Computer Networks 64 (2014) 26–37

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Combining multi-path forwarding and packet aggregation for improved network performance in wireless mesh networks

Giovanni Di Stasi [a,*], Jonas Karlsson [b], Stefano Avallone [a], Roberto Canonico [a], Andreas Kassler [b], Anna Brunstrom [b]

[a] Dipartimento di Ingegneria Elettrica e Tecnologie dell'Informazione, Universita di Napoli Federico II, Via Claudio 21, 80125 Naples, Italy
[b] Department of Computer Science, Karlstad University, Universitetsgatan 2, 651 88 Karlstad, Sweden

ARTICLE INFO

Article history:
Received 22 December 2012
Received in revised form 5 November 2013
Accepted 25 January 2014
Available online 31 January 2014

Keywords:
Wireless mesh network
Multi-path routing
Packet aggregation

ABSTRACT

Wireless mesh networks (WMNs) based on the IEEE 802.11 standard are becoming increasingly popular as a viable alternative to wired networks. WMNs can cover large or difficult to reach areas with low deployment and management costs. Several multi-path routing algorithms have been proposed for such kind of networks with the objective of load balancing the traffic across the network and providing robustness against node or link failures. Packet aggregation has also been proposed to reduce the overhead associated with the transmission of frames, which is not negligible in IEEE 802.11 networks. Unfortunately, multi-path routing and packet aggregation do not work well together, as they pursue different objectives. Indeed, while multi-path routing tends to spread packets among several next-hops, packet aggregation works more efficiently when several packets (destined to the same next-hop) are aggregated and sent together in a single MAC frame. In this paper, we propose a technique, called *aggregation aware forwarding*, that can be applied to existing multi-path routing algorithms to allow them to effectively exploit packet aggregation so as to significantly increase their network performance. In particular, the proposed technique does not modify the path computation phase, but it just influences the forwarding decisions by taking the state of the sending queues into account. We demonstrated our proposed technique by applying it to Layer-2.5, a multi-path routing and forwarding paradigm for WMNs that has been previously proposed. We conducted a thorough performance evaluation by means of the ns-3 network simulator, which showed that our technique allows to increase the performance both in terms of network throughput and end-to-end delay.

1. Introduction

Wireless mesh networks (WMNs) are becoming increasingly popular thanks to their ability to provide Internet connectivity in wide or difficult to reach areas with low deployment and management costs. In order to be able to transmit simultaneously on different channels and therefore increase performance, mesh nodes are recently being equipped with multiple radios. The presence of multiple radios leads to the channel assignment problem, i.e. how to select the channels to set the radios on. In addition to the channel assignment problem there are also the routing and forwarding problems, i.e., respectively, how to compute paths between each couple of sending and receiving nodes and how to forward packets along the available paths. The channel assignment, the routing and forwarding problems have been shown to be

* Corresponding author. Tel.: +39 3336383732.
E-mail address: giovanni.distasi@unina.it (G. Di Stasi).

http://dx.doi.org/10.1016/j.comnet.2014.01.006
1389-1286/

inter-correlated and in need to be solved jointly [1]. In particular, routing (and forwarding) algorithms should be able to route packets in a way not to exceed the available bandwidth of each link. Such a strategy allows to better utilize the intrinsically scarce resources of WMNs and, in particular, not to overload links, which would cause increased contention and therefore a waste of resources. If properly designed, routing algorithms based on the multi-path paradigm are able to achieve such an objective, as shown in [2]. Moreover, multi-path routing algorithms have the advantage of being able to provide robustness against link or node failures thanks to the availability of different redundant end-to-end paths.

Packet aggregation also helps to improve the capacity of wireless networks by aggregating several packets into a single transmission unit. Such a strategy reduces the overhead due to the transmission of headers and to the inter-frame spaces provided by the contention-based DCF (Distributed Coordination Function), the channel access mechanism of IEEE 802.11 networks [3]. Recently, the IEEE 802.11n amendment has also adopted frame aggregation as a key feature for increasing performance [4].

Routing and forwarding paradigms for WMNs that adopt the multi-path paradigm cannot fully exploit the possibility of packet aggregation since they tend to spread packets among several next-hops. This diminishes the efficiency of packet aggregation, as packets that are sent to different next-hops cannot be aggregated by the packet aggregation procedure. For that reason, packet aggregation is far less effective in the case of multi-path routing and forwarding than in the case of single path routing [5].

Given these considerations, in this paper we present a technique, that we call *aggregation aware forwarding*, which can be applied to a large class of existing multi-path routing and forwarding paradigms to make them efficiently exploit packet aggregation. Our proposal consists of an aggregation sub-layer and a technique to influence the forwarding decisions with the objective of increasing the obtained packet aggregation. The aggregation sub-layer aggregates packets by making use of a nonwork-conserving approach in scheduling the transmission of packets. Indeed, the addition of a small delay at each hop of the network has been demonstrated to allow the aggregation of more packets, with a consequent significant increase in the network throughput in a not fully loaded network [6]. Forwarding decisions are instead altered by taking the state of the sending queues into account, so as to exploit *aggregation possibilities* when they arise. The concept of aggregation possibility is related to the fact that a certain sending queue can be chosen allowing the packet that is being forwarded to be aggregated with the packets already in the queue and sent together with them. We formally define the concept of aggregation possibility in Section 3.

The class of multi-path forwarding paradigms to which such technique is applicable includes algorithms that can select the next-hop among a set of *potential next-hops* depending on the values of a set of *weights* associated with the outgoing links. Examples of forwarding paradigms belonging to such class are Layer-2.5 [7], the MPLS-splitting forwarding strategy described in [8] and the *anypath* routing proposed in [9]. The technique we propose works by modifying such weights, on a per-packet basis, so as to increase packet aggregation. Other multi-path routing algorithms that do not use weights could however take advantage of the ideas described hereafter by modifying the way routing decisions are made in order to take packet aggregation into account.

To show the feasibility and the improvements that can be obtained with the proposed technique, we applied it to Layer-2.5 [7], a multi-path forwarding approach aiming to respect the bandwidth limits on links imposed by the channel assignment algorithm and to provide robustness against failures of single links and nodes. We conducted simulation studies by using the ns-3 network simulator to demonstrate the performance improvements enabled by taking packet aggregation opportunities into account in the Layer-2.5 forwarding decisions.

The rest of the paper is structured as follows. Section 2 introduces in general terms packet aggregation. Section 3 introduces the proposed aggregation-aware forwarding technique. Section 4 introduces Layer-2.5 and the forwarding paradigm we derived from it by applying the proposed technique. Section 5 presents the simulation studies we performed. Section 7 relates our work to current state-of-the-art. Finally, Section 8 concludes the paper and also presents future works.

2. Packet aggregation in wireless mesh networks

The basic idea of packet aggregation is to aggregate several packets into a single transmission unit, called *aggregation packet*. Such strategy allows to reduce the number of MAC layer transmissions and the related overhead thus significantly reducing contention for highly congested links. Packet aggregation can be performed either *end-to-end*, i.e. by the sender and the receiver, or *hop-by-hop*, i.e. by each host independently. In the former strategy, the sender aggregates its own packets and the destination disassembles them once received. In the latter strategy, instead, each hop disassembles incoming aggregation packets and then hands the resulting packets to the forwarding layer. The forwarding layer then selects a next-hop for each packet and passes the packets onto the lower layers that aggregate packets destined to the same next-hop and send them. The *hop-by-hop* aggregation strategy usually allows to obtain a higher amount of aggregation, since it can allow to aggregate packets originated from/destined to different sources/destination. Moreover, disassembling packets at each hop allows to increase packet aggregation by altering the forwarding decisions (which is what we try to exploit in our work).

The efficiency of the packet aggregation strategy can be improved by making use of a non-work-conserving approach when scheduling the transmission of packets, consisting in the addition of a small delay at each hop. Such strategy can be conveniently applied to delay-tolerant traffic, such as p2p traffic, video on-demand, ftp traffic, and emails, which indeed constitutes a major part of today's network traffic [10,11].

Our approach provides the introduction of a distinct queue (*sending queue*) for each next-hop. As a result of the forwarding decision, a packet is placed in one of such queues and likely delayed in the attempt to form an aggregation packet together with other packets being placed in the same queue. Instead, packets that cannot be delayed, such as routing messages or real-time traffic, are placed in a *priority queue* and sent as soon as possible.

3. Aggregation aware forwarding

Our proposal to efficiently exploit packet aggregation consists of two components: an *aggregation sub-layer* and a set of *aggregation multipliers*. The former is an additional sub-layer that can be used in conjunction with any multi-path routing protocol and serves the purpose of aggregating packets being forwarded to the same next-hop. The latter is a set of coefficients that can be adopted by multi-path routing protocols that make forwarding decisions based on some weights assigned to each of the next-hops. The goal of such coefficients is to influence the forwarding decisions in such a way to exploit and increase packet aggregation possibilities.

The aggregation multipliers can be computed based on the information provided by the aggregation sub-layer. In the following subsections, we describe the two components of our proposed approach.

3.1. Aggregation sub-layer

The aggregation sub-layer (Fig. 1) sits on top of the MAC layer and serves the purpose of aggregating packets coming from the upper layer and disassembling packets coming from the lower layer. In addition, it provides an API (Application Programming Interface) that can be queried by the upper layer to get some information about its internal state.

For each interface, the aggregation sub-layer keeps a sending queue for each next-hop reachable through that interface, and a priority queue. Each sending queue is used to store and aggregate delay-tolerant packets destined to the corresponding next-hop, while the priority queue is used to store non-delay-tolerant packets, e.g. control packets.

When a packet is received by the aggregation sub-layer, it is timestamped and put into one of the queues associated with the outgoing interface determined by the forwarding decision (the priority queue if the packet cannot be delayed, the sending queue corresponding to the selected next-hop otherwise). The timestamp is used later to determine how long the packet has been waiting in the queue.

A packet is kept in the sending queue until an *aggregation event* for that queue occurs. An aggregation event consists in aggregating as many packets as possible from the queue (respecting the order of packet arrivals) and handing the resulting aggregation packet to the MAC layer. An aggregation event occurs when either the sum of the packet sizes exceeds the maximum MSDU (MAC Service Data Unit) size or the first packet has stayed in the queue for a certain amount of time. Such amount of time, denoted as *AggregationMaxDelay*, is a parameter indicating the maximum delay a packet can experience due to the aggregation procedure. When the network traffic is low, this parameter introduces an artificial delay, while in case of high network load, no additional delay is likely added. Indeed, it is very likely that, under high load, an aggregation event is triggered because enough packets have arrived to exceed the maximum MSDU size or the packets have been delayed by the medium access layer for a time longer than the *AggregationMaxDelay* time.

3.2. Aggregation multipliers

We consider a multi-path routing protocol that makes forwarding decisions based on a set of weights, each assigned to one of the next-hops. In order to further increase the aggregation possibilities, we propose to scale each of such weights by an *aggregation multiplier*, i.e., a coefficient whose value is a function of the state of the sending queue of the corresponding next-hop. The proposed function takes as inputs the size of the packet to forward, the maximum MSDU size and the *available space* of the queue. The available space of a sending queue on node u for packets destined to neighbor v, denoted as $AS(u \to v)$, is defined as:

$$AS(u \to v) = max\_MSDU\_size - \sum_{p \in Q_{u \to v}} size(p) \quad (1)$$

where $Q_{u \to v}$ is the set of packets destined to node v still in the sending queue on node u. Such packets are stored in the sending queue, as previously described, either because

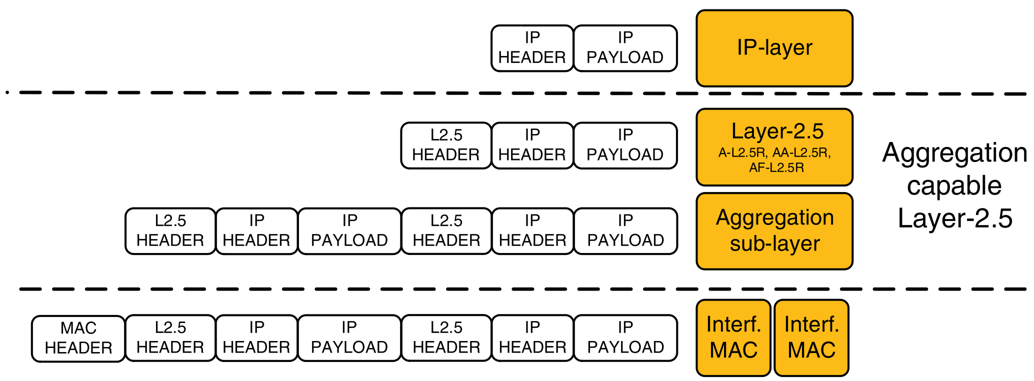


Fig. 1. Aggregation sub-layer.

the aggregation maximum delay has not elapsed yet or because the MAC layer has not yet managed to send them (because of, e.g., network congestion).

We observe preliminarily that if the available space of a queue (Eq. (1)) exceeds the size of the packet to be forwarded, the packet can be aggregated with the packets already in that queue and sent together with them in a single MAC frame (as the sum of their lengths is less than the maximum MSDU size), thus saving a transmission. We define this situation as an *aggregation possibility*. Such an aggregation possibility, if exploited, allows to send the packet without requiring an additional transmission for it.

Incidentally, we observe that it is not always beneficial to exploit aggregation possibilities, since that might lead to often select a next-hop (associated with the chosen sending queue) that is on a long path, thus wasting resources. Indeed, the forwarding scheme called AA-L2.5R (Aggregation-Aware Layer-2.5) [5] always exploit aggregation possibilities and, as shown in the performance evaluation section, routes packets on paths that are on average much longer than required. The technique we propose tries instead to balance the achieved packet aggregation and the resulting path lengths.

If we denote by $w(u \rightarrow v)$ the weight that node u associates with its neighbor v, the modified weight that node u associates with node v when forwarding a packet p is, according to our proposed technique:

$$w'(u \rightarrow v) = w(u \rightarrow v) \cdot AggrMul(AS(u \rightarrow v), p)$$

where

$$AggrMul(x,p) = \begin{cases} 1 & 0 < x < size(p) \\ \gamma & size(p) \leqslant x < max\_MSDU\_size \\ \delta & x = max\_MSDU\_size \end{cases} \quad (2)$$

and

$$1 \leqslant \delta \leqslant \gamma$$

The objective of the *AggrMul* function is to increase the probability to choose a sending queue that offers an aggregation possibility, i.e. the available space in the queue is between the packet size and the maximum MSDU size. Such a strategy allows to prefer sending queues where the packet will be aggregated with other packets (the ones already in the queue) and sent together with them in a single transmission.

Secondarily, empty sending queues, i.e. those where the available space equals the maximum MSDU size, are preferred over sending queues where the available space is not enough to fit the current packet. It is better not to choose a sending queue where the available space is less than the packet size, because selecting such a sending queue triggers an aggregation event and implies the loss of the available space of the sending queue. Such available space can instead be saved for (smaller) packets yet to come.

The γ and δ parameters control the extent to which the original weights are altered by the aggregation multipliers. The higher their values, the bigger the importance given to the aggregation possibility.

4. Multi-path routing in wireless mesh networks

The aggregation aware forwarding technique described in the previous section can be applied to a large class of existing multi-path forwarding paradigms. In order to demonstrate its effectiveness, we show how it can be applied to Layer-2.5, a multi-path forwarding paradigm presented in [7]. In the following, we first briefly present the operations of Layer-2.5 and then describe how the proposed packet aggregation technique can be applied to Layer-2.5.

4.1. Layer-2.5

Layer-2.5 (L2.5R) is a multi-path routing and forwarding strategy that aims at utilizing links in proportion to their available bandwidth (as determined by the channel assignment algorithm), while guaranteeing that packets reach the destination in at most a predefined number of hops. Such maximum number of hops equals the length of the shortest path between the source and the destination times an α coefficient ($\alpha \geqslant 1$), which is a configuration parameter of the algorithm. It turns out that a packet sent by node A and destined to B can take *any* path between A and B having a length not greater than α times the length of the shortest path between A and B.

In order to enforce the constraint on the maximum path length, each packet carries an HC^{max} (*hop count max*) field, in the additional Layer-2.5 header, which is initialized to the maximum number of hops allowed when the packet enters the mesh network and is decremented at each hop. The HC^{max} field is used to determine the number of hops that the packet is still allowed to make.

For each destination d, each node u partitions its neighbors into three sets: $\mathfrak{N}_d^+(u)$, which is the set of neighbors which are one additional hop away from the destination (than the node itself); $\mathfrak{N}_d^=(u)$, which is the set of neighbors which have the same distance to the destination; $\mathfrak{N}_d^-(u)$ the set of neighbors which are one hop closer to the destination (than the node itself). When node u has to take a forwarding decision for a packet destined to d, only the neighbors whose distance from the destination is not greater than the HC^{max} value of the packet are considered as candidate next-hops. Then, all the links to such candidate next-hops are considered and the selected next-hop is the one that maximizes the following $\Delta(v)$ function:

$$\Delta_u(v) = \frac{\beta_{uv}\left(\frac{HC_u^{max}}{HC_u(d)+1}\right) \cdot f(u \rightarrow v)}{\sum_{\forall u \rightarrow i} \beta_{ui}\left(\frac{HC_u^{max}}{HC_u(d)+1}\right) \cdot f(u \rightarrow i)} - \frac{b(v)}{\sum_{\forall u \rightarrow i} b(i)} \quad (3)$$

where $f(u \rightarrow v)$ is the *flow-rate* between node u and the generic neighbor v and represents the available bandwidth of that link (as determined by the channel assignment algorithm), $HC_u(d)$ is the distance of node u to destination d and $b(v)$ is the amount of bytes sent to the generic neighbor v (in the last considered period). If we neglect the β function for a moment, $\Delta_u(v)$ represents the gap between the desired link utilization (the link flow-rate), and the actual link utilization. Sending a packet over the link with the largest $\Delta_u(v)$ allows to reduce that gap, in the attempt to keep the actual utilization of all the links close to their desired utilization.

The β function is defined as:

$$\beta_{uv}(x) = \begin{cases} \left(\frac{x-1}{x}\right)^{2\beta} \cdot \mathbf{1}_{[1,+\infty]}(x) & \text{if } v \in \mathfrak{N}_d^+(u), \\ \left(\frac{x-1}{x}\right)^{\beta} \cdot \mathbf{1}_{[1,+\infty]}(x) & \text{if } v \in \mathfrak{N}_d^=(u), \\ 1 & \text{if } v \in \mathfrak{N}_d^-(u) \end{cases}$$

and is introduced to weigh the flow-rates. In particular, such a function increases the flow-rates of links to neighbors which are closer to the destination, so as to increase the probability for those links to be selected. Thus, the β function has the objective to reduce the average path length. Without the β function, as demonstrated in [7], most of the times packets would take paths as long as the maximum length allowed, which would waste network resources (because of the increased number of transmissions required). An example illustrating the operations of Layer-2.5 is shown in Fig. 2.

4.2. AF-L2.5R

We denote by AF-L2.5R (Aggregation and Flow-rate aware Layer-2.5) the forwarding paradigm resulting from applying the proposed packet aggregation technique to Layer-2.5. To this end, we modify the $\Delta(v)$ function (Eq. (3)) used to choose the outgoing link among the feasible ones. In Layer-2.5, a flow-rate representing the bandwidth available on a link to a neighbor is weighted by means of the β function with the objective of decreasing the average path length taken by packets. In AF-L2.5R, a flow-rate is further weighted by the aggregation multiplier associated with the corresponding neighbor in the attempt to exploit the aggregation possibilities and then reduce the number of MSDUs handed by the aggregation sub-layer to the MAC layer.

In the case where both the parameters γ and δ used in the definition of the aggregation multipliers (Eq. (2)) are equal to 1, AF-L2.5R behaves as the original Layer-2.5 (in the sense that forwarding decisions are not altered), with the difference that packet aggregation is performed (due to the presence of the aggregation sub-layer). In the following, we refer to this strategy as Aggregation-Layer2.5 (A-L2.5R). In the next section we compare AF-L2.5R, AA-L2.5R and A-L2.5R to analyze how changing the way forwarding decisions are made can increase packet aggregation and the overall network performance.

5. Performance evaluation

In order to evaluate the performance of AF-L2.5R, we performed simulations by means of ns-3 [12].

The model used for the physical layer is the standard given by ns-3, which is based on the SINR (Signal To Noise Ratio) seen at the receiver. We set the gain of the radio channel between two nodes to be equal to the reciprocal of the square of the distance between them and set the thermal noise to be −20 dbm. Moreover, the SINR thresholds at the receiver are set to allow a rate of 54 Mbps when the nodes are within 30 m, 48 Mbps within 32 m, 36 Mbps within 37 m, 24 Mbps within 45 m, 18 Mbps within 60 m, 12 Mbps within 69 m, 9 Mbps within 77 m, and 6 Mbps within 90 m. The transmission power is set for every radio statically to 15 dbm, while the transmission rate and channel is selected on a link basis according to the output of the FCRA (Flow-based Channel and Rate Assignment algorithm) [7]. We suppose the availability of 6 different orthogonal channels in the IEEE 802.11a band. This assumption derives from considering half of the twelve theoretically available orthogonal channels, as defined by the standard, in order to avoid the ACI (Adjacent Channel Interference) [13] between two consecutive channels.

5.1. Initial experiments

The first series of experiments serves the purpose of evaluating how the results vary for different γ and δ values. The experiments were performed on the topology of Fig. 3, made of 25 nodes in a 300 × 300 m area. Each node was equipped with two or three wireless interfaces. We selected, as indicated in the figure, four nodes as senders and three other nodes as receivers.

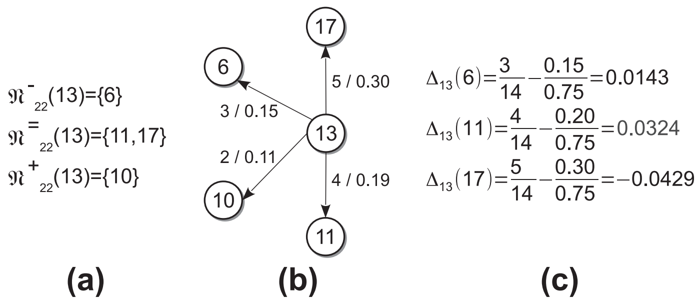


Fig. 2. Example illustrating the operations of L2.5R. Consider the topology shown in Fig. 3 and assume node 13 has to forward a packet destined to node 22 and having an HC^{max} value of 5. (a) Based on the minimum hop count to the destination, the neighbors of node 13 are partitioned in three sets. Node 6 is 3 hops away from the destination, nodes 13, 11 and 17 are 4 hops away and node 10 is 5 hops away. (b) The flow-rate (Mbps) and the amount of megabits sent since the beginning of the current period are shown next to each link. (c) Given the HC^{max} value of the packet, only neighbors in $\mathfrak{N}_d^-(u)$ and $\mathfrak{N}_d^=(u)$ are considered. The neighbor with the largest gap between the desired and actual utilization is node 11, which is selected as next-hop for the packet (we neglected the β function in this calculation). The Δ values are then updated to take into account the last forwarding decision.

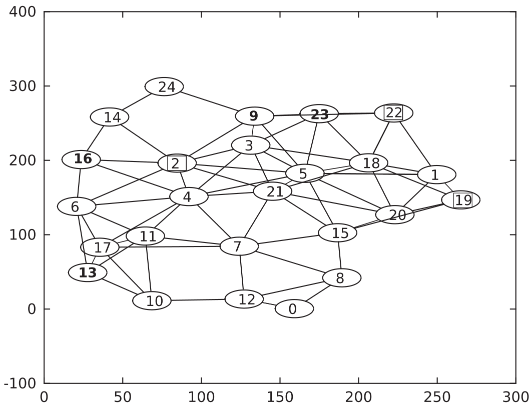


Fig. 3. Wireless mesh topology used for the experiments (senders in bold, receivers inside a square).

To reduce the impact on the end-to-end delay we used an *AggregationMaxDelay* of 3 ms. Such value is able to increase the achieved amount of packet aggregation while having at the same time a limited impact on the end-to-end delay, as shown hereafter. Two backlogged TCP connections, one for each direction, between each couple of sender and receiver nodes was established. We set the MSS (Maximum Segment Size) of TCP to 500 bytes in order for the aggregation sub-layer to be able to aggregate packets before handing them to the wireless interfaces. Such value allows the sender nodes to generate packets with a more realistic average packet size (500 instead of the default 1500). Indeed, the distribution of packet sizes seen in real networks usually shows a high variability and a high amount of small packets (see [14] as an example).

In Fig. 4 the average throughput, delay and aggregation ratio over 20 different repetitions is shown. The standard deviation (not represented for clarity) was very low. The results show how AF-L2.5R is able to improve performances for a wide range of values of the γ and δ parameters compared with A-L2.5R (Aggregation-Layer-2.5), which is represented in the graph by the $\gamma = \delta = 1$ configuration, i.e. the original flow-rates are not altered. In particular, the throughput (Fig. 4) increases of about 20% and the delay decreases of about 55% for γ and δ both equal to 1.2. For the same values, the average size of aggregation packets increases of about 25% (Fig. 4(c)), due to the fact that AF-L2.5R increases packet aggregation by exploiting aggregation possibilities. For this configuration of the parameters other relevant measures, such as the amount of traffic distributed on links, not reported for brevity, do not significantly differ from the same values obtained by running the original Layer-2.5 algorithm.

It is interesting to note, instead, how results get steadily worse while the parameters are increased. This can be explained by the fact that the higher the values of the parameters, the more the original flow-rates are altered in the attempt to increase aggregation. Such an alteration of the flow-rates leads to choose in average longer paths, as shown in Fig. 4(d), and to respect less the bandwidth limits represented by the original flow-rates, which result in reduced performances.

Thanks to this first experiment, we are able to select the best combination of parameters that will be used for the following experiments ($\gamma = \delta = 1.2$).

5.2. Analysis through real traffic traces

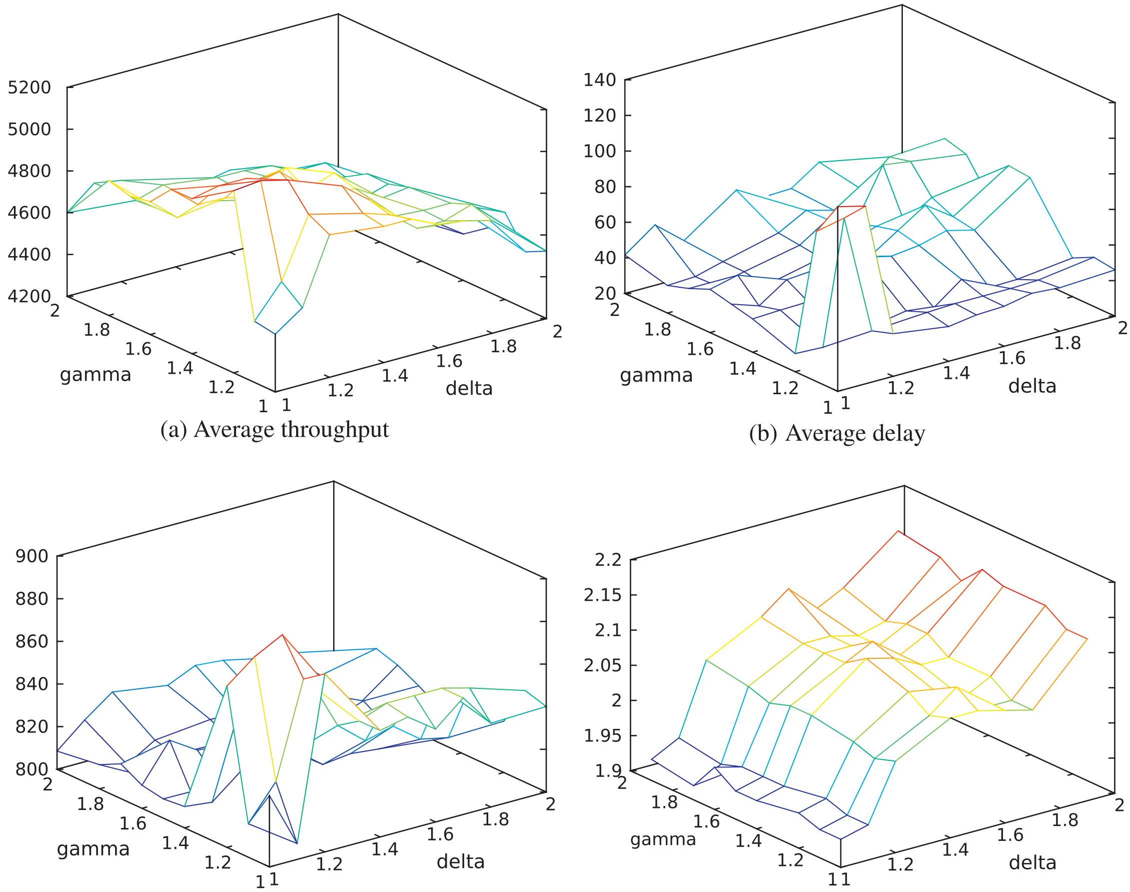
We performed other simulation studies where the traffic injected into the network is based on real traffic traces. We considered six traffic traces collected at the gateway router of the wireless network at the UCSD (University of California, San Diego) Computer Science building [14]. Each of such traces records the traffic collected in one hour. For each trace, we only considered TCP packets and classified each of them as *upstream* or *downstream*. We identified all the TCP SYN segments and recorded the corresponding 4-tuple (IP source address, source port, IP destination address, destination port). Then, all the TCP packets matching a 4-tuple have been marked as upstream (TCP connections have been likely opened by the hosts of the wireless network), while TCP packets having source and destination IP addresses and ports swapped with respect to a 4-tuple have been marked as downstream.

We considered the same topology of the previous experiment and selected three of the nodes as receivers and other three as senders. We established two UDP flows between each couple of sender and receiver nodes. The UDP flow in the sender to receiver direction was generated by considering the packet sizes and the inter-packet departure times of the upstream packets of a trace file; the flow in the opposite direction by considering the downstream packets of the same trace file. In such a way, it is as though the wireless mesh network were used as a back-bone network carrying the traffic generated by the hosts of the UCSD wireless network (with the receivers acting as gateways).

Simulations were performed for values of *AggregationMaxDelay* ranging from 0 to 10 ms. The results are normalized against the result obtained by the standard Layer-2.5 algorithm without packet aggregation, in order to give a better idea on the improvements that can be obtained by employing aggregation (i.e. A-L2.5R) and then by applying our aggregation aware technique (i.e. AF-L2.5R).

Fig. 5(a) reports the throughput obtained by the different forwarding paradigms which shows that AF-L2.5R is able to increase the resulting throughput compared with A-L2.5R, AA-L2.5R and Layer-2.5. In particular, for *AggregationMaxDelay* = 6 ms AF-L2.5R gives, respectively, 15%, 12%, 5% more throughput than AA-L2.5R, Layer-2.5 and A-L2.5R.

As far as delay is concerned, Fig. 5(c) shows that AF-L2.5R and A-L2.5R, despite the use of the non-work conserving approach, do not increase the end-to-end delay significantly (compared with Layer-2.5). Moreover, AF-L2.5R does not increase the delay compared with A-L2.5R, that shows that altering the flow-rates to increase packet aggregation does not lead to additional end-to-end delays. On the contrary, AA-L2.5R increases the delay significantly, mainly because of the use of long paths (see Fig. 5(b)), choosen in the attempt to increase packet aggregation. Overall, the best results of AF-L2.5R can be explained by the increased aggregation packet sizes it can achieve



(a) Average throughput (b) Average delay

(c) Average aggregation packet size (d) Average path length

Fig. 4. Average throughput, end-to-end delay and average packet size achieved for different values of the γ and δ parameters.

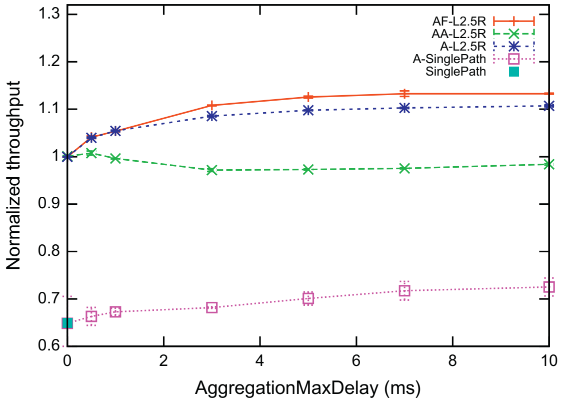
(Fig. 5(d)), and to the fact that, at the same time, the average packet length is kept to a value close to Layer-2.5 and A-L2.5R. Interestingly, when using A-L2.5R and AF-L2.5R, the normalized one way delay decreases with large aggregation max delays. This is because aggregation reduces the number of packets in flight, and thus reduces MAC layer contention, which decreases latency [6]. This beneficial effect is offset by the disadvantageous use of very long paths when using AA-L2.5R.

Note, that when using aggregation in addition to the standard Layer 2.5 forwarding (i.e. A-L2.5R), normalized path length is impacted by the aggregation delay value. This is due to the design of Layer-2.5 routing which tries to forward packets to obey the flow rates as determined by the channel assignment. When enabling standard aggregation on top of Layer 2.5, packet sizes (and thus per neighbor link utilization) are quite different than when not aggregating. As a result, the forwarding decisions as to which neighbor a packet to send are different, if aggregation is enabled or not. As a consequence, normalized path length also changes but the impact is quite small as can be seen from Fig. 5(b).

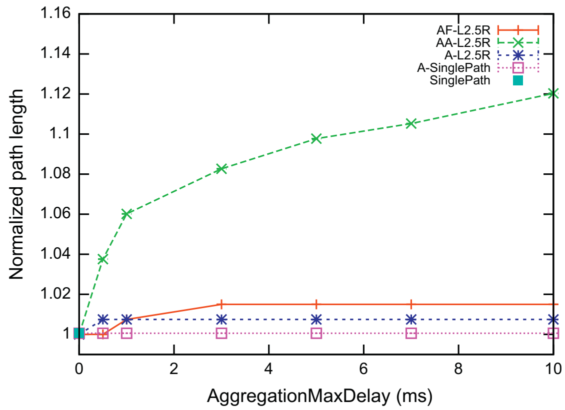
We also compared the multi-path forwarding schemes against two single-path alternatives, denoted as *SinglePath* and *A-SinglePath*. Such forwarding schemes use only the shortest path between each couple of sender and receiver nodes. In addition to that, *A-SinglePath* (Aggregation-SinglePath) also employs packet aggregation, thanks to the use of the aggregation sub-layer. The results related to such forwarding paradigms show a significant reduction in throughput (see Fig. 5(a)) with respect to the multi-path forwarding schemes. Such results can be explained by the fact that the multi-path forwarding schemes are able to exploit different paths between source and destination pairs, which allows to increase the achieved throughput. Moreover, they are able to distribute traffic on links in proportion to the flow-rates assigned to them. Such flow-rates, as previously stated, represent the bandwidth assigned to links by the channel assignment algorithm. On the other hand, single-path routing uses only the shortest paths and does not take into account the capacity of such paths, which may result in congestion for low-bandwidth links and a reduced performance.

5.3. FTP results

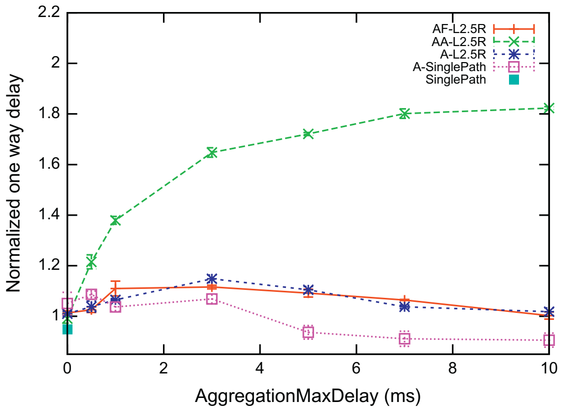
In the following we describe simulations using FTP traffic. Differently from the first experiment, in this case we



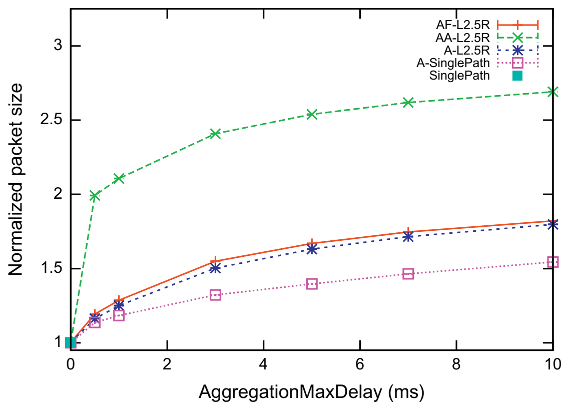
(a) Average throughput measured in all the experiments



(b) Average path length measured in all the experiments



(c) Average delay measured in all the experiments



(d) Average packet sizes measured in all the experiments

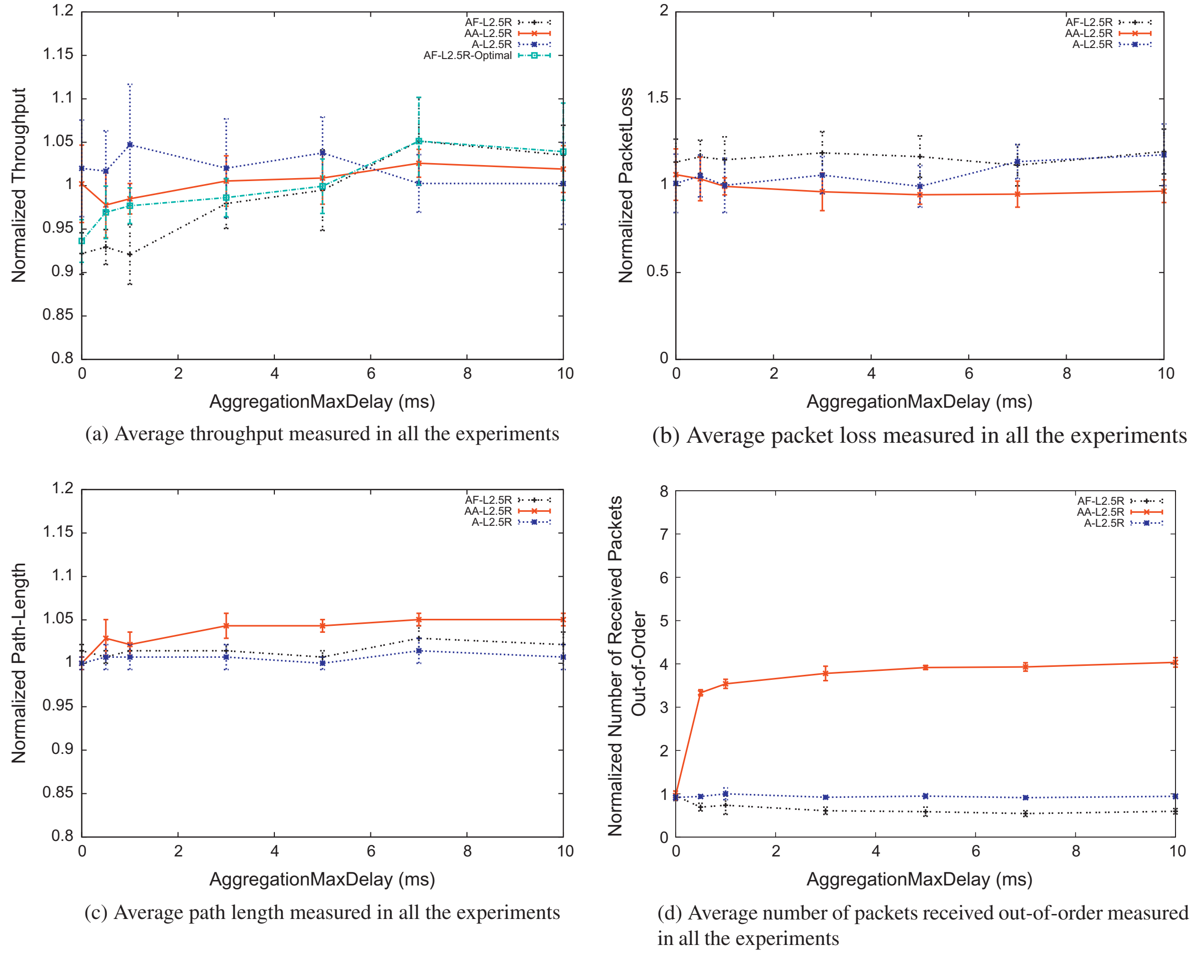
Fig. 5. Simulations with real traffic traces while varying the *AggregationMaxDelay*.

use a MSS of 1400 bytes. As only one TCP data packet plus a few TCP ack packets can be sent in a single aggregation packet, there are very low possibilities for packets aggregation. Moreover, with these settings the network will be fully loaded and the more loaded the network is, the more important it is to respect the flow-rates for good network throughput. Therefore, these simulations represent a worst case scenario for AF-L2.5R, as there is only limited benefit of aggregation and any deviation from the assigned flow-rates can reduce the throughput, due to increased interference.

The simulations were performed for values of *AggregationMaxDelay* ranging from 0 to 10 ms. All the results are averaged over 10 different repetitions and normalized against the result obtained by the standard Layer-2.5 algorithm (without packet aggregation). We also report (in Fig. 6(a)), for each *AggregationMaxDelay*, the results obtained by AF-L2.5R when the values of γ and δ that gave the best throughput are used (we denote this approach as AF-L2.5R-Optimal), while AF-L2.5R uses fixed values ($gamma = delta = 1.2$). We wanted to assess the improvement that could be achieved when the parameters of the algorithms are finely tuned (depending on the topology and the traffic).

Fig. 6(a) shows that, in a scenario where aggregation is of scarce importance, all the algorithms give a very similar throughput. There are minor differences, as for instance for *AggregationMaxDelay* $\leqslant 3$ ms , where A-L2.5R gives a slight throughput advantage, basically because it fulfills the flow-rates and still can utilize the aggregation opportunities that occur. AF-L2.5R has, for *AggregationMaxDelay* > 3 ms, similar throughput as AF-L2.5R-Optimal with $\gamma = 1.2$ and $\delta = 1.2$

In Fig. 6(b), the normalized packet-loss is presented. All strategies except AA-L2.5R lead to an increase of packet loss rates for *AggregationMaxDelay* > 0.5 ms compared with L2.5R. When using AA-L2.5R with an *AggregationMaxDelay* of 5 ms, the packet loss is reduced of 5% compared to L2.5R. Since AA-L2.5R is the proposal that gives the highest importance to aggregation possibilities, it is also the proposal that achieves the highest aggregation ratio. As we are using a TCP MSS of 1400 bytes, it is not possible to aggregate two TCP data-packets together. However, since AA-L2.5R tries to increase future aggregation possibilities, each TCP data-packet is put into a queue to a separate neighbor. This will effectively spread the TCP data-packets on as many paths as possible, which can increase the packet reordering. This can be observed from Fig. 6(d), where



(a) Average throughput measured in all the experiments

(b) Average packet loss measured in all the experiments

(c) Average path length measured in all the experiments

(d) Average number of packets received out-of-order measured in all the experiments

Fig. 6. Simulations with FTP traffic while varying the *AggregationMaxDelay*.

the number of packets received out of order is around four times higher for AA-L2.5R when compared to (A-)L2.5R. However, since AF-L2.5R balances the path length and the aggregation possibilities during its forwarding decision, AF-L2.5R tries to send packets to the same next hop. This results in more packets which are sent on the same path thus effectively controlling the number of out-of-order packets.

A-L2.5R and L2.5R prioritize shorter paths while all other proposals prioritize to some extent aggregation over path length. AF-L2.5R and, especially, AA-L2.5R can thereby potentially increase the average path length. This can reduce the available network resources as longer paths consume more network resources than shorter paths. In Fig. 6(c) we show the average path length normalized against to L2.5R. Since AA-L2.5R does not take path length into account, it has the overall highest path length. A-L2.5R prioritizes path length and is therefore on average using the shortest paths than the other proposals. AF-L2.5R uses slightly longer paths than A-L2.5R, but can reduce the average path length compared to AA-L2.5R, as it considers the path length as one optimization metric.

For an *AggregationMaxDelay* higher than 3 ms, all proposals have similar throughput. However, while AA-L2.5R reduce the consumed network resources by having the lowest amount of packet loss it also uses longest paths and has the highest amount of reordered packets, which wastes network resources. In this respect, AF-L2.5R is able to give a good balance between the throughput, and the amount of network resources consumed in the form of average path length used and the packet loss ratio.

We also run the same set of experiments by employing the single path routing algorithm, both with or without aggregation. The results showed that the singlepath algorithm is in average 10, or 15% slower than the multi-path versions, basically, as seen for the trace based case, for congestion on the used paths.

6. Stability and convergence analysis

We performed some additional tests to evaluate the behavior of the proposed AF-L2.5R algorithm with regards to routing stability and convergence. The purpose is to assess if the modified Layer 2.5 algorithm still shows a stable behavior in distributing traffic on link, despite the use of aggregation multipliers to modify flow-rates of links.

Such tests were performed on the topology of Fig. 3, with the same setup of previous experiments, apart from the use of constant bitrate UDP flows between each couple of sender and receiver nodes and the choice of a fixed *AggregationMaxDelay* of 3 ms.

In Fig. 7 we report the average throughput on four network links, calculated in time windows of both 30 s (dashed lines) and 1 s (continuous lines). Such selected four links were the ones whose throughput resulted to be the most variable (in the sense specified hereafter). In the follows we analyze the link with the most variable throughput among them, i.e. link between nodes 2 and 3. Such link has an average throughput, in the time window 10–120 s, of 3272 kbit/s. The average throughput in 30 s time windows, instead, has a minimum, in the time windows 10–40, of 2995 kbit/s, and a maximum, in the time window 70–100, of 3464 kbit/s. The difference among such values and the average throughput is respectively −277 kbit/s and 192 kbit/s, which represent a deviation from that average of respectively 8.5% and 5.8%.

We believe that such small maximum value (below 10%) for the deviation from the average throughput is a good indicator of the stability of the algorithm. Indeed, such limited variability of throughput can be ascribed to contention (regulated by the IEEE 802.11 DCF) due to the sharing of common transmission channels by links.

7. Related work

Packet aggregation, as previously stated, is a well known technique to improve network throughput by reducing the amount of small frames and has been studied in several works [3,15–17]. To increase the obtained aggregation ratio, some works adopt a non-work-conserving approach, consisting in adding a small forced delay at each hop [18]. In our work we adopt a similar strategy, by defining an aggregation sub-layer which differentiates among delay and non-delay tolerant traffic and in addition supports the providing of information to the upper layer to improve the forwarding process.

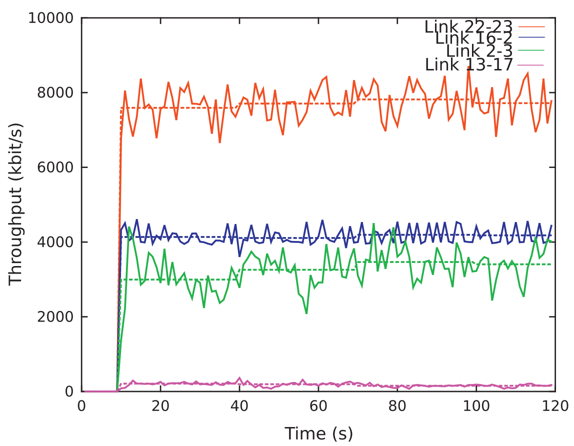


Fig. 7. Throughput over time for different network links in windows of 1 and 30 s.

The IEEE 802.11n amendment [4] supports frame aggregation, which can be done in two steps. The first step is to aggregate multiple MSDUs to form an A-MSDU whose maximum size is 7935 bytes. Depending on the available PHY layer, a second step is possible, in which multiple MPDUs are aggregated in a single A-MPDU whose maximum size is 65,535 bytes. The IEEE 802.11n frame aggregation and our proposed aggregation techniques are not alternatives, but complement each other. Indeed, the former can only aggregate frames destined to the same receiver and the aggregation multipliers we defined just have the objective to send packets to the same neighbor in order to increase packet aggregation. On the other hand, our approach benefits from the presence of the 802.11n frame aggregation because the overhead associated with the transmission of frames is further reduced.

Aggregation with QoS support has been shown to be beneficial in multi-hop scenarios with mixed UDP and TCP traffic, where VOIP capacity increased greatly [19]. Similar to our algorithm, the algorithm proposed in [19], uses multiple queues to buffer packets. However, in [19], packets are divided according to their priority class, ranging from best effort to high priority packets and therefore restrict aggregation to packets within the same class. In this work we evaluate multi-path routing and combined with packet aggregation using both TCP and UDP.

The benefits of robust header compression combined with packet aggregation is evaluated in [20]. The experiments showed up to 10 times improvement by combining header compression and packet aggregation in a small WMN. The authors used UDP packets of 20 bytes, which are easily aggregated. In this work we focus on much larger (up to 1400 bytes) UDP and TCP packets, and focus on the impact of combining multi-path routing and packet aggregation.

The impact of packet aggregation on TCP performance with different traffic patterns was evaluated in [18]. Aggregation can substantially improve TCP throughput and reduce TCP round trip times.

In [5], the authors studied jointly the problems of multi-path forwarding and packet aggregation. The experimental evaluation showed promising results but, due to the choice of giving absolute priority to aggregation, the proposal limited the possibility to effectively load-balance traffic according to flow-rates,i.e. link weights. In this work we have extended that idea and generalized it in order to combine forwarding and packet aggregation while at the same time respecting the flow-rates resulting from the channel assignment algorithm.

8. Conclusions and future work

In this work we propose a novel technique that can be applied to existing multi-path routing and forwarding paradigms to allow them to effectively exploit packet aggregation and therefore increase network performances. In particular, the proposed technique does not require to modify the path construction phase but just to slightly change the way the forwarding decisions are made so as to take the state of the sending queues into account. Moreover, the technique provides the usage of an aggregation sub-layer, which is able to perform packet aggregation and to provide information through an API to improve the forwarding process. To the best of our knowledge, ours is the first attempt to combine multi-path routing and packet aggregation to improve performances.

Extensive simulation demonstrate that the technique is able to increase network throughput and reduce the end-to-end delay in several scenarios. In particular, in case of backlogged TCP connections with a packet size of 500 bytes, the technique can increase network throughput of about 20% and decrease the delay up to 55%. In case of UDP flows generated from real traffic traces, the technique also allows to improve performance, by up to about 10%. In case packet aggregation cannot be performed, i.e. packets of size that approximate the maximum MAC MSDU, the technique gives results which are in line with the ones obtained by the original algorithm.

The technique, as currently implemented, consists on altering the weights associated to the outgoing links by means of a piecewise constant function (*AggrMul*), which loosely depends on the state of the sending queues. As a future work, we plan to modify such a function by making it more finely consider the state of the sending queues (by altering, as an example, the flow-rates depending on the amount of available space of the sending queue). Exploiting the same concept, a future work may also consist in altering the original weights in dependence of other MAC or physical layer attributes, such as the quality of the physical links (by decreasing, as an example, the weights for low-quality links).

Finally, we argue that the performance of our proposed technique can be further improved if we have knowledge that the 802.11n frame aggregation mechanism is available at the MAC layer. In such a case, we might exploit the fact that multiple MSDUs destined to the same receiver are aggregated at the MAC layer and, for instance, alter the order in which packets are handed to the MAC layer, so that consecutive packets have to be sent to the same neighbor. Investigating such improvements is left for future work.

Acknowledgments

This research is supported by Grant YR2009-7003 from Stiftelsen för internationalisering av högre utbildning och forskning (STINT).

References

[1] A. Raniwala, K. Gopalan, T. Chiueh, Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks, ACM SIGMOBILE Mob. Comput. Commun. Rev. 8(2), 2004.

[2] S. Avallone, F.P. D'Elia, G. Ventre, Layer-2.5 routing in multi-radio wireless mesh networks, in: Proceedings of IEEE WiMesh, IEEE, 2009, pp. 19–24. doi:http://dx.doi.org/10.1109/SAHCNW.2009.5172942.

[3] A. Jain, M. Gruteser, M. Neufeld, D. Grunwald, Benefits of packet aggregation in ad-hoc wireless network, Technical Report CU-CS-960-03, Department of Computer Science, University of Colorado at Boulder, August 2003.

[4] IEEE Standard for Information Technology – Telecommunications and Information Exchange between Systems – Local and Metropolitan Area Networks – Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughput, IEEE Std 802.11n-2009, 2009, pp. 1–502.

[5] J. Karlsson, G. Di Stasi, A. Kassler, S. Avallone, An aggregation aware multi-path forwarding paradigm for wireless mesh networks, in: Proceedings of the 8th IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (MASS'11), Valencia, Spain, 2011.

[6] M. Castro, P. Dely, J. Karlsson, A. Kassler, Capacity increase for voice over IP traffic through packet aggregation in wireless multihop mesh networks, in: Proceedings of the Future Generation Communication and Networking (FGCN'07), Jeju, Korea, 2007.

[7] S. Avallone, I.F. Akyildiz, G. Ventre, A channel and rate assignment algorithm and a Layer-2.5 forwarding paradigm for multi-radio wireless mesh networks, IEEE/ACM Trans. Network. 17 (1) (2009) 267–280.

[8] S. Avallone, G. Di Stasi, A new MPLS-based forwarding paradigm for multi-radio wireless mesh networks, IEEE Trans. Wirel. Commun. 12 (8) (2013) 3968–3979, http://dx.doi.org/10.1109/TWC.2013.071113.121529.

[9] R. Laufer, H. Dubois-Ferriere, L. Kleinrock, Multirate anypath routing in wireless mesh networks, in: Proceedings of the 28th IEEE Conference on Computer Communications (INFOCOM '09), Rio de Janeiro, Brazil, 2009.

[10] T. Karagiannis, A. Broido, N. Brownlee, K. Claffy, M. Faloutsos, Is p2p dying or just hiding?, in: Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM'04), Dallas, USA, 2004.

[11] A. Dainotti, A. Pescapè, K. Claffy, Issues and future directions in traffic classification, IEEE Network 26(1), 2012.

[12] T. Henderson, S. Roy, S. Floyd, G. Riley, ns-3 project goals, in: Proceeding from the 2006 Workshop on ns-2: The IP Network Simulator (WNS2 '06), Pisa, Italy, 2006.

[13] M. Castro, A. Kassler, S. Avallone, Measuring the impact of ACI in cognitive multi-radio mesh networks, in: Proceedings of the 72nd IEEE Vehicular Technology Conference (VTC '10), Ottawa, Canada, 2010.

[14] Y.-C. Cheng, Crawdad data set ucsd/cse (v. 2008-08-25).

[15] D. Kliazovich, F. Granelli, Packet concatenation at the IP level for performance enhancement in wireless local area networks, Springer Wirel. Netw. 14(4), 2008.

[16] Y. Xiao, J. Rosdahl, Throughput and delay limits of IEEE 802.11, IEEE Commun. Lett. 6(8), 2002.

[17] K. Kim, S. Hong, VoMESH: voice over wireless mesh networks, in: Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC'06), Las Vegas, USA, 2006.

[18] J. Karlsson, A. Kassler, A. Brunstrom, Impact of packet aggregation on TCP performance in wireless mesh networks, in: Proceedings of the 10th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WOWMOM'09), Kos, Greece, 2009.

[19] R. Riggio, D. Miorandi, F.D. Pellegrini, F. Granelli, I. Chlamtac, A traffic aggregation and differentiation scheme for enhanced QoS in IEEE 802.11-based wireless mesh networks, Elsevier Comput. Commun. 31(7), 2008.

[20] S. Jung, S. Hong, P. Park, Effect of RObust Header Compression (ROHC) and packet aggregation on multi-hop wireless mesh networks, in: Proceedings of the 6th IEEE International Conference on Computer and Information Technology (CIT'06), Seoul, Korea, 2006.

Giovanni Di Stasi is a post-doc at the Computer Engineering department of University of Napoli "Federico II". He received the Laurea degree in Computer Engineering from University of Napoli Federico II in 2007. In 2008 he worked at CINI (Consorzio Interuniversitario Nazionale per l'Informatica) for the European Research Project ONELAB. In 2009 he was a visiting member of the ONELAB engineering group at INRIA (Sophia Antipolis, France) under the supervision of Dr. Thierry Parmentelat, technical director of the ONELAB project. His current research interests include experimental research infrastructures and testbeds, routing and channel assignment algorithms for wireless mesh networks, peer-to-peer traffic optimization, network emulation and simulation. He is a member of IEEE.

Jonas Karlsson is a research assistant at Karlstad University, Sweden. He received a B.Sc. and M.Sc. in Computer Science and Engineering from Karlstad University, in 2003 and 2006, respectively. In 2011 he completed a Licentiate degree in engineering at the Computer Science Department at Karlstad University. He was a visiting member of the Network Research Lab at University of California, Los Angeles, CA and Computer Networking Group at University of Napoli "Federico II", Italy, in 2009 and 2011 respectively. His current research interests include performance enhancing techniques in wireless mesh networks, multi-path transport layer algorithms, mobile and tactical networks, network emulation and simulation.

Stefano Avallone is Assistant Professor at the Department of Computer Engineering and Systems of the University of Napoli "Federico II". He received the M.S. degree in Telecommunications Engineering (2001) and the Ph.D. degree in Computer Networks (2005) from the University of Napoli "Federico II". His research interests include computer networks, traffic engineering, QoS routing, wireless mesh networks. He was a visiting researcher at the Delft University of Technology (2003–2004) and at the Georgia Institute of Technology (2005). In 2004 he was awarded a research funding from the European Doctoral School of Advanced Topics in Networking (SATIN), the instrument employed by E-NEXT (an EU FP6 Network of Excellence) to invest in education of researchers for the European Research Area.

Roberto Canonico is Associate Professor at University of Napoli Federico II. He received the Laurea degree (cum laude) in Electronic Engineering from University of Napoli Federico II in 1995, and a Ph.D. in Computer Engineering from the same University in 2000. In 2000, he was a visiting member of the Distributed Multimedia Research Group at Lancaster University, UK. Roberto has been involved in several European Research Projects, such as CADENUS, INTERMON and ONELAB, and Networks of Excellence, such as E-NEXT and CONTENT. His current research interests include experimental research infrastructures and testbeds, network virtualization, cloud computing, peer-to-peer traffic optimization, overlay networks, network emulation and simulation.

Andreas J. Kassler is Professor of Computer Science at Karlstad University, Karlstad, Sweden, that he joined in 2005. From 2003 to 2004, he was Assistant Professor at the School of Computer Engineering, Nanyang Technological University, Singapore. His research interests are in the area of Wireless Meshed Networks, Ad-Hoc Networks, Multimedia Networking, Quality of Service, and P2P systems. He has published over 100 conference and journal papers, and several book chapters. He received the Docent title in Computer Science from Karlstad University in 2006, the Ph.D. degree in Computer Science from Universit Ulm, Germany, in 2002 and an M.S. degree from Universit Augsburg, Germany. Prof. Kassler is a Member of the IEEE.

Anna Brunstrom (1967) received a B.Sc. in Computer Science and Mathematics from Pepperdine University, CA, in 1991, and a M.Sc. and Ph.D. in Computer Science from College of William & Mary, VA, in 1993 and 1996, respectively. She joined the Department of Computer Science at Karlstad University, Sweden, in 1996, where she is currently a Full Professor and Research Manager for the Distributed Systems and Communications Research Group. She has a background in distributed systems, but her main area of work over the last years has been in computer networking with a focus on transport protocol design, QoS issues, cross-layer interactions, wireless communication and network security. She has authored/coauthored ten book chapters and over 100 international journal and conference papers.