Competition: Towards Low-Power Wireless Networking that Survives Interference with Minimal Latency

Beshr Al Nahas, Olaf Landsiedel
Department of Computer Science and Engineering
Chalmers University of Technology, Sweden
beshr, olafl @chalmers.se

Abstract

Low-power wireless networking needs to survive interference in order to accommodate the requirements of serious applications of Internet of Things. Synchronous transmission techniques like Glossy and Chaos perform well under normal operating conditions. However, their data delivery latency suffers under interference even when extended to use channel hopping. In this paper we discuss our techniques to enhance the robustness of synchronous flooding while keeping the latency and power consumption minimal.

1 Introduction

Dependable low-power wireless protocols are a key enabler for serious applications of Internet of Things. Data is too valuable to be lost, timeless delivery is atleast favorable and low-power operation is crucial for battery driven applications. However, interference from ubiquitous wireless devices or from malicious agents deeply challenge the low-power wireless communication as it causes data loss, increased latency and wasted energy.

In this paper, we try to incorporate what we learned from our previous participation in the competition [1], and we present a simple protocol that tries to achieve and maintain a low-latency and low-power communication scheme that is robust against interference. We first present the synchronous flooding technique in §*2* that is the basis of the design of our protocol, then we discuss frequency diversity techniques to increase robustness against interference in §*3*. In §*4* we discuss the trade-offs that we need to consider when communicating asynchronous events with our synchronous protocol, and we conclude in §*5*.

2 Synchronous Flooding in A Nutshell

Synchronous flooding is organized in periodic rounds of slotted operation. Nodes communicate in the active slots portion of the round and sleep until the start of the next period.

International Conference on Embedded Wireless
Systems and Networks (EWSN) 2017
20–22 February, Uppsala, Sweden
© 2017 Copyright is held by the authors.
Permission is granted for indexing in the ACM Digital Library
ISBN: 978-0-9949886-1-4

In each slot, the nodes decide autonomously to send, receive or turn the radio off. One node –which we designate as the initiator– starts the flooding round. Each other node waits to hear a packet, learns the flooding interval from this packet, sets the acknowledgment flag if it is in the designated set of destinations and decides whether to repeat the packet it heard in the next slot. Slowly, nodes in the network all get the data and join the flood, and they propagate the acknowledgment flags in the same time. With this strategy we achieve a one-to-many data dissemination.

The flood will stop either a) after repeating for a maximum number of slots, or b) after hearing a packet signaling acknowledgment.

2.1 Why does it work? Capture effect.

In synchronous flooding, multiple nodes transmit in the same time. This causes packet collisions and could cause packet corruption. However, wireless receivers can recover one of the colliding packets under specific conditions concerning the signal strength and collision timing. The chance to recover the packet is much higher if the packets meet in the air during the start-of-frame delimiter (SFD), and if one of the packets has a higher signal strength (2-3 dB) than the others. This is usually referred to as capture effect. The capture effect was documented earlier in FM receivers [6], and was recently employed by state-of-art low-power wireless protocols such as Chaos [5]. Moreover, if the colliding packets have exactly the same contents, and if they are tightly synchronized, then the packets have a higher chance of recovery due to non-destructive interference [2]. This effect was exploited in the design of a family of protocols that Glossy [3] started and inspired the design of Chaos [5] as well.

2.2 Why is it robust?

Flooding is robust by design. Network flooding is a greedy strategy that exploits all the possible paths toward all possible destinations. It does not need to keep track of links status, routing information or topology changes. If there is a path to a node, the flood will eventually pass it.

2.3 What makes it low-power?

In the target platform, the radio is the most power consuming device; thus, we duty-cycle the radio; *i.e.*, we turn the radio off when we are done sending or receiving, when we do not have data to transmit or when we do not receive a valid packet within the start of the slot. Moreover, this

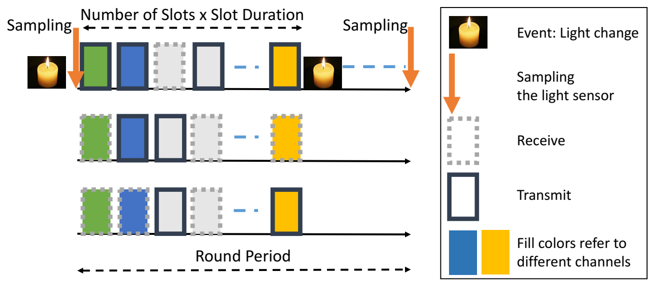
Sampling
Number of Slots x Slot Duration
Sampling
Round Period
Event: Light change
Sampling the light sensor
Receive
Transmit
Fill colors refer to different channels

Figure 1. Illustration of operation: The initiator samples the light sensor and reports the light changes. It communicates the change in the start of the next period. The nodes that hear the packet relay it and they hop the channel in each slot according to a shared hopping sequence.

protocol does not have an overhead for *e.g.*, route discovery, and the nodes correct their synchronization based on the data packets they exchange without needing extra signaling.

3 Surviving Interference

A common way to disrupt wireless communication is to interfere the operating frequencies. In the case of synchronous flooding, this interference might block the communication for sometime, but it will not incur additional overhead for path discovery or similar recovery mechanisms. However, if the interference blocks the communication for a reasonably long period, the nodes in the network might have their clocks drifting away and they need to re-synchronize with the initiator. Nevertheless, we try to avoid the interfered channels by using multiple channels for communication as we explain next.

3.1 Channel Hopping

Inspired by Bluetooth and WirelessHART [4], we extend the synchronous flooding mechanism to use multiple channels where the nodes rapidly change the communication channel for every time slot. To avoid the overhead of channel rendezvous, we use a common pseudo-random hopping-sequence in which the nodes hop to a specific channel according to the slot number.

How does this help? If a node is not reachable on one or more channels due to interference, but there are other clear channels, then it is a matter of time until the neighbors hit a clear channel to reach that node. So whenever there exists a path to a node on any channel, we will eventually find it although it might take a large number of tries; thus, increasing latency.

3.2 Extreme Channel Hopping

Using one channel for the whole network have the following drawbacks; a) nodes could suffer from internal interference, and b) a large-scale external interference on one channel could block the flood progress in the whole network.

To overcome these limitations we divide the network into clusters and let the clusters have their own hopping sequences. However, planning the clusters dynamically will incur a large signaling overhead. Instead, we want the clusters to form autonomously. To achieve this, we propose using multiple channels in parallel in the network where the nodes autonomously and randomly choose the channel they want to communicate on. This might lead to corner cases where only one node is active on a specific channel and nobody can communicate to it in a specific time-slot. However, due to randomization, this situation will probably solve itself in the next slots. With this technique we increase the resilience against interference since the network is more frequency diverse, and it allows the flood to progress faster on multiple channels in parallel.

4 Communicating Asynchronous Events

Our approach is using a synchronous protocol to handle asynchronous events as illustrated in Figure *1*. In the best case, the event might happen just before starting the communication round; thus, resulting in latency in the order of the number of time slots needed to for the packet to reach the destination in the case of successful communication. On the other hand, the event might happen just after ending the communication slots in the round; resulting in latency in the order of the round period. On average, the latency is

$$Latency = \frac{RoundPeriod}{2} + NumberOfSlots \times SlotDuration$$

This leads us to the following trade offs: a) To achieve a lower latency, we need to communicate more often, b) to achieve more robustness against interference, we need more communication slots, and c) to achieve a lower energy usage, we need to communicate less often, and we need to use less communication slots per round. However, both the average and maximum rate of events are known which will make it easier to balance this trade off.

5 Conclusion

In this paper, we present a synchronous flooding protocol that incorporates channel hopping to survive interference and to decrease latency. We keep the design simple to limit the overhead of the protocol and because we believe simplicity helps minimizing the number of surprising bugs.

6 Acknowledgments

This work was supported by the project *ChaosNet: Distributed Computing for Low-Power Wireless Networks* that was funded by the Swedish Research Council (VR).

7 References

[1] B. Al Nahas and O. Landsiedel. Competition: Towards low-latency, low-power wireless networking under interference. In *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks*, EWSN '16, pages 287–288, USA, 2016. Junction Publishing.

[2] M. S. Chun-Hao Liao, Yuki Katsumata and H. Morikawa. Revisiting the so-called constructive interference in concurrent transmission. In *Proceedings of the 41st IEEE Conference on Local Computer Networks*, 2016.

[3] F. Ferrari, M. Zimmerling, L. Thiele, and O. Saukh. Efficient network flooding and time synchronization with glossy. In *IPSN: Proc. of the ACM/IEEE International Conference on Information Processing in Sensor Networks*, 2011.

[4] HART Communication Foundation. HCF_SPEC-065, 2.4GHz DSSS O-QPSK Physical Layer Specification. *HCF_SPEC-065, Rev 1.0*, 2007.

[5] O. Landsiedel, F. Ferrari, and M. Zimmerling. Chaos: Versatile and efficient all-to-all data sharing and in-network processing at scale. In *SenSys: Proc. of the ACM Conference on Embedded Networked Sensor Systems*, 2013.

[6] K. Leentvaar and J. H. Flint. The capture effect in FM receivers. *IEEE Transactions on Communications*, 24(5):531–539, 1976.