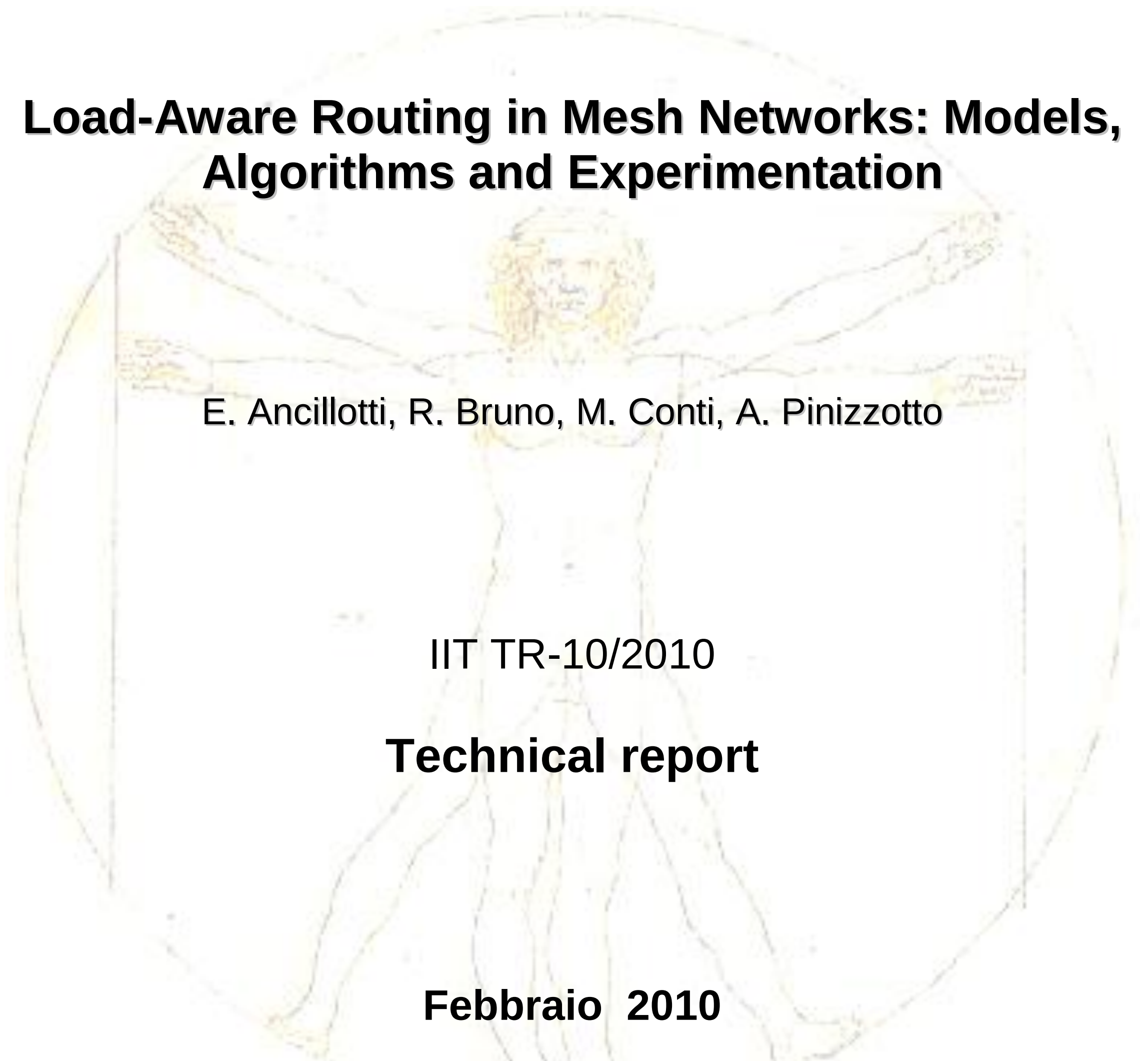
Consiglio Nazionale delle Ricerche



Load-Aware Routing in Mesh Networks: Models, Algorithms and Experimentation

E. Ancillotti, R. Bruno, M. Conti, A. Pinizzotto

IIT TR-10/2010

Technical report

Febbraio 2010

Istituto di Informatica e Telematica

Load-Aware Routing in Mesh Networks: Models, Algorithms and Experimentation

Emilio Ancillotti[a], Raffaele Bruno[a], Marco Conti[a], Antonio Pinizzotto[a]

[a]*Italian National Research Council (CNR)*
Institute for Informatics and Telematics (IIT),
Via G. Moruzzi 1, 56124 Pisa - ITALY

Abstract

In this paper we consider wireless mesh networks (WMNs) used to share the Internet connectivity of sparsely deployed fixed lines with heterogeneous capacity, ranging from ISP-owned high-speed links to subscriber-owned low-speed connections. If traffic is routed in the mesh without considering the load distribution and the bandwidth of Internet connections, some gateways may rapidly get overloaded because they are selected by too many mesh nodes. This may cause a significant reduction of the overall network capacity. To address this issue, in this paper we firstly develop a queuing network model that predicts the residual capacity of network paths, and identifies network bottlenecks. By taking advantage of this model, we design a novel Load-Aware Route Selection algorithm, named LARS, which improves the network capacity by allocating network paths to upstream Internet flows so as to ensure a more balanced utilization of wireless network resources and gateways' Internet connections. Using simulations and a prototype implementation, we show that the LARS scheme significantly outperforms the shortest-path first routing protocol using a contention-aware routing metric, providing up to 240% throughput improvement in some network scenarios.

Key words: Wireless mesh networks, routing protocols, queuing models, performance evaluation, test-bed and experimentation.

1. Introduction

802.11-based wireless mesh networks are emerging as a key technology to provide cost-effective ubiquitous access to the Internet [1]. Normally, in mesh networks only a subset of routers, referred to as *gateways*, has a high-speed Internet connection, while Internet access is shared among all the other mesh nodes by exploiting the ad hoc routing capabilities of the mesh routers [2, 3]. However, this vision is rapidly changing. Real-world mesh networks have been recently deployed, which are used to share a potentially large number of low-speed Internet connections (i.e., DSL fixed lines) available at the customers' premises. Examples of such networks are Meraki-based deployments in urban areas [4], or the Ozone's network in Paris, which is composed of 400 mesh routers, most of them using standard DSL links as Internet backhaul, while only ten gateways are provided with an ISP-owned fiber link [5]. In a broader sense, wireless mesh networks are evolving into a *converged infrastructure* used to share the Internet connectivity of sparsely deployed fixed lines with *heterogeneous capacity*, ranging from ISP-owned broadband links to subscriber-owned low-speed connections [6].

Being mesh networks primarily used for Internet access, both traffic routing and Internet gateway selection play a crucial role in determining the overall network performance, and in ensuring the optimal utilization of the mesh infrastructure [7, 8]. For instance, if too many mesh nodes select the same gateway

Email addresses: `emilio.ancillotti@iit.cnr.it` (Emilio Ancillotti), `raffaele.bruno@iit.cnr.it` (Raffaele Bruno), `marco.conti@iit.cnr.it` (Marco Conti), `antonio.pinizzotto@iit.cnr.it` (Antonio Pinizzotto)

as egress point to the Internet, congestion may increase excessively on the wireless channel, or the Internet connection of the gateway can get overloaded. This is especially important in the heterogeneous mesh networks we consider in this work, because low-speed Internet gateways may easily become a *bottleneck*, limiting the achievable capacity of the entire network. In addition a load-unaware gateway selection can lead to an unbalanced utilization of the gateways' backhaul links, and, eventually, to the underutilization of network resources.

To improve load balancing and increase capacity of WMNs, previous studies suggested to use balanced tree structures rooted at the gateways, and to route the traffic along the tree paths. For instance, an heuristic for calculating load-balanced shortest path trees taking into account flow load is proposed in [9]. In [10], approximated solutions are defined for calculating load-balanced trees that allocate the same bandwidth to all the nodes, using both single-path and multi-path approaches. An alternative strategy is proposed in [11], where the complexity of finding optimal routes is mitigated by considering only delay optimal routing forests, i.e., unions of disjoint trees routed at the gateway nodes. However, tree-based routing structures are less reliable to link failures than mesh-based structures. Furthermore, the admission of a new flow usually triggers complex reconfiguration procedures for the entire tree.

A simpler approach to improve network performance is to define routing metrics for traditional shortest-path first routing protocol capable of discovering high-throughput paths and/or facilitating load balancing. Initially proposed metrics (e.g., ETX [12] and ETT [13]), focused only on link characteristics (e.g., frame loss and transmission rates), and they do not balance the load. Recent studies proposed to introduce in the metric computation estimates of inter-flow and intra-flow interference (e.g., IRU [14]), location-dependent contention (e.g., CATT [15], ETP [11]) or load-dependent cost (such as the queue length in WCETT-LB [16], or the number of per-link admitted flows in LAETT [17]). Although these metrics have been demonstrated to work quite well in mesh networks, and to provide higher throughput performance than simple hop count, they are completely unaware of the available resources at the gateways. On the contrary, significant performance improvements might be obtained by considering residual capacity of gateways' Internet connections, as well as load distributions, when routing traffic flows. However, there is a complex interdependence between the way traffic flows are routed in the network and the utilization of network resources, which makes quite difficult to define simple heuristics to estimate the remaining capacity of a network path or a gateway.

To address this problem, in this paper we make the following two main contributions. First of all, we develop a queuing-based model of an heterogeneous mesh network, which incorporates the interdependencies between packet loss rates at the physical layer, random access MAC protocol, traffic routing and load distribution. This model is used to estimate the network capacity, and to identify network bottlenecks, due to either congestion on the wireless channels or overloading of Internet fixed lines. Then, we propose a novel *Load-Aware Route Selection* algorithm, named *LARS*, which integrates traffic routing with gateway selection. The goal of LARS is to improve network capacity, and to avoid underutilization of gateways' resources. The idea behind the design of the LARS algorithm is to allow each mesh node to distribute the traffic load among multiple gateways to ensure evenly utilization of Internet connections. To this end, mesh nodes select the routes towards the gateways taking into account the residual capacity of the paths, and the utilization of the gateways' fixed lines. We exploit the proposed queuing model to predict the residual capacity of each network path, and to discard paths or gateways that cannot accept additional demands.

It is important to point out that previous studies have proposed to use queuing models to investigate system performance of CSMA-based ad hoc networks. However most of these studies have applied queuing theory to the analysis of *single-hop* ad hoc networks [18, 19, 20]. To the best of our knowledge, in literature a few examples exist which deal with the multi-hop case. In [21], the authors model random access multi-hop wireless networks as open $GI/G/1$ queuing networks to analyze the average end-to-end delay and maximum achievable per-node throughput. However, the formulation proposed in [21] can be applied only to random networks, and it does not incorporate flow-level behaviors. Our objective is different from [21], because we consider arbitrary topologies and routing strategies, and we focus on per-flow performance. Previous papers [22, 23] have also developed queuing models to analyze the network capacity of heterogeneous WMNs. This paper extends those analytical studies to incorporate packet losses in the channel modeling.

We evaluate the performance gains provided by the LARS scheme over the shortest-path first routing algorithm using a contention-aware routing metric performing both simulations under large-scale network scenarios, and experiments in a realistic 5-node outdoor mesh network. Our simulations demonstrate the accuracy of the proposed modeling framework over a wide range of network settings. Furthermore, the numerical results show that proposed route and gateway selection algorithms significantly outperforms the shortest-path first routing using a contention-aware routing metric, providing up to 240% throughput improvement in some network scenarios. We have also implemented a prototype of the LARS solution as a proof-of-concept. The small-scale experiments conducted in our trial mesh network confirm the performance gains with respect to the shortest-path first routing protocol, and they demonstrate the practicality and feasibility of using load-aware route and gateway selection in WMNs.

The remaining of this paper is organized as follows. Section 2 introduces the overall system architecture. Section 3 develops the capacity analysis for a multi-hop heterogeneous mesh network. In Section 4, we describes the LARS algorithm. Section 5 validate the analysis and evaluate the performance gains of proposed algorithms using both simulations and a prototype implementation in a realistic mesh network. Finally, conclusions and future extensions are discussed in Section 6.

2. System architecture overview

Figure 1 illustrates the reference network architecture adopted in this work. Differently from traditional mesh architectures, which generally assume a limited number of mesh gateways connected to the Internet through unlimited-bandwidth fixed links, we recognize that mesh gateways can have connections to the speed with highly heterogeneous capacity. Specifically, there are a few gateways that have an high-speed backhaul connection (e.g., optical fiber or fixed broadband wireless) to the Internet. Since such high-capacity links are usually installed by network providers, we refer to this category of gateways as *provider gateways*. However, there might be a number of mesh routers directly owned by mesh network subscribers, which can share their low-speed Internet connection (e.g., ADSL cable) with other mesh users. Thus, we refer to this second category of mesh gateways as *residential gateways*. On the user side, wireless clients have a direct wireless connection to one mesh router, which acts as aggregator point for the user-generated traffic.

In this architecture we assume that QoS provisioning is managed by a centralized entity, hereafter called *network manager*. This entity is responsible for the admission of a new arriving traffic flow, and for the efficient selection of routes satisfying the QoS demands of that flow. The network manager implements a set of components to perform its tasks. First of all, it generates a connectivity map and interference characterization of the mesh network based on the statistics collected from each mesh node. Moreover, this manager node maintains a list of the admitted traffic flows and the network paths used to route their traffic. Interference map and load information are then used to construct the capacity utilization model, and to drive the route and gateway selection process for a newly arriving flow. Although a distributed approach for network management would provide better network resilience, there are many advantages in adopting such an architecture that *centralizes* certain functions of the routing protocol. Firstly, it facilitates the deployment of intelligent algorithms that exploit a global knowledge of the network status (e.g., connectivity and interference maps, offered loads, resource utilization, etc.) to provide network optimizations (e.g., topology and channel management, or QoS policies). In addition, several commercial solutions of mesh networking have already a centralized controller where CAC and special routing functionalities can be deployed. Finally, this architecture is also compatible with one of the deployment scenarios recently proposed by the CAPWAP working group [24] for the management, monitoring, and control of large number of interconnected wireless access devices.

In the following two sections we detail the operations performed by the capacity utilization model, and the gateway and route computation module. In Section 5.2 we discuss the approach we adopted to implement such architecture in an experimental mesh network.

3. Modeling network capacity

In this section we develop the capacity utilization model of the heterogeneous WMN described in Section 2. This model will be used to estimate every node's available capacity, which is needed to compute a feasible

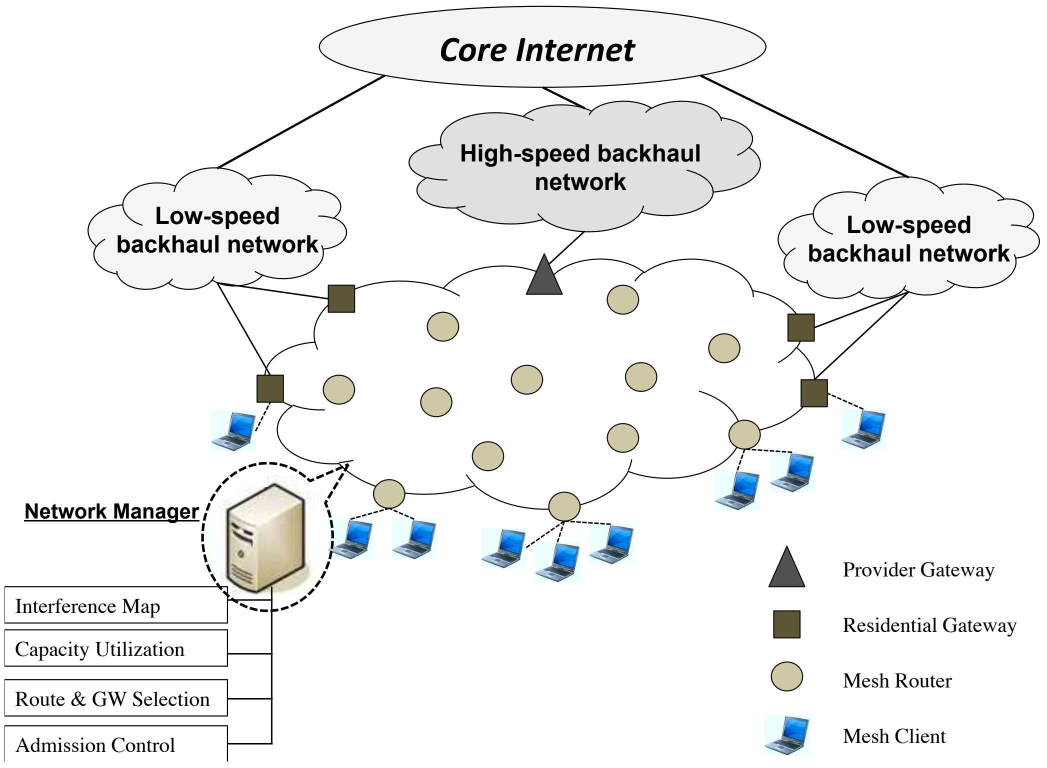


Figure 1: Reference network architecture.

route for a new arriving traffic flow. For the sake of clarity, Table 1 summarizes the notation used in the following analysis.

3.1. Network model

The key idea behind the proposed modeling framework is to convert the physical network model into an equivalent queuing network model. Both these network models can be represented as graphs. In the physical network model the vertexes of the network graph are the mesh nodes, and the edges characterize the physical network connectivity relationships (e.g., in terms of channel bandwidth and packet transmission error rate of each link) that exist between the communicating entities. On the other hand, in the equivalent queuing network model each mesh node is represented using an equivalent queuing station, and the edges characterize the packet forwarding process between two queues.

Formally, let $\mathcal{G}_r$, $\mathcal{G}_p$ and $\mathcal{M}$ be the set of residential gateways, provider gateways and mesh routers, as defined in Section 2. Let n_w, n_r and n_p be the cardinality of the $\mathcal{M}$, $\mathcal{G}_r$, and $\mathcal{G}_p$ sets, respectively, with $n = n_w + n_r + n_p$. Then, the physical network model can be described using a mixed graph $G(V \cup \{a\}, E_w, E_g)$, where the graph vertexes V ($|V| = n$) represent the mesh nodes (i.e., $V = \mathcal{G}_r \cup \mathcal{G}_p \cup \mathcal{M}$) and a is a virtual vertex that corresponds to the fixed infrastructure (i.e., the Internet). We denote by E_w the set of edges representing the wireless links between mesh nodes, while E_g is the set of edges representing the backhaul links between the gateways and the infrastructure. The neighborhood of node $v \in V$, denoted by $N(v)$, is the set of nodes to which node v is physically connected. If node v is a gateway, the virtual node a is included in the neighborhood of v.

To construct the link-layer connectivity map of the WMN, we assume that the *transmission range* of each wireless transmitter is fixed and equal to R_{tx}. Each edge $e_{i,j}$ between node $i, j \in G$ is labelled with a pair of values, $C_{i,j}$ and $r_{i,j}$. The former parameter represents the nominal capacity associated to that link, while the latter represents the probability that a packet transmitted over that link is corrupted by channel errors.

variable	definition
$\mu_{i,l}$	mean service rate of packets at the queue l of the station i
λ_i^e	arrival rate of packets from outside (i.e.,mesh clients) to station i
$\lambda_{i,l}$	overall arrival rate of packets at the queue l of the station i
$\lambda_{i;j}^f$	mean rate of packets transferred from station i to station j
$\lambda_{i;j}^t$	mean rate of packets served at station i and heading towards station j
$\lambda_{i;j}^r$	mean rate of packets that are corrupted when transmitted on the wireless link from station i to station j
λ_i^r	overall rate of packets to be retransmitted that are inserted in the wireless queue of station i
$p_{i,l;j,s}$	*routing probability*: the probability that a job is transferred to queue s of node j after service completion at queue l of node i.
$r_{i;j}$	*retransmission probability*: the probability that a job served at the wireless queue of node i is corrupted when transferred to node j. It holds that $\lambda_{i;j}^r = r_{i;j} \cdot \lambda_{i;j}^t$.

Table 1: Model notation

Moreover, we assume that probability $r_{i,j}$ is time-invariant, which is equivalent to assume that the packet errors at the PHY layer can be modeled using a stationary random process. In general, the $r_{i,j}$ process can be derived by integrating a specific PHY layer model into the channel characterization. Alternatively, actual link-layer measurements can be used to extract the statistics of the packet transmissions error rates and to allow a trace-driven emulation of the physical network. For the sake of generality, in the following we do not model the statistics of the $r_{i,j}$ process, but we only assume that it is stationary.

For simplicity, we assume that each $e_{i,j} \in E_w$ has a fixed and constant transmission rate C_w, i.e., no rate adaptation is used on the wireless links. Moreover, the low-speed backhaul link from a residential gateway $i \in \mathcal{G}_r$ to the wired infrastructure a has fixed capacity C_r, while the high-speed backhaul link between a provider gateway $i \in \mathcal{G}_p$ and the wired infrastructure a has fixed capacity C_p. Generally, it holds that $C_r \ll C_p$.

From the graph representation $G(V \cup \{a\}, E_w, E_g)$ of the heterogeneous WMN, we are able to derive an equivalent queuing network model $G'(Q, L)$, where Q indicates the set of queuing systems in the network, for brevity *stations*, and L is the set of connections between stations. Intuitively, jobs in the queuing network represent packets in the physical network[1]. Owing to the analogy between the physical network and an equivalent queuing network, each mesh node $i \in V$ is modeled through a service station $k \in Q$. In general, this equivalent queuing station may include several queues. This allows us to model mesh nodes with multiple interfaces such as the gateways, which are equipped with wired and wireless interfaces. Furthermore, the internal queuing structure of the queuing station is also exploited to provide the model with the flexibility to characterize different routing strategies. Hereafter, $q(j)$ indicates the number of internal queues at station j. For simplicity, in this study we assume that all queues have infinite size and serve packets according to a FCFS discipline.

It is intuitive to note that, being the WMN composed of two classes of mesh nodes, gateway and mesh routers, at least two different queuing station models should be specified for the analysis. For ease of explanation, Figure 2 exemplifies the structure of the queuing stations used to model mesh nodes. Specifically, Figure 2(a) illustrates a station modeling a wireless mesh router i ($i \in \mathcal{M}$) not connected to the wired infrastructure. This station consists of a singe queue (i.e., $q(i) = 1$), say $q_{i,1}$, which models the transmissions on the wireless channel[2]. Let us denote with $\mu_{i,1}$ the mean service rate of queue $q_{i,1}$. As observed in Section 2, each mesh node i aggregates the traffic flows originated from the mesh clients associated with it. We model this aggregated traffic through its average packet arrival rate, say λ_i^e. In addition to locally generated jobs, station i can receive jobs from each of its neighboring stations, with average arrival rate $\lambda_{j;i}^f$ ($j \in N(i)$). On the other hand, station i transfers jobs to its neighboring stations. Let $\lambda_{i;j}^t$ be the overall rate of jobs served

[1]In the following, the terms job and packet are used equivalently.

[2]For simplicity, we consider a single wireless interface. The extension to multiple wireless interfaces is straightforward, but it would require the incorporation in the model of a channel assignment algorithm, which is out of the scope of this study.

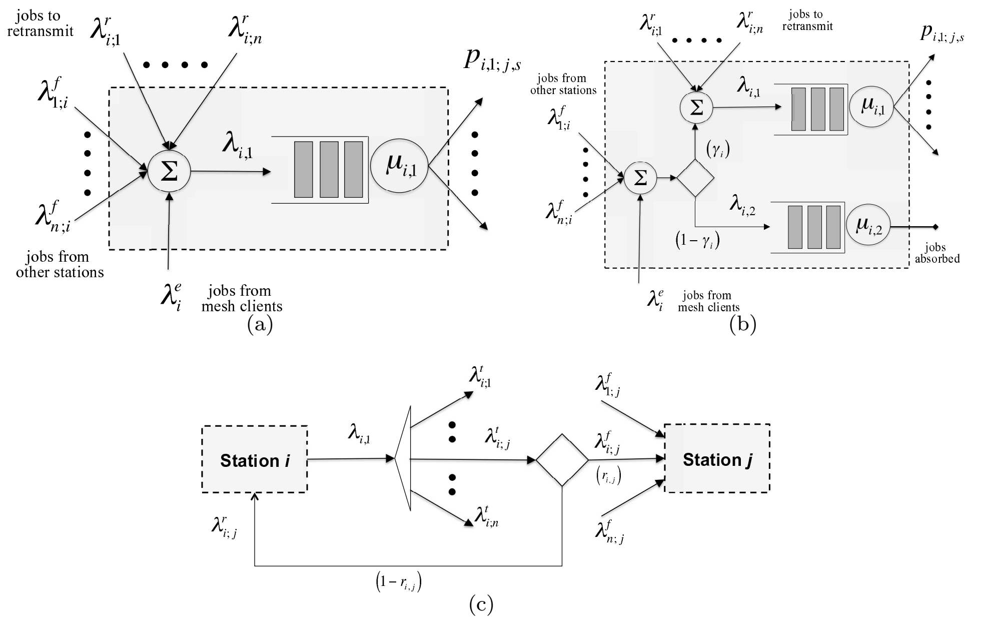


Figure 2: Components of the queuing network model: *a*) structure of a non-gateway station, *b*) structure of a gateway station, and *c*) wireless channel model.

at queue $q_{i,1}$ that are headed towards stations j. A fraction $r_{i;j}$ of these jobs will be retransmitted because corrupted by channel errors. These retransmissions can be modeled as an additional ingress traffic for queue $q_{i,1}$, with mean rate $\lambda^r_{i;j}$ equal to $\lambda^t_{i;j} \cdot r_{i;j}$. Consequently, the overall rate of packets that are re-inserted in queue $q_{i,1}$ after been served is $\lambda^r_i = \sum_k \lambda^r_{i;k}$. For clarity, the described channel model is also illustrated in Figure 2(c). Owing to the previous considerations, it holds that the total arrival rate at queue $q_{i,1}$ can be expressed as $\lambda_{i,1} = \sum_j \lambda^f_{j;i} + \sum_k \lambda^r_{i;k}$, while the mean rate of jobs that are transferred from station i to station j can be computed as $\lambda^f_{i;j} = (1 - r_{i,j}) \cdot \lambda^t_{i;j}$.

Figure 2(b) describes the internal structure of a station modeling a gateway node i ($i \in \mathcal{G}_r \cup \mathcal{G}_p$). In this case, the queuing station structure consists of two queues: $q_{i,1}$, which models wireless transmissions, and $q_{i,2}$, which models the transmissions on the gateway's backhaul link. Similarly to non-gateway stations, each gateway station can receive packets forwarded by neighboring stations, with average arrival rate $\lambda^f_{j;i}$ ($j \in N(i)$), as well as packets generated by associated mesh clients, with average arrival rate λ^e_i. Then, the logic implemented internally to the gateway station determines what fraction of the arriving jobs is transferred to $q_{i,1}$ or $q_{i,2}$. It is important to note that in this work we are primarily concerned with *upstream Internet traffic*. In other words, we assume that traffic flows are originated from mesh nodes and destined for the Internet (i.e., the virtual node a). Hence, when a job reaches a gateway station, it could be directly transferred to queue $q_{i,2}$, and then leave the network after being served. However, a residential gateway may have a low-speed upstream connection to the Internet, which rapidly becomes a bottleneck as the traffic received on the wireless interface builds up, limiting the achievable capacity of the whole mesh network. To make this limitation less severe, the residential gateway may take advantage of the available wireless bandwidth to behave as a relay node, and further forwarding the traffic to one of its neighbors, which may be less congested, or closer to a provider gateway. To model this capability we introduce the *re-forwarding* probability γ_i. Specifically, a job received by gateway i is routed through the wireless queue $q_{i,1}$ with probability γ_i, or directly through the upstream wired queue $q_{i,2}$ with probability $(1 - \gamma_i)$. The design of the γ_i function depends on the routing and resource allocation strategies implemented in the mesh network. Finally, similarly to non-gateway stations, jobs transmitted by queue $q_{i,1}$ can be corrupted by channel errors and they will be retransmitted. These retransmitted jobs will contribute to the overall arrival rate at queue

$q_{i,1}$ with an additional flow of jobs having mean arrival rate equal to λ_i^r.

Before concluding this section, it is important to describe the wireless channel access-control mechanisms we adopt for the MAC layer, because they significantly affect the estimate of the $\mu_{i,1}$ values. In this study, we assume that a simplified CSMA-based MAC protocol is used by the wireless transmitters to coordinate simultaneous transmissions of interfering nodes. This basic MAC scheme implements an idealized collision avoidance mechanism. More precisely, we assume that each node has an instantaneous knowledge of the communication state (i.e., idle, receiving or transmitting) of other interfering nodes, so as to ensure that it starts transmitting only when its transmitted packets does not cause a collision. This is somehow equivalent to determine a collision-free random transmission schedule among contending nodes. To model the interference relationships between contending mesh nodes we use the Protocol Model as in [21, 22]. In other words, a transmission from mesh station i to mesh station j, with $i, j \in Q$, is *admissible* if the following conditions are satisfied: 1) $dist_{i,j} \leq R_{tx}$ and 2) for every other transmitting node k, $dist_{k,j} \geq (1+\Delta) \cdot R_{tx}$, where $dist_{i,j} istheeuclideandistancebetweennodeiandnodej, and \Delta$ is a positive constant that represents a guard zone in the Protocol Model. Note that not all the admissible transmissions are successful, because we incorporate in the analysis a retransmission probability $r_{i,j}$. The collision-less MAC protocol used in our study might be considered somehow restrictive, especially because we neglect the detailed protocol implementation of collision avoidance and resolution mechanisms, such as 802.11-like backoff schemes. However, though in a simplified form, this MAC scheme captures the fundamental aspects of location-dependent contention inherent to multi-hop environments, which is due to differences in the number of contending nodes at both endpoints of each communication link. In other words, in this study we are more concerned on modeling the link capacity degradation due to location-dependent contention, rather than precisely incorporating in the analysis all the features of the IEEE 802.11 MAC protocol.

3.2. Node utilization

In this section we develop the analysis to determine if a given throughput allocation in $G(V \cup \{a\}, E_w, E_g)$ is *feasible*. Before formally defining when a throughput allocation is feasible, and describing our analytical methodology, it is useful to introduce some notation.

Let us denote with λ^e the overall arrival rate of a job from *outside* (i.e., from mesh clients associated to mesh nodes) to the mesh network. Furthermore, let $p_{i,l}^e$ be the probability that a job from outside the network enters queue l of station i. This implies that the job arrival rate from outside to queue l of station i is $\lambda_{i,l}^e = \lambda^e \cdot p_{i,l}^e$. For brevity, we introduce the *probability matrix of external arrivals* defined as $\mathbf{P^e} = \{p_{i,l}^e, i \in Q, l \in [1, q(i)]\}$. Note that this notation conforms to the network model formulated in Section 3.1.

Definition 1. Throughput allocation. *A throughput allocation for $G(V \cup \{a\}, E_w, E_g)$ is any assignment for the rate λ^e and the probability matrix $\mathbf{P^e}$.*

Definition 2. Feasible throughput allocation. *A throughput allocation is feasible if every queue $q_{i,l}$ ($i \in Q$ and $l \in [1, q(i)]$) has a bounded time-average number of packets. This is equivalent to state that arrival process at queue $q_{i,l}$ is admissible with rate $\lambda_{i,l}$.*

From a mathematical point of view, Definition 2 implies that a throughput allocation is feasible if all the queues in the system are stable, i.e., the number of jobs waiting in queue does not grow indefinitely. From a more practical perspective, to determine if a throughput allocation is feasible is equivalent to verify that the allocation of a given set of flows on a given set of network paths does not violate the network capacity constraints. To verify the queue stability we have to compute the queue's *utilization factor* [25]. From elementary queuing theory this requires the evaluation of the first moments of the packet arrival and service processes at each queue of the network. Specifically, the utilization $\rho_{i,l}$ of the queue l at station i (i.e., $q_{i,l}$) is $\rho_{i,l} = \lambda_{i,l}/\mu_{i,l}$. By definition, an infinite-size queue is stable if and only if $\rho_{i,l} < 1$.

A fundamental parameter to compute the arrival rate at every queue $q_{j,s}$ is the *routing probability* $p_{i,l;j,s}$ defined as the probability that a job is transferred from the queue l of station i (i.e., $q_{i,l}$) to queue s of station j (i.e., $q_{j,s}$). Following the equivalency between the real WMN and the queuing network, the $p_{i,l;j,s}$ value expresses the probability that mesh node i selects mesh node j as next-hop to reach the Internet. The queue indexes are used to specify if the packet is transmitted using the wireless links or the wired fixed lines. For the sake of brevity, we introduce the network *routing matrix* defined as $\mathbf{R_{fwd}} = \{p_{i,l;j,s}, i,j \in Q, l \in [1, q(i)], s \in [1, q(j)]\}$, which is the probabilistic representation of the underlying routing process. Note that in statistical equilibrium the rate of departure from a queue is equal to the rate of arrival, and the overall arrival rate at queue $q_{i,l}$ can be written as:

$$\lambda_{i,l} = \lambda^e \cdot p^e_{i,l} + \sum_{\substack{j=1 \\ j\neq i}}^{n} \sum_{s=1}^{q(j)} \lambda_{j,s} \cdot p_{j,s;i,l} + \begin{cases} \sum\limits_{\substack{k=1 \\ k\neq i}}^{n} \lambda^t_{i;k} \cdot r_{j,k} & l=1 \\ 0 & l=2 \end{cases} \quad , \text{ for } i \in Q, l \in [1, q(i)] \ . \tag{1}$$

It is intuitive to observe that the specific formulation of the routing matrix depends on several factors including the routing algorithm used in the WMN, the network topology, the throughput allocation and the retransmission probabilities.

The following lemma provides a methodology to compute the routing matrix $\mathbf{R_{fwd}}$ without solving the system defined in Equation 1, but considering a simplified queuing network.

Lemma 1. *Let $\overline{p}_{i,l;j,s}$ the routing probability of a simplified queuing network $\overline{G'}(Q,L)$ obtained from $G'(Q,L)$ by setting $r_{i,j}=0$ for $i,j \in Q$, while leaving unmodified all the other characteristics, i.e., λ^e, $\mathbf{P^e}$, and how the jobs are routed between the stations. Then, it holds that*

$$p_{i,l;j,s} = \begin{cases} \dfrac{\overline{p}_{i,l;j,s}}{1+\sum\limits_{\substack{k=1 \\ k\neq i}}^{n} \frac{r_{i,k}}{1-r_{i,k}} \cdot \overline{p}_{i,l;j,s}} & i \neq j \\ 1 - \sum\limits_{\substack{k=1 \\ k\neq i}}^{n} p_{i,l;k,s} & i=j \end{cases} \tag{2}$$

Proof. Without loss of generality, we prove expression 2 for $i,j \in \mathcal{M}$, i.e., when the communication endpoints are two mesh routers. In this case, $q(i)=q(j)=1$. The other cases can be easily derived following the same line of reasoning.

By definition, it holds that

$$p_{i,1;j,1} = \frac{\lambda^f_{i;j}}{\lambda_{i,1}} \qquad , \qquad \overline{p}_{i,1;j,1} = \frac{\overline{\lambda}^f_{i;j}}{\overline{\lambda}_{i,1}} \ ,$$

where $\overline{\lambda}^f_{i;j}$ and $\overline{\lambda}_{i,1}$ are the mean rate of jobs transferred from station i to station j in $\overline{G'}(Q,L)$, and the overall rate of jobs entering queue $q_{i,1}$ in $\overline{G'}(Q,L)$, respectively. Since corrupted packets transmitted over the wireless link between station i and station j do not enter into queue $q_{j,1}$, it is intuitive to observe that $\lambda^f_{i;j} = \overline{\lambda}^f_{i;j}$. In other words the same rate of packets are transferred from station i to station j in both $\overline{G'}(Q,L)$ and $G'(Q,L)$. This implies $\lambda^f_{i;j} = \overline{p}_{i,1;j,1} \cdot \overline{\lambda}_{i;j}$, and that

$$\lambda_{i,1} = \overline{\lambda}_{i,1} + \sum_{\substack{k=1 \\ k\neq i}}^{n} \lambda^r_{i;j} \ . \tag{3}$$

Now, considering the channel model illustrated in Figure 2(c) we can write the following two equalities:

$$\lambda^r_{i;j} = r_{i;j} \cdot \lambda^t_{i;j} \qquad , \qquad \lambda^r_{i;j} = \lambda^t_{i;j} - \lambda^f_{i;j} \ .$$

With simple algebraic transformations, the above expressions can be written as

$$\lambda^r_{i;j} = \frac{r_{i;j}}{1-r_{i;j}} \lambda^f_{i;j} = \frac{r_{i;j}}{1-r_{i;j}} \overline{p}_{i,1;j,1} \overline{\lambda}_{i;j} \ . \tag{4}$$

By substituting expression 4 in formula 1, after simple manipulations we obtain equation 2, and this concludes the proof. □

Now, the average arrival rate $\lambda_{i,l}$ can be computed from λ^e, $\mathbf{P^e}$ and $\mathbf{R_{fwd}}$ by solving a system of linear equations obtained by writing the flow balance condition at each queue of the system as in equation 1. The mean service rates for the queues modeling transmissions on wired links can be easily derived by observing that in switched communication technologies there is no contention. Hence, average service times depend only on the nominal link capacity and the packet size. Then, under the assumption that the packet size is constant and equal to P bits, it holds that $\mu_{i,2} = P/C_r$ if $i \in \mathcal{G}_r$, and $\mu_{i,2} = P/C_p$ if $i \in \mathcal{G}_p$. On the other hand, the derivation of the average service rate for the queues modeling transmissions on the wireless channel is more involved because it is necessary to take into account the location-dependent contention, the distributions of active queues (i.e., queues with at least a packet to serve) and the channel access coordination procedures implemented by the MAC protocol. Several stochastic models have been developed to analyze the access delays of CSMA-based MAC protocols used in multi-hop environments. Recall from Section 3.1 that in this work we consider a basic collision-free CSMA-based MAC protocol, and we assume that each mesh node has an instantaneous knowledge of the communication state of other interfering nodes. Then, following the footprints of [26] and our previous work [22], we can model the impact on the channel access of location-dependent contention by employing an *average value analysis*, and considering only the long-term fraction of time each mesh node spends in one of three potential states: transmission state, receiving state, and idle state. This modeling approach will lead to a mathematically manageable and reasonably accurate analysis.

To compute the $\mu_{i,1}$ parameter we analyze the channel events during the $X_{i,1}$ period, defined as the interval from the time instant a job reaches the head of queue $q_{i,1}$ to the time instant in which its service is completed. Under the assumption that the transmission events are i.i.d., it holds that $\mu_{i,1} = 1/E[X_{i,1}]$, where $E[\cdot]$ is the expectation operator. To simplify the derivation of the $E[X_{i,1}]$ expression we condition to the possible destinations of a job served at queue $q_{i,1}$. Specifically, owing to the conditional expectation theory we can write that

$$E[X_{i,1}] = \sum_{j=1}^{n} \sum_{s=1}^{q(j)} E[X_{i,1;j,s}] \cdot p_{i,1;j,s} \ , \tag{5}$$

where $X_{i,1;j,s}$ is the time needed by queue $q_{i,1}$ to complete the service of a job heading to queue $q_{j,1}$. This time will mainly depend on the level of contention around the transmitting station i and the receiving station j, i.e., on the distribution of interfering nodes in the network, as well as on their activity level, i.e., the fraction of time these nodes contend for the channel access. More precisely, due to the random access scheme the this packet transmission can be preceded by a number $z_{i,1;j,s}$ of transmissions performed by other contenting stations[3]. Let us denote with $E[B_{i,1;j,s}]$ the average period of channel time occupied by other stations' packet transmissions, which precedes the service of the packet at the head of queue $q_{i,1}$, given that this packet is heading towards queue $q_{j,s}$. Then, under the assumption of fixed packet size, it is straightforward to derive that

$$E[B_{i,1;j,s}] = P \cdot E[z_{i,1;j,s}]/C_w . \tag{6}$$

This yields to the following expression for the $E[X_{i,1;j,s}]$ parameter.

$$E[X_{i,1;j,s}] = P \cdot (1 + E[z_{i,1;j,s}])/C_w \ . \tag{7}$$

To derive a closed expression for the $E[z_{i,1;j,s}]$ parameter, the key approximation of our analysis is to assume that station i attempts to transmit a packet to station j immediately after the channel becomes idle again with a constant (state independent) probability equal to $\tau_{i,1;j,s}$. This approximation is commonly adopted when modeling CSMA-based random access schemes, and it also known as *decoupling* approximation [27]. While in single-hop networks it is generally assumed that all nodes have the same *transmission probability*, in our study the location-dependent contention is modeled by admitting different values of the $\tau_{i,1;j,s}$ probabilities. The decoupling approximation yields that $z_{i,1;j,s}$ is geometrically distributed with parameter $\tau_{i,1;j,s}$, that is

$$Pr\{z_{i,1;j,s} = h\} = (1 - \tau_{i,1;j,s})^h \tau_{i,1;j,s} \ . \tag{8}$$

[3]In our idealized MAC scheme transmission attempts are not preceded by backoff delays.

Now, it is straightforward to derive that

$$E[B_{i,1;j,s}] = \frac{(1-\tau_{i,1;j,s})}{\tau_{i,1;j,s}} \cdot S \ , \tag{9}$$

and formula (7) can be rewritten as $E[X^r_{i,1;j,s}] = P/(C_w \cdot \tau_{i,1;j,s})$.

The following lemma provides an explicit expression for the transmission probability $\tau_{i,1;j,s}$.

Lemma 2. *Under the assumption that reception and transmission events in $G'(Q,L)$ are mutually independent, it holds that*

$$\tau_{i,1;j,s} = \prod_{h\in\mathcal{E}_i} \prod_{u=1}^{q(h)} (1-\phi_{h,u}\cdot\omega_{h;j}) \cdot \prod_{k\in\mathcal{E}_j\cup\{j\}} (1-\psi_{k,1}) \ , \tag{10}$$

where

- *$\phi_{h,u}$ is the long-term fraction of time spent by queue $q_{h,u}$ receiving packets;*
- *$\psi_{k,1}$ is the long-term fraction of time spent by queue $q_{k,1}$ transmitting packets;*
- *$\omega_{h,u;j}$ is the fraction of wireless queues that are neighbors of station h, but they are not interferers for station j, and which have a not-null routing probability towards queue $q_{h,u}$;*
- *$\mathcal{E}_i$ is the set of mesh nodes in the interference region of node i (formally, $\mathcal{E}_i = \{h : dist_{h,j} \leq (1+\Delta)\cdot r \ , \ h \in G\}$).*

Proof. The proof follows the same line of reasoning of Lemma 1 in [22]. □

In summary, the analytical methodology we adopt to determine the feasibility of a throughput allocation consists of the following steps. First of all, from the WMN topology $G(V\cup\{a\}, E_w, E_g)$ we extract the equivalent queuing network $G'(Q,L)$. Then, given the throughput allocation (λ_o and $\mathbf{P_o}$), and the routing matrix $\mathbf{R_{fwd}}$, we can determine the overall arrival rate at each queue solving the linear system defined with equation (1). From the $\lambda_{i,l}$ values we compute the $\phi_{i,l}$ and $\psi_{i,1}$ parameters, and the $\tau_{i,1;j,s}$ probabilities using Lemma 2. This allows us to derive the average service times of each queue in the network, and to check the feasibility of the throughput allocation.

4. Load-aware algorithms for route and gateway selection

In the previous section we have developed a analytical framework to determine if a given routing matrix leads to an unfeasible throughput allocation. In this section we develop a practical *Load-Aware Route Selection* (*LARS*) algorithm, which exploits this framework to construct a routing matrix that avoids unevenly utilization of gateways' backhaul links and network bottlenecks, while ensuring that the resulting throughput allocation is feasible. A key feature of our solution, is to implement a simple and efficient strategy to discover and select feasible paths (i.e., paths with sufficient remaining capacity to accommodate the bandwidth demands of new flows). As a matter of fact, it is unrealistic to perform an exhaustive search because there are exponentially many paths between a source/destination pair, and a brute force strategy does not scale. For these reasons, in the literature various solutions have been proposed for reducing the complexity of this problem. A popular approach is to consider only disjoint and braided paths [28], but it is still computationally intensive to construct multiple disjoint paths. An alternative strategy is proposed in [11], where the complexity of finding optimal routes is mitigated by considering only routing forests, i.e., unions of disjoint trees rooted at the gateway nodes. However, tree-based structures are less reliable to link failures than mesh-based structures. The authors in [29] propose to transform the original network graph into an edge graph, where multiple links are aggregated into segments. This approach results into a reduction in the number of possible paths to check for feasibility, depending on the adopted segment size.

In our routing scheme we adopt a simpler approach by constructing a *routing mesh* from each mesh node to the available gateways. More precisely, for each mesh node i we compute the minimum cost paths towards each gateway j (with $j \in \mathcal{G}_r \cup \mathcal{G}_p$). Note that minimum cost path can be efficiently computed in a loop-free manner using Dijkstra and Bellman-Ford algorithms if the routing metric is *isotonic* [30]. Thus, the number of paths to check for feasibility grows linearly with the number of gateways and mesh nodes. The penalty we pay for this simplicity is that occasionally the routing process may not find a feasible route although it exists.

Algorithm 1: Pseudo-code of the LARS algorithm.

```
Input: $G(Q,L)$, $\mathcal{F}^{(k)}$, $\mathbf{P}^{e^{(k)}}$, $\mathbf{R}^{(k)}_{\mathbf{fwd}}$, and $f^{(k+1)}$.
Output: stable, $\mathbf{P}^{e^{(k+1)}}$, $\mathbf{R}^{(k+1)}_{\mathbf{fwd}}$.

1  stable ← $true$ ;
2  admitted ← $false$ ;
3  $\overline{\lambda^{e}}^{(k+1)} \leftarrow \lambda^{e^{(k)}} + b^{(k+1)}$ ;
4  $\overline{\mathbf{P}^{\mathbf{e}}}^{(k+1)} \leftarrow$ Update($\mathbf{P}^{\mathbf{e}^{(k)}}, f^{(k+1)}$) ;
5  $\mathcal{Q}_s \leftarrow \mathcal{G}_r \cup \mathcal{G}_p$ ;
6  while ($\mathcal{Q}_s \neq \emptyset$) or !admitted do
7      $g \leftarrow$ ExtractClosest($v, \mathcal{Q}_s$) ;
8      $path_{s,g} \leftarrow$ MinimumCostPath($s, g, G(Q,L)$) ;
9      $\overline{\mathbf{R}}^{(k+1)}_{\mathbf{fwd}} \leftarrow$ Update($path_{s,g}, \mathbf{R}^{(k)}_{\mathbf{fwd}}$) ;
10     stable ← IsStable($\overline{\lambda^{e}}^{(k+1)}, \overline{\mathbf{P}^{\mathbf{e}}}^{(k+1)}, \overline{\mathbf{R}}^{(k+1)}_{\mathbf{fwd}}$) ;
11     if stable then
12         Consolidate($\overline{\lambda^{e}}^{(k+1)}, \overline{\mathbf{P}^{\mathbf{e}}}^{(k+1)}, \overline{\mathbf{R}}^{(k+1)}_{\mathbf{fwd}}$) ;
13         $\mathcal{F}^{(k+1)} \leftarrow \mathcal{F}^{(k)} \cup f^{(k+1)}$ ;
14         admitted ← $true$ ;
15     else
16         $\mathcal{Q}_s \leftarrow \mathcal{Q}_s \setminus \{g\}$ ;
17     end
18 end
```

To facilitate the description of the LARS solution, Algorithm 1 shows the pseudo-code that it is used to determine a feasible route for a newly arrived traffic flow. Let us assume that at time t a set $\mathcal{F}^{(k)}$ of k upstream Internet flows, $f^{(1)}, f^{(2)}, \ldots, f^{(k)}$ has been already admitted in the network. Each of these flow has demanded a certain average bandwidth to satisfy its QoS requirements. Formally, let $b^{(i)}$ denote the mean arrival rate of packet generated by the i-th flow in the set $\mathcal{F}^{(k)}$. Thus, the overall offered load $\lambda^{e^{(k)}}$ is equal to $\lambda^{e^{(k)}} = \sum_{i=1}^{k} b^{(k)}$, and $\mathbf{P}^{\mathbf{e}(k)}$ is the related throughput allocation. Finally, let $\mathbf{R}^{(k)}_{\mathbf{fwd}}$ be the equivalent routing matrix used to forward these flows.

Now, let as assume that at time $t+1$ arrives a new flow $f^{(k+1)}$, originated at mesh node $s \in V$, which demands a bandwidth equal to $b^{(k+1)}$. We denote with $\mathcal{Q}_s$ the set of gateways that node s can use to access the Internet. Without loss of generality, we can assume that $\mathcal{Q}_s$ contains all the gateways deployed in the network. Then, we update the throughout allocation vector to include the additional demand of this flow. The core of LARS scheme is the selection of the *best* gateway for s in the set $\mathcal{Q}_s$ of potential Internet gateways. To this end, LARS algorithm finds the gateway g in the set $\mathcal{Q}_s$ that is at the least distance from node s. The minimum cost path between s and g, denoted with $path_{s,g}$, is computed using the MinimumCostPath$(s, g, G(Q,L))$. Then, the routing matrix is updated assuming the new flow is routed on $path_{s,g}$. Finally, the algorithm checks if the *tentative* forwarding matrix $\mathbf{R}^{(k+1)}_{\mathbf{fwd}}$ obtained after adding this new flow from s to g generates a bottleneck in the mesh network. To this end, the analytical framework developed in Section 3 is used to perform a *stability test* that checks the feasibility of the throughput allocation. If the stability test is positive, the algorithm confirms the flow allocation using the `Consolidate` function. On the other hand, if gateway g is a bottleneck it can not be used as an egress point to the

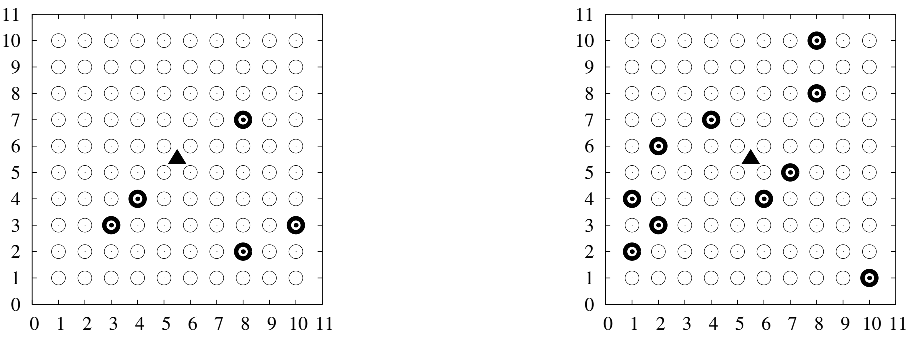

(a) $n_r/n = 5\%$ (b) $n_r/n = 10\%$

Figure 3: Illustrative network topologies. Bold circles represent residential gateways, while filled triangles are provider gateways.

Internet for this new flow and it is removed from the set $\mathcal{Q}_s$. If there are still available gateways in the set $\mathcal{Q}_s$, LARS will repeat the steps in the *while* cycle, testing the feasibility of a new gateway g.

5. Performance evaluation

We compare the performance of the LARS scheme an the shortest-path first routing protocol using a contention-aware routing metric using both numerical simulations and preliminary test-bed experiments. We take advantage of the different evaluation environments to investigate the proposed approach from various point of views, including practicality and feasibility.

5.1. Numerical simulations

The goal of this section is twofold. First of all we compare the analytical results of our model with the outcome of an event-based simulator to validate our analysis. Then, we compare the throughput performance gains of the LARS scheme over a shortest-path first routing algorithm using the IRU metric. For brevity, in the following we refer to this second scheme as *SPF-IRU*. For the sake of clarity, before describing the simulation environment and set-up, we present a brief description of the IRU metric as reported in [14]. Specifically, to capture inter-flow interference, the IRU metric for link l is defined as, $IRU_l = ETT_l \times N_l$, where N_l denotes the number of mesh nodes with which the transmission on link l interferences, while ETT_l [13] is the expected transmission time on link l. Hence, the IRU cost captures the aggregated channel time that transmissions on link l consume on neighboring nodes, which essentially represents the inter-flow interference. Note that results in [14] show that the IRU metric is able to substantially improve total network throughput in mesh networks by balancing network load.

In the following experiments, the nodes are deployed in a square area of size 1 Km. In the center of the simulated area we place as single provider gateway (i.e., $n_p = 1$), which has a symmetric high-speed Internet connection with $C_p = 1$ Gbps. Then, other 100 nodes (mesh routers and residential gateways) are deployed on a grid layout, with grid points separated by 100m. More precisely, we randomly pick up n_w grid points where we place mesh routers, and in the remaining n_r grid points we place residential gateways ($n = n_w + n_r = 100$). This ensures a sufficient degree of randomness in the locations of the gateways. To simulate residential gateways with low-speed backhaul links, we set $C_r = 5$ Mbps. Finally, we model diverse levels of network heterogeneity by varying the percentage n_r/n of residential gateways over mesh routers. For the sake of clarity, two illustrative network topologies are plotted in Figure 3.

The interference in the network is simulated using the Protocol Model, and the transmission range and interference range of each node are fixed and equal to 100m and 200m, respectively. Regarding the MAC

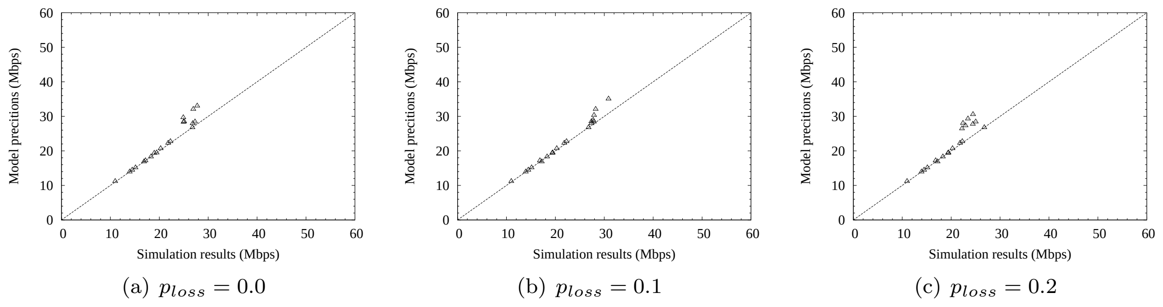


Figure 4: Comparison between predicted and measured network capacity using the LARS scheme with $n_r/n = 0.05$.

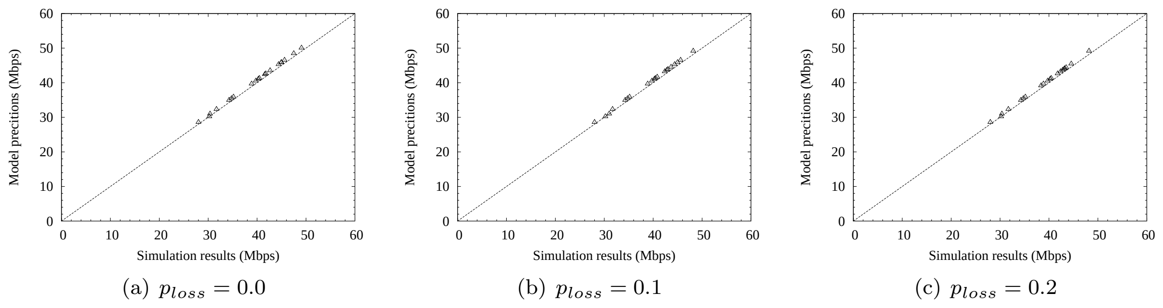


Figure 5: Comparison between predicted and measured network capacity using the LARS scheme with $n_r/n = 0.1$.

protocol, we have implemented the collision-free CSMA-based access scheme described in Section 3.1. A more realistic MAC protocol, using practical collision avoidance mechanisms (e.g., 802.11-based backoff algorithm) will be considered in future work. However, to take into account that we abstract away the MAC-layer signaling issue, i.e., a node is instantly informed about the success of its transmissions, we set the effective wireless channel bandwidth to $C_w = 30$ Mbps. Finally, to model channel errors to each wireless link is assigned a constant packet loss rate, in accordance with the physical layer model used in Section 3.

Regarding the traffic model, in this study we use UDP as the transport protocol for generating data traffic. We consider *upstream* Internet flows established from randomly selected mesh nodes towards the wired infrastructure. Following the notation introduced in Section 4, $b^{(k)}$ is the average uplink bandwidth demand of flow $f^{(k)}$. If not otherwise specified, in the following tests $b^{(k)}$is a random value uniformly selected in the range [100kbps, 200kbps], while the inter-packet arrival time is exponentially distributed.

5.1.1. Capacity estimation

In this section we validate the analysis by comparing the network capacity predicted using our model, and the network capacity measured through simulations in different network scenarios. Without loss of generality, in the following experiments we assume that all the wireless links have the same packet loss rate, say p_{loss}. Following the Definition 2, to compute the network capacity we use randomly generated traffic traces. More precisely, a traffic trace is composed of a large number of independent traffic flows, and the originator of each flow is randomly selected among the mesh nodes. During each simulation run, flows are sequentially injected into the network, and the maximum network capacity is obtained when a new flow cannot be accepted without saturating one of the queues in the network. In the following graphs, we report the results related to the network capacity estimation using the LARS algorithm. The LARS scheme permits to inject in the network a larger number of flows than SPF-IRU (results presented later in Section 5.1.2).

Panel captions and axis labels:

This allows us to validate the analysis in conditions of higher contention on both the wired and the wireless channel, which is very helpful to check the accuracy of the analysis in a larger set of network conditions.

Figures 4 and Figures 5 show a set of scatter plots comparing the network capacity predicted by our model and the one measured through simulations for different numbers of residential gateways and packet loss rates. Twenty different traffic traces are tested per each network topology. The plots show that the correspondence between analytical and simulation results is quite good in most of the considered scenarios. By inspecting the results we discovered that the slight discrepancies between the model predictions and the simulation results occurs primarily when the saturation of a wireless queue is responsible for the limitation of network capacity. This suggests that the model accuracy can be improved by further refining the average value analysis developed to characterize the service times of wireless queues (see Lemma 2).

A number of additional important observations can be derived from the shown results. First, network capacity is greatly dependent on the location of residential gateways, and the traffic pattern. This is even more evident with SPF-IRU[4] since it uses the gateways' resources in a less efficient way than LARS. In general, the higher the n_r/n value, the higher the network capacity. This is expected because adding more gateways increases the aggregate bandwidth available to access the Internet. Moreover, we can observe that increasing the packet loss rate the network capacity may decrease because the retransmitted packets consume more wireless bandwidth. However, many network scenarios are negligibly affected by an increase of packet loss rates. This is especially true for scenarios characterized by poor network capacity because, in this case, it is likely that the network bottleneck is a particularly disadvantaged gateway. Thus, the effect of a small increase in the number of retransmitted frames is dominated by the inefficiency in the use of the gateways' low-speed backhaul links.

Both the above observations motivate the LARS design in which the route selection algorithm takes into account the locations of gateways, as well as the remaining capacity of fixed lines and wireless links.

5.1.2. Capacity gain

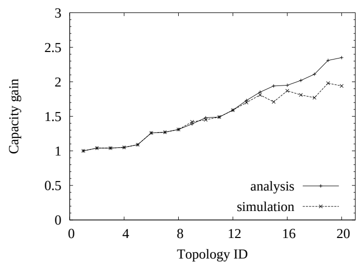
In this section, we evaluate the efficiency of the LARS solutions in terms of the provided *throughput gain* G, i.e., the ratio between the maximum network capacity they obtain and the one achieved by SPF-IRU routing algorithm. Since network capacity measurements have a high dispersion over different topologies and traffic traces, rather than using mean or standard deviation as comparison metric, we show the throughput gain obtained in each network scenario. More precisely, Figure 6 and Figure 7 show the performance gain provided by LARS over SPF-IRU for each of the topologies considered in Figure 4 and Figure 5, respectively. For the sake of clarity, in the graphs we sort the topologies from the maximum network capacity obtained by the shortest-path routing algorithm to the minimum one. Moreover, for each topology we plot the performance gain measured using simulations and the one predicted by the analysis. For these scenarios, LARS significantly outperforms shortest path routing providing a throughput improvement that range from 10% up to 240%. By analyzing more in depth these results we have found out that the performance gain is higher for the most disadvantaged topologies, i.e., for the topologies where the SPF-IRU scheme obtained the lowest performance. This further underlines that the key property of the LARS solution is to anticipate the emergence of network bottleneck and to avoid paths passing though such bottleneck. It is also interesting to observe that higher performance gains are achieved for $n_r/n = 0.05$ than $n_r/n = 0.1$. This can be explained by observing that the higher the density of residential gateways, and the higher the probability that a mesh client has a close gateway. Thus, the inefficiency of a shortest-path first routing algorithm may diminish.

The remarkable throughput improvements provided by LARS can be explained by considering the ability of this algorithm to evenly distribute the network load among all the available gateways. This observation will be further discussed in the following sections, where we use the prototype LARS implementation in a small-scale realistic mesh network to perform a more fine grained study of gateways' resource utilization.

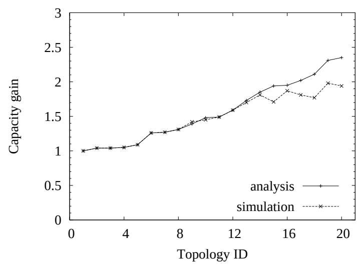
5.2. Preliminary test-bed evaluation

We have developed a proof-of-concept prototype implementing the proposed LARS solution. The goal of the following small-scale experiments is to confirm the performance gains obtainable with the proposed

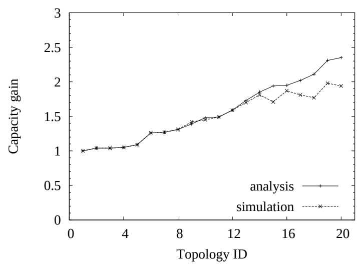
[4]Simulation results related to SPF-IRU are not reported here due to space limitations



(a) $p_{loss} = 0.0$

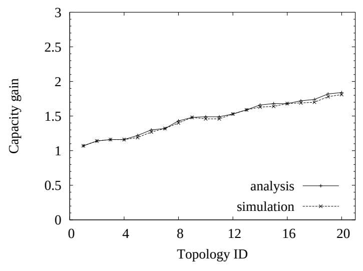


(b) $p_{loss} = 0.1$

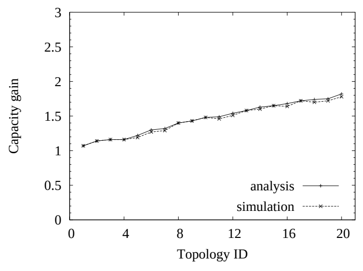


(c) $p_{loss} = 0.2$

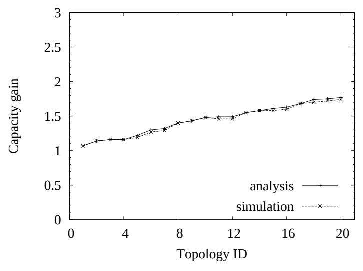
Figure 6: Capacity gain of LARS over SPF-IRU for $n_r/n = 0.05$. The topologies in each graph are ordered from the one with maximum network capacity to the one with minimum network capacity.



(a) $p_{loss} = 0.0$



(b) $p_{loss} = 0.1$



(c) $p_{loss} = 0.2$

Figure 7: Capacity gain of LARS over SPF-IRU for $n_r/n = 0.1$. The topologies in each graph are ordered from the one with maximum network capacity to the one with minimum network capacity.

approach, and to demonstrate the practicality and feasibility of using load-aware route and gateway selection in WMNs.

5.2.1. Test-bed description

The following experimental results have been collected in a trial outdoor mesh network deployed in the CNR's campus area in Pisa, Italy. This mesh test-bed consists of five Soekris-based mesh routers deployed on the rooftops of various buildings, which are equipped wit both directional and omni-directional antennas[5]. Figure 8 illustrates the network topology and connectivity graph of our test-bed. Solid lines indicate links between directional antennas, while dashed lines are used to represent links between omni-directional antennas. As shown in the diagram, all mesh nodes except node $GW1$ are equipped with omni-directional antennas of various gains (8 dBi for nodes $GW3$ and A, and 15 dBi for nodes $GW2$ and B). Mesh node $GW1$ is equipped with one 15 dBi Yagi directional antenna pointing to node $GW2$, and one 19 dBi Grid directional antenna pointing to node $GW3$, while nodes $GW3$ and $GW2$ are equipped with one 19 dBi Grid directional antenna and one 15 dBi Yagi directional antenna, respectively, both pointing to node $GW1$. The shortest link in our network is from A to B, which is 80-meter long, while the longest link is from $GW1$ to $GW3$, which extends over 280 meters. These differences in link distances and antenna characteristics ensure a reasonable variability of link qualities. In our mesh test-bed, nodes $GW1$, $GW2$ and $GW3$ are connected to a high-speed wired infrastructure, thus they function as gateways for the other mesh nodes. In order to emulate backhaul links with various bandwidths, we have used *netem* [32], a linux tool that provides network emulation functionalities for testing protocols, including rate control to limit the input/output transmission

[5]A more detailed description of the hardware architecture of our mesh routers is reported in [31]

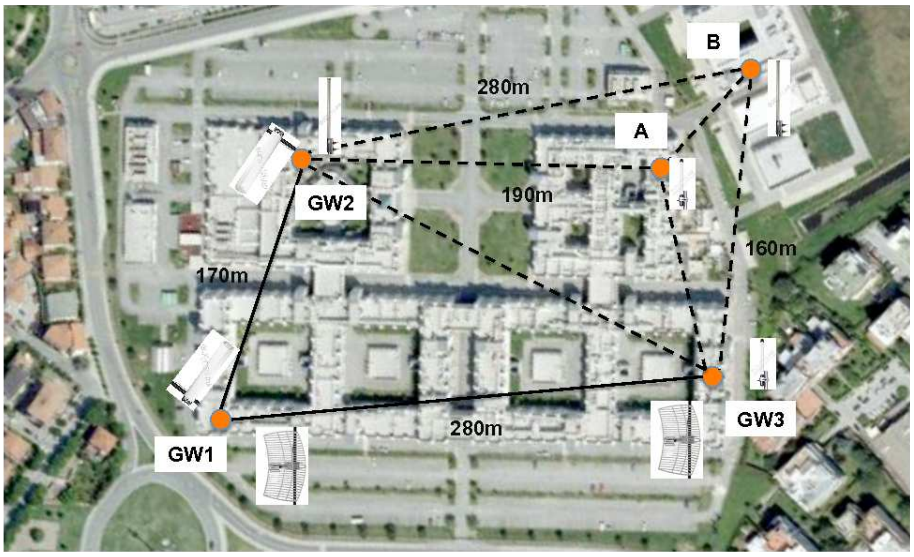


Figure 8: Connectivity graph of our experimental outdoor mesh network. Solid lines are directional links, while dashed lines are omni-directional links.

speed of a network interface. Using netem, we have set up the transmission speed of the wired link at $GW1$ to 2 Mbps, while the wired links at both $GW2$ and $GW3$ are set up to a lower transmission speed equal to 0.5 Mbps. Concerning the wireless interfaces, autorate capabilities are disabled and the data rate is fixed to 11 Mbps. Note that the bandwidth of gateways' fixed links has been set to a lower value than the wireless interfaces to investigate the case that a gateway is the bottleneck node limiting the capacity of the entire network.

5.2.2. LARS software architecture

In order to gain a more clear insight on the issues related to jointly perform admission control, gateway and route selection using flow-based routing in a real mesh network, it is useful to briefly describe the software architecture we have adopted to implement the LARS scheme. Our reference software architecture is an open source implementation of the ad hoc routing protocol OLSR [33]. The OLSR protocol is pro-active, table driven and utilizes an optimized technique called multipoint relaying for efficient message flooding. The current OLSR daemon also implements an optional link quality extension to compute the link costs according to the ETX metric [12].

To implement the LARS scheme, we have developed a set of additional software components which have been integrated into the OLSR daemon. For the sake of clarity, the diagram reported in Figure 9 illustrates the overall software architecture on both the mesh-node side and the network-manager side, as well as the communications between the different modules we have developed. As shown in the figure, the LARS implementation consists of two separate system components, one running in the kernel space and the other running in the user space, which communicates using the *Generic Netlink* communications channel. The kernel component has been developed using the *netfilter* framework, which permits to easily implement both stateless and stateful packet filtering. More precisely, each packet received by a mesh node on the wireless interface used to aggregate the traffic generated by mesh clients associated to that mesh node, is intercepted by the LARS module in the kernel *prerouting* chain. If this packet belongs to a new arriving traffic flows, this flow is classified by the LARS module[6], and, if it is an upstream Internet ow, it is added to the list of *pending flows* waiting for approval by the admission control module. More precisely, the LARS component implemented in the user space periodically (every 4 seconds in our implementation), polls the

[6]Traffic flows are classified using source and destination IP addresses and transport ports.

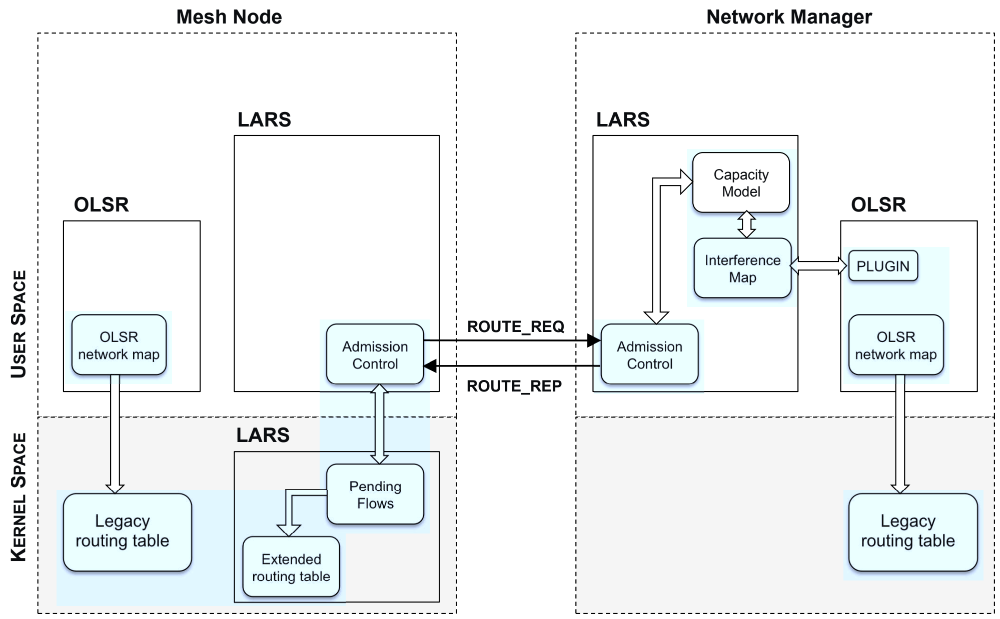


Figure 9: Software architecture of the LARS scheme.

kernel component to check if a pending flow exists. In this case, the mesh node sends a `ROUTE_REQ` message to the network manager to discover a feasible path for this new flow. The network manager replies with a `ROUTE_REP` message, which either contains the route the flow must follow when forwarded in the network[7]. The key issue here is that the legacy kernel routing table does not provide a native support for flow-based routing but only for hop-by-hop routing (i.e., routing based on the knowledge of the next hop for each IP destination). For these reasons we have implemented our own extended kernel routing process supporting flow-based routing, which runs in parallel to the legacy kernel routing process. Note that if a flow is not classified as an upstream Internet flow it passes through the LARS kernel module without any processing, and it is handled in a standard way using the unmodified kernel routing tables.

The LARS component deployed at the network manager is responsible for the computation of the feasible routes, if any, to be allocated to the newly arriving flows. Thus, it implements an admission control module for signaling exchange with the correspondent module collocated at every mesh node. However, to perform the LARS decision process as specified in Section 4, it is necessary to know the packet loss rates for each wireless link of the mesh network. To collect this information, we have developed a new *OLSR plugin*, which is a library that can be dynamically linked to the OLSR daemon using a standard API, enabling the generation/processing of OLSR messages, as well as the access to internal functionalities of the OLSR daemon. More precisely, our plugin access the internal OLSR routing table to read the link quality measurements and to build a complete interference map of the wireless mesh network. This map, along with the link loads due to previously admitted flows, is used to execute the LARS algorithm and to determine the feasible route, if any, for the flow that originated the initial `ROUTE_REQ` message. Note that in our prototype we do not reject flows whose bandwidth demands cannot be fulfilled because this would require more sophisticated CAC mechanisms to communicate with the application. On the contrary, the flow is admitted but routed using the legacy OLSR protocol.

Finally, it is worthwhile to discuss how forwarding is implemented in case of flow-based routing. In principle, the network manager should instruct all the mesh nodes traversed by a flow about the path

[7]Various options are possible for specifying this route. For instance, we can specify the complete route, which is equivalent to support classical flow-based routing. A simpler design choice would be to provide only the identity of the egress router that must be used to exit the mesh network, delegating to OLSR the routing decisions. In our prototype, we have implemented the first option.

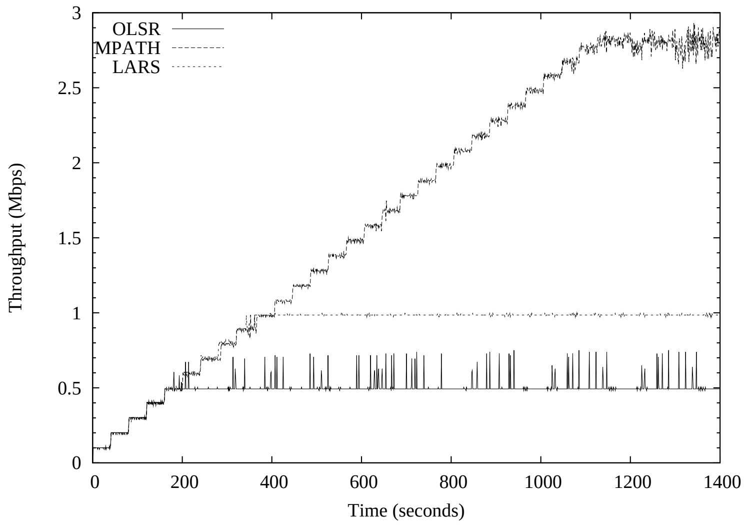


Figure 10: Experimental results when all the traffic flows have node B as source.

that flow should use. Then, a local routing cache should be used to store per-flow routing decisions at intermediate mesh nodes. However, to make easier the implementation of our prototype, we decided to use a simpler approach taking advantage of the IP options fields. Specifically, at the source mesh node, the IP address of each mesh node the packet should traverse is added to the packet IP-header as an additional IP source-route option. The disadvantage of such solution is the typical one of any source-routing based scheme, i.e., an additional protocol overhead is added to the routing process, which is directly proportional to the path length. However, in our small scale network this routing overhead is almost negligible compared to the performance gains ensured by the LARS solution.

5.2.3. Experiments

To gain a better understanding of the advantages and disadvantages of our LARS prototype, and to evaluate the performance limits of the proposed algorithm, we have conducted two distinct sets of experiments.

The first set of experiments aims at validating the correct implementation of the designed mechanisms, and to evaluate the impact of gateways' locations on the system performance. In these tests, every 40 seconds we inject in the network a new traffic flow. A traffic flow is an UDP connection generating packets with a constant rate equal to 100 Kbps and fixed payload equal to 1400 Bytes. For these set of experiments *all the flows are originated at mesh node B*, while the flow destination is a server located in the external Internet. As shown in Figure 8, both low-speed gateways $GW2$ and $GW3$ are one-hop distant from mesh node B, while it is necessary to traverse at least two wireless hops to reach the high-speed gateway $GW1$ from node B. Thus, the OLSR algorithm will select one of the two closest gateways as default gateway for the egress traffic generated by mesh node B. The gateway choice if not fixed, but it depends on the current measurements of link qualities for link $GW2 \leftrightarrow B$ and $GW3 \leftrightarrow B$. However, it is straightforward to observe that, in this case, the upstream throughput for mesh node B is mainly limited by the bandwidth of a single slow-speed fixed line (in our case 0.5 Mbps). Thus, it is reasonable to expect that a performance improvement could be easily achieved by using multiple gateways' upstream links in parallel. This configuration is achievable in a linux-based device, because the Linux kernel permits to easily set up a default route as a multipath route, and to balance the traffic over multiple upstream links. In the considered scenario, we can configure mesh node B to equally use the two gateways $GW2$ and $GW3$. Note that it is not possible to also add a default route to $GW1$ because this default route would necessarily share wireless links with the other default routes generating a cycle in a routing decision process, which uses only information on the next hop when forwarding packets.

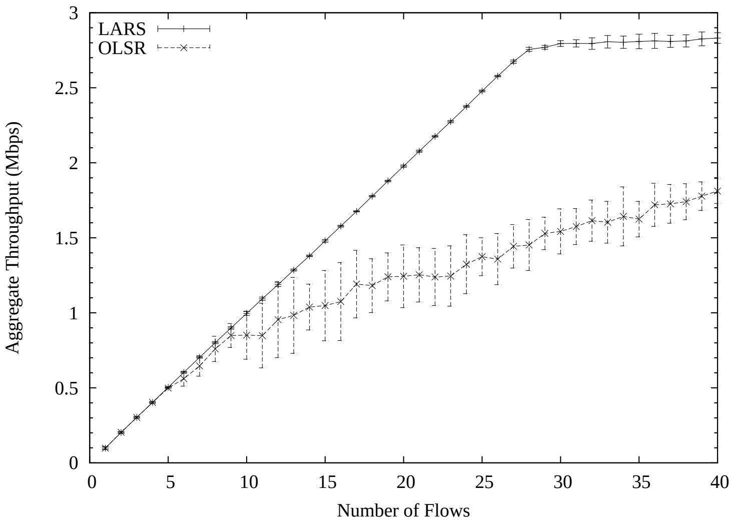


Figure 11: Experimental results when the source of each traffic flow is randomly selected.

Figure 10 shows that instantaneous throughout obtained by mesh node B when thirty-five flows are progressively added to the network for three routing strategies: LARS, standard OLSR and static routing with multiple default routes, for brevity MPATH. Note that the overall capacity of the gateways' upstream links is 3 Mbps, which should permit to fulfil the bandwidth demands of at most thirty flows without packet losses. The experimental results clearly indicate that LARS scheme is significantly more efficient than the other two tested algorithms because it ensure the maximum utilization of the resources available at the three gateways. MAPTH also ensures better performance than OLSR because it permits to fully utilize the bandwidth of $GW2$ and $GW3$, but not the bandwidth of gateway $GW1$. Finally, OLSR is the worst among the tested algorithms because it limits mesh node B to use the closest gateway for all the flows it originates. It is interesting to note that the throughput of mesh node B is not bounded to 0.5 Mbps, although that is the maximum speed of the fixed line at both $GW2$ and $GW3$, but sporadic higher peaks can be observed. These apparently surprising results can be explained by observing that OLSR suffers from route oscillations due to well-known instability of its link quality measurements [34]. Due to these oscillations, the transmission queues at both gateways $GW2$ and $GW3$ can contain packets generated by mesh node B. This may cause a sort of multiplexing effect on links $GW2 \leftrightarrow B$ and $GW3 \leftrightarrow B$, because there might be short period of times during which both gateways are simultaneously using their wired links to serve node B's traffic.

We have carried out a second set of experiments to extend the previous results to a more general scenario in which all the mesh nodes can be originators of traffic flows. More precisely, a traffic flow is an UDP connection generating packets with a constant rate equal to 100 Kbps, which is injected into the network every 40 seconds, as for the first set of experiments. However, differently from the previous experiments, now the originator of each new flow is randomly selected between the five mesh nodes (i.e., we assume that also the gateways can have mesh clients directly associated to them). The performance metric used to compare alternatives schemes is the overall network throughput computed over all the mesh nodes. Since the number of active flows in the network changes during the experiment, we have computed the mean aggregate network throughput by averaging the instantaneous throughput values measured between two consecutive flow arrivals. In this way, we can univocally associate a throughput measure to a given network offered load. Finally, to have statistical meaningful results, we have used five different traffic traces, each one composed of forty traffic flows.

Figure 11 shows the measured average aggregate throughputs, and their 95% confidence intervals, as a function of the network load expressed in terms of number of active flows. We have conducted experiments using both the LARS prototype and the standard OLSR, but not the MAPTH scheme. As described

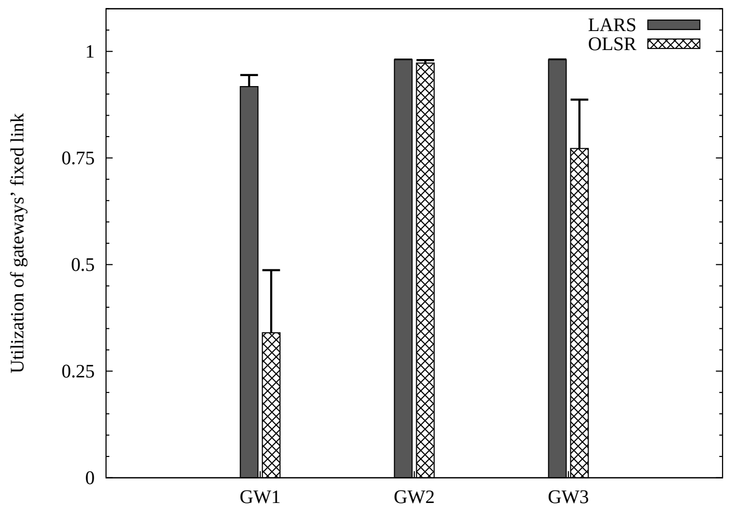


Figure 12: Utilization of gateways' backhaul links when in the network there are 30 flows.

previously, with multiple default routes it is difficult to avoid route cycles in a multi-hop wireless network with multiple flow originators. From the shown experimental results, we can observe that the LARS scheme is able to maximize the network capacity and to fully utilize the network resources. More precisely, given the bandwidth limitations of gateways' fixed lines at most thirty upstream Internet flows could be supported without introducing packet losses on the gateways' transmission buffers. The LARS curve reported in Figure 11 confirms that, in the considered network scenario, our route and gateway selection algorithm is able to satisfy the bandwidth demands of the first thirty flows injected into the network, almost independently of the specific traffic pattern. On the contrary, the standard OLSR performs a blind gateway selection, which quickly introduces inefficiency and significant packet losses. Moreover, with OLSR the network capacity is noticeably dependent on the traffic patterns and gateways' locations. This explains the large confidence intervals that affects the throughput measurements for OLSR.

To better explain the essential reasons why LARS outperforms so significantly the standard OLSR protocols, at least in the considered network scenarios, Figure 12 reports the utilization of the gateways' fixed line when in the mesh network there are thirty upstream Internet flows with randomly selected source mesh nodes. Since this number of flows generates an offered load equal to the overall bandwidth of gateways' uplink connections, it is intuitive to acknowledge that at least one of the gateways must be fully used. However, LARS attempts to distribute the load in an uniform way over all the available gateways, while OLSR always selects the closest gateway, leading to a very unbalanced and inefficient use of the network resources.

6. Conclusions

Differently from other studies on WMNs, in this paper we have considered heterogeneous WMNs where gateways' backhaul links may have various speeds. Focusing on this scenario, we have developed a queuing network model to analyze the network capacity as a function of several system parameters, including locations of gateways, traffic patterns, link bandwidths and packet loss rates. By exploiting this predictive tool, we have designed LARS, a load-aware route and gateway selection algorithm that improves the network capacity by ensuring a more balanced utilization of the network and gateways' resources. Using simulations and a prototype implementation in a realistic small-scale mesh network, we have shown that the LARS scheme significantly outperforms the shortest path routing using a contention-aware routing metric, providing up to 240% throughput improvement in some network scenarios.

Although our analysis considers packet losses due to channel errors, we have used an idealized CSMA-based MAC protocol, which primarily captures location-dependent contention issues due to differences in the number of contending nodes at both endpoints of each communication link. Although this basic CSMA model can provide accurate expressions, the extension of our analysis to a real MAC protocol implementing practical collision avoidance mechanisms is a challenge that needs to be addressed. Furthermore, traffic flows can express their QoS demands using various metrics. For instance, end-to-end delay may be a more important metric to use for real-time traffic. However, jointly considering capacity and end-to-end delay constraints in the routing process is a complex issue. Finally, to integrate specific fairness models in the gateway selection is also an interesting research direction.

References

[1] R. Karrer, A. Sabharwal, E. Knightly, Enabling Large-Scale Wireless Broadband: The Case for TAPs, ACM SIGMOBILE Comp. Comm. Review 34 (1) (2004) 27–34.

[2] R. Bruno, M. Conti, E. Gregori, Mesh Networks: Commodity Multihop Ad Hoc Networks, IEEE Commun. Mag. 43 (3) (2005) 123–131.

[3] I. Akyildiz, X. Wang, W. Wang, Wireless mesh networks: a survey, Computer Networks 47 (4) (2005) 445–487.

[4] Meraki Networks Inc., Residential & MDU Case Studies, http://meraki.com/.

[5] Ozone, French Wireless ISP, http://wwwo.zone.net/.

[6] V. Siris, I. Askoxylakis, M. Conti, R. Bruno, Enhanced, Ubiquitous and Dependable Broadband Access using MESH Networks, ERCIM News 73 (2008) 50–51.

[7] B. Liu, Z. Liu, D. Towsley, On the Capacity of Hybrid Wireless Networks, in: Proc. of IEEE INFOCOM'03, Vol. 2, 2003, pp. 1543–1552.

[8] P. Zou, X. Wang, R. Rao, Asymptotic Capacity of Infrastructure Wireless Mesh Networks, IEEE Trans. Mobile Comput. 7 (8) (2008) 1011–1024.

[9] J. He, J. Chen, S.-H. Chan, Extending WLAN coverage using infrastructureless access points, in: Proc. of IEEE HPSR 2005, 2005, pp. 162–166.

[10] Y. Bejerano, S.-J. Han, A. Kumar, Efficient load-balancing routing for wireless mesh networks, Computer Networks 51 (10) (2007) 2450–2466.

[11] V. Mhatre, F. Baccelli, H. Lundgren, C. Diot, Joint MAC-aware routing and load balancing in mesh networks, in: Proc. ACM CoNEXT'07, New York, USA, 2007, pp. 1–12.

[12] D. De Couto, D. Aguayo, J. Bicket, R. Morris, A High-Throughput Path Metric for Multi-Hop Wireless Routing, in: Proc. of ACM MobiCom, San Diego, CA, USA, 2003, pp. 134–146.

[13] R. Draves, J. Padhye, B. Zill, Routing in Multi-Radio, Multi-Hop Wireless Mesh Networks, in: Proc. of ACM MobiCom'04, 2004, pp. 114–128.

[14] Y. Yang, J. Wang, R. Kravets, Load-balanced Routing For Mesh Networks, ACM Mobile Comp. and Comm. Review (M2CR) 1.

[15] M. Genetzakis, V. Siris, A Contention-Aware Routing Metric for Multi-Rate Multi-Radio Mesh Networks, in: Proc. IEEE SECON'08, San Francisco, CA, USA, June 16–20, 2008, pp. 242–250.

[16] L. Ma, M. Denko, A Routing Metric for Load-Balancing in Wireless Mesh Networks, in: Proc. of IEEE AINAW '07, Vol. 2, 2007, pp. 21–23.

[17] H. Aiache, L. Lebrun, V. Conan, S. Rousseau, A load dependent metric for balancing Internet traffic in Wireless Mesh Networks, in: Proc. IEEE MeshTech'08, Atlanta, GA, USA, September 29, 2008, pp. 629–634.

[18] F. Alizadeh-Shabdiz, S. Subramaniam, A Finite Load Analytical Model for IEEE 802.11 Distributed Coordination Function MAC, in: Proc. ACM WiOpt'03, Sophia-Antipolis, France, 2003.

[19] M. Özdemir, A. McDonald, An M/MMGI/1/K queuing model for IEEE 802.11 ad hoc networks, in: Proc. IEEE PE-WASUN'04, Venice, Italy, 2004, pp. 107–111.

[20] O. Tickoo, B. Sikdar, Modeling Queueing and Channel Access Delay in Unsaturated IEEE 802.11 Random Access MAC Based Wireless Networks, IEEE/ACM Trans. Networking 16 (4) (2008) 878–891.

[21] N. Bisnik, A. Abouzeid, Queuing network models for delay analysis of multihop wireless ad hoc networks, Ad Hoc Networks 7 (1) (2009) 79–97.

[22] R. Bruno, M. Conti, A. Pinizzotto, A Queuing Modeling Approach for Load-Aware Route Selection in Heterogenous Mesh Networks, in: Proc. of IEEE WoWMoM'09, Kos , Greece, 2009.

[23] R. Bruno, M. Conti, A. Pinizzotto, Capacity-Aware Routing in Heterogeneous Mesh Networks: An Analytical Approach, in: Proc. of IEEE MsWiM'09, Tenerife , Canary Islands, Spain, 2009.

[24] P. Calhoun, M. Montemurro, D. Stanley, Control And Provisioning of Wireless Access Points (CAPWAP) Protocol Specification, RFC 5415 (March 2009).
URL `http://www.ietf.org/rfc/rfc5415.txt`

[25] G. Bolch, S. Greiner, H. de Meer, K. S. Trivedi, Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications, WileyBlackwell, 2006.

[26] Y. Gao, D.-M. Chiu, J. C. Lui, Determining the end-to-end throughput capacity in multi-hop networks: methodology and applications, SIGMETRICS Perform. Eval. Rev. 34 (1) (2006) 39–50.

[27] A. Kumar, E. Altman, D. Miorandi, M. Goyal, New insights from a fixed-point analysis of single cell IEEE 802.11 WLANs, IEEE/ACM Trans. Networking 15 (3) (2007) 588–601.

[28] S. Waharte, R. Boutaba, Totally Disjoint Multipath Routing in Multihop Wireless Networks, in: Proc. IEEE ICC'06, Istanbul, Turkey, 2006.

[29] A. Kashyap, S. Ganguly, A. Das, S. Banerjee, VoIP on Wireless Meshes: Models, Algorithms and Evaluation, in: Proc. IEEE INFOCOM'07, Anchorage, USA, 2007, pp. 2036–2044.

[30] Y. Yang, J. Wang, R. Kravets, Designing Routing Metrics for Mesh Networks, in: Proc. of IEEE WiMesh, 2005, Santa Clara, CA, USA, 2005.

[31] E. Ancillotti, R. Bruno, M. Conti, Design and Performance Evaluation of Throughput-Aware Rate Adaptation Protocols for IEEE 802.11 Wireless Networks, Performance Evaluation.

[32] T. L. Fondation, Netem Tools (October 2009).
URL `http://www.linuxfoundation.org/`

[33] T. Clausen, P. Jaquet, Optimized Link State Routing Protocol (OLSR), RFC 3626 (October 2003).
URL `http://www.ietf.org/rfc/rfc3626.txt`

[34] K. Ramachandran, I. Sheriff, E. Belding, K. Almeroth, Routing Stability in Static Wireless Mesh Networks, in: Proc. PAM'07, Louvain-la-neuve, Belgium, 2007, pp. 73–82.