2011 Eighth IEEE International Conference on Mobile Ad-Hoc and Sensor Systems

# An Aggregation Aware Multi-path Forwarding Paradigm for Wireless Mesh Networks

Jonas Karlsson, Giovanni Di Stasi, Andreas Kassler, Stefano Avallone

***Abstract*—In multi-radio wireless mesh networks, a network device simultaneously transmits packets over different channels by using multiple radios. Such frequency diversity not only increases throughput but makes multi-path routing approaches extremely interesting. This is because the channel diversity reduces the risk for intra- and inter-flow interference. A fundamental problem to solve is the forwarding strategy which determines which packets to be sent over what multi-path segments at any given time. Ideally, the forwarding strategy should schedule flows according to the capacity constraints imposed by the channel assignment. However, the possibility to improve MAC layer efficiency by aggregating small packets into larger ones is reduced when packets are forwarded to different next-hops. In this paper, we develop a novel packet forwarding strategy for multi-radio mesh networks that combines the benefits of multi-path routing with packet aggregation. In our cross-layer approach, we effectively trade-off aggregation opportunities with channel diversity. Simulation results show that our approach can improve network throughput and delay by up to 15 percent and 25 percent, respectively, compared with aggregation unaware forwarding strategies.**

***Keywords*-Wireless Mesh Networks, Multi-channel, Multi-radio, Multi-path, Packet Aggregation.**

## I. Introduction

Wireless mesh networks (WMNs) are considered to be a new and promising technique to provide Internet connectivity for, among others, cities, rural areas and user-communities. In this work, we focus on IEEE 802.11 [1] based mesh networks, which share many properties and problems with multi-hop ad-hoc networks such as intra- and inter flow interference and high overhead for sending small packets due to the properties of the MAC layer.

Interference can be reduced by using multiple radios tuned to different channels. IEEE 802.11b/g and IEEE 802.11a have, theoretically, three and thirteen (as defined by FCC) orthogonal (non-overlapping) channels [2]. Using multiple channels greatly improves the network capacity by reducing contention on highly occupied links. In addition, multi-path routing allows to better utilize the channel diversity as a flow can be sent over multiple paths, which further reduces the risk of intra and inter-flow interference if the path segments are on different frequencies. In order to improve network throughput, multi-path routing algorithms can take into account the capacity constraints given by the channel assignment algorithm [3].

J. Karlsson and A. Kassler are with Computer Science Department, Karlstad University, Universitetsgatan, 65188 Karlstad, Sweden (e-mail: jonas.karlsson | andreas.kassler@kau.se).

G. Di Stasi and S. Avallone are with Computer Science Department, University "Federico II" of Naples, Via Claudio 21, 80125 Naples, Italy (e-mail: giovanni.distasi | stefano.avallone@unina.it).

Another technique which allows to increase network capacity is packet aggregation. Here, several packets are aggregated into a single transmission unit. This is particularly efficient when the overhead for a single transmission is high. The benefits of packet aggregation have been shown in several works, such as [4], [5] and [6]. The recent IEEE 802.11n [7] standard has also adopted aggregation as an optional feature to improve performance.

Deploying packet aggregation in multi-channel multi-path environments may result in suboptimal performance. This is because multi-path routing algorithms typically spread packets among different next-hop neighbors in the attempt to achieve efficient load-balancing, whereas only packets that must be forwarded to the same next hop can be aggregated. Due to this trade-off, it is important to effectively combine multi-path routing and packet aggregation.

The key contribution of this paper is a novel packet scheduling method for multi-path forwarding over multi-channel, multi-radio wireless mesh networks. While previous approaches try to either satisfy the link rates as given by the channel assignment or try to balance the load over different paths, our scheme takes into account both the capacity constraints given by the channel assignment algorithm and packet aggregation opportunities. Our algorithm exploits cross-layer information, such as the status of the sending queues, and makes the forwarding decision so as to improve packet aggregation and hence the MAC layer efficiency. To the best of our knowledge, this is the first attempt to combine multi-path routing and packet aggregation in an effective way for multi-radio multi-channel mesh networks. Simulation results show that our approach can improve network throughput by up to 15 percent and average delay by up to 25 percent.

The rest of the paper is structured as follows. Section II introduces the channel assignment, multi-path routing and packet aggregation algorithms in wireless multi-hop networks. Section III presents our simulation results. Section IV relates our work to state-of-the-art. Finally, section V concludes the paper.

## II. Architecture

In our approach, we explicitly distinguish between routing and forwarding. The routing algorithm builds the routing tables and finds one or multiple paths for any given source-destination pair. In contrast to single path routing, our algorithm calculates multiple candidate next-hops at every node for each source-destination pair. The forwarding algorithm selects then for each packet the next-hop among the *candidate next-hop*s.

978-0-7695-4469-4/11 $26.00 © 2011 IEEE
DOI 10.1109/MASS.2011.85

### A. Channel Assignment and Routing

The channel assignment algorithm ensures that the network is connected and that, if certain relative flow-rates are respected between links, the network capacity is maximized. It goes through three phases: i) calculation of a set of pre-computed flow-rates, which represent the desired utilization of the links. The calculation is performed using the max-flow algorithm [8], with the objective of maximizing the aggregate network throughput; ii) the channels to be assigned to the wireless interfaces are calculated trying to make the pre-computed flow-rates schedulable; iii) if the second step does not succeed, the pre-computed flow-rates are scaled down, in order to obtain a set of schedulable flow-rates. The output of the channel assignment algorithm is the computed set of flow-rates and a set of wireless channels to be assigned to nodes. The routing algorithm determines the potential paths between each source-destination pair using a hop count metric. More information on the channel assignment and routing can be found in [3].

### B. Packet Aggregation

The basic idea of packet aggregation is to improve MAC layer efficiency by aggregating several packets into a single transmission unit. This reduces the number of MAC-layer transmissions and the related overhead and significantly reduces contention for highly congested links. We use hop-by-hop aggregation, as it provides more aggregation opportunities than end-to-end aggregation [9]. Here, every node independently aggregates packets which should be transmitted to the same next hop. At the receiving interface, the node de-aggregates an aggregated packet and inserts the de-aggregated packets into the local network stack.

A single transmission unit is called *an aggregation packet*. Such aggregation packets can be link layer frames or IP packets, depending on the particular technique used. In this work we have used IP packets, as in [10] and [11]. since this allows to deploy aggregation without requiring changes or native support at the MAC layer. Therefore, our system can be used with 802.11a/b/g interfaces which, in contrast to 802.11n, do not support MAC layer aggregation. In our approach, an aggregation packet will contain several IP packets. An additional aggregation header allows to distinguish between aggregated and un-aggregated packets. A packet *is aggregated* when it is combined with at least *one other packet* inside an aggregation packet. We measure the efficiency of our method by calculating the aggregation ratio, which denotes the percentage of packets which *are aggregated*.

We implemented packet aggregation in an *aggregation module,* which extends the functions of a network interface. Such aggregation module stores and aggregates packets before passing them to the real network interface for transmission. Internally, it stores the packets in different queues, one for each next hop neighbor that can be reached through the network interface. When a packet is received by the aggregation module, it is timestamped and put into the appropriate queue, i.e. the queue of the next-hop the packet is destined to. The time-stamp is used later to determine how long the packet has been queued already. If there are more packets in a queue that can fit inside one MAC frame, an aggregation event is triggered, i.e. the aggregation module aggregates as many packets as possible from the queue while respecting the packet order. The resulting aggregation packet is sent as soon as the interface becomes ready. The remaining packets are left in the queue for the next aggregation event. An aggregation event is also triggered when the network interface becomes ready to transmit and one of the queues has packets to send. If more queues are ready, the queue with the *oldest* packet is served first to avoid starvation.

Packets can be purposely delayed in order to increase the aggregation ratio. The maximum amount of such artificial delay is controlled by the *AggregationMaxDelay* parameter. An aggregation event is triggered when the first packet in (any) queue has stayed for at least *AggregationMaxDelay* time. When the network traffic is low, this parameter induces artificial delay, which increases the number of packets in the queue and thereby increases the aggregation ratio. When traffic is high, typically a queue contains enough packets to fill one MAC frame and packets are normally not delayed. Note that slightly delaying packets for aggregation may actually reduce the total end-to-end delay in a multi-hop environment. The reason is that the reduced contention implies reduced back-off times and fewer packet re-transmissions. This has been shown to be beneficial even for VoIP services, where the controlled delay introduced to aggregate packets increased the total achievable mean opinion score (MOS) [12].

The main idea of this paper is to exploit information on aggregation opportunities at the forwarding layer (see next section), which decides for each packet which next hop to use. Therefore, we inform the packet scheduler about the available space left in each aggregation queue and the remaining time for each aggregated packet to be sent.

### C. Forwarding Strategies

In this section we describe our forwarding strategies, which decide the next hop for each packet, among the candidate next-hops as determined by the routing module.

*1) L2R (Layer 2 Routing):* L2R distributes traffic on links in proportion to the flow-rate, i.e. desired utilization, given by the channel assignment algorithm. In order to do so, each node $u$ records the amount of bytes sent on each link to its neighbors, and chooses for each packet the neighbor $v$ among the candidate next-hops ($C$) with the minimum cost. Therefore, it first calculates:

$$\Delta_u(v) = \frac{f(u \to v)}{\sum_{\forall u \to i, i \in C} f(u \to i)} - \frac{b(v)}{\sum_{\forall u \to i} b(i)} \qquad (1)$$

where $f(u \to i)$ represents the flow-rate of the link between $u$ and $i$, $b(i)$ represents the bytes sent on link between $u$ and $i$ and$\Delta_u(v)$ represents the difference between the desired and the actual utilization of the link between $u$ and $v$. The actual cost is then calculated by weighting the flow rates in order to reduce the average path length. More details can be found in [3].

**Algorithm 1** Aggregation Aware L2R

AGGRAWARE-L2R$(C, p)$

```
1  A ← ∅
2  for each n ∈ C
3      if isNotEmpty(n) and spareSpace(n) >= dim(p)
4          addElement(A, n)
5  if dim(A) > 0
6      q = findFlowrateQueues(A)
7      enqueue(p, q)
8  else
9      F = findFreeQueues(C)
10     if dim(F) > 0
11         q = findFlowrateQueues(F)
12         enqueue(p, q)
13     else q = findFlowrateQueues(C)
14         enqueue(p, q)
```

FINDFREEQUEUES$(B)$

```
1  F ← ∅
2  for each q ∈ B
3      if isEmpty(q)
4          addElement(F, q)
5  return F
```

*2) AA-L2R (Aggregation-Aware L2R):* In order to exploit the information on aggregation opportunities at the forwarding layer, we have developed the AA-L2R algorithm, which prioritizes in its forwarding decision the increase of aggregation ratio over the fulfillment of flow-rates. The key idea is to first find all potential queues related to next-hop candidates who allow the packet to be aggregated and then from all those queues to pick the one which best fulfills the flow-rates. If a neighbor associated to one of these queues is selected as next-hop node, the packet can be aggregated with packets already in the queue. This reduces MAC layer overhead, because of the saved transmission as packets are aggregated before sending.

In Algorithm 1 we show the pseudo-code of AA-L2R. We define the *aggregation set A* for the given packet as the potential set of next-hops associated with queues that offer an aggregation opportunity, i.e. which allow to aggregate. A next-hop belongs to the set *A* if it belongs to the candidate next-hops and the following conditions hold: i) the associated queue is not empty; and ii) the *spare space* (SP) of the associated queue is greater than the packet size. The *spare space* of the generic queue $i$ is defined as in the following:

$$SP = MTU - \sum_{p \in Q_i} p_{size} - H \qquad (2)$$

where *MTU* is the Maximum Transmission Unit, *H* the aggregation header size, $Q_i$ the set of packets in queue $i$.

If the aggregation set is empty, the packet cannot be aggregated at the moment. However, it could be aggregated with packets yet to come. This happens because the queues are empty or because the queues which are not empty do not have enough spare space to aggregate the packet. Here, the *free aggregation set F*, which contains the empty queues, is evaluated (see *FindFreeQueues* function). By choosing a queue of the free aggregation set, we leave unchanged the queues which hold at least a packet and can potentially aggregate a packet which arrives later on. In addition, the selected empty queue becomes as well a potential source of aggregation. If also the free aggregation set is empty, all queues hold some packets with a spare space smaller than the packet size. In this case, all the queues belonging to the candidate next-hops are considered eligible for sending the current packet. Once the eligible set of next-hops has been chosen, the L2R criterion is applied in selecting among the specific next-hops to try to approximate the flow-rates (by applying Equation 1).

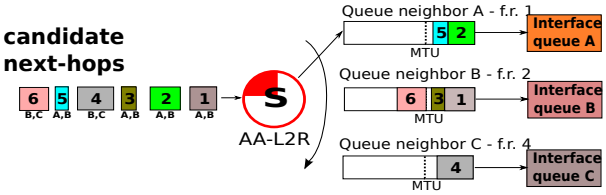


Figure 1. Forwarding process example (f.r. stands for flow-rate). On the bottom of each packet the set of candidate next-hops is shown.

In Figure 1 we report an example of the forwarding process. A mesh node with three neighbors and 6 packets to forward is shown. As can be seen, packets are placed in the different queues so as to maximize the aggregation ratio.

*3) RR (Round-Robin):* Another multi-path forwarding strategy is *round-robin* (RR). Here, the traffic is load-balanced equally between all candidate next-hops. This will give the least possibilities for aggregation and most reordering since the packets have the largest possible spread among the next-hop candidates.

## III. EVALUATION

We performed a number of NS-2 [13] simulation studies to evaluate the performance of the considered forwarding paradigms. Unless otherwise noted we used the default NS-2.32 settings. The MAC/PHY layer was configured to simulate an IEEE 802.11 MAC/PHY layer with the MAC MTU set to 2304 bytes, in order to make the aggregation capabilities compliant with IEEE 802.11a/b/g standards [1].

In the simulations we use 54 Mbps PHY layer speed, and TCP Newreno[14] with selective acknowledgment (Sack) [15]. We considered a randomly generated topology of 25 nodes placed in an area of 300x300 meters (see Figure 2). Each node was equipped with a maximum of 3 radio interfaces. We randomly selected three source and sink nodes placed at opposite sides in the network. In future work, we will use significantly larger topologies and traffic configurations.

### A. Methodology

We evaluated the behavior of the considered forwarding paradigms under two different traffic classes. The first class consisted of three UDP flows between each source-destination

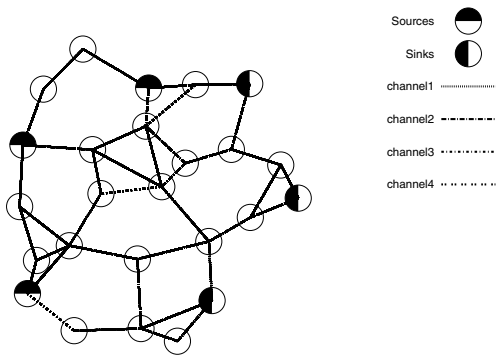


Figure 2. Simulated topology

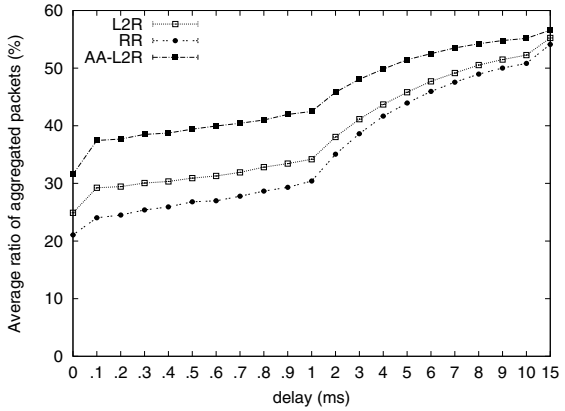


Figure 3. Average Aggregation ratio for UDP traffic scenario.

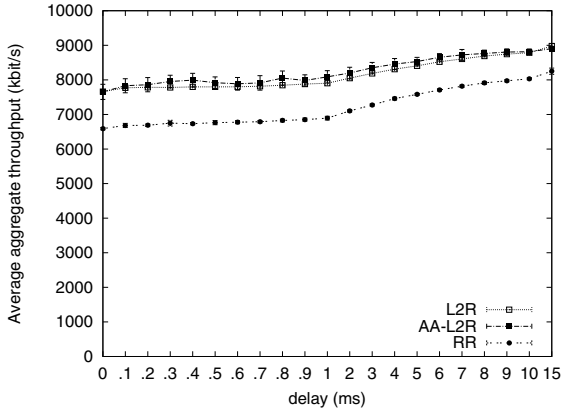


Figure 4. Average aggregate throughput for UDP traffic scenario.

pair, resulting in 27 flows in total. Each flow had an exponential ON-OFF behavior, an average "on" time of 5 seconds and an average "off" time of 1 second.

The packet size was different for each flow: 200 bytes for the first, 700 for the second and 1400 for the third. The inter-packet departure time was set so as to have 380 kbit/s of bitrate for each flow, which implied the generation of 10 Mbit/s of bitrate in total by the sources. The second class consisted of TCP flows which were generated between each pair of source and sink nodes. We assumed FTP type traffic with infinite backlog to simulate large file transfers with a segment size of 1460, corresponding to a common IP packet size [16]. With a segment size of 1460 bytes, only one TCP DATA packet can fit inside each aggregation packet, which means that there is less room for improvement by aggregation. However, multiple TCP ACKs can be aggregated together with both TCP DATA and other TCP ACK packets.

Traffic generation was started after 9s of delay to allow the network to stabilize routing. Each simulation was run for 310 seconds with 25 repetitions. The output of the simulations was statistically collected and analyzed using the tool from [17]. The results are shown relative to the value of the *AggregationMaxDelay* parameter (x-axis), i.e. the amount of artificial delay, as inserted by the aggregation algorithm. In all simulations only shortest hop paths were used.

### B. Simulation Results

*1) UDP:* In case of a highly loaded network, there is basically a high amount of contention among nodes for accessing the shared medium. This wastes network resources, since the nodes spend time in trying to obtain the medium, rather than actually sending packets. Figure 3 shows the average aggregation ratio with varying *AggregationMaxDelay*. The aggregation ratio is up to 48 percent higher for AA-L2R compared to RR and up to 23 percent better than L2R. AA-L2R also slightly improves the end-to-end throughput and significantly reduces end-to-end delay up to 25 percent over RR and 17 percent over L2R (see Figure 4 and 5). Packet aggregation can greatly help to reduce contention by reducing the number of packets to be sent. By exploiting the knowledge on the internal state of the queues made available by the aggregation module, AA-L2R can reduce the amount of contention compared to strategies that do not consider aggregation possibilities. We can also observe that both AA-L2R and L2R have a significantly better throughput than RR, which can be explained by the fact that RR neither approximates the flow-rates nor tries to increase the aggregation ratio.

*2) TCP:* TCP simulation results, which we omit due to space constraints, show that the aggregation ratio is almost identical for all three schemes when using TCP traffic with the full segment size of 1460 bytes. This follows earlier results in e.g. [5] where it is shown that large packets limit the improvement due to aggregation. Furthermore, as the aggregation is TCP un-aware, the artificial delay can increase TCP RTT as TCP DATA packets will be delayed but rarely can be aggregated, since we only have TCP DATA packets flowing in one direction.

When *AggregationMaxDelay* is smaller than $< 1ms$, both TCP round-trip time (RTT) and packet loss (including re-ordered packets) are similar or slightly lower for AA-L2R compared to both RR and L2R (omitted due to space constraints). The slightly lower RTT is reflecting the improved MAC layer efficiency due to the more effective aggregation of TCP ACKs with AA-L2R. When *AggregationMaxDelay* is

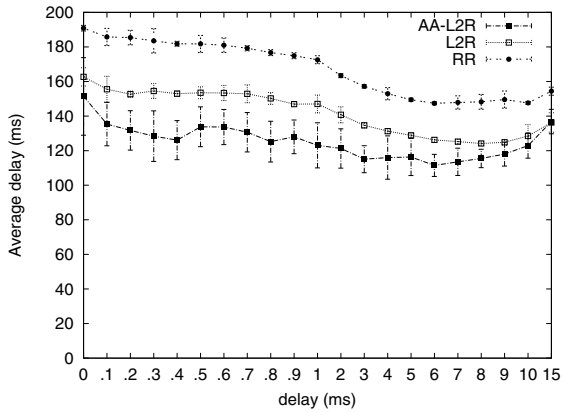


Figure 5. Average delay for UDP traffic scenario.

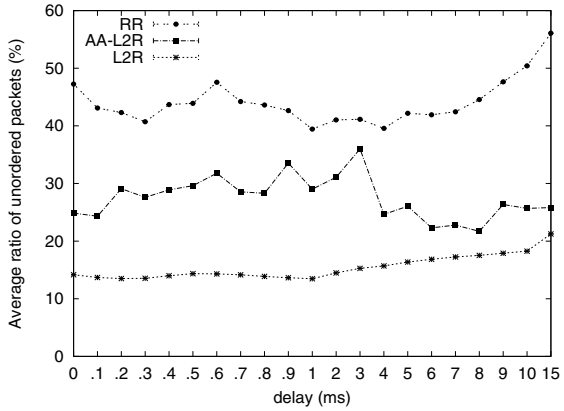


Figure 6. Packet reordering with TCP traffic

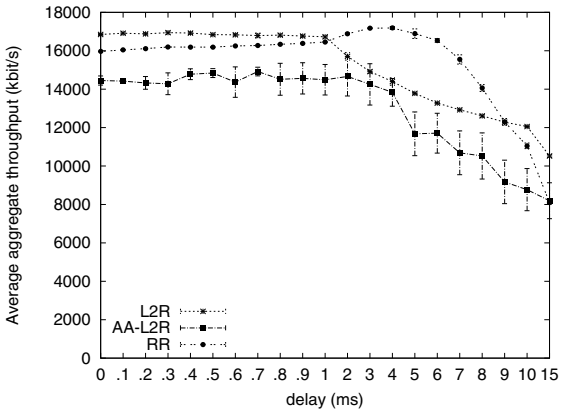


Figure 7. Average aggregate TCP throughput

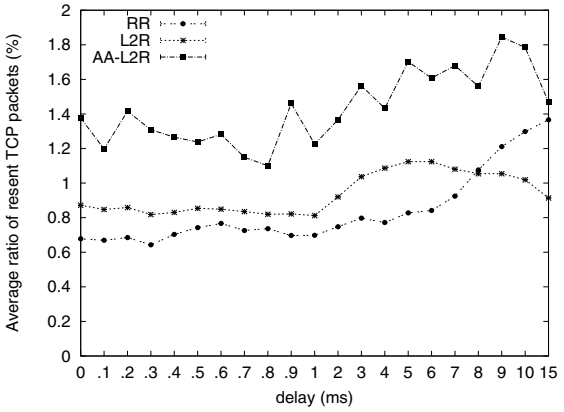


Figure 8. TCP Retransmitted packets

increased, the aggregation awareness of AA-L2R becomes a disadvantage.

With large TCP DATA packets and no reverse flows, TCP DATA can NOT be aggregated with TCP ACKs, AA-L2R will thus place them in empty queues. Therefore, TCP DATA packets will be delayed longer than when using AA-L2R compared to RR and L2R where more often a already occupied queue will be used and hence trigger an aggregation event. Although no TCP DATA packets will be aggregated due to this aggregation event it will make the first TCP DATA packet in the queue to be forwarded before the *AggregationMaxDelay* timer is due and therefore reduce packet delay. The preference of queues that are empty will also make AA-L2R switch next hop neighbors more often for packets within the same TCP flow than L2R, causing up to two times the number of reordered packets compared to L2R (see Figure 6). The highest amount of reordered packets however is experienced by RR. However, simulation results omitted due to space constraints show that the packet displacement is higher for AA-L2R indicated by a higher delay variation. This is due to the more aggressive use of empty queues by AA-L2R whereas RR uses all queues in a uniform manner.

As can be seen from Figure 7, TCP throughput is slightly lower for AA-L2R compared to both RR and L2R. This follows from the higher number of retransmitted TCP packets that can be observed in Figure 8. Since we use the TCP sack option, both the amount of reordering and the magnitude of packet displacement impact the amount of resent TCP packets. Additional simulation results, not shown here, indicate that RR has the lowest amount of TCP timeouts and that the increase in throughput for RR when *AggregationMaxDelay* is varied between 2 and 4 ms is due to a slight reduction of the number of lost packets and a maintained TCP RTT, compared to AA-L2R and L2R, in these simulations. This effect is due to a synchronization effect where a slightly higher amount of packets, not necessarily within the same flow, use the same next-hop neighbour with RR. This increases the aggregation ratio and reduces the contention and therefore the time packets spend in queue.

## IV. Related work

Several studies have considered the usefulness of packet aggregation on wireless networks in a great variety of operational conditions. For instance, the performance measurement study done in [18] has shown an improvement of up to 160 percent of throughput in a single-hop scenario with mixed traffic. In [5], TCP performance in small topologies with no hidden nodes was studied and an improvement of up to 73

percent was shown compared to not using aggregation. In [4] packet aggregation was studied with the objective of deriving the dependence of the throughput to the size of aggregation packets and to the link quality. Modern IEEE 802.11 WiFi cards change modulation schemes to compensate for changes in BER. For this reason, in this work we focused on the integration between packet aggregation and routing and have only used "good" links.

The recent WiFi standard IEEE 802.11n [7] exploits packet aggregation to improve performance [19]. In particular, the standard defines two different strategies for aggregation, one where the aggregation is done when packets enter the MAC layer (A-MSDU) and one when the packets leave the MAC layer (A-MPDU). The aggregation strategy most similar to the one performed by our aggregation module is the A-MSDU. We have however limited the size of the aggregated packets to 2304 bytes in order to be compliant to IEEE 802.11a/b/g devices whereas an IEEE 802.11n A-MSDU is allowed to be 7935 bytes. However, even if several works have considered the effect of packet aggregation on wireless networks, no study, to the best of our knowledge, is available which combines aggregation aware routing and forwarding for multi-radio WMNs.

## V. Conclusions and future works

In this paper we propose and evaluate a new forwarding strategy for multi-radio multi-channel WMNs. The forwarding strategy selects for each packet a next-hop based on a set of candidate nodes trying to balance between aggregation efficiency and the flow-rate criteria given by the channel assignment algorithm. Simulation results show that the proposed aggregation aware forwarding paradigm is able to significantly increase the aggregation possibilities leading to a lower delay and packet loss for user traffic. However, packet re-ordering might negatively impact TCP performance.

When using multi-path routing, packet reordering is a well known problem for TCP and causes severe performance degradation [20], [21]. In a general purpose WMN, it can be anticipated that most clients will run standard TCP variants that have limited mechanisms to handle reordering. As future work, we will therefore investigate approaches to minimize the effect of packet reordering, e.g. using TCP flow aware forwarding strategies.

## VI. Acknowledgment

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

## References

[1] *IEEE Standard 802.11, IEEE standard for wireless LAN medium access control (MAC) and physical layer (PHY) specifications*, IEEE Std., 1999.

[2] A. Adya, P. Bahl, J. Padhye, A. Wolman, and L. Zhou, "A multi-radio unification protocol for IEEE 802.11 wireless networks," 2004.

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

[4] R. Raghavendra, A. P. Jardosh, E. M. Belding, and H. Zheng, "IPAC: IP-based Adaptive Packet Concatenation for Multihop Wireless Networks," in *Proc. Fortieth Asilomar Conference on Signals, Systems and Computers ACSSC '06*, 2006, pp. 2147–2153.

[5] J. Karlsson, A. Kassler, and A. Brunstrom, "Impact of packet aggregation on TCP performance in Wireless Mesh Networks," in *WOWMOM*. IEEE, 2009, pp. 1–7.

[6] P. Dely, A. Kassler, N. Bayer, and D. Sivchenko, "An Experimental Comparison of Burst Packet Transmission Schemes in IEEE 802.11-Based Wireless Mesh Networks," in *GLOBECOM*, vol. 2010, 2010, pp. 1–5.

[7] IEEE, "IEEE standard 802.11n," *IEEE Std 802.11n-2009 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, and IEEE Std 802.11w-2009)*, pp. 1–502, 2009.

[8] R. Ahuja, T. Magnanti, and J. Orlin, "Network flows: theory, algorithms, and applications. 1993."

[9] A. Jain, M. Gruteser, M. Neufeld, and D. Grunwald, "Benefits of Packet Aggregation in Ad-Hoc Wireless Network," Department of Computer Science, University of Colorado at Boulder, Boulder, CO 80309, CU-CS-960-03, August 2003. [Online]. Available: http://www.cs.colorado.edu/department/publications/reports/docs/CU-CS-960-03.pdf

[10] M. Castro, P. Dely, J. Karlsson, and A. Kassler, "Capacity Increase for Voice over IP Traffic through Packet Aggregation in Wireless Multihop Mesh Networks," *Future Generation Communication and Networking*, vol. 2, 2007.

[11] R. Riggio, D. Miorandi, F. De Pellegrini, F. Granelli, and I. Chlamtac, "A traffic aggregation and differentiation scheme for enhanced QoS in IEEE 802.11-based Wireless Mesh Networks," *Computer communications*, vol. 31, no. 7, pp. 1290–1300, 2008.

[12] P. Dely, A. Kassler, N. Bayer, H. Einsiedler, and D. Sivchenko, "FUZ-PAG: A Fuzzy-Controlled Packet Aggregation Scheme for Wireless Mesh Networks," in *Proc. 7th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD'10), Yantai, China*, 2010.

[13] S. McCanne, S. Floyd, K. Fall, K. Varadhan *et al.*, "Network simulator ns-2," *The Vint project, available for download at http://www.isi.edu/nsnam/ns*.

[14] S. Floyd, T. Henderson, and A. Gurtov, "RFC 3782-The NewReno modification to TCP's fast recovery algorithm," IETF, 2004.

[15] E. Blanton, M. Allman, K. Fall, and L. Wang, "RFC 3517: A Conservative Selective Acknowledgment (SACK)-Based Loss Recovery Algorithm for TCP," IETF, 2003.

[16] W. John and S. Tafvelin, "Analysis of internet backbone traffic and header anomalies observed," in *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*. New York, NY, USA: ACM, 2007, pp. 111–116.

[17] C. Cicconetti, E. Mingozzi, and G. Stea, "An integrated framework for enabling effective data collection and statistical analysis with ns-2," in *WNS2 '06: Proceeding from the 2006 workshop on ns-2: the IP network simulator*. New York, NY, USA: ACM, 2006, p. 11.

[18] D. Kliazovich and F. Granelli, "Packet concatenation at the IP level for performance enhancement in wireless local area networks," *Wirel. Netw.*, vol. 14, no. 4, pp. 519–529, 2008.

[19] D. Skordoulis, Q. Ni, U. Ali, and M. Hadjinicolaou, "Analysis of Concatenation and Packing Mechanisms in IEEE 802.11n," in *Proceedings of the 6th Annual Postgraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting (PGNET'07)*, 2007.

[20] K. Leung, V. Li, and D. Yang, "An overview of packet reordering in transmission control protocol (TCP): problems, solutions, and challenges," *IEEE Transactions on Parallel and Distributed Systems*, pp. 522–535, 2007.

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