Opportunistic Routing for Interactive Traffic in Wireless Networks

Tianji Li*†, Douglas Leith*
*Hamilton Institute
National University of Ireland Maynooth, Ireland
Email: {tianji.li, doug.leith}@nuim.ie

Lili Qiu†
†Computer Science Department
University of Texas at Austin, USA
Email: lili@cs.utexas.edu

***Abstract*—To take advantage of the broadcast nature of wireless communication, a number of opportunistic routing protocols have recently been proposed. In order to manage the extra signaling overhead associated with operation of the opportunistic routing, these schemes work in terms of 'batches' that consist of multiple packets. While these opportunistic protocols can dramatically improve the total throughput, the use of batches means that they are best suited to bulk UDP transfer. However, in the Internet and wireless networks, the vast majority of the traffic is interactive (e.g., TCP/VoIP which requires close interactions and feedback between the two communicating end points). To effectively support interactive traffic, we develop a new opportunistic routing protocol, called RIPPLE. RIPPLE uses an expedited multi-hop transmission opportunity mechanism to achieve low signaling overhead and eliminate re-ordering, and uses a two-way packet aggregation technique to further reduce overhead. We implement the RIPPLE in NS-2 along with several existing routing protocols, including predetermined routing, shortest path routing, the early version of ExOR, MCExOR, and an IEEE 802.11n-like single-hop packet aggregation scheme called AFR. We compare their performance for long- and short-lived TCP transfers and VoIP traffic over a wide range of network conditions, including varying wireless channel states, collision levels, and types of network topologies. Our results show that the RIPPLE scheme consistently achieves 100% – 300% performance gains over other approaches.**

***Index Terms*—Medium access control (MAC), Opportunistic Routing, IEEE 802.11, Wireless LANs (WLANs), Wireless Mesh Networks.**

I. Introduction

Wireless communication is inherently broadcast by nature. Any transmission (including unicast) can be heard not only by the target destination, but also by all the stations within the communication range of the transmitter. However, only the target destination decodes the transmissions it hears, whereas all the other stations simply drop the received transmissions as they are not the intended recipients. To take advantage of the broadcast nature of wireless medium, several opportunistic routing protocols, such as [16], [8], [9], [28], [25], have been proposed where multiple forwarders cooperatively relay overhead traffic. Such opportunistic routing schemes can result in significant performance gain under lossy wireless medium.

A key issue in designing opportunistic routing is the signaling overhead. In classical predetermined routing, once the routing tables have been constructed, there is no additional per packet signaling overhead. However, in opportunistic schemes, multiple forwarders typically overhear a packet transmission. Due to the stochastic nature of channel noise, this set of forwarders varies from one packet to another. It is thus necessary for the forwarders to acknowledge whether they hear a particular packet. One simple approach is for the forwarders or the final destination to transmit a MAC acknowledgment (ACK) upon receiving a packet. These MAC ACKs should also be scheduled sequentially in order to avoid collisions. This approach is used in the early version of ExOR [7], which we refer to as preExOR to distinguish it from the later work in [8]. Clearly, the sequential ACKs in preExOR is inefficient if there are many forwarders. To achieve efficient use of network resources, it is important to minimize the per packet signaling overhead. ExOR [8] mitigates this overhead by amortizing the signaling overhead over a batch of packets. Several recent opportunistic routing protocols, such as [9], [17], [21], further leverages network coding to reduce signaling overhead. MCExOR [28] also proposes an approach where forwarders can prematurely stop waiting for MAC ACKs from the destination and highly ranked forwarders and send their MAC ACKs.

While these opportunistic routing protocols can dramatically improve total throughput for bulk UDP transfers, none of them considers supporting interactive traffic, such as TCP and VoIP. Effectively supporting interactive traffic is important since the vast majority (up to 80%-90% according to [27], [26]) of network traffic is TCP; moreover VoIP is becoming increasingly popular.

Interactive traffic is different from bulk UDP transfers because they are highly sensitive to delay and re-ordering. Existing opportunistic schemes, which use a fixed batch size to amortize the signaling overhead, are not appropriate for carrying interactive traffic especially when the number of packets in flight is smaller than the typical batch sizes. This is acknowledged by the authors of [8]. Since MORE [9] and the follow-up work [17] focus on network coding extensions to [8], they inherit similar issues. Therefore it is necessary to design an efficient opportunistic routing protocol that explicitly supports interactive traffic.

To address this issue, we design a novel opportunistic routing protocol, called RIPPLE. It uses an expedited multi-hop transmission opportunity (mTXOP) mechanism to achieve low signaling overhead and eliminate re-ordering, and uses a two-way packet aggregation technique to further reduce overhead. We implement the RIPPLE in NS-2 along with several existing routing protocols, including predetermined routing, shortest path routing, preExOR, MCExOR, and an IEEE 802.11n-like single-hop packet aggregation scheme called AFR [19]. We compare their performance for long- and short-lived TCP transfers and VoIP traffic over a wide range of network conditions, including different wireless channel states, collision levels, and network topologies, such as the Wigle and Roofnet topologies. Our results show that RIPPLE consistently achieves significant performance gains over the existing approaches, with 100% – 300% throughput improvement. We expect RIPPLE to benefit a range of static wireless networks, including both infrastructure WLANs and multi-hop wireless mesh networks. In WLANs, clients can use RIPPLE to help each other efficiently communicate with the AP, and nodes in mesh networks cooperatively forward traffic for each other to provide Internet services.

II. Motivation

A routing protocol normally consists of three parts: route discovery, packet forwarding, and route maintenance. Let *S*, *R* and $F = F_1, \ldots, F_n$ denote source, destination, and the set of forwarders in a wireless network. For an opportunistic routing protocol, the task of route discovery and route maintenance involves selecting and prioritizing the forwarders F. Several existing opportunistic routing schemes, such as ExOR [8] and MORE [9], select forwarders based on ETX [14], which quantifies the number of transmissions required for sending traffic from the source to the destination. As will be seen in Section IV, the proposed RIPPLE scheme can easily incorporate any forwarder selection schemes. Our focus in this paper is on the second task, i.e., packet forwarding. Current forwarding techniques can be classified into two categories: predetermined and opportunistic forwarding.

A. Predetermined Forwarding

In predetermined forwarding, each transmission has exactly one intended forwarder. All the stations except the intended forwarder simply drop the overheard transmission. Therefore, the forwarding path followed by a packet is known in advance and is updated periodically.

B. Opportunistic Forwarding

Opportunistic forwarding allows multiple forwarders to participate in relaying overheard transmissions to improve network performance. We consider two of the existing methods: preExOR and MCExOR since neither of them uses batches and are potentially more suitable for supporting interactive traffic.

In preExOR, after the source transmits a data packet, forwarders send MAC ACKs sequentially to avoid collisions. This is achieved by having each forwarder defer for a period that is sufficient to allow the destination and all the higher priority forwarders to transmit their ACKs.

The MCExOR scheme uses a compressed acknowledging mechanism, where a forwarder of rank i waits for i SIFS intervals before transmitting a MAC ACK. If it detects an ACK transmission during its waiting period, it will not transmit its ACK since the ACK reception indicates that a higher ranked forwarder has received the packet.

C. Comparison

In the context of opportunistic routing, the destination R is able to hear from the source S but the link quality between them is typically poor. The link quality between the source S and the forwarders in F and that between the forwarders in F and the destination R is relatively better. Thus, a properly designed opportunistic routing scheme should take advantage of transmissions that either directly reach R or reach some forwarders and maximize the progress of each transmission to improve network efficiency. However, preExOR and MCExOR do not work well for interactive traffic, such as TCP and VoIP, due to signaling overhead and packet reordering.

1) Signaling Overhead: The preExOR and MCExOR schemes frequently incur higher signaling overhead than predetermined schemes. Specifically, transmissions from the source S are received with the highest probability by the first forwarder F_1, and transmissions from F_1 are most frequently received by the second forwarder F_2, and so on. Therefore both the opportunistic and predetermined routing schemes tend to use this same route (i.e., $S \rightarrow F_1 \rightarrow \ldots \rightarrow F_n \rightarrow R$). That is, even if opportunistic routing methods are used, the most probable route is the same as that the one used by predetermined routing. On the other hand, opportunistic routing incurs a higher signaling overhead than predetermined routing due to the need to identify the set of forwarders that have received each packet transmission and select the best forwarder to relay the transmission. This may cause significant degradation in total throughput compared with predetermined routing.

In more details, let $T_{backoff}$ denote the time to perform random backoff, T_{ACK} denote the time to transmit MAC ACKs, T_{phyhdr} denote the time to send the physical layer header, T_{SIFS} denote SIFS duration, and T_{DIFS} denote DIFS duration. Using the predetermined routing, for a packet that is relayed by $n-1$ forwarder(s) to be received by the destination, it takes $n(T_{backoff} + T_{DATA} + T_{DIFS} + T_{SIFS} + T_{ACK} + 2T_{phyhdr})$, where there are two T_{phyhdr}: one for the data packet and another for ACK. In comparison, preExOR and MCExOR take $n(T_{backoff} + T_{DATA} + T_{DIFS} + T_{phyhdr}) + \sum_1^n (T_{ACK} + T_{SIFS} + T_{phyhdr})$ and $n(T_{backoff} + T_{DATA} + T_{DIFS} + T_{ACK} + 2T_{phyhdr}) + \sum_1^n T_{SIFS}$, respectively, to deliver a packet to the destination due to the use of sequential ACKs and the compressed sequential ACKs. Therefore preExOR and MCExOR require longer time to deliver a transmission end-to-end. For example, Fig. 2 illustrates transmission timeline of two packets for flow 1 in the topology in Fig. 1. The predetermined route used for flow 1 is called PRR, where packets in flow 1 follow the route $0 \rightarrow 1 \rightarrow 2 \rightarrow 3$. In comparison, preExOR takes $6 * (T_{ACK} + T_{SIFS})$ longer than PRR in this example, whereas MCExOR takes $6 * T_{ACK}$ less time than preExOR but still $6 * T_{SIFS}$ longer than PRR. That is, for the most probable transmission sequence, the preExOR and

T_{SIFS} (μs)	16
Idle slot duration (μs)	9
Packet size (bytes)	1000
PHY data rate (Mbps)	216
PHY basic rate (Mbps)	54
Interface queue (packets)	50
T_{phyhdr} (μs)	20
Simulation time (s)	10

TABLE I
SIMULATION PARAMETERS USED IN THIS PAPER.

MCExOR schemes incur extra signaling overhead over PRR due to the signaling requirements associated with operation of opportunistic routing.

2) Packet Re-ordering: The previous analysis focuses on the most probable transmission sequence. In opportunistic routing, there are several other possible transmission sequences: some are shorter while others are longer. To compare performance, we implement the preExOR and MCExOR schemes in NS-2 and compare them with predetermined methods.

We observed that even considering other transmission sequences, the performance of preExOR and MCExOR is still worse than that of predetermined routing. For example, consider a TCP flow from station *0* to station *3*, which lasts for 10 seconds and Table I shows other parameters. The total throughput under Shortest Path Routing (SPR), preExOR, and MCExOR are 6.7, 5.9 and 5.85 Mbps, respectively. Refer to our technical report [20] for more details.

A closer look of the simulation results reveals that part of degradation comes from the signaling overhead discussed above, and part of degradation comes from packet re-ordering. Packet re-ordering happens frequently under both preExOR and MCExOR. For example, under preExOR, among 10766 TCP packets received by the destination, 2862 are out of order, which corresponds to 26.58% re-ordered packets. Under MCExOR, there are 3122 packets out of 11191 packets are re-ordered, yielding 27.9% re-ordered packets. Re-ordering in preExOR and MCExOR occurs due to the random backoff mechanism in IEEE 802.11 and the unpredictable channel state. To see this, consider an example where the source has two packets i and $i+1$ to send. Suppose it sends packet i first, which is received by forwarder j but not by the destination. Both the source and forwarder j then initiate a random backoff to compete for the channel access, but the source may choose a shorter random backoff time than forwarder j and so transmits packet $i+1$ before forwarder j transmits packet i. If packet $i+1$ is heard by the destination, then re-ordering occurs. Packet re-ordering degrades TCP performance because the congestion control algorithm in TCP responds to re-ordered packets by reducing its sending rate.

III. The RIPPLE Scheme

A. The Main Idea

In this section, we describe RIPPLE to address the two key issues in the existing opportunistic routing protocols – packet re-ordering and signaling overhead.

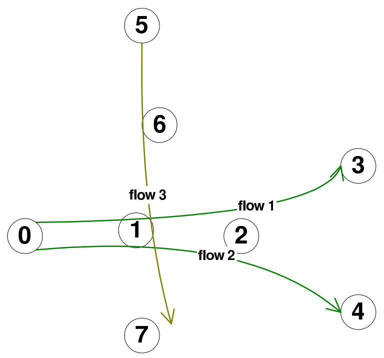


Fig. 1. A multiple-flow topology. There are three flows altogether in this topology. Flows 1 and 2 share stations 0, 1 and 2, and flow 3 intersections with the other flows at station 1.

1) Resolving Re-ordering: The reason of packet re-ordering, as introduced in the last section, is the time difference between transmissions of new packets by the source and transmissions of old packets by the forwarders. To solve this issue, we do not let the forwarders cache any overheard frames while still letting them help forward transmissions. We design an atomic operation between the source and the destination within which re-ordering can be completely eliminated. We call this kind of operation a multi-hop transmission opportunity, where a transmission opportunity in IEEE 802.11 consists of a DIFS interval, a backoff period, a data transmission, a SIFS interval and a MAC ACK transmission (mTXOP). Below we describe our approach in more details.

- *Multi-hop Transmission Opportunity*. Let forwarder 1 denote the highest priority forwarder, forwarder 2 denote the next highest priority forwarder, and so on. In RIPPLE, the destination acknowledges reception of a frame after a $T = T_{SIFS}$ time, where T_{SIFS} is the SIFS duration [1]. Forwarder i ($i \geq 1$) relays a received data frame only after detecting the channel is idle for $T = i \times T_{Slot} + T_{SIFS}$, where T_{Slot} is a slot time in IEEE 802.11 [1]. This results in a prioritized opportunistic acknowledging scheme whereby the highest priority forwarder that receives a data frame relays the packet while lower priority forwarders defer and make no transmission (See Sections III-B1 and III-B2 for details about priority assignments). Therefore, high priority forwarders can help relay whenever they overhear the transmissions, thereby improving performance. Note that in MCExOR, a similar premature waiting mechanism is used, after which the MAC ACK will be sent to the source. In comparison, our scheme forwards the overheard data frame towards to destination.
- *Two-way Opportunistic Forwarding*. Upon receiving a data frame, the destination replies with a MAC ACK. Since MAC ACKs are important to network performance, we let the forwarders help relay MAC ACKs in a similar manner to relaying data frames so that the MAC ACKs will be received by the source with a high probability. That is, forwarder i ($i \geq 1$) relays a received MAC

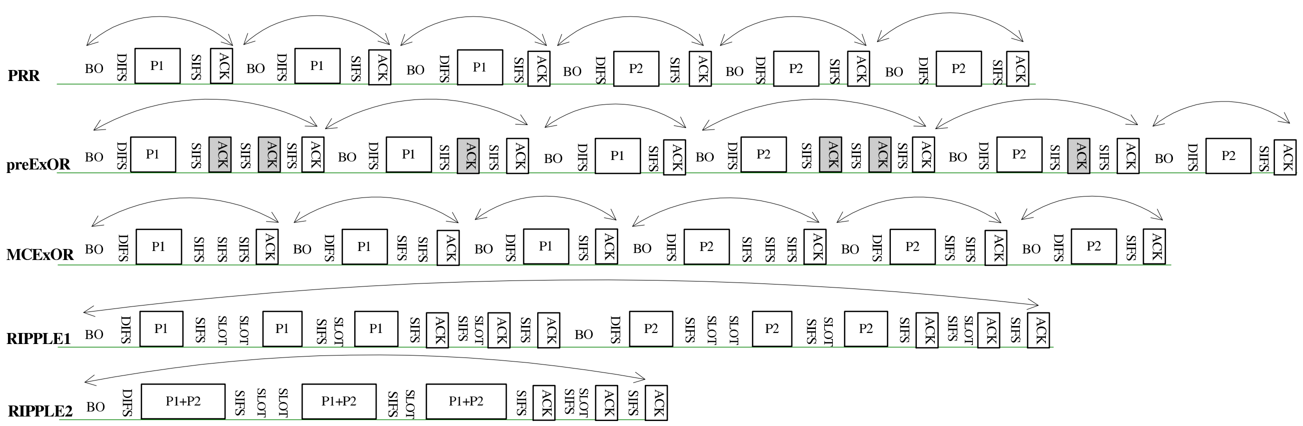


Fig. 2. Transmissions of two packets (P1 and P2) with predetermined route (i.e., $0 \rightarrow 1 \rightarrow 2 \rightarrow 3$), preExOR, MCExOR and RIPPLE. BO is the abbreviation of backoff. In the preExOR scheme, shadowed ACKs indicate that the source is waiting for an ACK which is not transmitted. Each arc line indicates one transmission opportunity.

ACK frame after detecting the channel to be idle for $T = (i-1) \times T_{Slot} + T_{SIFS}$. Since there is no acknowledgment for the MAC ACKs, a forwarder defers one less slot in relaying a MAC ACK than relaying a data frame.

- *Multi-hop Retransmission*. Forwarders do not cache any transmissions and only relay overheard transmissions at most once, i.e., if a forwarder hears a data (or a MAC ACK) frame but does not hear the corresponding transmissions from higher priority stations, it will start relaying, otherwise it discards the overheard frame. Retransmission is thus performed on an end-to-end basis, with the source retransmitting if it does not receive a MAC ACK within a timeout time. In this way, re-ordering caused by relaying from forwarders will never happen.

We revisit the example in Section II using the mTXOP mechanism. As before, for illustrative purposes we assume the same transmission order as in the PRR scheme. The transmission timeline for packets *P1* and *P2* is shown in Fig. 2 (see RIPPLE1). When packet $P1$ is transmitted by station *0*, it is received by station *1* but not by stations *2* or *3*. Station *1* waits for one SIFS and 2 slot intervals, where SIFS is for a possible ACK transmission from station *3*, one slot time is for station *2*, and another slot time to provide time to turn itself from receiving to sending state before forwarding $P1$. After hearing *1*'s transmission, station *2* relays in a similar way but defers one SIFS and one slot time as it is only one hop from the destination. Finally, *P1* arrives at station *3*. The MAC ACK is then sent and forwarded in a similar way, and then the same sequence repeats for Packet *P2*.

2) Reducing Overhead: To mitigate the MAC overhead, we propose a two-way packet aggregation mechanism that works as follows.

- When the source (re)transmits, we allow multiple packets (each protected using a separate CRC) to be aggregated in the (re)transmitted frame. In the RIPPLE scheme, multiple packets can be transmitted in a single frame. For clarity, we use packet to refer to the transmission from the upper layer to the MAC, and use frame to refer to the transmission from the MAC to PHY layer. Thus, overhead is incurred only once for the large frame under aggregation, whereas without aggregation overhead is incurred for every small packet. In the above example, using the packet aggregation (RIPPLE2 in Fig. 2) leads to approximately 50% overhead reduction in comparison to the non-aggregated version (i.e., RIPPLE1 in Fig. 2). As in [2], [19], we select 16 as the maximum number of packets that can be aggregated into a frame.
- Aggregation can be performed in a bi-directional manner. If there are data packets waiting to be transmitted from the destination to the source, the destination also aggregates packets into large frames. This seemingly simple mechanism can lead to significant efficiency gains for two-way flows such as TCP, where TCP ACKs in the reverse direction have to be sent.
- If there is local traffic at forwarders, a forwarder can aggregate local packets and relayed packets in order to save bandwidth.

B. Remarks

1) Forwarder Lists: The selection of forwarder lists and their priority assignment belongs to the task of route discovery. Existing routing schemes (e.g., ExOR [8] and MORE [9]) use ETX [14] towards the destination to select forwarders. This paper focuses on packet forwarding, which is orthogonal to forwarder selection. RIPPLE can easily support different forwarder list selection and priority assignments. As we will show in Section IV, for any given pre-selected path, RIPPLE can consistently improve performance over the existing routing protocols.

2) Priority Assignments: Upon hearing a transmission, a station checks the forwarding list to decide whether it is a forwarder. For forwarders, we use an implicit rule to assign their priorities. In particular, all stations know that the forwarding list is located between the MAC header and the frame body. We mandate that the forwarders that are closer to the MAC header have higher priorities. The destination is always the highest priority forwarder, and thus the closest one to the MAC header.

3) How mTXOPs Break: The mTXOPs used in the RIPPLE scheme are potentially longer than transmission opportunities in other schemes. If such mTXOPs are stopped prematurely and frequently, performance will degrade. We find that even though this situation can arise, its impact is likely to be insignificant.

Broken mTXOPs can be due to channel noise and collisions. Opportunistic routing schemes are mainly designed for mitigating the former issue, i.e., when the link between the source and the destination is error prone, and links between the source and forwarders and the links between the forwarders and the destination are more reliable.

As for the latter issue, collisions may be classified into intra-path or inter-path collisions. Intra-path collisions refer to collisions between stations (including the source and the destination) that are on the path from the source to the destination, whereas inter-path collisions refer to the collisions between stations on and off the path.

Intra-path collisions introduce two issues. First, two transmissions from on-path stations can overlap when stations cannot hear each other's transmissions (for example a lower priority forwarder does not hear a relayed transmission by a higher priority forwarder). This is essentially a hidden-terminal scenario. To mitigate such effects, we can use a small number of forwarders (see remark (4) below), and our results indicate that using up to 5 forwarders works well over a wide range of network conditions. In this case, the collision rate is not high. Second, there may be local traffic at a forwarder waiting for transmission. To reduce collisions between relayed and local traffic, when relaying an overheard frame, a forwarder aggregates local packets (if the frame is not large enough) so that both multi-hop and local packets are sent in one transmission.

Inter-path collisions can happen either because the transmitters are hidden terminals or happen to use the same backoff timer and starts transmissions at the same time. We call the former collision as hidden collisions and the latter collisions as regular collisions. Both hidden and regular collisions can degrade network performance. Regular collisions happen infrequently due to randomized backoff timer selection. Hidden terminals are more damaging. But fortunately, recently measurement studies in real networks show that hidden collisions only account for less than 5% of all losses (Fig. 9 in [13]). Interestingly, under both regular and hidden collisions, RIPPLE can out-perform the other schemes, as shown in Section IV-B due to packet aggregation.

4) Maximum Number of Forwarders: Using a small number of forwarders can limit intra- and inter-path collisions. If the number of forwarders is large, collision can become frequent, as reported in [15] (i.e., if there are too many forwarders, collisions can be so frequent that final performance is worse than that of using single path routing approaches). The number of forwarders refers to the actual number of forwarders used on a given path, not the number of potential forwarders in the network. The forwarder selection method (e.g., [8] [14]) ensures that a short path is selected, i.e., not all potential forwarders will be chosen as forwarders. We leave the maximum number of forwarders as an open design parameter. In this paper, we use 5 as the default maximum forwarders since it works well under a wide range of network conditions. In Section IV-C, we also consider up to 7 forwarders.

5) Self-Adaptability to Traffic Demands: Packet aggregation requires multiple packets in the queue. Waiting for some time before transmission allows more packets to be accumulated in the queue and increases the opportunity for packet aggregation. However, it also increases delay. To limit the delay and better support interactive traffic, we use zero waiting time [19]. With zero waiting, the source simply aggregates and transmits the packets currently in the sending queue (if it does not exceed the maximum allowed number). This simple scheme adapts automatically to changing network conditions. Specifically, when the network is heavily loaded, a queue backlog will develop, which allows us to inject larger frames to the network, thereby improving efficiency and alleviating overhead. If the network load decreases, smaller frame sizes will be automatically selected due to a smaller number of packets available in the queue.

6) MAC Layer Queues: There are two types of queues at the MAC layer: a sending queue (Sq) and a receiving queue (Rq). There is no queue used at the forwarders. With the Sq, we can i) aggregate packets into large frames, ii) keep packets until they are acknowledged. With the Rq, we can keep incoming packets if they arrive out of order, i.e., we only pass in-order packets to the upper layer. Note that a new type of packet re-ordering may happen without Rq due to the use of packet aggregation. Specifically, with packet aggregation, multiple packets can be transmitted in a large frame. Due to channel noise, some packets in the same frame may be corrupted but others are correctly received by the destination. If the corrupted packets have lower sequence number than correct packets, Rq is required to cache the correct packets temporarily and wait for the corrupted packets to be retransmitted.

7) Piggyback on MAC ACKs: After receiving a data transmission from the source, the destination first replies with a MAC ACK, then sends a data packet back to the source if there are any packets in the Sq at the destination. Potentially, it is possible to aggregate the MAC ACK and new data packet into the same frame in order to reduce transmission overhead. However, such aggregation creates signaling difficulties: if we piggy-back data packets on MAC ACKs, ACK transmission time now depends on how much data packets we piggy-back, which makes it hard to set the timeout at the source. Therefore, in this paper we do not piggy data packets onto MAC ACKs.

8) Co-existence With IEEE 802.11: Similarly to IEEE 802.11n, the RIPPLE scheme basically uses packet aggregation whenever possible while keeping other aspects of IEEE 802.11 (e.g., backoff, DIFS, etc.) unchanged. It thus can coexist well with IEEE 802.11.

IV. Evaluation

We implemented RIPPLE in NS-2 along with several existing routing protocols, including predetermined routing, shortest path routing, preExOR, MCExOR, and an IEEE 802.11n-like single-hop packet aggregation scheme called AFR [19]. All results presented are averages over multiple runs. Due to packet re-ordering and signaling issues introduced in Section II, the performance of the preExOR and MCExOR schemes is consistently worse than predetermined routing schemes, so we omit their results in interest of brevity.

To evaluate performance under erroneous and colliding channel states (including intra-path and inter-path collisions, as described in Section III Remark (3)), we use a combination of frame and bit error models.

For the frame models, the Shadowing model of NS-2 is used in which frame losses are proportional to the distance between stations. Note that this model assumes that losses between the source and different forwarders are independent. Therefore, intra-path and inter-path collisions occur in a random manner. The selected shadowing model parameters are as follow: path loss exponent 5, shadowing deviation 8, transmission power 281 mW.

The type of bit error models is important for us to fairly compare other schemes with packet aggregation schemes, since the effectiveness of partial retransmissions depends on bit error model. We use a widely used independent and identically distributed (i.i.d.) BER model. Since TCP congestion control views packet losses as an indicator of congestion, TCP throughput is strongly dependent on the link loss rate (e.g., [11] [12]) and too high a loss rate may result in low utilization of the wireless channel. To study the impact of channel noise, we use a BER of 10^{-5} and 10^{-6} to simulate a "noisy" and a "clear" channel state, respectively.

We first present the performance using the topology in Fig. 1 (We will further show results for two larger topologies in Section IV-F). There are initially three flows in this topology: the sources of flows 1, 2 and 3 are station 0, 0 and 5, while the destinations are 3, 4 and 7. We consider three sets of predetermined routes for these flows and call them ROUTE0, ROUTE1 and ROUTE2. In Table II we list these routes. For example, if we use ROUTE0, flow 1 would have a route from 0 to 3 via stations 1 and 2. In the results figures (Figs. 3 and 4) we report results when only flow 1, both flows 1 and 2, and all flows 1, 2 and 3 are activated at the same time, respectively.

For predetermined routing methods (i.e., using routes ROUTE0, ROUTE1 and ROUTE2), we use the standard IEEE 802.11 DCF and the AFR scheme (which is similar to IEEE 802.11n) [19] for packet aggregation. The AFR scheme is a packet aggregation extension of IEEE 802.11 DCF, and the maximum number of packets that can be aggregated into large frames is 16, which is the same as that used in the RIPPLE scheme. In this way, we can fairly compare AFR and RIPPLE and distinguish improvements due to packet aggregation and due to mTXOPs.

	Flow 1	Flow 2	Flow 3
ROUTE0	0 1 2 3	0 1 2 4	5 6 1 7
ROUTE1	0 1 3	0 1 4	5 6 7
ROUTE2	0 2 3	0 2 4	5 1 7

TABLE II
The used paths for the topology in Fig. 1.

A. Long-lived TCP Transfers

We first consider long-lived TCP transfers, which persistently send traffic throughout the simulation. To show that mTXOPs are necessary, we locate the stations and tune carrier/receiving ranges in such a way that one-hop routing is inefficient. That is, a single TCP flow's throughput with SPR (directly from station *0* to station *3*) is 0.76 Mbps when the BER is 10^{-6}, using the parameters in Table I. But, when ROUTE0 is used, throughput increases to 7.04 Mbps (see Fig. 3 for results of SPR and ROUTE0 running multiple flows).

With this network configuration, both intra- and inter-flow collisions can occur. So we evaluate the performance when collisions arise from both hidden and non-hidden stations. Furthermore, by using the BER model, we can assess the effect of channel noise that is independent from collisions.

We first compare the RIPPLE scheme when packet aggregation is turned off with the AFR scheme from [19]. These two schemes correspond to pure mTXOP without aggregation and pure aggregation without mTXOP, respectively. Fig. 3(a) shows the results when the ROUTE0 route is used. We can see that in comparison with the IEEE 802.11 DCF (marked as D), the pure mTXOP (marked as R1) and pure aggregation (marked as A) schemes achieve slightly higher and almost twice the throughput of DCF, respectively.

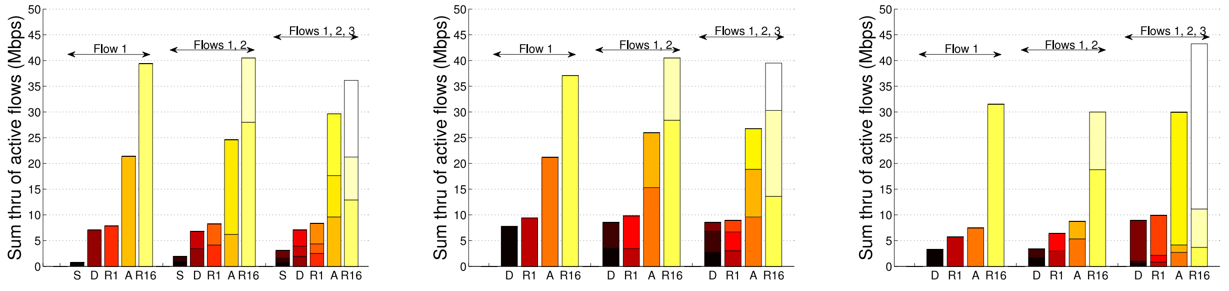
We then turn on the packet aggregation of the RIPPLE scheme. As shown in Fig. 3(a), the RIPPLE scheme (marked as R16) is able to take advantage of both mTXOP and packet aggregation and achieve even higher throughput. This indicates that the effectiveness of the RIPPLE scheme is due to both mTXOPs and packet aggregation.

To show that the RIPPLE scheme is able to support various selection of forwarder lists and priority assignments, Figs. 3(b) and 3(c) plot the results when the ROUTE1 and ROUTE2 routes are used. Comparing Figs. 3(a) 3(b) and 3(c), we note that the performance of RIPPLE is similar on both ROUTE0 and ROUTE1, while a significantly lower throughput is achieved on ROUTE2. Interestingly, regardless of the routes, the RIPPLE scheme consistently outperforms the other approaches.

Furthermore, channel noise may have a major impact on any MAC layer schemes. Fig. 4 shows the performance when the BER 10^{-5}. As before, RIPPLE consistently out-performs the other schemes.

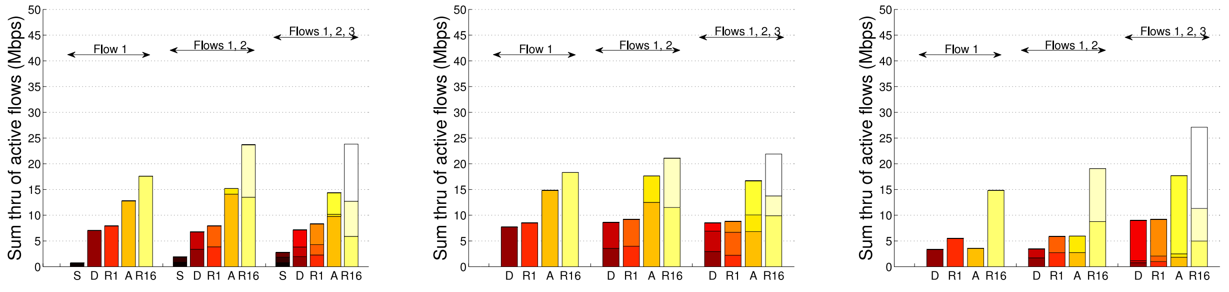
B. Effects of Regular and Hidden Collisions

Packet collisions can significantly degrade network performance. To show the effect of regular collisions, we use the topology shown in Fig. 5(a), where all the flows are within



(a) ROUTE0 (b) ROUTE1 (c) ROUTE2

Fig. 3. Measured throughput in Mbps for the topology in Fig. 1. 'S', 'D', 'R1', 'A' and 'R16' represent the SPR (directly from station *1* to station *3*) with DCF, IEEE 802.11 DCF, RIPPLE without packet aggregation, AFR and RIPPLE (with packet aggregation) schemes respectively. BER is 10^{-6} and other parameters listed in Table I.



(a) ROUTE0 (b) ROUTE1 (c) ROUTE2

Fig. 4. Measured throughput in Mbps for the topology in Fig. 1. 'S', 'D', 'R1', 'A' and 'R16' represent the SPR (directly from station *1* to station *3*) with DCF, IEEE 802.11 DCF, RIPPLE without packet aggregation, AFR and RIPPLE (with packet aggregation) schemes respectively. BER is 10^{-5} and other parameters listed in Table I.

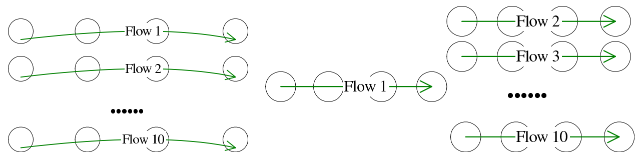
communication range of each other. Infrequent inter- and intra-path collisions can still happen due to the use of the Shadow model. Fig. 6(a) shows that the total throughput of all flows drops as expected when the number of flows increases. Moreover, RIPPLE outperforms the DCF and AFR schemes.

We further use the topology shown in Fig. 5(b) to evaluate the effect of hidden terminals. In that topology, the sources of flows 2–10 are completely hidden from the source of flow 1 (but not from the forwarders). Therefore, when the hidden traffic load becomes heavy, flow 1 can be throttled, as we see in Fig. 6(b) where the throughput of flow 1 is plotted as the number of hidden flows (flows 2–10, each sending 5×10^6 packets during the simulations) is increased from 0 to 9. Again, the RIPPLE scheme behaves better for less than 7 hidden flows. When there are 7, 8 and 9 hidden flows, the performance of RIPPLE is slightly worse than the other two schemes. This is because the hidden flows in RIPPLE use mTXOPs and can cause longer hidden collisions than DCF and AFR; when hidden collisions happen frequently, performance can be penalized. Note that in this extreme region, no scheme can achieve more than 3 Mbps throughput even though the physical layer rate is 216 Mbps.

C. Maximum Hops with Cross Traffic

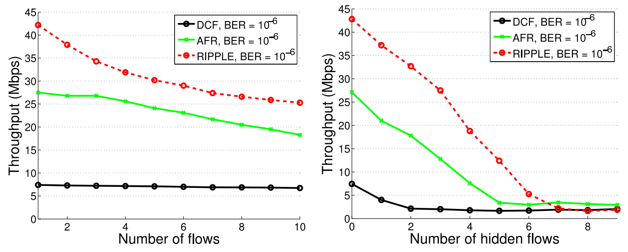
In this paper, we mostly use 5 as the maximum number of forwarders. In this section, we briefly introduce results when up to 7 hops are used since the maximum reported hops that we can find in the literature is 7 (see [8]).

For this aim, we use a line topology and increase the line



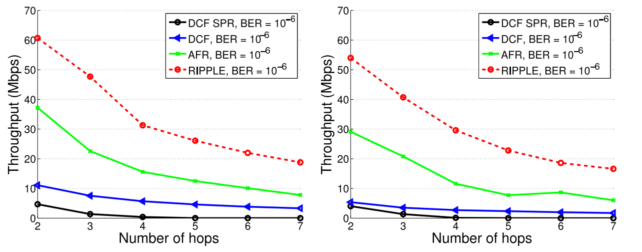
(a) For regular collisions (b) For hidden collisions

Fig. 5. Topologies used for illustrating the impact of regular and hidden collisions.



(a) Regular collisions (b) Hidden collisions

Fig. 6. Measured throughput in Mbps for regular and hidden collisions. BER is 10^{-6} and other parameters listed in Table I.



(a) Without Cross Traffic (b) With Cross Traffic

Fig. 7. Throughput in Mbps for up to 7 hops. BER is 10^{-6} and other parameters listed in Table I.

length from 2 to 7 hops with and without a 3-hop flow (sending 5×10^6 packets) intersecting it. As expected (see Figs. 7(a) and 7(b)), the throughput drops with increased distance with again the RIPPLE scheme achieving the best performance. Note that when the source/destination are 6 and 7 hops apart, they are not able to hear each other. This means that in the RIPPLE scheme, direct sending/receiving between the two ends of the path is not possible, and thus its performance depends entirely on the forwarders help. Results show interestingly that the forwarders do well.

D. Short-lived TCP Transfers: Web Traffic

In this section, we present results for short TCP transfers which mimic realistic web traffic ([23]). Web traffic consists of ON/OFF periods. During the ON time, a web user visits some web pages, and in the OFF time the user is reading what he/she just downloaded. To run the simulations in a realistic manner, traffic generated in ON periods should be long-range-dependent, i.e., resembles the aggregation of many ON-OFF sources with heavy-tailed ON periods. We use a transfer whose size follows a Pareto distribution with mean 80KB and shape parameter 1.5 in the ON time. In the OFF periods, no traffic is generated. The length of the OFF periods follows an exponential distribution with mean duration of one second.

The topology used for web traffic is the same as that used for long-lived TCP transfers (i.e., Fig. 1). However, there are now 10 short transfers between each source/destination pair. Namely, between stations 0 and 3, 0 and 4, and 5 and 7, are flows 1–10, 11–20, and 21–30, respectively. In Fig. 8, we show the total throughput of all active flows. As we can see, RIPPLE outperforms the other two approaches even in presence of short web transfers.

E. VoIP

To investigate RIPPLE's ability of supporting interactive traffic, we further test it with VoIP traffic. VoIP is sensitive to both losses and delay/jitter, and thus is more challenging than TCP traffic. The standard evaluation metric for VoIP is Mean Opinion Score (MoS), which ranges from 1–5, where 1, 2, 3, 4, 5 correspond to that the perceived VoIP quality is *impossible*, *very annoying*, *annoying*, *fair* and *perfect*, respectively. MoS is commonly estimated from an R-factor as: 1, if $R < 0$; 4.5,

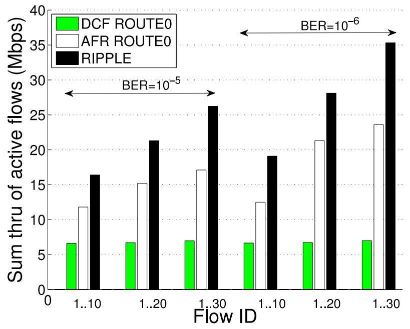


Fig. 8. Web traffic results in Mbps (sum throughput of all active flows) for the topology in Fig. 1. Parameters used are listed in Table I.

	BER=10^{-5}			BER=10^{-6}		
Flows	1..10	1..20	1..30	1..10	1..20	1..30
DCF ROUTE0	3.82	1.19	1.18	4.13	1.56	1.20
AFR ROUTE0	4.11	1.21	1.00	4.12	1.42	1.01
RIPPLE	4.11	2.49	1.75	4.14	2.82	2.09

TABLE III
MoS FOR VoIP TRAFFIC IN FIG. 1. THE PHYSICAL LAYER DATA AND BASIC RATES USED ARE BOTH 6MBPS, AND OTHER PARAMETERS LISTED IN TABLE I.

if $R > 100$; and $1 + 0.035 \times R + 710^{-6}R(R-60)(100-R)$, otherwise. The R-factor is obtained (as per [6]) from the expression $94.2 - 0.024d - 0.11(d - 177.3)H(d - 177.3) - 11 - 40 \log(1 + 10e)$, where d is the mouth-to-ear delay including coding/network/buffering delays, e is the total loss rate including losses in the network and those due to late arrivals, and $H(x) = 1$ if $x > 0$; 0 otherwise.

To simulate VoIP traffic, we model a 96kbs on-off traffic stream with on and off periods exponentially distributed with mean 1.5 seconds. Similar to [6], we aim to achieve a 177 ms mouth-to-ear delay, and a 52 ms delay for the wireless part. That is, packets arrived but with a >52 ms delay in wireless part are considered as losses. We use the same topology as for the web traffic for our evaluation. Table III summarizes the results. As we can see, the MoS achieved by the RIPPLE scheme is consistently higher than DCF and AFR schemes.

F. Large Scale Topologies with Low Rates

We now consider a typical Wigle topology, shown in Fig. 9, obtained from the Wigle database which contains measurements of real AP locations (the topology used corresponds to the connected part of Fig 3. in [22]). The main network consists of 8 wireless stations, and we added two additional stations S and R in order to simulate the impact of hidden collisions. The hidden traffic used is a TCP flow from S to R sending 1×10^6 packets.

Figs. 10(a), 10(c), 10(b), and 10(d) show throughput for TCP flows of eight randomly picked pairs of stations. We show the results with and without hidden stations (i.e. with and without traffic between stations S and R). Due to the small diameter of the network topology, most of the flows traverse 1–3 hop(s). The results show that the RIPPLE consistently

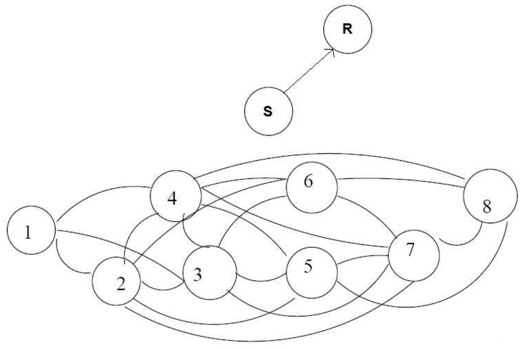


Fig. 9. A typical topology from the Wigle database, adapted from Fig. 3 of [22]

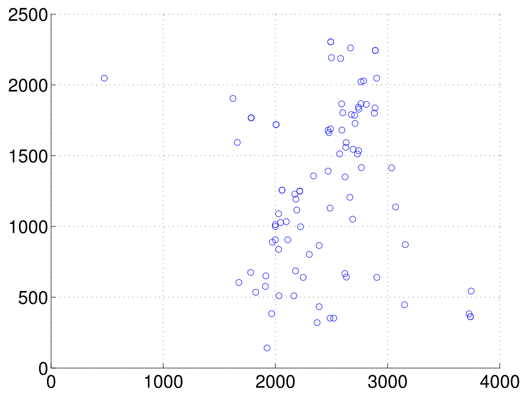


Fig. 11. The Roofnet topology. The unit for both x-axis and y-axis is meter.

outperforms predetermined routing with the DCF and AFR schemes, with up to 200% improvement (e.g., flow 8-7-5 in Fig. 10(a)).

Finally we consider the topology (see Fig. 11) of the MIT Roofnet, derived from the GPS coordinates file [4]. This topology is relatively large so we focus on transmissions between stations that are 4 or 5 hops apart. After picking station pairs to use as sources and destination, two more nearby stations are selected to act as the hidden terminals. As shown in Fig. 12, the RIPPLE scheme consistently outperforms the other two schemes, with up to 300% improvement (e.g., flow 5(1) in Fig. 12(a)).

V. Conclusions and Future Work

In this paper, we introduced a novel opportunistic routing, called RIPPLE. RIPPLE uses an expedited multi-hop transmission opportunity mechanism to ensure low signaling overhead and eliminate re-ordering, and uses a two-way packet aggregation technique to further reduce overhead. We implement the RIPPLE and other routing schemes in NS-2 and compare their performance under different traffic (e.g., long/short TCP transfers and VoIP) and over a wide range of network conditions (e.g., different wireless channel states, collision levels, and network topologies). Our results show that the RIPPLE scheme consistently delivers 100% – 300% performance gains over other approaches. As part of future work, we plan to further analyze the RIPPLE scheme and extend it to take advantage of multiple PHY data rates.

Acknowledgment

This research is supported by Irish Research Council for Science, Engineering and Technology, Science Foundation Ireland Grant 07/IN.1/I901, and National Science Foundation grants CNS-0546755 and CNS-0916106.

References

[1] *Part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications*, IEEE 802.11.

[2] *Part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications: Amendment 4: Enhancements for Higher Throughput*, IEEE 802.11n.

[3] http://pdos.csail.mit.edu/roofnet

[4] http://www.pdos.lcs.mit.edu/roofnet/roofnet-coords

[5] V. Bahl, Ranveer Chandra, Patrick P. C. Lee, Vishal Misra, Jitendra Padhye, Dan Rubenstein, and Yan Yu. "Opportunistic Use of Client Repeaters to Improve Performance of WLANs," Proc. of CoNext, 2008.

[6] A. Balasubramanian, R. Mahajan, A. Venkataraman, B. N. Levine, and J. Zahorjan, "Interactive WiFi Connectivity For Moving Vehicles," Proc. of ACM SIGCOMM, 2008.

[7] S. Biswas and R. Morris, "Opportunistic Routing in Multi-Hop Wireless Networks," Proc. of HotNets, 2003.

[8] S. Biswas and R. Morris, "ExOR: Opportunistic MultiHop Routing for Wireless Networks," Proc. of ACM SIGCOMM, 2005.

[9] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading Structure for Randomness in Wireless Opportunistic Routing," Proc. of ACM SIGCOMM, 2007.

[10] T. Bu, Y. Liu, and D. Towsley, "On the TCP-Friendliness of VoIP Traffic," Proc. of IEEE INFOCOM, 2006.

[11] N. Celandroni, "Comparison of FEC types with regard to the efficiency of TCP connections over AWGN satellite channels," *IEEE Trans. on Wireless Commun.* vol. 5, no. 7, pp. 1735-1745, Jul. 2006.

[12] N. Celandroni, F. Davoli, E. Ferro, and A. Gotta, "Long-Lived TCP Connections Via Satellite: Cross-Layer Bandwidth Allocation, Pricing, and Adaptive Control," *IEEE/ACM Trans. Network*, vol. 14, no. 5, pp. 1019-1030, Oct. 2006.

[13] Y.-C. Cheng, et. al., "Jigsaw: Solving the Puzzle of Enterprise 802.11 Analysis," Proc. of ACM SIGCOMM, 2006.

[14] D. S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. "A high-throughput path metric for multi-hop wireless routing," Proc. of ACM MOBICOM, 2003.

[15] Y. Ganjali, and A. Keshavarzian, "Load Balancing in Ad hoc Networks: Single-path Routing vs. Multi-path Routing", Proc. of IEEE INFOCOM, 2004.

[16] Y. Ganjali, and A. Keshavarzian, "Selection Diversity Forwarding in a Multihop Packet Radio Network with Fading Channel and Capture," *ACM MC2R*, 2001.

[17] S. Katti, D. Katabi, H. Balakrishnan, and M. Medard, "Symbol-level Network Coding for Wireless Mesh Networks," Proc. of ACM SIGCOMM, 2008.

[18] M. Kurth, A. Zubow, and J. P. Redlich, "Cooperative Opportunistic Routing using Transmit Diversity in Wireless Mesh Networks," Proc. of IEEE INFOCOM, 2008.

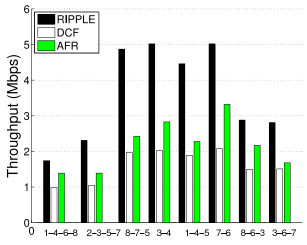
[19] T. Li, Q. Ni, D. Malone, D. Leith, T. Turletti, and Y. Xiao, "Aggregation with Fragment Retransmission for Very High-Speed WLANs," *IEEE/ACM Transactions on Networking*, Apr. 2009.

[20] T. Li, D. Leith, and L. Qiu, "Opportunistic Forwarding for Interactive Traffic in Multi-hop Wireless Networks," *Technical Report*, 2009, http://www.hamilton.ie/tianji_li/ripple.tr.pdf.

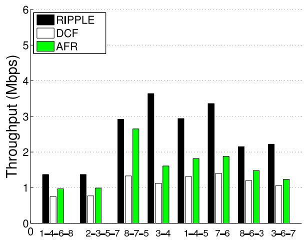
[21] Y. Lin, B. Li, and B. Liang, "CodeOR: Opportunistic routing in wireless mesh networks with segmented network coding," Proc. of ICNP 2008.

[22] A. Mishra, V. Shrivastava, D. Agarwal, S. Banerjee, and S. Ganguly, "Distributed Channel Management in Uncoordinated Wireless Environments," Proc. of ACM MobiCom, 2006.

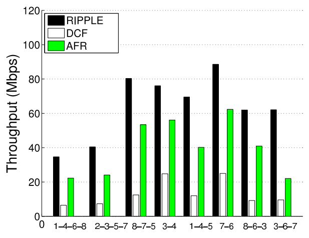
[23] R. S. Prasad, C. Dovrolis, and M. Thottan, "Router Buffer Sizing Revisited: The Role of the Output/Input Capacity Ratio," Proc. of CoNext, 2007.



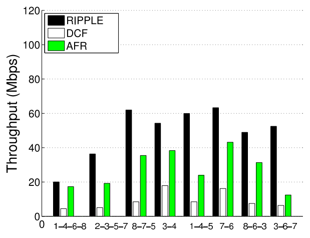
(a) 6Mbps



(b) 6Mbps with hidden terminals

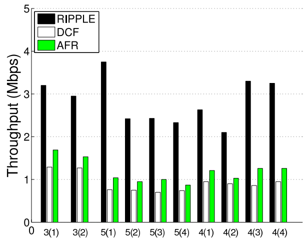


(c) 216Mbps

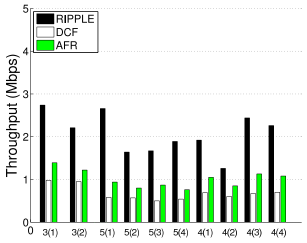


(d) 216Mbps with hidden terminals

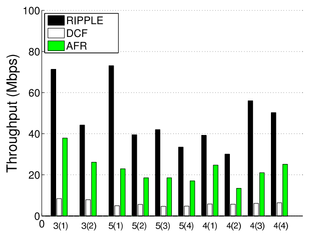
Fig. 10. Throughput measurements for the topology in Fig. 9. The labels on the x-axis indicate the flow path concerned. For example, '1-4-6-8' means that the flow is from station 1 to 4 with stations 4 and 6 as forwarders.



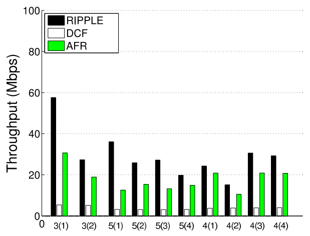
(a) 6Mbps



(b) 6Mbps with hidden terminals



(c) 216Mbps



(d) 216Mbps with hidden terminals

Fig. 12. Results for the Roofnet topology in Fig. 11. The labels on the x-axis indicate the number of hops and the number of each test. For example, '3(1)' means there are 3 hops between the source and the destination and this is the 1st 3-hop example, with '3(2)' meaning the 2nd 3-hop example.

[24] R. Ramanathan, "Challenges: a radically new architecture for next generation mobile ad hoc networks," Proc. of ACM MobiCom, 2005.

[25] E. Rozner, J. Seshadri, Y. A. Mehta, and L. Qiu, "SOAR: Simple Opportunistic Adaptive Routing Protocol for Wireless Mesh Networks," IEEE Transactions on Mobile Computing. 2009.

[26] D. Tang and M. Baker, "Analysis of A Local-Area Wireless Network," Proc. of ACM MobiCom, Aug. 2000.

[27] Z. Zhao, S. Darbha, and A. L. N. Reddy, "A Method for Estimating the Proportion of Nonresponsive Traffic At a Router," *IEEE/ACM Transactions on Networking*, vol. 12, no. 4, pp. 708–718, Aug. 2004.

[28] A. Zubow, M. Kurth, and J. P. Redlich, "Multi-Channel Opportunistic Routing," European Wireless Conference 2007.