Have Your Cake and Eat It Too! Enabling Frequency Diversity Through Opportunism

Edmund L. Wong and Sangmin Lee
The University of Texas at Austin

1. INTRODUCTION

The broadcast nature of wireless networks is both a boon and a bane. On the one hand, multiple receivers may overhear a single transmission on the same channel at no additional cost above a point-to-point transmission; this property has been exploited in many *opportunistic* applications (*e.g.*, [2]). On the other hand, the shared nature of the medium effectively forces wireless nodes to leverage frequency diversity and channelization for efficiency. Unfortunately, coordinating communication between nodes that may switch channels at any time is difficult and may in fact offset any performance gains from leveraging frequency diversity.

In this paper, we describe *lazily-assisted channel hopping* (LACH) which utilizes opportunism to enable more frequency diversity. LACH consists of two techniques: *flexible channel hopping*, which gives nodes nearly complete freedom in switching channels in order to efficiently leverage available channels; and *lazy opportunistic gossip*, which helps nodes find one another by having nodes gossip schedules when unacknowledged transmissions are opportunistically overheard. Through these techniques, LACH enables nodes to dynamically adjust to diverse workloads, flexibly allocate their bandwidth, and efficiently find other nodes of interest.

2. FLEXIBLE CHANNEL-HOPPING

LACH is a link-layer protocol that takes advantage of frequency diversity in IEEE 802.11 ad-hoc wireless networks by hopping between channels in a way similar to SSCH [1]. LACH divides up time into fixed-interval slots, and nodes, assumed to be roughly synchronized, switch to a specific channel at the beginning of each slot. Nodes advertise a compact schedule, which provides some hints as to which channels a node may switch to in the future, by broadcasting it once per slot. A node that has data to send uses this schedule to find a particular receiver.

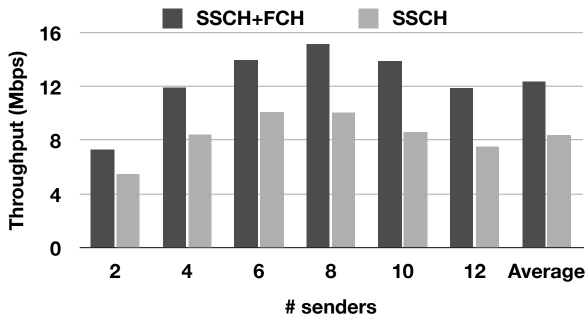

Figure 1: Throughput of SSCH, with and without FCH.

However, a LACH node is not required to follow its schedule. LACH nodes employ *flexible channel hopping* (FCH), which allows them to switch channels *at any time* in order to synchronize with peers they are interested in sending to. This flexibility enables nodes to dynamically adjust for different workloads and arbitrarily allocate bandwidth to receivers. For example, if a node only has a few packets to send to many receivers, it can quickly switch between channels within a slot and send to different receivers. Moreover, if a node ends up switching to the wrong channel, it can immediately try other channels without waiting for the next slot.

LACH's flexibility is in stark contrast to SSCH, in which nodes select a single channel for each slot and wait an entire slot duration before switching again. Although waiting can be reduced by shortening the slot duration, shorter slots actually degrade performance by (a) making it more difficult for senders to synchronize with receivers; and (b) increasing SSCH's overhead (*e.g.*, by increasing the number of schedule broadcasts). Because slots exist to *enable receivers to be found*, LACH nodes only consider switching channels

ACM CoNEXT Student Workshop, November 30, Philadelphia, USA.

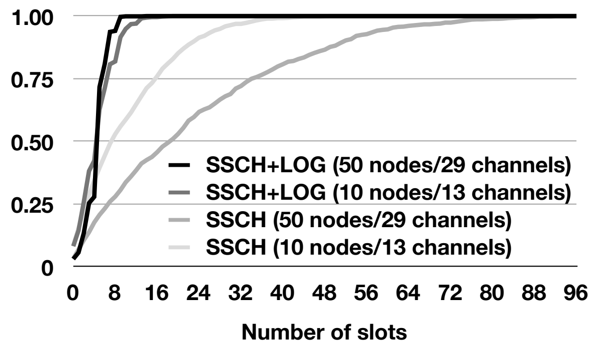

Figure 2: CDF showing how many slots a particular sender, with and without LOG, took to converge on a receiver in our simulation.

in the middle of slots in which they are receiving relatively little traffic.

Preliminary simulations show that this flexibility in channel hopping provides significant improvements in performance. Figure 1 shows the throughput achieved by SSCH, with and without FCH, in a QualNet simulation with 10 senders sending one 512-byte packet to every receiver every 2 ms on a 802.11a wireless network. Overall, FCH improved SSCH's throughput by about 48% compared to SSCH without FCH.

3. LAZY OPPORTUNISTIC GOSSIP

One possible downside of LACH's flexibility is that nodes may have a harder time finding other nodes, especially when nodes are continually adapting their schedules to communicate with many different nodes. Finding other nodes becomes even more difficult as the number of channels increase.

We address this difficulty through a novel technique called *lazy opportunistic gossip* (LOG). LOG leverages the broadcast nature of the wireless medium in order to gossip schedules on demand. Specifically, LACH nodes, using LOG, store a copy of every node's schedule, and any node (a *bystander*) that observes a transmission not acknowledged by its intended receiver assists the sender by broadcasting its own *view* of the receiver's schedule.

Because of the flexibility provided by LACH, nodes may have conflicting views about which channel a particular node is on. To determine which information is the most recent, a node maintains a version number which is incremented every time the node updates its own schedule. This version number is included whenever scheduling information is broadcast (from either the node itself or a bystander). Transmissions from a sender also embed the sender's view of the receiver's schedule and associated version number for this slot. Finally, a bystander only broadcasts its own view of some receiver's schedule (a) after backing off for a duration that depends on how recent it believes its view to be; and (b) if the bystander has not overheard a newer version of the schedule in the meantime.

Thus, in LOG, a node updates its view of another node's schedule by observing any transmission which contains a newer version of that node's schedule. In addition, if a node observes a transmission from another node that is not expected to be on its channel, the former node updates its view of the latter node's schedule.

To measure LOG's potential for improving node convergence, we perform a simple, idealized simulation of SSCH and LOG in an environment where there exists one receiver that every other node is trying to locate in order to send a single data packet. We start every node with a random schedule and assume that as soon as a sender overlaps on the same channel as the receiver, the sender successfully transmits its packet. Furthermore, we assume that when LOG is enabled, (a) a node that knows the receiver's schedule gossips it to all other nodes in the same channel; and (b) if a node has not transmitted its packet yet and learns the receiver's schedule from another node, it will switch to the receiver's channel in the following slot.

The CDF in Figure 2 shows our preliminary results on how many slots a particular sender takes to successfully transmit its packet. As expected, SSCH nodes take longer to find the receiver as the number of channels increase. However, as the number of nodes increases, LOG's ability to reduce convergence time improves, as a node is more likely to overlap with a node that knows the receiver's schedule. Overall, with 50 nodes/29 channels and 10 nodes/13 channels, LOG-enabled SSCH requires 5 slots in the median and 14 and 19 slots worst-case, respectively. Without LOG, SSCH performs significantly worse, requiring 19 and 8 slots in the median and 96 and 49 slots worst-case, respectively.

4. CONCLUSION

The airwaves by their very nature are shared by many nodes. Wireless networks exploit frequency diversity to efficiently utilize this precious resource. By giving nodes flexibility in how they allocate their bandwidth while providing opportunistic mechanisms to help nodes find one another, LACH significantly improves performance through efficient utilization of wireless spectrum.

5. REFERENCES

[1] P. Bahl, R. Chandra, and J. Dunagan. SSCH: Slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks. In *MobiCom '04*.

[2] S. Biswas and R. Morris. ExOR: Opportunistic multi-hop routing for wireless networks. *SIGCOMM CCR*, 35(4):133–144, 2005.