three CBR (Constant Bit Rate) flows, modeling audio, video and a background traffic. The choice for these three traffic models is due to their different characteristics, in terms of packet sizes. The audio traffic is configured to use small packets (120 bytes at a rate of 48 Kb/s), while the video traffic uses medium size packets (900 bytes sent every 40 ms) and the background has the largest packets (1400 bytes sent every 22.4 ms). Hence, we could observe how MARA-P and MARA-RP react in the presence of packets of different size classes.

![Fig. 8: Most frequently chosen routes by each metric in the Grid Topology.](image)

Fig. 8: Most frequently chosen routes by each metric in the Grid Topology.

![Fig. 9: End-to-end delay and packet loss rate results for the audio flow, obtained in the Random Topology.](image)

Fig. 9: End-to-end delay and packet loss rate comparison between MARA, MARA-P and MARA-RP in the Random Topology.

![Fig. 10: End-to-end delay between nodes 0 and 3 of the Indoor Topology: comparison between MARA-P and MARA.](image)

Fig. 10: End-to-end delay between nodes 0 and 3 of the Indoor Topology: comparison between MARA-P and MARA.

B. Practical Results

To validate our simulation results, we conducted a series of experiments in a real testbed using the implementation described in Section IV. This experiments took place in a topology depicted in Fig. 3 (after which the Indoor Topology used during the simulations was modeled). The real experiments used a methodology very similar to the one used in the simulations. The only difference was that an ICMP flow (using the ping tool) was added to each experiment to provide a measurement of the Round Trip Time (in replacement of the one-way delay of the main flow, which is difficult to measure in practice). Another difference between the practical and simulated experiments is in terms of compared proposals. In the practical experiments we could only compare MARA to combinations of the metrics ETX, ML and Hop Count with the rate adaptation algorithm ARF, because we did not have access to the source code of the wireless interface driver of our routers. Therefore, it was not possible to implement other rate adaptation proposals.
Fig. 11: Throughput and RTT of the TCP and ICMP flows between nodes 0 and 9 of the real testbed.

Fig. 11 shows the RTT (Round Trip Time) and the throughput between nodes 0 and 9. As in the simulations, MARA performed better than the other proposals. It achieved roughly twice the throughput of the second best proposal. In terms of RTT, MARA performed almost three times better. By repeating the experiment varying the sink node (from 1 to 9), we noticed the same trend observed in the simulations. As the distance from source to sink increases, the relative performance of MARA (with respect to the other proposals) also increases.

Fig. 12: Comparison between MARA, MARA-P and MARA-RP in the real testbed: RTT for the audio flow.

Fig. 12 shows the average RTT for the audio flow between nodes 0 and 9. Differently from what happened in the simulations, MARA-P and MARA-RP had different results. MARA-P was slightly better than the original MARA, while MARA-RP had the worst result. Considering the confidence intervals, nevertheless, it is not possible to state that MARA-P was definitively superior. For other pairs of nodes (sources and sinks) the results of MARA and MARA-P were also very close (which is consistent with the simulations). Besides presenting the worst result, it is interesting to notice that MARA-RP also had the highest variability. Although in the simulations MARA-P and MARA-RP were equivalent, the processing overhead of MARA-RP is much higher (which is not considered by ns-2 simulations). Hence, the processing overhead affects both the average performance and its variation.

VI. CONCLUSIONS

In this paper we presented MARA, a joint approach to the problems of automatic rate adaptation and routing metrics. Although these two problems have been frequently considered separately, we argue that they are closely related and dependent. Moreover, we propose a method of PER conversion among different transmission rates of a given link. This method allows MARA to obtain accurate statistics about each link, while maintaining low overhead. In addition to proposing MARA, we also present two possible variations, MARA-P and MARA-RP, which take into consideration the size of the packet to be transmitted in the rate and route selection decisions.

Our simulations show that MARA is consistently superior to several combinations of routing metrics and automatic rate adaptation algorithms of the literature in various topologies. The simulation results also show that MARA can maintain a good balance between end-to-end delay and packet loss rate, and that the performance of MARA, relative to the other proposals, increases with the distance between source and sink nodes. At the most extreme example, MARA was 8 times better than the second best proposal. With respect to the variations, in our simulations they showed little or no benefit. In most cases, MARA-P and MARA-RP were not considerably superior to MARA. More interestingly, perhaps, is the fact that MARA-P and MARA-RP were completely equivalent in all our simulations.

To validate our simulation results, we present an implementation of MARA and its variations in a real wireless mesh network composed by 10 nodes. With this implementation, we reproduced our simulations, obtaining results very similar to those of our simulations. Again, MARA was considerably superior to other proposals (two times better, in terms of throughput and 3 times better, in terms of delay, considering the two most distant nodes). One difference between simulations and practical results was the performance of MARA-RP. While in the simulations, MARA-RP was totally equivalent to MARA-P, during the practical experiments, MARA-RP’s processing overhead resulted in low performance and high variability.

Future work includes a deeper analysis (using analytical methods) of MARA-P and MARA-RP to determine whether the two proposals are indeed equivalent (as suggested by our simulation results). This analysis may also help to understand the little benefit obtained with these variations, with respect to the original MARA proposal. We also intend to improve the cost model of MARA, by taking into consideration propagation effects, intra-flow interference and inter-flow interference.

REFERENCES


Diego Passos  Diego Passos received the B.S. M.S degrees in computer science from Fluminense Federal University (UFF), Rio de Janeiro, Brazil, in 2007 and 2009. Currently, he is a D.S. student at UFF. His major research interests are multihop wireless networks, network coding, and wireless routing.

Célio V. N. Albuquerque  Célio V. N. Albuquerque (S’94–M’00) received the B.S. and M.S. degrees in electrical and electronics engineering from Universidade Federal do Rio de Janeiro, Brazil, in 1993 and 1995, and the M.S. and Ph.D. degrees in information and computer science from the University of California at Irvine in 1997 and 2000, respectively. From 2000 to 2003, he served as the networking architect for Magis Networks, designing high-speed wireless medium access control. Since 2004 he has been an Associate Professor at the Computer Science Department of Universidade Federal Fluminense, Brazil. His research interests include Internet architectures and protocols, wireless networks, multicast and multimedia services, and traffic control mechanisms.