Demo: Floodcasting, A Data Dissemination Service Supporting Real-Time Actuation and Control

Ye-Sheng Kuo, Pat Pannuto, and Prabal Dutta
Electrical Engineering and Computer Science Department
University of Michigan
Ann Arbor, MI 48109
{samkuo,ppannuto,prabal}@umich.edu

ABSTRACT

Packet collisions have gone from something to be avoided to something that can be embraced. We build upon recent results that employ intentional packet collisions for synchronized flooding to show how multi-hop wireless networks can support real-time actuation and control. First, we show how a network of nodes can synchronize their LED transmissions to extend the range of a visual light communication (VLC) system. Second, we show how buffer-free, streaming audio is possible over a multi-hop wireless mesh network.

Categories and Subject Descriptors

B.4.1 [**HARDWARE**]: Input/Output and Data Communications—*Input/Output Devices*; C.3 [**COMPUTER-COMMUNICATION NETWORKS**]: Special-Purpose and Application-Based Systems

General Terms

Design, Experimentation, Measurement, Performance

Keywords

Software-Defined Radio, IEEE 802.15.4, Wireless MAC Protocol, Constructive Interference, Real-Time

1. INTRODUCTION

Recent work exploited the concurrent transmission of hardware generated ACKs to achieve a low-power MAC layer protocol [1]. Follow-on work by Ferrari et al. debuted Glossy, a MAC protocol that attempted to exploit constructively interfering data packets to provide a cheap, efficient flooding protocol [2]. Practical Glossy implementation was stymied, however, by the inability of commercial radios to decode interfering full-length packets due to the envelope effect, severely limiting the data rate [3].

In this demo we present *Floodcasting*, a MAC protocol that provides a completely buffer-free, near real-time, high bandwidth, and tightly synchronized flood channel. We leverage our software defined radio platform, μSDR ([3]), to overcome the limitations of commercial radios and enable a continuous stream of constructively interfering full length IEEE 802.15.4 packets.

MobiSys'13, June 25–28, 2013, Taipei, Taiwan.
ACM 978-1-4503-1672-9/13/06.

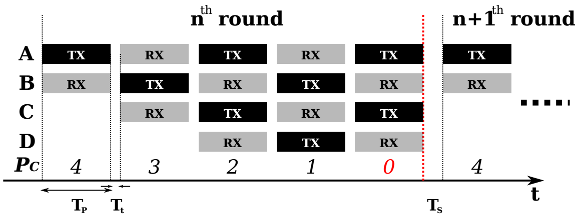

Figure 1: *Countdown Flooding*—Nodes A, B, C and D are all 1-hop from one another, connected linearly. In this example, node A initiates a countdown flood counting down from $P_C = 4$. At first, only node B receives the packet which it re-broadcasts after a turnaround time T_t with countdown value $P_C = 3$. This packet is then re-broadcast by nodes A and C with $P_C = 2$. This packet constructively interferes at node B and is simply received at node D. The re-broadcasting repeats until nodes A and C transmit a packet received by nodes B and D with $P_C = 0$. Upon completion of transmission/reception of the $P_C = 0$ packet, nodes A, B, C, and D simultaneously process the data packet.

2. DEMONSTRATION

To demonstrate the efficacy of Floodcasting, we demo two demanding applications built on our countdown flooding primitive (Fig 1) and deploy them on a testbed of μSDRs.

The first application is a tightly synchronized array of visual light communication (VLC) frontends. We show that by leveraging Floodcasting, the VLC drivers themselves can be made to generate constructively interfering waveforms to improve the range and coverage of VLC networks. We then build a buffer-free, real-time audio streaming application on top of a multi-hop wireless mesh network using Floodcasting. In this demonstration, we show that we can achieve sub-μs synchronization and high bandwidth communication across a multi-hop wireless mesh network.

3. REFERENCES

[1] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J. M. Liang, and A. Terzis. Design and evaluation of a versatile and efficient receiver-initiated link layer for low-power wireless. In *SenSys '10.*

[2] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. Efficient network flooding and time synchronization with glossy. In *IPSN '11.*

[3] Y.-S. Kuo, P. Pannuto, T. Schmid, and P. Dutta. Reconfiguring the software radio to improve power, price, and portability. In *SenSys '12.*